GOVERNMENT OF TAMILNADU

ہائر سیکنڈری سال اول

HIGHER SECONDARY FIRST YEAR

کمپیوٹر سائنس

COMPUTER SCIENCE

(URDU MEDIUM)

Department of School Education

Untouchability is Inhuman and a Crime

11th_CompSci_FM_UM.indd 1 20-04-2023 14:46:08

Government of Tamil Nadu

First Edition - 2023

Reprint - 2024

(Published under New syllabus)

Content Creation

State Council of Educational
Research and Training
© SCERT 2023

Printing & Publishing

Tamil NaduTextbook and Educational
Services Corporation

www.textbooksonline.tn.nic.in

11th_CompSci_FM_UM.indd 2 20-04-2023 14:46:08

پیش لفظ

انسانی تہذیب نے کمپیوٹر کی نشوونما کے ساتھ کافی ترقی کی ہے اسی لیے اس زمانہ کو" کمپیوٹر کا زمانہ" کہا جاتا ہے۔ آج کے زمانہ میں پڑھے لکھے وہی لوگ ہوتے ہیں جن کو کمپیوٹر کے استعمال کا طریقہ معلوم ہوتا ہے، جبکہ دوسرے لوگوں کو حاصل شدہ دیگر ڈگریوں کے باوجود ان پڑھ سمجھا جاتا ہے۔ اس وقت قوم کی نشوونما نوجوانوں کے ہاتھوں میں ہے، لہذا اس کتاب کے مضامین کو نوجوانوں کے مستقبل کی ضروریات کو مد نظر رکھتے ہوئے تیار کیا گیا ہے ، تاکہ نوجوان نسل اس سے بھرپور فائدہ حاصل کرسکے۔

کتاب کا استعمال کیسے کریں؟

- اس کتاب میں کمپیوٹر ٹکنالوجی میں پیشگی معلومات کی ضرورت نہیں ہے۔
- ہر یونٹ سادہ سرگرمیوں اور مظاہروں پر مشتمل ہے جو اساتذہ اور طلبا آسانی سے حل کر سکتے ہیں ۔
- تکنیکی تفہیمات کو آسانی سے سمجھنے کے لیے لغات میں اس کی وضاحت کردی گئی ہے۔
- کیا آپ جانتے ہیں؟ " بکس" اضافی معلومات کے ساتھ قاری کے علم کو تقویت بخشتے ہیں۔
- سافٹ ویئر ایپلی کیشنز کا استعمال کرتے ہوئے مشقوں کو حل کرنے کے لئے ورکشاپس متعارف کروائی گئیں ہیں۔
- QR کوڈ کو اضافی مدد سے جوڑنے کے لئے استعمال کیا جاتا ہے۔
- مواد ڈیجیٹل شکل میں۔•

!کوڈ استعمال کریں QR آیئے نصابی کتابوں میں

ZNYP5

- o نصابی کتاب میں QR کوڈز کو اسکین کرنے کے لیے QR کوڈ آئیکن پر کلک کریں۔
- o آلہ کی طرف اشارہ کریں اور QR کوڈ پر فوکس کریں۔
- o اسکین مکمل ہونے پر، QR کوڈ سے منسلک مواد درج ہو جاتا ہے۔

نوٹ: ICT کارنر کے لیے، Digi Links QR کوڈز کوئی دوسرا QR سکینر استعمال کرتے ہیں۔

11th_CompSci_FM_UM.indd 3 20-04-2023 14:46:08

CAREER GUIDANCE AFTER 12TH

COURSES	COLLEGES/ UNIVERSITIES	PROFESSION
B.E / B.Tech	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Software Engineer, Hardware Engineer, Software Development, Healthcare Section, IT & ITEs
Science and Humanities		
B.Sc (Computer Science) BCA B.Sc (Maths, Physics, Chemistry, Bio-Chemistry, Geography, journalism, Library Sciences, Political Science, Travel and Tourism)	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Government Job and Private Company BPO, Geologist, Journalist
LAW		
LLB B.A+LLB B.Com BBM+LLB BBA+LLB	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Lawyer, Legal Officer, Govt Job
CA	The Institute of Chartered Accountant of India (ICAI)	CA Private and Govt.
Diploma	Government Polytechnic and Self-financing colleges	Junior Engineer (Government and Private)
Commerce Courses		
B.com-Regular, B.com-Taxation & Tax Procedure, B.com-Travel &Tourism, B.com-Bank Management, B.com-Professional, BBA/BBM-Regular, BFM- Bachelors in Financial Markets, BMS-Bachelors in Management Studies, BAF- Bachelors in Accounting & Finance, Certified Stock Broker & Investment Analysis, Certified Financial Analyst, Certified Financial Planner, Certified Investment Banker	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Private Organization , Government ,Banking sectors and prospects for self – employment.

11th_CompSci_FM_UM.indd 4 20-04-2023 14:46:08

COURSES	COLLEGES/ UNIVERSITIES	PROFESSION
Management Courses		
Business Management Bank Management Event Management Hospital Management Human Resource Management Logistics Management	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Private Organization , Government ,Banking sectors and prospects for self – employment.
LAW		
LLB B.A+LLB B.Com BBM+LLB BBA+LLB	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Lawyer, Legal Officer, Private Organization , Government, Banking sectors and prospects for self – employment
CA-Chartered Accountant CMA-Cost Management Accountant. CS-Company Secretary (Foundation)	The Institute of Chartered Accountant of India (ICAI)	CA, Private Organization, Government ,Banking sectors and prospects for self – employment.
Science and Humanities		
B.Sc.Botany B.Sc.Zoology B.Sc.Dietician & Nutritionist B.Sc.Home Science B.Sc.Food Technology B.Sc.Dairy Technology B.Sc. Hotel Management B.Sc. Fashion Design B.Sc. Mass Communication B.Sc. Multimedia B.Sc. -3D Animation	All University and their affiliated Colleges and Self financing Colleges in India and Abroad	Government Job and Private Company BPO, Geologist, Journalist
LAW		
LLB B.A+LLB B.Com BBM+LLB BBA+LLB	All University and their affiliated Colleges and Self financing Colleges in India and Abroad.	Lawyer, Legal Officer, Govt Job
CA	The Institute of Chartered Accountant of India (ICAI)	CA Private and Govt.
Diploma	Government Polytechnic and Self-financing colleges	Junior Engineer (Government and Private)

11th_CompSci_FM_UM.indd 5 20-04-2023 14:46:08

فہرست مضامین

باب نمبر	عنوان	صفحہ	مہینہ
	اکائی I - فنڈامینٹل آف کمپیوٹر اور ٹیپیکل آپریٹنگ سسٹم کے ساتھ کام کرنا (ونڈوز اور لینکس)		
1	**کمپیوٹرز کا تعارف**	1	جون
2	نمبر سسٹم	14	جون
3	کمپیوٹر تنظیم	41	جون
4	**آپریٹنگ سسٹم کے نظریاتی تصورات**	50	جولائی
5	**ونڈوز آپریٹنگ سسٹم کے ساتھ کام کرنا**	57	جولائی
	اکائی II - الگورتھمک مسئلہ کا حل		
6	تفصیلات اور خلاصہ	76	جولائی
7	کمپوزیشن اور ڈی کمپوزیشن (Composition and Decomposition)	88	اگست
8	آئی ٹریشن و ریکرشن (دہراؤ اور اعادہ) (Iteration and recursion)	102	اگست
	اکائی III - کا تعارف C++		
9	C++ کا تعارف	115	ستمبر
10	فلو آف کنٹرول (Flow of Control)	152	ستمبر
11	فنکشنس (Functions)	180	اکتوبر
12	اریز اور اسٹرکچرس (Arrays and Structures)	206	اکتوبر
	اکائی IV - C++ کے ساتھ آبجیکٹ اورینٹیڈ پروگرامنگ		
13	آبجیکٹ اورین ٹیڈ پروگرامنگ تکنیکس کا تعارف	228	اکتوبر
14	کلاسز اور آبجیکٹس (Classes and objects)	233	نومبر
15	پولیمورفزم (Polymorphism)	249	نومبر
16	انہیریٹینس (Inheritance)	260	دسمبر
	اکائی V - کمپیوٹر اخلاقیات اور سائبر سیکورٹی۔		
17	کمپیوٹر اخلاقیات اور سائبر سیکیورٹی	275	دسمبر
18	تمل کمپیوٹنگ	286	دسمبر
	عملی مشق (Praticals)	298	

W0J9P

E - book

HIWV1

Assessment

11th_CompSci_FM_UM.indd 6 20-04-2023 14:46:08

باب 1

کمپیوٹرز کا تعارف

سیکھنے کے مقاصد (Learning Objectives)

اس باب میں تصورات سیکھنے کے بعد ، طلباء

- کمپیوٹر کے بارے میں جاننے۔
- کمپیوٹر کے بنیادی کاموں کو سمجھنے۔
- اجزاء اور ان کے افعال کو جاننے۔
- کمپیوٹر کے بوٹنگ کے بارے میں جاننے کے قابل ہوجائیں گے۔

Father of Computer

کیا آپ جانتے ہیں؟

چارلس بیبج (Charles Babbage) کو 1837 میں اپنی ایجاد اور تجزیاتی انجن کے تصور کے لیے کمپیوٹر کا موجد سمجھا جاتاہے۔ تجزیاتی انجن میں ایک ریاضی منطقی یونٹ (ALU)، بنیادی بہاؤ کنٹرول، اور مربوط میموری شامل تھی۔ جس کی وجہ سے پہلے عام مقصد کے لیے کمپیوٹر تصور کی ترقی ہوئی۔



506BD

1.1 کمپیوٹر زکا تعارف (Introduction of Computers)

کمپیوٹر ہمارے اردگرد ہر جگہ نظر آتے ہیں، زندگی کے تمام شعبوں میں، تعلیم، تحقیق، سفر اور سیاحت، موسم کی پیشن گوئی، سوشل نیٹ ورکنگ، ای کامرس وغیرہ۔ کمپیوٹرز اب ہماری زندگی کا ناگزیر حصہ بن چکے ہیں۔ کمپیوٹرزنے اپنی درستگی اور کام کو انجام دینے کی رفتار سے ہماری زندگیوں میں انقلاب برپا کر دیا ہے، یہ واقعی قابل ذکر ہے۔ آج کوئی بھی ادارہ کمپیوٹر کے بغیر کام نہیں کر سکتا۔ درحقیقت مختلف ادارے پیپر لیس ہو چکے ہیں۔ کمپیوٹر سالوں کے دوران ایک سادہ کیلکولیشن ڈیوائس سے تیز رفتار پورٹیبل کمپیوٹرز تک تیار ہوئے ہیں۔

تیزی سے حساب لگانے کی ضرورت کے ساتھ کمپیوٹر انڈسٹری کی ترقی کا آغاز ہوا۔ کمپیوٹنگ کا دستی طریقہ سست اور غلطیوں کا شکار تھا۔ لہٰذا، تیز رفتار کیلکولیشن ڈیوائسز تیار کرنے کی کوششیں کی گئیں، یہ سفر پہلے معلوم کیلکولیشن ڈیوائس (Abacus) سے شروع ہوا جس کی وجہ سے آج ہم ایک انتہائی تیز رفتار کیلکولیشن ڈیوائسز تک پہنچ گئے ہیں۔

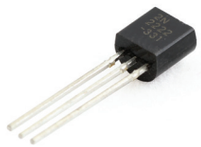
1.2 کمپیوٹرز کی نسلیں (Generations of Computers)

کمپیوٹر انڈسٹری میں ترقی کا تعین ٹیکنالوجی کی ترقی سے ہوتا ہے۔

ترقی کے مختلف مراحل کی بنیاد پر، کمپیوٹرز کو مختلف نسلوں میں تقسیم کیا جا سکتا ہے۔

نمبر شمار	نسل	دورانیہ	اہم جز کا استعمال	خوبیاں / نقصان
1	پہلی نسل	1940-1956	 ویکیوم ٹیوبس	• سائز میں بڑا • بجلی کا زیادہ استعمال ۔ • زیادہ گرمی کی وجہ سے خرابی۔ • مشینی زبان کا استعمال

11th_Comp Science_UM_Ch 1.indd 1 18-04-2023 11:51:11

پہلی نسل کے کمپیوٹرز - 1 UNIVAC ، EDVAC ،ENIAC ENIAC کا وزن تقریباً ۲۷ ٹن ہے، جس کا سائز 8 فٹ × 100 فٹ × 3 فٹ ہے اور تقریباً 150 واٹ بجلی استعمال ہوتی ہے۔				
2	دوسری نسل	1956-1964	 ٹرانزسٹر	• پہلی نسل کے مقابلے میں چھوٹا • کم حرارت پیدا کی گئی۔ • پہلی نسل کے مقابلے میں کم بجلی کی کھپت • پنچڈ کارڈز (Punched Cards) کا استعمال کیا گیا۔ • پہلا ترقی یافتہ آپریٹنگ سسٹم کا وجود- بیچ پروسیسنگ اور ملٹی پروگرامنگ آپریٹنگ سسٹم۔ • مشینی زبان کے ساتھ ساتھ اسمبلی زبان بھی استعمال کی گئی۔
دوسری نسل کے کمپیوٹرز IBM 1401، IBM 1620، UNIVAC 1108				
3	تیسری نسل	1964 -1971	 انٹیگریٹڈ سرکٹس (IC)	• کمپیوٹر چھوٹے، تیز اور زیادہ قابل اعتماد تھے۔ • کم بجلی کی کھپت • اعلیٰ سطحی زبانیں استعمال کی گئیں۔
تیسری نسل کے کمپیوٹرز IBM 360 سیریز، ہنی ویل 6000 سیریز				
4	چوتھی نسل	1971-1980	 مائکرو پروسیسر بہت بڑے پیمانے پر مربوط سرکٹس (VLSI)	• چھوٹا اور تیز • مائیکرو کمپیوٹر سیریز جیسے آئی بی ایم اور ایپل تھے۔ • ترقی یافتہ پورٹ ایبل کمپیوٹرز متعارف کرائے گئے۔
5	پانچویں نسل	1980 - تا حال	 الٹرا لارج اسکیل کا انضمام (ULSI)	• متوازی پروسیسنگ • سپر کنڈکٹرس • کمپیوٹر کا سائز کافی حد تک کم کر دیا گیا تھا۔ • امیجز اور گرافکس کو پہچان سکتا ہے۔ • مصنوعی ذہانت اور ماہرانہ نظام کا تعارف • فیصلہ سازی اور منطقی استدلال سمیت اعلیٰ پیچیدہ مسائل کو حل کرنے کی صلاحیت۔

11th_Comp Science_UM_Ch 1.indd 2 18-04-2023 11:51:12

6	چھٹی نسل	مستقبل میں		• متوازی اور تقسیم شدہ کمپیوٹنگ • کمپیوٹر بن چکے ہیں۔ ہوشیار، تیز اور چھوٹے۔ • روبوٹکس کی ترقی • قدرتی زبان کی پروسیسنگ • آواز کی شناخت کے لئے سافٹ ویئر کی ترقی

ٹیبل 1.1 کمپیوٹرز کی نسلیں۔

کیا آپ جانتے ہیں؟

پہلا ڈیجیٹل کمپیوٹر

ENIAC (الیکٹرانک عددی انٹیگریٹر اور کیلکولیٹر) J. Presper Eckert اور John Mauchly نے پنسلوانیا یونیورسٹی میں ایجاد کیا تھا۔اور 1943 میں تعمیر شروع ہوئی اور 1946 تک مکمل نہیں پائی۔اس نے تقریباً 1,800 مربع قدم پر قبضہ کیا اور تقریباً 18,000 ویکیوم ٹیوبس استعمال کیں، جن کا وزن تقریباً 50 ٹن تھا۔ ENIAC پہلا ڈیجیٹل کمپیوٹر تھا کیونکہ یہ مکمل طور پر فعال تھا۔



1.3 چھٹی نسل کی کمپیوٹنگ
(Sixth Generation Computing)

چھٹی نسل میں، کمپیوٹرز کو مصنوعی اعصابی نیٹ ورکس پر مبنی ذہین کمپیوٹرز کے دور کے طور پر بیان کیا جا سکتا ہے۔ چھٹی نسل میں سب سے زیادہ ڈرامائی تبدیلیوں میں سے ایک وائڈ ایریا نیٹ ورکنگ (WAN) کی دھماکہ خیز ترقی ہوگی۔ نیچرل لینگویج پروسیسنگ (NLP) مصنوعی ذہانت (AI) کا ایک حصہ ہے۔ یہ انسانی زبان کو سمجھنے کے لیے کمپیوٹر پروگرام تیار کرنے کی صلاحیت فراہم کرتا ہے۔

کیا آپ جانتے ہیں؟

آپٹیکل کریکٹر کی پہچان (Optical Grapheme Recognition) انجن انڈس اسکرپٹس کے لیے ڈیپ لرننگ نیورل نیٹ ورکس (Deep Learning Neural Networks) (مصنوعی ذہانت کا ذیلی فیلڈ) کا استعمال کرتے ہوئے تیار کیا گیا۔وادی سندھ کی تہذیب کے نمونے کی تصاویر، اسکین، یا کسی بھی تصویری فیڈ کو دیکھتے ہوئے، سسٹم تصویر سے نوشتہ جات (علامت / گرافیم کی ترتیب) کو پہچاننے کے قابل ہوں۔ کل ۴۱۷ علامتیں / گرافیم / کریکٹرز انڈس اسکرپٹ میں پائے جاتے ہیں اور مشین کو سیکھنے اور مہارت حاصل کرنے کے لیے ڈیٹا کے صرف +3700 ٹیکسٹ انکرپشنز (Text Inscriptions) استعمال ہوتے ہیں۔

4C7HS



1.4 ڈیٹا اور معلومات
(Data and Information)

ہم سب جانتے ہیں کہ کمپیوٹر کیا ہے؟ یہ ایک الیکٹرانک ڈیوائس ہے جو اس کو فراہم کردہ ہدایات کے مطابق ان پٹ پر کارروائی کرتا ہے اور مطلوبہ آؤٹ پٹ بہت تیز رفتار سے دیتا ہے۔

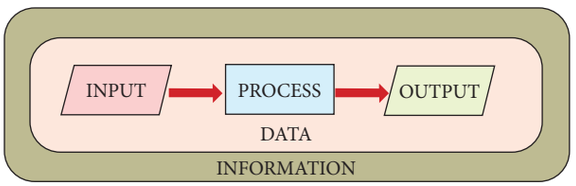
11th_Comp Science_UM_Ch 1.indd 3 18-04-2023 11:51:13

کمپیوٹرز بہت ہمہ گیر (Versatile) ہوتے ہیں کیونکہ وہ بہت سے مختلف کام کرتے ہیں جیسے ڈیٹا اسٹور کرنا، موسم کی پیشن گوئی کرنا، ایئر لائنز کی بکنگ، ریلوے یا مووی ٹکٹ اور یہاں تک کہ گیمز کھیلنا۔

ڈیٹا (Data) : ڈیٹا کو خام حقائق کے غیر پروسیس شدہ مجموعہ کے طور پر بیان کیا جاتا ہے، جو مواصلات، تشریح یا پروسیسنگ کے لیے موزوں ہے۔

مثال کے طور پر، 134، 16 'Kavitha'، 'C' ڈیٹا ہیں۔ اس سے کوئی معنی خیز پیغام نہیں ملے گا۔

معلومات (Information): معلومات حقائق کا ایک مجموعہ ہے جس سے نتیجہ اخذ کیا جا سکتا ہے۔ آسان الفاظ میں ہم کہہ سکتے ہیں کہ ڈیٹا خام حقائق ہیں جن پر بامعنی، ترتیب شدہ یا منظم معلومات دینے کے لیے کارروائی کی جاتی ہے۔ مثال کے طور پر کویتا کی عمر ۱۶ سال ہے۔ یہ معلومات کویتا کے بارے میں ہے اور کچھ معنی بیان کرتی ہے۔ ڈیٹا کی معلومات میں اس تبدیلی کو ڈیٹا پروسیسنگ (Data Processing) کہتے ہیں۔



تصویر 1.1 ڈیٹا اور معلومات

کمپیوٹر ایک الیکٹرانک ڈیوائس ہے جو صارف سے خام ڈیٹا (غیر پروسیس شدہ) کو بطور ان پٹ لیتا ہے اور اسے ہدایات کے ایک سیٹ (جسے پروگرام کہا جاتا ہے) کے کنٹرول میں پروسیس کرتا ہے، نتیجہ (آؤٹ پٹ) پیدا کرتا ہے، اور اسے مستقبل کے استعمال کے لیے محفوظ کرتا ہے۔

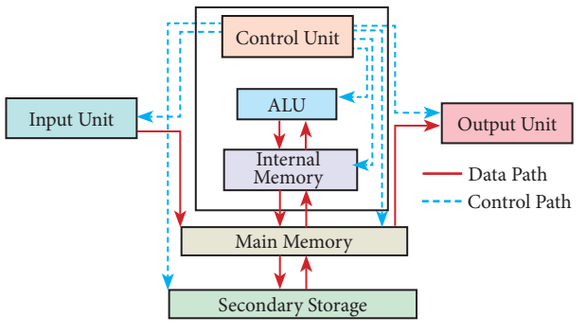
1.5 کمپیوٹر کے اجزاء

کمپیوٹر ہارڈ ویئر (Hardware) اور سافٹ ویئر (Software) کا مجموعہ ہے۔ ہارڈ ویئر کمپیوٹر کا جسمانی جزو ہے جیسے مدر بورڈ (Motherboard)، میموری ڈیوائسز (Memory Devices)، مانیٹر (Monitor)، کی بورڈ (Keyboard) وغیرہ، جبکہ سافٹ ویئر پروگرام یا ہدایات سیٹ ہے۔ ہارڈ ویئر اور سافٹ ویئر دونوں مل کر کمپیوٹر سسٹم کو کام کرنے کے قابل بناتے ہیں۔



تصویر 1.2: کمپیوٹر

آئیے پہلے کمپیوٹر کے فنکشنل اجزاء (Functional Componentes) پر ایک نظر ڈالتے ہیں۔ کمپیوٹر کو دیا جانے والا ہر کام ایک ان پٹ-پروسیس-آؤٹ پٹ سائیکل (IPO سائیکل) کی پیروی کرتا ہے۔ اسے مخصوص ان پٹ کی ضرورت ہوتی ہے، ایسے عمل جو ان پٹ اور مطلوبہ آؤٹ پٹ تیار کرتے ہیں۔ ان پٹ یونٹ ان پٹ لیتا ہے، سینٹرل پروسیسنگ یونٹ ڈیٹا کی پروسیسنگ کرتا ہے اور آؤٹ پٹ یونٹ آؤٹ پٹ تیار کرتا ہے۔ میموری یونٹ پروسیسنگ کے دوران ڈیٹا اور ہدایات محفوظ رکھتا ہے۔



تصویر 1.3 کمپیوٹر کے اجزاء

1.5.1 ان پٹ یونٹ (Input Unit)

ان پٹ یونٹ کا استعمال کسی بھی قسم کے ڈیٹا کو کمپیوٹر میں جمع کرنے کے لیے کیا جاتا ہے، جسے مزید پروسیسنگ کے لیے میموری یونٹ میں محفوظ کیا جا سکتا ہے۔ مثال: کی بورڈ، ماؤس وغیرہ۔

11th_Comp Science_UM_Ch 1.indd 4 18-04-2023 11:51:13

1.5.2 سنٹرل پروسیسنگ یونٹ (CPU)

سی پی یو ایک بڑا جزو ہے جو سافٹ ویئر کی ہدایات کی تشریح اور عمل کرتا ہے۔ یہ دیگر تمام اجزاء جیسے میموری، ان پٹ اور آؤٹ پٹ یونٹس کے آپریشن کو بھی کنٹرول کرتا ہے۔ یہ بائنری ڈیٹا کو بطور ان پٹ قبول کرتا ہے، ہدایات کے مطابق ڈیٹا پر کارروائی کرتا ہے اور نتیجہ آؤٹ پٹ کے طور پر فراہم کرتا ہے۔

CPU کے تین اجزاء ہیں جو کہ کنٹرول یونٹ ،ارتھ میٹک اور لوجک یونٹ (ALU) اور میموری یونٹ ہیں۔

1.5.2.1 ریاضی اور منطقی یونٹ (Arithmetic and Logic Unit - ALU)

ALU سی پی یو کا ایک حصہ ہے جہاں ڈیٹا پر کمپیوٹنگ کے مختلف افعال انجام دیے جاتے ہیں۔ ALU ریاضی کی کارروائیاں کرتا ہے جیسے کہ جوڑ، گھٹاؤ، ضرب، تقسیم اور منطقی کارروائیاں۔ آپریشن کا نتیجہ CPU کی اندرونی میموری میں محفوظ ہوتا ہے۔ ALU کی منطقی کارروائیاں کمپیوٹر کی فیصلہ سازی کی صلاحیت کو فروغ دیتی ہیں۔

1.5.2.2 کنٹرول یونٹ (Control Unit)

کنٹرول یونٹ سی پی یو،میموری اور ان پٹ / آوٹ پٹ آلات کے درمیان ڈیٹا کے بہاؤ کو کنٹرول کرتا ہے۔ یہ کمپیوٹر کے پورے آپریشن کو بھی کنٹرول کرتا ہے۔

1.5.3 آؤٹ پٹ یونٹ (Output)

آؤٹ پٹ یونٹ کسی بھی ہارڈ ویئر کا جزو ہے، جو صارفین تک معلومات کو قابل فہم شکل میں پہنچاتا ہے۔ مثال: مانیٹر، پرنٹر وغیرہ۔

1.5.4 میموری یونٹ (Memory Unit)

میموری یونٹ دو قسم کا ہوتا ہے: جو کہ بنیادی میموری (Primary Memory) اور سیکنڈری میموری (Secondary Memory) ہیں۔ بنیادی میموری جب ہدایات پر عمل درآمد کے لیے تیار ہوں تو عارضی طور پر ڈیٹا اور پروگرام کو جمع کرنے کے لیے استعمال کی جاتی ہے۔ثانوی میموری کا استعمال ڈیٹا کو مستقل طور پر محفوظ کرنے کے لیے کیا جاتا ہے۔

پرائمری میموری غیر مستحکم ہے، یعنی جب بجلی کی فراہمی بند ہو جاتی ہے تو مواد ضائع ہو جاتا ہے۔ رینڈم ایکسیس میموری (RAM) مرکزی میموری کی ایک مثال ہے۔ سیکنڈری میموری مستحکم ہے، یعنی بجلی کی فراہمی بند ہونے کے بعد بھی مواد دستیاب ہوتا ہے۔ ہارڈ ڈسک (Hard Disk، CD-ROM اور DVD ROM) سیکنڈری میموری کی مثالیں ہیں۔

1.5.5 ان پٹ اور آؤٹ پٹ ڈیوائسز

ان پٹ ڈیوائسز:

(1) کی بورڈ (Keyboard): کی بورڈ (وائرڈ / وائرلیس، ورچوئل) آج کل استعمال ہونے والا سب سے عام ان پٹ ڈیوائس ہے۔ حروف، اعداد اور خصوصی حروف کے لیے انفرادی کیز (Individual Keys) کو اجتماعی طور پر کریکٹر کیز (Character Keys) کے نام سے جانا جاتا ہے۔ یہ کی بورڈ لے آؤٹ اصل ٹائپ رائٹر کے کی بورڈ سے اخذ کیا گیا ہے۔ ڈیٹا اور ہدایات کی بورڈ پر ٹائپ کرکے کمپیوٹر کو ان پٹ کے طور پر دی جاتی ہیں۔ حروف تہجی اور عددی کیز کے علاوہ، اس میں مختلف افعال انجام دینے کے لیے فنکشن کیز بھی ہیں۔ کی بورڈ میں کیز کے مختلف سیٹ دستیاب ہیں جیسے کریکٹر کیز، موڈیفائر کیز (Modifier Keys)، سسٹم اور جی یو آئی GUI کیز، انٹر اور ایڈیٹنگ کیز، فنکشن کیز، نیویگیشن کیز (Navigation Keys)، عددی کی پیڈ اور لاک کیز۔



تصویر 1.4 کی بورڈ

(2) ماؤس (Mouse): ماؤس (وائرڈ/وائرلیس) ایک اشارہ کرنے والا آلہ ہے جسے ڈسپلے اسکرین پر کرسر کی حرکت

11th_Comp Science_UM_Ch 1.indd 5 18-04-2023 11:51:13

کو کنٹرول کرنے کے لیے استعمال کیا جاتا ہے۔ اسے آئیکنس، مینوس، کمانڈ بٹنس منتخب کرنے یا کمپیوٹر پر کسی چیز کو چالو کرنے کے لیے استعمال کیا جا سکتا ہے۔ کچھ ماؤس کے کام یہ ہیں ۔موو (Move)، کلک (Click)، ڈبل کلک (Double Click)، رائٹ کلک، ڈریگ اور ڈراپ (Drag and Drop) ۔

دستیاب ماؤس کی مختلف اقسام ہیں: مکینیکل ماؤس (Mechanical Mouse)، آپٹیکل (Optical)، لیزر ماؤس (Laser Mouse)، ایئر ماؤس (Air Mouse)، 3D ماؤس، ٹیکٹائل ماؤس (Tactile Mouse)، ایرگونومک ماؤس (Ergonomic Mouse) اور گیمنگ ماؤس (Gaming Mouse) ۔

ماؤس کس نے ایجاد کیا؟

کیا آپ جانتے ہیں؟



کمپیوٹر ماؤس جیسا کہ آج ہم جانتے ہیں کہ ڈگلس اینجل بارٹ (Douglas Engelbart) نے بل انگلش (Bill English) کی مدد سے 1960 کی دہائی کے دوران ایجاد اور تیار کیا تھا۔ اور ۱۷ نومبر 1970 کو سند حق ایجاد (Patented) دیا گیا۔

C8OW1

(3) اسکینر (Scanner): اسکینرس معلومات کو براہ راست کمپیوٹر میموری میں داخل کرنے کے لیے استعمال کیا جاتا ہے۔ یہ ڈیوائس زیروکس مشین کی طرح کام کرتی ہے۔ سکینر کسی بھی قسم کی پرنٹ یا تحریری معلومات بشمول تصاویر کو ڈیجیٹل فارمیٹ میں تبدیل کرتا ہے، جسے کمپیوٹر کے ذریعے ہیرا پھیری کیا جا سکتا ہے۔



تصویر 1.5 اسکینر

(4) فنگر پرنٹ اسکینر (Fingerprint Scanner): فنگر پرنٹ اسکینر ایک فنگر پرنٹ شناخت کرنے والا (Fingerprint Recognition) ڈیوائس ہے جسے کمپیوٹر سیکیورٹی کے لیے استعمال کیا جاتا ہے،فنگر پرنٹ ریکگنیشن فیچر سے لیس ہے جو بائیو میٹرک ٹیکنالوجی کا استعمال کرتا ہے۔ پاس ورڈ استعمال کرنے کے بجائے جو دھوکہ دہی کا خطرہ اور یاددہانی کے لیے مشکل ہے ، جبکہ فنگر پرنٹ ریڈر / سکینر سیکیورٹی کے لیے بہت محفوظ اور آسان ڈیوائس ہے۔



تصویر 1.6 فنگر پرنٹ سکینر

(5) ٹریک بال (Track Ball): ٹریک بال ماؤس کی طرح ایک الٹا ڈیزائن ہے ۔صارف بال کو براہ راست منتقل کرتا ہے،جبکہ آلہ خود ساکن رہتا ہے۔ صارف اسکرین کی حرکت کو کسی بھی سمت (Navigate) کرنے کے لیے بال کو مختلف سمتوں میں گھماتا ہے۔



تصویر 1.7 ٹریک بال

(6) ریٹینل اسکینر (Retinal Scanner): یہ ریٹینل(آنکھ کی پتلی) اسکین کرتا ہے جو کہ ایک بائیو میٹرک تکنیک ہے جو کسی شخص کے ریٹینل خون کی نالیوں پر منفرد نمونوں کا استعمال کرتی ہے۔



تصویر 1.8 ریٹینل سکینر

(7) لائٹ پین (Light Pen): لائٹ پین اشارہ کرنے والا آلہ ہے جس کی شکل پین کی طرح ہے اور مانیٹر سے منسلک ہے۔ لائٹ پین کی نوک پر مشتمل حساس روشنی کا عنصر ہے جو اسکرین سے روشنی کا پتہ لگاتا ہے جس سے



تصویر 1.9 لائٹ پین

11th_Comp Science_UM_Ch 1.indd 6 18-04-2023 11:51:14

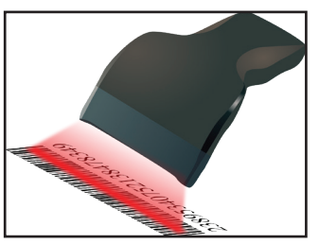
کمپیوٹر کو اسکرین پر قلم کی جگہ کی شناخت کرنے میں مدد ملتی ہے۔ لائٹ پین براہ راست اسکرین پر ڈرائنگ کا فائدہ دیتا ہے، لیکن یہ استعمال کرنا مشکل ہو جاتا ہےاور درست بھی نہیں ہے۔

(8) آپٹیکل کریکٹر ریڈر (Character Optical Reader): یہ ایک ایسا آلہ ہے جو او سی آر (OCR) کے ساتھ کاغذ پر چھپی ہوئی یا لکھی ہوئی حروف کا پتہ لگاتا ہے، صارف کتاب سے کسی صفحے کو اسکین کر سکتا ہے۔ کمپیوٹر صفحہ میں حروف کو حروف اور اوقاف کے نشانات کے طور پر پہچانے گا اور محفوظ کرے گا۔ اسکین شدہ دستاویز کو ورڈ پروسیسر کے ذریعے ایڈٹ کیا جا سکتا ہے۔



تصویر 1.10 آپٹیکل کریکٹر ریڈر

(9) بار کوڈ / کیو آر کوڈ ریڈر (Bar Code / QR Code Reader): بار کوڈ ایک پیٹرن ہے جو مختلف موٹائی کی لائنوں میں پرنٹ کیا جاتا ہے۔ بار کوڈ ریڈر، بار کوڈز کی معلومات کو اسکین کرتا ہے اور مزید پروسیسنگ کے لیے کمپیوٹر کو بھیجتا ہے۔ سسٹم تیز رفتار اور غلطی سے پاک اندراج فراہم کرتا ہے۔



تصویر 1.11 بار کوڈ ریڈر۔

کیو آر (فوری رد عمل) کوڈ (QR – Quick Response- Code): کیو آر کوڈ دو جہتی بار کوڈ ہے جسے کیمرے کے ذریعے پڑھا جا سکتا ہے اور تصویر کی تشریح کے لیے کارروائی کی جا سکتی ہے۔

(10) وائس ان پٹ سسٹمز (Voice Input Systems): مائیکروفون وائس ان پٹ ڈیوائس کے طور پر کام کرتا ہے۔ یہ صوتی ڈیٹا حاصل کرتا ہے اور اسے کمپیوٹر کو بھیجتا ہے۔ اسپیچ ریکگنیشن سافٹ ویئر کے ساتھ مائکروفون کا استعمال کمپیوٹر میں معلومات داخل کرنے کے لیے بالکل نیا طریقہ پیش کر سکتا ہے۔



تصویر 1.12 وائس ان پٹ سسٹم

(11) ڈیٹیجل کیمرا (Digital Camera): تصاویر / ویڈیوز کو براہ راست ڈیجیٹل شکل میں کھینچتا ہے۔ یہ ایک سی سی ڈی CCD (چارج کپلڈ ڈیوائس) (Charge Coupled Device) الکٹرانک چپ کا استعمال کرتا ہے۔ جب روشنی لینس کے ذریعہ چپ پر پڑتی ہے تو یہ روشنی کی شعاعوں کو ڈیجیٹل شکل میں بدل دیتی ہے۔



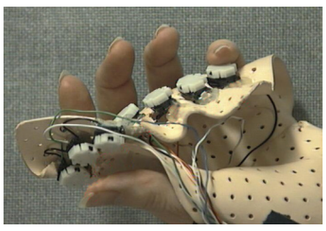
تصویر 1.13 ڈیٹیجل کیمرا

(12) ٹچ اسکرین (Touch Screen): ٹچ اسکرین ایک ڈسپلے ڈیوائس ہے جو صارف کو انگلی کا استعمال کرکے کمپیوٹر کے ساتھ بات چیت کرنے کی اجازت دیتی ہے۔ یہ گرافیکل یوزر انٹر فیس (GUI) کی سمت شناسی کے لیے ماؤس یا کی بورڈ کے متبادل کے طور پر کافی کارآمد ثابت ہوسکتا ہے ۔ ٹچ اسکرین کا استعمال مختلف آلات جیسے کمپیوٹر، لیپ ٹاپ، مانیٹر، سمارٹ فون، ٹیبلٹ، کیش رجسٹر اور معلوماتی کیوسک (Kiosks) پر کیا جاتا ہے۔ کچھ ٹچ اسکرینز ٹچ حساس ان پٹ کو استعمال کرنے کے بجائے انگلی کی موجودگی کو محسوس کرنے کے لیے انفراریڈ بیم کا گرڈ (Grid of Infrared Beams) استعمال کرتی ہیں۔



تصویر 1.14 ٹچ اسکرین۔

(13) کیئر (Keyer): کیئر ایک آلہ ہے جو ہاتھ سے سگنل کرنے کے لیے ہے، ایک یا زیادہ سوئچ کو دبانا ہے۔ جدید کیئرس کے بہت سے سوئیچز ہیں، لیکن ایک مکمل کی بورڈ کے سائز کے برابر نہیں ہیں۔ عام طور پر یہ تعداد 4 اور 50 کے درمیان ہے۔ ایک کیئر کی بورڈ سے مختلف ہوتا ہے،



تصویر 1.15 کیئر

11th_Comp Science_UM_Ch 1.indd 7 18-04-2023 11:51:15

جس میں "کوئی بورڈ نہیں" (No Board) ہوتا، لیکن کیز کو ایک مجموعہ میں ترتیب دیا جاتا ہے۔

آؤٹ پٹ ڈیوائسز (Output Devices):

(1) مانیٹر (Monitor):



تصویر 1.16 مانیٹر

مانیٹر معلومات کو ظاہر کرنے کے لیے سب سے زیادہ استعمال ہونے والا آؤٹ پٹ ڈیوائسز (آلہ) ہے۔ یہ دیکھنے میں ٹی وی کی طرح ہے۔ مانیٹر پر تصویریں تصویری عناصر کے ساتھ بنتی ہیں جنہیں PIXELS کہتے ہیں۔ مانیٹر یا تو مونوکروم ہو سکتے ہیں،جو کالے یا سفید رنگ میں ٹیکسٹ یا تصاویر دکھاتا ہے یا رنگین ہو سکتا ہے، جس کے نتائج متعدد رنگوں میں ظاہر ہوتے ہیں۔بہت سے قسم کے مانیٹر دستیاب ہیں جیسے CRT (کیتھوڈ رے ٹیوب - Cathode Ray Tube)، LCD (لیکوڈ کرسٹل ڈسپلے - Liquid Crystal Dispaly) اور ایل ای ڈی LED (Light Emitting Diodes– لائٹ امیٹنگ ڈائیوڈس)۔ مانیٹر VGA (ویڈیو گرافکس ارے – Video Graphics Array) کارڈ کے ساتھ کام کرتا ہے۔ ویڈیو گرافکس کارڈ اسکرین کے ساتھ بات چیت کرنے کے لیے کی بورڈ (Keyboard) کی مدد کرتا ہے ۔ یہ کمپیوٹر اور ڈسپلے مانیٹر کے درمیان ایک انٹرفیس کے طور پر کام کرتا ہے۔ عام طور پر حالیہ مدر بورڈز بلٹ ان (Built-in) ویڈیو کارڈ کو شامل کرتے ہیں۔

پہلا کمپیوٹر مانیٹر زیروکس آلٹو کمپیوٹر سسٹم (Xerox Alto Computer Systems) کا حصہ تھا، جسے 1 مارچ 1973 کو جاری کیا گیا تھا۔

(2) پلاٹر (Plotter): پلاٹر ایک آؤٹ پٹ آلہ ہے جو کاغذات پر گرافیکل آؤٹ پٹ پیدا کرنے کے لیے استعمال ہوتا ہے۔ یہ تصویریں کھینچنے کے لیے سنگل کلر (ایک رنگ) یا ملٹی کلر (کئی رنگ) پین کا استعمال کرتا ہے۔



تصویر 1.17 پلاٹر

(3) پرنٹرز (Printers): پرنٹرز کا استعمال کاغذات پر معلومات کو پرنٹ کرنے کے لیے کیا جاتا ہے۔ پرنٹرز کو دو اہم اقسام میں تقسیم کیا گیا ہے:

- اپیکٹ پرنٹرز (Impact Printers)
- نان اپیکٹ پرنٹرز (Non Impact Printers)

اپیکٹ پرنٹرز

یہ پرنٹرز ربن پر ہتھوڑے یا پنوں کی زور کے ساتھ پرنٹ کرتے ہیں۔ یہ پرنٹرز مکینیکل پریشر کا استعمال کرتے ہوئے کثیر حصے پر (کاربن پیپرز کا استعمال کرتے ہوئے) پرنٹ کر سکتے ہیں۔مثال کے طور پر، ڈاٹ میٹرکس پرنٹرز (Dot Matrix Printers) اور لائن میٹرکس پرنٹرز (Line Matrix Printers) اپیکٹ پرنٹرز ہیں۔



تصویر 1.18 اپیکٹ پرنٹرز

ایک ڈاٹ میٹرکس پرنٹر جو پنوں یا تاروں کی ایک مقررہ تعداد کا استعمال کرتے ہوئے پرنٹ کرتا ہے۔ ہر نقطے کو ایک چھوٹی دھاتی چھڑی سے تیار کیا جاتا ہے، جسے "تار" (Wire) یا "پن" (Pin) بھی کہا جاتا ہے، جو ایک چھوٹے برقی مقناطیس (Electro Magnet) یا سولینائڈ (Solenoid) کی طاقت سے براہ راست یا چھوٹے لیورز کے سیٹ کے ذریعے کام کرتا ہے۔ عام طور پر ایک وقت میں متن کی ایک لائن پرنٹ کرتا ہے۔ ان پرنٹرز کی پرنٹنگ کی رفتار 30 سے 1550 CPS (کریکٹر فی سیکنڈ) تک ہوتی ہے۔

لائن میٹرکس پرنٹرز پرنٹنگ کے لیے ایک متعین پرنٹ ہیڈ استعمال کرتے ہیں۔ بنیادی طور پر یہ نقطوں کی ایک صفحہ وسیع لائن پرنٹ کرتا ہے۔ لیکن یہ نقطوں کی لائنوں کو پرنٹ کرکے متن کی ایک

11th_Comp Science_UM_Ch 1.indd 8 18-04-2023 11:51:15

لائن بناتا ہے۔ لائن پرنٹرز 1000 لائنز فی منٹ سے زیادہ پرنٹ کرنے کی صلاحیت رکھتے ہیں، جس کے نتیجے میں فی گھنٹہ ہزاروں صفحات بنتے ہیں۔ یہ پرنٹرز ملٹی پارٹ (کاربن پیپرز کا استعمال کرتے ہوئے) پر پرنٹ کرنے کے لیے مکینیکل پریشر بھی استعمال کرتے ہیں۔

نان امپیکٹ پرنٹرز

یہ پرنٹرز پرنٹنگ کے لیے اسٹرائیکنگ میکانزم (Striking Mechanism) کا استعمال نہیں کرتے ہیں۔ وہ الیکٹرو سٹیٹک (Electrostatic) یا لیزر ٹیکنالوجی (Laser Technology) استعمال کرتے ہیں۔ ان پرنٹرز کا معیار اور رفتار امپیکٹ پرنٹرز سے بہتر ہے۔ مثال کے طور پر، لیزر پرنٹرز اور انکجیٹ پرنٹرز نان امپیکٹ پرنٹرز ہیں۔

لیزر پرنٹرز

لیزر پرنٹرز زیادہ تر اسی طرح کی ٹیکنالوجی کے ساتھ کام کرتے ہیں جو فوٹو کاپیئرز (Photocopiers) استعمال کرتے ہیں۔ یہ پرنٹر کے اندر ایک ڈرم پر آگے پیچھے لیزر بیم اسکین کرکے ایک پیٹرن بناتا ہے۔ یہ گرافک امیجز کی بہت اچھی کوالٹی تیار کر سکتا ہے۔ لیزر پرنٹر کی اہم خصوصیات میں سے ایک ان کا ریزولوشن (Resolution) – کتنے ڈاٹ فی انچ (DPI) ہے۔ دستیاب ریزولوشن کی حد تقریباً 1200 ڈی پی آئی ہے۔ یہ تقریباً 100 صفحات فی منٹ پرنٹ کر سکتا ہے (PPM)



تصویر 1.19 لیزر پرنٹر

انک جیٹ پرنٹرز:

انک جیٹ پرنٹرز رنگین کارٹریجیس (Cartridges) استعمال کرتے ہیں جس میں رنگین ٹونر بنانے کے لیے میجنٹا (Magenta)، پیلا اور سیان (Cyan) سیاہی مل جاتی ہے۔ ایک سیاہ کارٹریجیس بھی مونوکروم (Monochrome) آؤٹ پٹ کے لیے استعمال ہوتا ہے۔ انک جیٹ پرنٹرز کاغذ کی شیٹ پر آئنائزڈ (Ionised) سیاہی چھڑک کر کام کرتے ہیں۔ انک جیٹ پرنٹرز کی رفتار عام طور پر 1 -20 PPM (صفحہ فی منٹ) تک ہوتی ہے۔



تصویر 1.20 انک جیٹ پرنٹر

وہ سیاہی کو گرم کرکے فائر نگ کی ٹکنالوجی (Technology of firing) کا استعمال کرتے ہیں تاکہ یہ بلبلوں میں کاغذ کی طرف پھٹ جائے یا پیزو الیکٹرسٹی (Piezoelectricity) کا استعمال کرتے ہوئے جس میں الیکٹرانک سرکٹس کے ذریعے کنٹرول کی جانے والی چھوٹی برقی کرنٹ کو پرنٹر کے اندر جیٹ کی رفتار سے سیاہی پھیلانے کے لیے استعمال کیا جاتا ہے۔ ایک انک جیٹ پرنٹر ہر ایک سیکنڈ میں کاغذ پر لاکھوں نقطوں کی سیاہی پھیلا سکتا ہے۔

اسپیکر (Speaker): اسپیکر آواز کے نتائج Voice output – (Audio) تیار کرتے ہیں۔ اسپیچ سنتھیسس سافٹ ویئر (Speech Synthesis Software) کا استعمال اسپیکر کے ساتھ کرکے کمپیوٹر آواز کے نتائج فراہم کر سکتا ہے۔ یہ ایئر لائنز (Airlines)، اسکولوں، بینکوں، ریلوے اسٹیشنوں وغیرہ جیسی جگہوں پر بہت عام ہو گیا ہے۔



تصویر 1.21 اسپیکر

ملٹی میڈیا پروجیکٹرس (Multimedia Projecters):

ملٹی میڈیا پروجیکٹر س کو بڑی اسکرین پر کمپیوٹر آؤٹ پٹ تیار کرنے کے لیے استعمال کیا جاتا ہے۔ یہ میٹنگ ہالز یا کلاس رومز میں کسی چیز کو پیش کرنے کے لیے استعمال ہوتے ہیں۔



تصویر 1.22 ملٹی میڈیا پروجیکٹرس

11th_Comp Science_UM_Ch 1.indd 9 18-04-2023 11:51:16

1.6 کمپیوٹر کی بوٹنگ
(Booting of Computer)

ایک آپریٹنگ سسٹم (OS) ایک بنیادی سافٹ ویئر ہے جو کمپیوٹر کو کام کرنے کے قابل بناتا ہے۔ جب کمپیوٹر کو چالوکیا جاتا ہے تو اس کی RAM میں کوئی معلومات نہیں ہوتی۔ ساتھ ہی ROM میں پہلے سے لکھے ہوئے پروگرام کو POST (Power on Self Test) کہا جاتا ہے۔ یہ پروگرام چیک کرتا ہے کہ آیا RAM کی بورڈ وغیرہ جیسے آلات صحیح طریقے سے جڑے ہوئے ہیں اور کام کرنے کے لیے تیار ہیں۔ اگر یہ آلات تیار ہیں، تو BIOS (بنیادی ان پٹ آؤٹ پٹ سسٹم) پر عمل درآمد ہو جاتا ہے۔ اس عمل کو بوٹنگ کہتے ہیں۔ اس کے بعد، "بوٹس ٹریپ لوڈر" (Bootstrap Loader) نامی ایک پروگرام OS کو ہارڈ ڈسک سے مین میموری میں منتقل کرتا ہے۔ اب OS لوڈ ہو جاتا ہے (ونڈوز/لینکس، وغیرہ) اور اس پر عمل درآمد ہو جائے گا۔بوٹنگ کا عمل دو طرح کا ہوتا ہے۔

1) کولڈ بوٹنگ (Cold Booting)

2) وارم بوٹنگ (Warm Booting)

کولڈ بوٹنگ: جب سسٹم ابتدائی حالت سے شروع ہوتا ہے یعنی اسے آن کیا جاتا ہے تو ہم اسے کولڈ بوٹنگ یا ہارڈ بوٹنگ کہتے ہیں۔ جب صارف پاور بٹن دباتا ہے، تو بوٹنگ کے عمل کو شروع کرنے کے لیے ROM سے ہدایات پڑھی جاتی ہیں۔

وارم بوٹنگ: جب سسٹم دوبارہ شروع ہوتا ہے یا جب ری سیٹ بٹن دبایا جاتا ہے تو ہم اسے وارم بوٹنگ یا سافٹ بوٹنگ (Soft Booting) کہتے ہیں۔ سسٹم ابتدائی حالت سے شروع نہیں ہوتا ، اس لیے اس معاملے میں تمام تشخیصی ٹیسٹ کرنے کی ضرورت نہیں ہے۔ ڈیٹا ضائع ہونے اور سسٹم کو نقصان پہنچنے کے امکانات ہیں ،کیونکہ ڈیٹا کو صحیح طریقے سے محفوظ نہیں کیا گیا ہو گا۔

یاد رکھنے کے نکات:

- کمپیوٹر ہمارے ارد گرد ہر جگہ، زندگی کے تمام شعبوں میں نظر آتے ہیں۔
- یہ ایک الیکٹرانک ڈیوائس ہے جو اس کو فراہم کردہ ہدایات کے سیٹ کے مطابق ان پٹ پر کارروائی کرتا ہے اور انتہائی تیز رفتاری سے مطلوبہ آؤٹ پٹ فراہم کرتا ہے۔
- ترقی کے مختلف مراحل کی بنیاد پر، کمپیوٹرز کو چھ مختلف نسلوں میں تقسیم کیا جا سکتا ہے۔
- کمپیوٹر ہارڈ ویئر اور سافٹ ویئر کا مجموعہ ہے۔
- ہارڈ ویئر کمپیوٹر کا جسمانی جزو ہے۔
- ان پٹ یونٹ کا استعمال کمپیوٹر کو ڈیٹا کے کسی بھی شکل کے لیے کیا جاتا ہے۔
- CPU سافٹ ویئر کی ہدایات کی تشریح اور ان پر عمل درآمد کرتا ہے۔
- CPU ALU کا ایک حصہ ہے جہاں ڈیٹا پر کمپیوٹنگ کے مختلف افعال انجام دیے جاتے ہیں۔
- کنٹرول یونٹ CPU، میموری اور I/O آلات کے درمیان ڈیٹا کے بہاؤ کو کنٹرول کرتا ہے۔
- آؤٹ پٹ یونٹ کسی بھی ہارڈ ویئر کا جزو ہے، جو صارف کو قابل فہم شکل میں ایک یا زیادہ لوگوں تک معلومات پہنچاتا ہے۔
- میموری یونٹ دو طرح کا ہوتا ہے ،جو کہ بنیادی میموری اور سیکنڈری میموری ہیں۔

مشق

طالب علم کی سرگرمی

1. کمپیوٹر کی درجہ بندی کی وضاحت کریں۔
2. مدر بورڈ کے ناموں، 1993، 1995، 2005، 2008، 2016 میں استعمال ہونے والی RAM کی صلاحیت کی تفصیلات دیں۔
3. دو نئے ان پٹ اور آؤٹ پٹ آلات کا ذکر کریں جو اس باب میں نہیں دیے گئے ہیں۔

11th_Comp Science_UM_Ch 1.indd 10 18-04-2023 11:51:16

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. پہلی نسل کے کمپیوٹر کا استعمال
(a) ویکیوم ٹیوبس
(b) ٹرانزسٹر
(c) انٹیگریٹڈ سرکٹس
(d) مائکرو پروسیسرز

2. غیر مستحکم میموری کا نام بتائیں۔
ROM (a) PROM (b)
RAM (c) EPROM (d)

3. آؤٹ پٹ ڈیوائس کی شناخت کریں۔
(a) کی بورڈ (b) میموری
(c) مانیٹر (d) ماؤس

4. ان پٹ ڈیوائس کی شناخت کریں۔
(a) پرنٹر (b) ماؤس
(c) پلاٹر (d) پروجیکٹر

5. ..... آؤٹ پٹ ڈیوائس بلڈنگ پلان پرنٹ کرنے کے لیے استعمال ہوتی ہے۔
(a) تھرمل پرنٹر
(b) پلاٹر
(c) ڈاٹ میٹرکس
(d) انک جیٹ پرنٹر

6. مندرجہ ذیل میں سے کون سا ATM مشینوں میں استعمال ہوتا ہے۔
(a) ٹچ اسکرین (b) اسپیکر
(c) مانیٹر (d) پرنٹر

7. جب سسٹم دوبارہ شروع ہوتا ہے تو کس قسم کی بوٹنگ استعمال ہوتی ہے۔
(a) وارم بوٹنگ
(b) کولڈ بوٹنگ
(c) ٹچ بوٹ
(d) ریئل بوٹ۔

8. POST کو واضح کیجیے۔
(a) Post on self Test
(b) Power on Software Test
(c) Power on Self Test
(d) Power on Self Text

9. مندرجہ ذیل میں سے کون سی اہم میموری ہے؟
ROM (a) RAM (b)
Flash drive (c) Hard disk (d)

10. کمپیوٹر کی کونسی نسل نے آئی سی کا استعمال کیا؟
(a) پہلی (b) دوسری
(c) تیسری (d) چوتھی

سیکشن - B

بہت ہی مختصر جواب دیں۔

1. کمپیوٹر کیا ہے؟
2. ڈیٹا اور معلومات کے درمیان فرق بیان کریں۔
3. CPU کے اجزاء کیا ہیں؟
4. ALU کا کام کیا ہے؟
5. کنٹرول یونٹ کے افعال لکھیں۔
6. میموری کا عمل کیا ہے؟
7. ان پٹ اور آؤٹ پٹ یونٹ میں فرق بیان کریں۔
8. پرائمری اور سیکنڈری میموری میں فرق بیان کریں۔

سیکشن - C

مختصر جوابات دیں۔

1. اکمپیوٹر کی خصوصیات کیا ہیں؟
2. کمپیوٹر کی ایپلی کیشنز لکھیں۔
3. ایک ان پٹ ڈیوائس کیا ہے؟ دو مثالیں دیں۔
4. کسی بھی تین آؤٹ پٹ آلات کا نام بتائیں۔
5. آپٹیکل اور لیزر میں فرق کریں۔
6. امپیکٹ پرنٹر پر مختصر نوٹ لکھیں۔
7. چھٹی نسل کی خصوصیات لکھیں۔
8. مانیٹر کی خصوصیات لکھیں۔

11th_Comp Science_UM_Ch 1.indd 11 18-04-2023 11:51:16

سیکشن - D

UVBOW

تفصیلی جواب دیں۔

1. وضاحتی خاکہ کے ساتھ کمپیوٹر کے بنیادی اجزاء کی وضاحت کریں۔
2. کمپیوٹر کی مختلف نسلوں پر بحث کریں۔
3. درج ذیل کی وضاحت کریں۔

a. انک جیٹ پرنٹر b. ملٹی میڈیا پروجیکٹر c. بار کوڈ / کیو آر کوڈ ریڈر

حوالہ جات

(1) Fundamentals of Computers – V. Rajaraman – PHI Publications

(2) Computer Science text book – NCERT, New Delhi

انٹرنیٹ وسائل

(1) www.wikipedia.org

(2) https://www.computerhope.com/jargon/c/computer.html

کیس اسٹڈی

رفتار، میموری، سائز، بجلی کی کھپت اور دیگر خصوصیات کے حوالے سے ماضی اور حال کے مختلف کمپیوٹرز کا تقابلی مطالعہ تیار کریں۔

لغات

کمپیوٹر	یہ ایک الیکٹرانک ڈیوائس ہے جو ان ہدایات کے مطابق ان پٹ پر کام کرتا ہے جو اسے فراہم کیاجاتاہے اور بہت تیز رفتار سے مطلوبہ آؤٹ پٹ دیتا ہے۔
ویکیوم ٹیوب	ویکیوم ٹیوبس میں الیکٹرون کے بہاؤ کو کنٹرول کرنے کے لیے الیکٹروڈس (Electrodes) ہوتے ہیں اور ابتدائی کمپیوٹرز میں سوئچ یا ایمپلیفائر (Amplifier) کے طور پر استعمال ہوتے تھے۔
ٹرانزسٹر	ٹرانزسٹر – Transistor "ٹرانسفر ریسسٹنس)سیمی کنڈکٹرز سے بنا ہے۔ یہ ایک ایسا جزو ہے جو الیکٹرانک سگنل کی ایمپلی فیکیشن /ماڈیولیشن کے لیے استعمال ہونے والے کرنٹ یا وولٹیج کی مقدار کو کنٹرول کرنے کے لیے استعمال ہوتا ہے۔
پنچڈ کارڈز	پنچ کارڈز جسے ہولیریتھ کارڈز (Hollerith Cards) بھی کہا جاتا ہے وہ کاغذی کارڈ ہوتے ہیں جن میں کئی پنچ یا سوراخ والے سوراخ ہوتے ہیں جنہیں ڈیٹا کی نمائندگی کرنے کے لیے ہاتھ یا مشین سے پنچ کیا جاتا ہے۔
مشینی زبان	مشینی زبان بائنری ہندسوں یا بٹس کا مجموعہ ہے جسے کمپیوٹر پڑھتا اور تشریح کرتا ہے۔
اسمبلی کی زبان	اسمبلی لینگویج ایک نچلی سطح کی پروگرامنگ لینگویج ہے۔

11th_Comp Science_UM_Ch 1.indd 12 18-04-2023 11:51:16

انٹیگریٹڈ سرکٹس.	IC ایک پیکج ہے جس میں بہت سے سرکٹس، پاتھ ویز، ٹرانجسٹرز اور دیگر الیکٹرانک اجزاء شامل ہیں جو ایک خاص فنکشن یا افعال کی ایک سیریز کو انجام دینے کے لیے مل کر کام کرتے ہیں۔
مائیکرو کمپیوٹر	مائیکرو کمپیوٹر کا استعمال معیاری پرسنل کمپیوٹر کی وضاحت کے لیے کیا جاتا ہے۔
اعلیٰ درجے کی زبانیں۔	ایک اعلی سطحی زبان ایک کمپیوٹر پروگرامنگ زبان ہے جو کمپیوٹر کے ذریعہ محدود نہیں ہے، کسی خاص کام کے لیے ڈیزائن کی گئی ہے، اور سمجھنے میں آسان ہے۔
نیچرل لینگویج پروسیسنگ (NLP)	قدرتی زبان کی پروسیسنگ ایک ایسا طریقہ ہے جو مصنوعی ذہانت میں انسانی زبان سے معنی اخذ کرنے اور عمل کرنے کے لیے استعمال ہوتا ہے
روبوٹکس	روبوٹ ایک اصطلاح ہے جو کیرل کیپیک (Karel Capek) نے 1921 میں RUR (Rossum's Robots Universal) کھیلنے کے لیے بنائی تھی۔ اس کا استعمال کمپیوٹرائزڈ مشین کو بیان کرنے کے لیے کیا جاتا ہے جسے دستی طور پر یا اس کے ارد گرد سے موصول ہونے والے ان پٹ کا جواب دینے کے لیے ڈیزائن کیا گیا ہے۔
نینو ٹیکنالوجی	نینو ٹیکنالوجی ایک انجینئرنگ، سائنس اور ٹیکنالوجی ہے جو مشینیں تیار کرتی ہے یا ایک ایٹم یا ایک مالیکیول کے ساتھ کام کرتی ہے جو 100 نینو میٹر یا اس سے چھوٹا ہے۔
بائیو انجینئرنگ	ایک ایسا نظم جو حیاتیاتی نظام اور بائیو میڈیکل ٹیکنالوجیز پر ڈیزائن اور تجزیہ کے انجینئرنگ اصولوں کا اطلاق کرتا ہے.

11th_Comp Science_UM_Ch 1.indd 13 18-04-2023 11:51:16

باب 2

نمبر سسٹم

سیکھنے کے مقاصد۔

- یہ جاننا کہ کمپیوٹر میموری میں ڈیٹا کی ترجمانی اور ذخیرہ کیسے کرتا ہے۔
- مختلف ڈیٹا کی نمائندگی اور بائنری ارتھمیٹک (ریاضی) سیکھنا۔
- مختلف نمبر سسٹمز کے درمیان تبدیلی کو سیکھنا۔

2.1 تعارف

ڈیٹا کی اصطلاح لفظ ڈیٹم (Datum) سے آتی ہے، جس کا مطلب ایک خام حقیقت ہے۔ ڈیٹا لوگوں، مقامات یا کچھ اشیاء کے بارے میں ایک حقیقت ہے۔

مثال:

'نام'، 'عمر'، 'کلاس'، 'مارکس' اور 'موضوع' کو کچھ متعین متغیر ہونے دیں۔ اب، ہم ان میں سے ہر ایک متغیر کی ایک قدر متعین کرتے ہیں۔

Name	=	Rajesh
Age	=	16
Class	=	XI
Mark	=	65
Subject	=	Computer Science

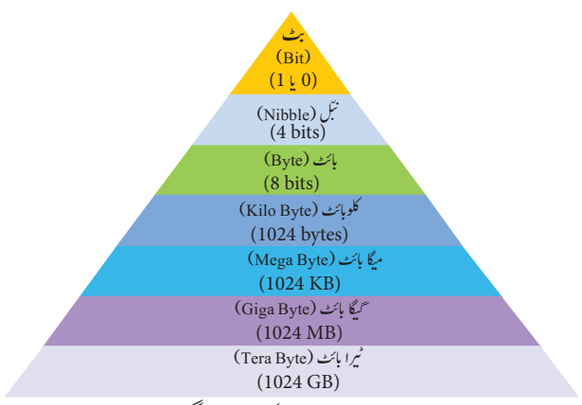
تصویر 2.1 ڈیٹا کی مثال

مندرجہ بالا مثال میں پانچ مختلف متغیرات کی اقدار متعین کی گئی ہے، اسی کو ڈیٹا کہاجاتاہے۔ جب مندرجہ بالا ڈیٹا پر کارروائی کی جاتی ہے، تو ہمیں ایک معلومات ملتی ہے کہ راجیش 16 سال کا ہے اور کلاس XI میں پڑھتاہے۔ کمپیوٹر سائنس کے مضمون میں 65 نمبر حاصل کیا ہے۔

2.2 ڈیٹا کی نمائندگی (Data Representations)

کمپیوٹر ڈیٹا کو '0' (Zero) اور '1' (one) کی شکل میں سنبھالتا ہے۔ کسی بھی قسم کے ڈیٹا جیسے نمبر، حروف تہجی، خصوصی حروف کو '0' یا '1' میں تبدیل کیا جانا چاہئے جسے کمپیوٹر سمجھ سکتا ہے۔ '0' اور '1' جسے کمپیوٹر سمجھ سکتا ہے اسے مشینی زبان کہتے ہیں۔ '0' یا '1' کو 'بائنری ڈیجٹس' 'Binary Digits' (BIT) کہا جاتا ہے۔ اس لیے کمپیوٹر میں ڈیٹا کی نمائندگی کا مطالعہ ضروری ہے۔

- ایک بٹ بائنری ہندسوں کی مختصر شکل ہے جو '0' یا '1' ہوسکتی ہے۔ یہ کمپیوٹر میں ڈیٹا کی بنیادی اکائی ہے۔
- ایک نبل (Nibble) 4 بٹس (بائنری ہندسوں) کا مجموعہ ہے۔
- 8 بٹس کا مجموعہ بائٹ (Byte) کہلاتا ہے۔ بائٹ کو کمپیوٹر میں میموری کے سائز کے پیمائش کی بنیادی اکائی سمجھا جاتا ہے۔
- لفظ کی لمبائی (Word length) سے مراد کمپیوٹر کے CPU کے ذریعے پروسیس شدہ بٹس کی تعداد ہے۔ مثال کے طور پر، ایک لفظ کی لمبائی میں 8 بٹس، 16 بٹس، 32 بٹس اور 64 بٹس ہو سکتے ہیں (موجودہ کمپیوٹرز 32 بٹس یا 64 بٹس استعمال کرتے ہیں۔)



تصویر 2.2 ڈیٹا کی نمائندگی

11th_Comp Science_UM_Ch 2.indd 14 18-04-2023 12:47:09

کمپیوٹر میموری (مین میموری اور سیکنڈری اسٹوریج) کو عام طور پر کلو بائٹ (KB) یا میگا بائٹ (MB) کے لحاظ سے دکھایا جاتا ہے۔ ڈیسیمل سسٹم میں، 1 کلو 1000 کی نمائندگی کرتا ہے، یعنی 10^3۔ بائنری سسٹم میں، 1 کلو بائٹ 1024 بائٹس کی نمائندگی کرتا ہے جو کہ 2^{10} ہے۔ درج ذیل ٹیبل میموری کے مختلف سائز کی نمائندگی کرتا ہے۔

ٹیبل 2.1 میموری سائز (Read 2^10 as 2 Power 10)

Name	Abbr.	Size
کلو (Kilo)	K	2^10 = 1,024
میگا (Mega)	M	2^20 = 1,048,576
گیگا (Giga)	G	2^30 = 1,073,741,824
ٹیرا (Tera)	T	2^40 = 1,099,511,627,776
پیٹا (Peta)	P	2^50 = 1,125,899,906,842,624
ایکسا (Exa)	E	2^60 = 1,152,921,504,606,846,976
زیٹا (Zetta)	Z	2^70 = 1,180,591,620,717,411,303,424
یوٹا (Yotta)	Y	2^80 = 1,208,925,819,614,629,174,706,173

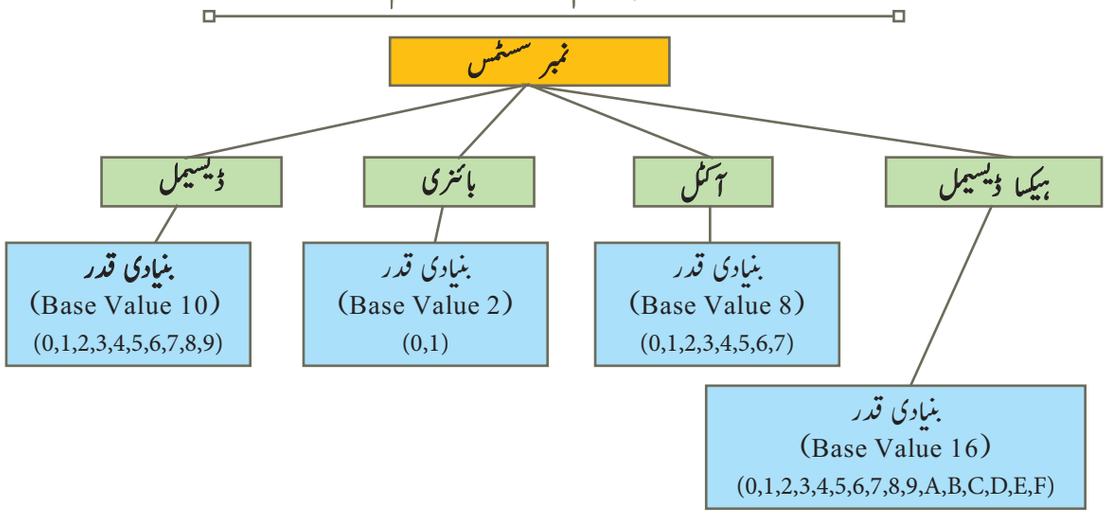
9GLYA

بائٹس کا استعمال متن میں حروف کی نمائندگی کے لیے کیا جاتا ہے۔ کریکٹر سیٹ اور نمبرز کی نمائندگی کرنے کے لیے مختلف قسم کی کوڈنگ اسکیمیں استعمال کی جاتی ہیں۔ سب سے زیادہ استعمال ہونے والی کوڈنگ اسکیم امریکن اسٹینڈرڈ کوڈ فار انفارمیشن انٹرچینج (American Standard Code for Information Interchange) (ASCII) ہے۔ 0 اور 127 کے درمیان ہر بائنری قدر ایک مخصوص کریکٹر کی نمائندگی کے لیے استعمال ہوتی ہے۔ (خالی جگہ) کے لیے ASCII قدر 32 ہے اور عددی 0 کی ASCII قدر 48 ہے۔ چھوٹے حروف تہجی (Lower Case) کے لیے ASCII اقدار کی حد 97 سے 122 تک ہے اور بڑے حروف تہجی (Upper Case) کے لیے ASCII اقدار کی حد 65 سے 90 ہے۔

کیا آپ جانتے ہیں؟

کمپیوٹر کی رفتار کا انحصار بٹس کی تعداد پر ہوتا ہے جس پر وہ ایک ساتھ کارروائی کر سکتا ہے۔ مثال کے طور پر، ایک 64 بٹ کمپیوٹر ایک آپریشن میں 64 بٹ نمبروں کو پروسیس کر سکتا ہے، جبکہ ایک 32 بٹ کمپیوٹر 64 بٹ نمبروں کو چھوٹے ٹکڑوں میں توڑ دیتا ہے، جس سے یہ دھیما ہو جاتا ہے۔

2.3 نمبر سسٹم کے مختلف اقسام



تصویر 2.3 نمبر سسٹم

11th_Comp Science_UM_Ch 2.indd 15 18-04-2023 12:47:10

نمبر سسٹم نمبروں کی نمائندگی کرنے کا ایک طریقہ ہے۔ حقیقی زندگی میں سب سے زیادہ استعمال ہونے والا نمبرنگ سسٹم ڈیسیمل نمبر سسٹم ہے۔ دوسرے نمبر سسٹم بائنری، آکٹل، ہیکسا ڈیسیمل نمبر سسٹم ہیں۔ ہر نمبر کا نظام منفرد طور پر اس کی بنیادی قدر یا ریڈکس (Base value or radix) سے پہچانا جاتا ہے۔ ریڈکس یا بیس ہر نمبر سسٹم میں ہندسوں کی تعداد کی گنتی ہے۔ ریڈکس یا بیس پوزیشنل نمبرنگ سسٹم کے پیچھے عمومی خیال ہے۔

2.3.1 ڈیسیمل نمبر سسٹم

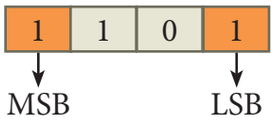
یہ 0,1,2,3,4,5,6,7,8,9 (10 ہندسوں) پر مشتمل ہے۔ یہ ہماری روزمرہ کی زندگی میں استعمال ہونے والا سب سے قدیم اور مقبول نمبر سسٹم ہے۔ پوزیشنل نمبر سسٹم میں ہر اعشاریہ ہندسہ نمبر میں اس کی پوزیشن کے لحاظ سے وزن کیا جاتا ہے۔اس کا مطلب یہ ہے کہ نمبر میں ہر ہندسہ کو 10 سے ضرب دیا جاتا ہے اور اس ہندسوں کی پوزیشن کے مطابق پاور میں اضافہ کیا جاتا ہے۔

مثال :

$$(123)_{10} = 1x10^2 + 2x10^1 + 3x10^0$$
$$= 100 + 20 + 3$$
$$= (123)_{10}$$

2.3.2 بائنری نمبر سسٹم (Binary Number System)

بائنری سسٹم میں صرف دو ہندسے ہیں، یعنی 0 اور 1۔ بائنری سسٹم میں نمبرز کو بیس 2 (Base 2) پر ظاہر کیا جاتا ہے اور پوزیشنل ملٹی پلائرز 2 کی پاورس (Powers) ہیں۔ بائنری نمبر میں سب سے زیادہ بایاں اہم بٹ Most Significant Bit (MSB) کو کہا جاتا ہے جو اس کا سب زیادہ پوزیشنل وزن ہے۔ سب سے زیادہ دائیں اہم بٹ Least Significant Bit (LSB) ہے جو اس کا سب سے چھوٹا پوزیشنل وزن ہے۔



مثال :

بائنری ترتیب $(1101)_2$ میں اعشاریہ کے مساوی ہے:

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$
$$= 8 + 4 + 0 + 1$$
$$= (13)_{10}$$

2.3.3 آکٹل نمبر سسٹم (Octal Number systems)

آکٹل نمبر سسٹم ہندسے0، 1، 2، 3، 4، 5، 6 اور 7 (8 ہندسوں) کو استعمال کرتا ہے۔ ہر آکٹل ہندسے کی اپنی پوزیشنل قدر یا وزن 8 کے پاور پر ہوتا ہے۔

مثال

آکٹل ترتیب $(547)_8$ کا اعشاریہ مساوی ہے:

$$(547)_8 = 5\times8^2 + 4\times8^1 + 7\times8^0$$
$$= 5\times64 + 4\times8 + 7\times1$$
$$= 320 + 32 + 7$$
$$= (359)_{10}$$

2.3.4 ہیکساڈیسیمل نمبر سسٹم (Hexadecimal Number system)

ایک ہیکساڈیسیمل نمبر (Hexadecimal Number) کو بیس 16 (Base 16) کا استعمال کرتے ہوئے دکھایا جاتا ہے۔ ہیکساڈیسیمل یا ہیکس نمبرز بائنری ترتیب کی شارٹ ہینڈ شکل کے طور پر استعمال ہوتے ہیں۔ اس نظام کو زیادہ کمپیکٹ (Compact) انداز میں ڈیٹا کی نمائندگی کرنے کے لیے استعمال کیا جاتا ہے۔ چونکہ 0 سے F تک 16 علامتیں استعمال ہوتی ہیں۔ اس اشارے کو ہیکساڈیسیمل کہا جاتا ہے۔ پہلی 10 علامتیں وہی ہیں جو اعشاریہ کے نظام میں ہیں، 0 سے 9 اور باقی 6 علامتیں حروف تہجی کی ترتیب کے پہلے 6 حروف سے لیے گئے ہیں، A سے F، جہاں A، 10 کی نمائندگی کرتا ہے، B، 11 ہے، C، 12 ہے، D، 13 ہے، E، 14 ہے اور F، 15 ہے۔

11th_Comp Science_UM_Ch 2.indd 16 18-04-2023 12:47:10

ٹیبل 2.2 بائنری، آکٹل، ہیکساڈیسیمل اعشاریہ کے برابر

Decimal	Binary	Octal	Hexadecimal
0	0000	000	0000
1	0001	001	0001
2	0010	002	0002
3	0011	003	0003
4	0100	004	0004
5	0101	005	0005
6	0110	006	0006
7	0111	007	0007
8	1000	010	0008
9	1001	011	0009
10	1010	012	A
11	1011	013	B
12	1100	014	C
13	1101	015	D
14	1110	016	E
15	1111	017	F

مثال

ہیکساڈیسیمل ترتیب $(25)_{16}$ میں اعشاریہ کے برابر ہے۔

$$(25)_{16} = 2\times16^1 + 5\times16^0$$
$$= 32+5$$
$$= (37)_{10}$$

ورکشاپ

1. درج ذیل نمبروں کے لیے نمبر سسٹم کی شناخت کریں۔

شمار نمبر	نمبر	نمبر سسٹم
1	$(1010)_{10}$	ڈیسیمل نمبر سسٹم
2	$(1010)_{2}$	
3	$(989)_{16}$	
4	$(750)_{8}$	
5	$(926)_{10}$	

2. بتائیں کہ درج ذیل نمبرز درست ہیں یا نہیں۔ اگر false ہے تو وجہ بتائیں۔

S.No.	بیانات	ہاں / نہیں	وجہ (اگر false ہے)
1.	786 ایک آکٹل نمبر ہے۔		
2.	101 بائنری نمبر ہے۔		
3.	آکٹل نمبر کا ریڈکس 7 ہے۔		

2.4 نمبر سسٹم کی تبدیلیاں
(Number System Conversions)

2.4.1 اعشاریہ سے بائنری تبدیلی

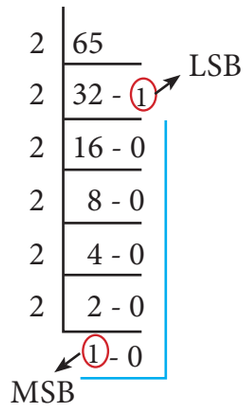
عام طور پر دو طریقے اپنائے جاتے ہیں۔

طریقہ 1: اعشاریہ کو بائنری میں تبدیل کرنے کے لیے "دوبارہ تقسیم بذریعہ 2" **(Repeated Division by 2)** طریقہ استعمال کیا جا سکتا ہے۔ کسی بھی اعشاریہ نمبر کو 2 سے تقسیم کرنے سے 0 یا 1 باقی رہ جائے گا۔ 2 سے دہرائی جانے والی تقسیم 0s اور 1s کی ترتیب چھوڑ دے گی جو اعشاریہ نمبر کے بائنری مساوی بن جائے گی۔ فرض کریں کہ اعشاریہ نمبر N کو بائنری شکل میں تبدیل کرنے کی ضرورت ہے، N کو 2 سے تقسیم کرتے ہوئے اعشاریہ سسٹم میں، ہم ایک حاصل تقسیم N1 اور بقیہ R1 حاصل کریں گے، جہاں R1 کی قدر 0 یا 1 ہو سکتی ہے۔ عمل کو اس وقت تک دہرایا جاتا ہے جب تک کہ حاصل تقسیم 0 یا 1 نہ ہو جائے۔ جب حاصل تقسیم '0' یا '1' ہو تو یہ آخری بقایا قدر ہے۔ حاصل کردہ آخری بقایا قدر سے شروع ہونے والی پہلی بقیہ قدر تک حتمی جواب لکھیں۔

مثال

$(65)_{10}$ کو اس کے مساوی بائنری نمبر میں تبدیل کریں۔

11th_Comp Science_UM_Ch 2.indd 17 18-04-2023 12:47:10



$(65)_{10} = (1\ 0\ 0\ 0\ 0\ 0\ 1)_2$

طریقہ 2: 2 کی اختیارات کا مجموعہ۔

اعشاریہ نمبر کو 2 کی اختیارات شامل کرکے اور پھر نمبر کی کل قدر حاصل کرنے کے لیے ضرورت کے مطابق بٹس شامل کرکے بائنری نمبر میں تبدیل کیا جاسکتا ہے۔

a) 2 کی سب سے بڑی اختیارات تلاش کریں جو 65 سے چھوٹی یا اس کے برابر ہو۔

$65_{10} > 64_{10}$

b) 64 کے بٹ کو 1 پر سیٹ کریں اور اصل نمبر سے 64 کو گھٹائیں۔

65-64=1

c) 32 بقیہ کل سے زیادہ ہے۔ لہذا، 32 کے بٹ کو 0 پر سیٹ کریں۔

d) 16 بقیہ کل سے بڑا ہے۔ لہذا، 16 کے بٹ کو 0 پر سیٹ کریں۔

e) 8 بقیہ کل سے بڑا ہے۔ لہذا، 8 کے بٹ کو 0 پر سیٹ کریں۔

f) 4 بقیہ کل سے بڑا ہے۔لہذا، 4 کے بٹ کو 0 پر سیٹ کریں۔

g) 2 بقیہ کل سے بڑا ہے۔ لہذا، 2 کے بٹ کو 0 پر سیٹ کریں۔

h) چونکہ باقی قدر 1 کے بٹ کے برابر ہے، اسے 1 پر سیٹ کریں۔

1-1=0

تبدیلی مکمل ہے $65_{10} = (1000001)_2$

مثال

تبادلوں کے اقدامات درج ذیل ہیں:

دیا ہوا نمبر: 65

مساوی یا قدر 2 کی پاور سے کم ہے: 64

(1) $65 - 64 = 1$

(2) $1 - 1 = 0$

Power's of 2	64	32	16	8	4	2	1
Binary Number	1	0	0	0	0	0	1

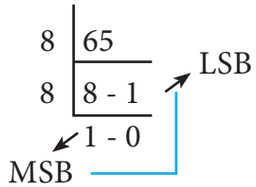
$65_{10} = (1000001)_2$

2.4.2 اعشاریہ سے آکٹل کی تبدیلی

اعشاریہ کو آکٹل میں تبدیل کرنے کے لیے، "بار بار تقسیم بذریعہ 8" (Repeated Division by 8) کا طریقہ استعمال کیا جا سکتا ہے۔ طریقہ وہی ہے جو ہم نے 2.4.1 میں سیکھا ہے، لیکن اس طریقہ میں، ہمیں دیئے گئے نمبر کو 8 سے تقسیم کرنا ہو گا۔

مثال

$(65)_{10}$ کو اس کے مساوی آکٹل نمبر میں تبدیل کریں۔



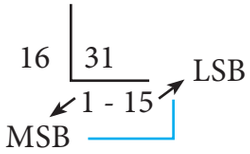
$(65)_{10} = (1\ 0\ 1)_8$

2.4.3 اعشاریہ سے ہیکساڈیسیمل کی تبدیلی

اعشاریہ کو ہیکساڈیسیمل میں تبدیل کرنے کے لیے، "بار بار تقسیم بذریعہ 16" (Repeated Division by 16) کا طریقہ استعمال کیا جا سکتا ہے۔ طریقہ وہی ہے جو ہم نے 2.4.1 میں سیکھا ہے، لیکن اس طریقہ میں، ہمیں دیے گئے نمبر کو 16 سے تقسیم کرنا ہو گا۔

مثال

$(31)_{10}$ کو اس کے مساوی ہیکساڈیسیمل نمبر میں تبدیل کریں۔



$(16)_{10} = (1F)_{16}$ (Refer Table 2.2 F=15)

11th_Comp Science_UM_Ch 2.indd 18 18-04-2023 12:47:10

2.4.4 فرکشنل ڈیسیمل کو بائنری میں تبدیلی
(Conversion of Fractional Decimal to Binary)

اس طرح کے اعشاریہ فرکشنل کو تبدیل کرنے کے لیے 2 سے بار بار ضرب (Repeated Multiplication by 2) لگانے کا طریقہ استعمال کرنا پڑتا ہے۔

2 سے بار بار ضرب کے طریقہ کار میں شامل اقدامات:

مرحلہ 1: اعشاریہ فرکشنل کو 2 سے ضرب دیں اور عددی حصے کو نوٹ کریں۔ عددی حصہ یا تو 0 یا 1 ہے۔

مرحلہ 2: پچھلے پروڈکٹ کے عددی حصے کو ضائع کر دیں۔ پچھلے پروڈکٹ کے جزوی حصے کو 2 سے ضرب دیں۔
مرحلہ 1 کو دہرائیں جب تک کہ وہی حصہ دوبارہ نہ آجائے یا ختم نہ ہوجائے (0)۔

مرحلہ 3: نتیجے میں آنے والا عددی حصہ0s اور1s کا ایک تسلسل بناتا ہے جو اعشاریہ فرکشن کے بائنری مساوی بن جاتا ہے۔

مرحلہ 4: حتمی جواب حاصل ہونے والے پہلے عددی حصے سے حاصل ہونے والے آخری عدد تک لکھا جانا ہے۔

عددی حصہ

$0.2 \times 2 = 0.4$ → 0 (first integer part obtained)
$0.4 \times 2 = 0.8$ → 0
$0.8 \times 2 = 1.6$ → 1
$0.6 \times 2 = 1.2$ → 1
$0.2 \times 2 = 0.4$ → 0 (last integer part obtained)

نوٹ: فریکشن دہرایا جاتا ہے، پروڈکٹ وہی ہے جو پہلے مرحلے میں ہے۔

مساوی فریکشنل بائنری نمبر حاصل کرنے کے لیے عددی حصوں کو اوپر سے نیچے تک لکھیں۔

لہذا

$(0.2)_{10} = (0.00110011\ldots)_2 = (0.00110011)_2$

ورکشاپ

3. درج ذیل اعشاریہ نمبروں کو اس کے مساوی بائنری، آکٹل، ہیکساڈیسیمل میں تبدیل کریں۔

1) 1920 2) 255 3) 126

2.4.5 بائنری سے اعشاریہ کی تبدیلی
(Binary to Decimal Conversion)

بائنری کو اعشاریہ میں تبدیل کرنے کے لیے ہم پوزیشنل اشارے کا طریقہ استعمال کر سکتے ہیں۔

مرحلہ 1: بائنری ہندسوں کو لکھیں اور 2 کی پاورس کو دائیں سے بائیں درج کریں (پوزیشنل نوٹیشن (Positional Notation))

مرحلہ 2: ہندسے کے لیے لکھے گئے ہر پوزیشنل اشارے کے لیے، اب مساوی وزن لکھیں۔

مرحلہ 3: ہر ہندسے کو اس کے متعلقہ وزن سے ضرب دیں۔

مرحلہ 4: تمام اقدار شامل کریں۔

ٹیبل 2.3 پوزیشنل اشارے اور وزن

وزن	پوزیشنل نوٹیشن
64	2^6
128	2^7
256	2^8
512	2^9
1024	2^{10}

وزن	پوزیشنل نوٹیشن
1	2^0
2	2^1
4	2^2
8	2^3
16	2^4
32	2^5

مثال

$(111011)_2$ کو اس کے مساوی اعشاریہ نمبر میں تبدیل کریں

وزن	32	16	8	4	2	1
پوزیشنل نوٹیشن	2^5	2^4	2^3	2^2	2^1	2^0
دئے گئے نمبر	1	1	1	0	1	1

11th_Comp Science_UM_Ch 2.indd 19 18-04-2023 12:47:10

$$32+16+8+0+2+1 = (59)_{10}$$
$$(111011)_2 = (59)_{10}$$

2.4.6 بائنری سے آکٹل کی تبدیلی

مرحلہ 1: دیئے گئے بائنری نمبر کو دائیں سے بائیں 3 بٹس میں گروپ کریں۔

مرحلہ 2: آپ 3 بٹس کا ایک گروپ بنانے کے لیے پہلے والے 0 کو شامل کر سکتے ہیں اگر بائیں زیادہ تر گروپ میں 3 سے کم بٹس ہوں۔

مرحلہ 3: "2 کے پاور پوزیشنل ویٹ طریقہ" (Power Positional weight method) کا استعمال کرتے ہوئے مساوی آکٹل قدر میں تبدیل کریں۔

ٹیبل 2.4 آکٹل نمبرز اور ان کے بائنری مساوی

بائنری مساوی	آکٹل
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

مثال

$(11010110)_2$ کو اوکٹل مساوی نمبر میں تبدیل کریں۔

مرحلہ 1: دیے گئے نمبر کو دائیں سے بائیں تین بٹس میں گروپ کریں۔

011 010 110

نوٹ: بائیں کے زیادہ تر گروپس 3 بٹس سے کم ہیں لہذا 0 کو بائیں طرف جمع کرنے کے بعد یہ مکمل طور پر تین بٹس کا گروپ بن جاتا ہے۔

مرحلہ- 2: ہر ایک کے آکٹل مساوی تلاش کریں گروپ

011	010	110
3	2	6

$(11010110)_2 = (326)_8$

2.4.7 بائنری سے ہیکساڈیسیمل کی تبدیلی

مرحلہ 1: دیے گئے نمبر کو دائیں سے بائیں 4بٹس میں گروپ کریں۔

مرحلہ 2: 4 بٹس کا گروپ بنانے کے لیے آپ پچھلے 0 کا اضافہ کرسکتے ہیں اگر بائیں گروپس کے زیادہ تر حصہ میں 4 بٹس سے کم ہوں۔

مرحلہ 3: مساوی ہیکساڈیسیمل کو تبدیل کریں۔ "2 کی پاور پوزیشنل وزن کا طریقہ" استعمال کرتے ہوئے، مساوی ہیکسا ڈیسیمل کو تبدیل کریں۔

مثال

$(1111010110)_2$ ہیکساڈیسیمل نمبر میں تبدیل کریں۔

مرحلہ 1: دیے گئے نمبر کو دائیں سے بائیں 4 بٹس میں گروپ کریں۔

0011 1101 0110

نوٹ: 0's کو سب سے زیادہ بائیں گروپ میں شامل کریں تاکہ 4 بٹس کا ایک گروپ بن جائے۔

0011	1101	0110
3	D	6

$(1111010110)_2 = (3D6)_{16}$

2.4.8 فریکشنل بائنری کو اعشاریہ مساوی میں تبدیلی

فرکشنل بائنری نمبر کو اعشاریہ مساوی میں تبدیلی کرنے کے اقدامات پر عمل کریں۔

مرحلہ 1: بائنری کے لازمی حصہ کو پوزیشنل اشارے کا طریقہ استعمال کرتے ہوئے اعشاریہ مساوی میں تبدیل کریں۔ (طریقہ کار وہی ہے جیسا کہ 2.4.5 میں زیر بحث آیا ہے۔)

مرحلہ 2: بائنری کے جزوی حصے کو اعشاریہ مساوی میں تبدیل کریں۔

مرحلہ 2.1: بائنری ہندسہ کے جزوی حصہ کو لکھیں۔

مرحلہ 2.2: تمام ہندسوں کے لیے 2 کی پاور کو بائیں سے دائیں میں $2^{-1}, 2^{-2}, 2^{-3} \ldots\ldots 2^{-n}$ یہاں سے شروعات کریں، اب مساوی وزن لکھیں۔

11th_Comp Science_UM_Ch 2.indd 20 18-04-2023 12:47:10

مرحلہ 2.3: ہر ہندسے کو اس کے متعلقہ وزن سے ضرب دیں۔

مرحلہ 2.4: وہ تمام اقدار شامل کریں جو آپ نے مرحلہ 2.3 میں حاصل کیا ہے۔

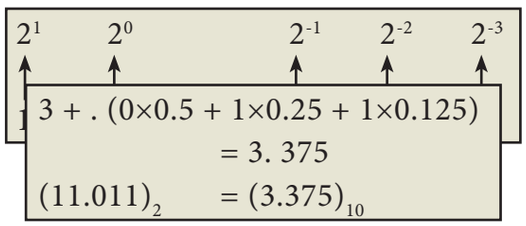
ٹیبل 2.5 پوزیشنل اشارے اور وزن

پوزیشنل اشارے	وزن
2^{-1} (1/2)	0.5
2^{-2} (1/4)	0.25
2^{-3} (1/8)	0.125
2^{-4} (1/16)	0.0625
2^{-5} (1/32)	0.03125
2^{-6} (1/64)	0.015625
2^{-7} (1/128)	0.0078125

مرحلہ 3: حتمی جواب حاصل کرنے کے لیے لازمی حصہ (تبدیلی کے بعد) لکھیں، اس کے بعد اعشاریہ پوائنٹ (.) اور جواب مرحلہ 2.4 میں آچکا ہے۔

مثال

دیئے گئے بائنری نمبر $(11.011)_2$ کو اس کے اعشاریہ مساوی عددی حصہ $3 = (11)_2$ میں تبدیل کریں۔ (بحوالہ ٹیبل 2.2)



ورکشاپ

4. دیے گئے بائنری نمبر کو اعشاریہ مساوی، آکٹل اور ہیکساڈیسیمل نمبر میں تبدیل کریں۔

1) 101110101 2) 1011010 3) 101011111

2.4.9 آکٹل سے اعشاریہ کی تبدیلی

اوکٹل کو اعشاریہ میں تبدیل کرنے کے لیے، ہم پوزیشنل اشارے کا طریقہ استعمال کر سکتے ہیں۔

1. آکٹل ہندسوں کو لکھیں اور 8 کی پاورس کی فہرست دائیں سے بائیں بنائیں (پوزیشنل نوٹیشن)۔
2. ہندسوں کے ہر پوزیشنل اشارے کے لئے مساوی وزن لکھیں۔
3. ہر ہندسے کو اس کے متعلقہ وزن کے ساتھ ضرب دیں۔
4. تمام اقدار شامل کریں۔

مثال

$(1265)_8$ کو مساوی اعشاریہ نمبر میں تبدیل کریں۔

وزن	512	64	8	1
پوزیشنل اشارے	8^3	8^2	8^1	8^0
دیئے گئے نمبر	1	2	6	5

$(1265)_8 = 512 \times 1 + 64 \times 2 + 8 \times 6 + 1 \times 5$

$= 512 + 128 + 48 + 5$

$(1265)_8 = (693)_{10}$

2.4.10 آکٹل سے بائنری کی تبدیلی (Octal to Decimal Converssion)

دیئے گئے نمبر میں ہر آکٹل ہندسے کو پوزیشنل اشارے کا استعمال کرتے ہوئے اس کے 3 ہندسوں کے بائنری مساوی لکھیں۔

مثال

$(6213)_8$ کو مساوی بائنری نمبر میں تبدیل کریں۔

6	2	1	3
↓	↓	↓	↓
110	010	001	011

$(6213)_8 = (110010001011)_2$

ورکشاپ

5. درج ذیل آکٹل نمبروں کو بائنری نمبرز میں تبدیل کریں۔

472(A) 145 (B) 347 (C)

6247 (D) 645 (E)

11th_Comp Science_UM_Ch 2.indd 21 18-04-2023 12:47:10

2.4.11 ہیکسا ڈیسیمل سے ڈیسیمل کی تبدیلی
(Hexadecimal to Decimal Conversion)

ہیکسا ڈیسیمل کو ڈیسیمل میں تبدیل کرنے کے لیے ہم پوزیشنل اشارے کا طریقہ استعمال کر سکتے ہیں۔

1. ہیکسا ڈیسیمل ہندسوں کو لکھیں اور دائیں سے بائیں 16 کے اختیارات درج کریں (پوزیشنل نوٹیشن)
2. ہر پوزیشنل اشارے ہندسے کے لیے لکھا گیا ہے ، اب مساوی وزن لکھیں۔
3. ہر ہندسے کو اس کے متعلقہ وزن کے ساتھ ضرب دیں۔
4. ایک حتمی قدر حاصل کرنے کے لیے تمام اقدار شامل کریں۔

مثال

$(25F)_{16}$ کو اس کے مساوی اعشاریہ نمبر میں تبدیل کریں۔

1	16	256	Weight
16^0	16^1	16^2	Positional Notation
F(15)	5	2	Given number

$$(25F)_{16} = 2\times256 + 5\times16 + 15\times1$$
$$= 512 + 80 + 15$$
$$(25F)_{16} = (607)_{10}$$

2.4.12 ہیکسا ڈیسیمل سے بائنری کی تبدیلی
(Hexadecimal to Binary Conversion)

دیے گئے ہر ایک ہیکساڈیسیمل ہندسے نمبر کے لیے پوزیشنل اشارے کا طریقہ استعمال کرتے ہوئے 4 بٹس بائنری مساوی لکھیں۔

مثال

$(8BC)_{16}$ کو مساوی بائنری نمبر میں تبدیل کریں۔

8	B	C
↓	↓	↓
1000	1011	1100

$$(8BC)_{16} = (100010111100)_2$$

ورکشاپ

6. درج ذیل ہیکسا ڈیسیمل نمبرز سے بائنری نمبرز کو تبدیل کریں۔

A6 (A)	BE (B)
9BC8 (C)	BC9 (D)

2.5 بائنری نمائندگی برائے سائنڈ نمبرز
(Binary Representation for Signed Numbers)

کمپیوٹر دونوں مثبت (Unsigned) اور منفی (Signed) نمبرز کو سنبھال سکتے ہیں۔ منفی بائنری نمبرز کی نمائندگی کرنے کے آسان ترین طریقہ کو سائنڈ میگنیٹیوڈ (Signed Magnitude) کہا جاتا ہے۔ سائنڈ میگنیٹیوڈ (Signed Magnitude) طریقہ میں سب سے زیادہ بایاں بٹ سب سے اہم بٹ (MSB) ہے، جسے سائن بٹ (signed bit) یا پیرائٹی بٹ (parity bit) کہا جاتا ہے۔
کمپیوٹر میں مختلف طریقوں سے نمبرز دکھایا جاتا ہے۔

- سائنڈ میگنیٹیوڈ (Signed Magnitude) کی نمائندگی۔
- 1 کی تکمیل
- 2 کی تکمیل

2.5.1 سائنڈ میگنیٹیوڈ (Signed Magnitude) کی نمائندگی۔

پورے نمبروں کی قدراس سے پہلے استعمال ہونے والے نشان سے طے کیا جائے۔ اگر نمبر پر '+' کا نشان ہے یا کوئی نشان نہیں ہوگا تو اسے مثبت سمجھا جاتا ہے۔ اگر نمبر پر '-' کا نشان ہے تو اسے منفی سمجھا جائے گا۔

مثال:

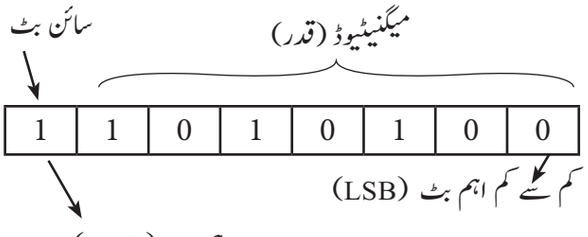
43 + یا 43 ایک مثبت نمبر ہے۔

43- ایک منفی نمبر ہے۔

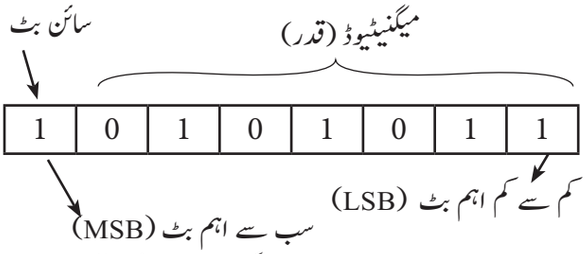
سائنڈ بائنری نمائندگی میں، سب سے زیادہ بائیں بٹ کو سائن بٹ سمجھا جاتا ہے۔ اگر یہ بٹ 0 ہے تو یہ ایک مثبت نمبر ہے اوراگر یہ 1 ہے تو یہ ایک منفی نمبر ہے۔ اس لیےایک سائنڈ بائنری نمبر میں 8 بٹس ہوتے ہیں، صرف 7 قدروں کو ذخیرہ (میگنیٹیوڈ) کرنے کے لیے استعمال ہونے والے بٹس اور 1 بٹ سائن کے لیے استعمال ہوتا ہے۔

11th_Comp Science_UM_Ch 2.indd 22 18-04-2023 12:47:11

+43 کی نمائندگی درج ذیل میموری میں کی جاتی ہے:



-43 کی نمائندگی درج ذیل میموری میں کی جاتی ہے:



2.5.2 1 کی تکمیلی نمائندگی
(1's Complement representation)

سائنڈ نمبروں کی نمائندگی کا یہ ایک آسان طریقہ ہے۔ یہ صرف منفی نمبر کے لیے ہے یعنی وہ نمبر جس کا MSB 1 ہے۔

1 کے عدد کی تکمیل کو تلاش کرنے کے لیے ذیل کے اقدامات پر عمل کرنا ہے :

مرحلہ 1: دیے گئے اعشاریہ نمبر کو بائنری میں تبدیل کریں۔

مرحلہ 2: چیک کریں کہ آیا بائنری نمبر 8 بٹس پر مشتمل ہے ،اگر کم ہو تو 0 کو سب سے زیادہ بائیں بٹ میں اسے 8 بٹس بنانے کے لیے شامل کریں۔

مرحلہ 3: تمام بٹس کو الٹ دیں (یعنی 1 کو 0 کے طور پر تبدیل کریں اور 0 کوبطور 1)

مثال

$(-24)_{10}$ کے لیے 1 کی تکمیل تلاش کریں۔

دیے گئے نمبر	بائنری نمبر	1 کی تکمیل
$(-24)_{10}$	00011000	11100111

2.5.3 2 کی تکمیلی نمائندگی
(2's Complement representation)

2 کی تکمیل کا طریقہ منفی نمبر کے لیے مندرجہ ذیل ہے:

a. بائنری میں تمام بٹس کو الٹ دیں ترتیب (یعنی، ہر 0 سے 1 اور ہر 1 سے 0 یعنی 1 کی تکمیل)

b. نتیجہ میں 1 کا اضافہ کم سے کم اہم بٹ (LSB) میں کریں۔

مثال

2 کی تکمیل $(-24)_{10}$ کی نمائندگی کرتی ہے۔

Binary equivalent of +24:	11000
8bit format:	00011000
1's complement:	11100111
Add 1 to LSB:	+1
2's complement of -24:	**11101000**

ورکشاپ

7. درج ذیل اعشاریہ نمبر کے لیے 1 کا تکمیلی نمبر اور 2 کا تکمیلی نمبر لکھیں:

-46 (D) -65 (C) -13 (B) 22 (A)

2.6 بائنری ارتھ میٹک (Binary Arithmetic)

اعشاریہ نمبر کے طور پر، بائنری نمبر بھی حساب کی اجازت دیتا ہے جیسے جوڑ، گھٹاؤ، ضرب اور تقسیم۔ مندرجہ ذیل سیشن صرف بائنری جوڑاور گھٹاؤ کے ساتھ عمل کرتا ہے۔

2.6.1 بائنری جوڑ (Binary Addition)

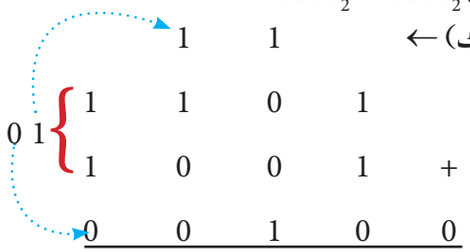
مندرجہ ذیل ٹیبل مفید ہے جب دو بائنری نمبرز کو شامل کیا جائے ۔

A	B	جمع (A + B)	کیری بٹ
0	0	0	-
0	1	1	-
1	0	1	-
1	1	0	1

10 = 1 + 1 میں، بطور جمع 0 کو سمجھا جاتا ہے اور 1 کو بطور کیری بٹ (Carry Bit)۔ اس کیری بٹ کو بٹ پیٹرن کی پچھلی پوزیشن کے ساتھ شامل کیا جاتا ہے۔

11th_Comp Science_UM_Ch 2.indd 23 18-04-2023 12:47:11

مثال: جوڑیں $1001_2 + 1011_2$



$$10100_2 = 1001_2 + 1011_2$$

مثال ذیل کے لیے بائنری جوڑ کو انجام دیں۔

$$12_{10} + 23_{10}$$

مرحلہ 1: 23 اور 12 کو بائنری شکل میں تبدیل کریں۔

23_{10}

2 کا پاور	16	8	4	2	1
بائنری نمبر	1	0	1	1	1

$00010111_2 = 23_{10}$

12_{10}

2 کا پاور	8	4	2	1
بائنری نمبر	1	1	0	0

$00001100_2 = 12_{10}$

مرحلہ 2: 23 اور 12 کا بائنری جوڑ:

							کیری بٹ ← Carry Bit
		1	1	1			
1	1	1	0	1	0	0	$23_{10} = 0$
0	0	1	1	0	0	0	$12_{10} = 0$
1	**1**	**0**	**0**	**0**	**1**	**0**	**$35_{10} = 0$**

2.6.2 بائنری گھٹاؤ (Binary Subtraction)

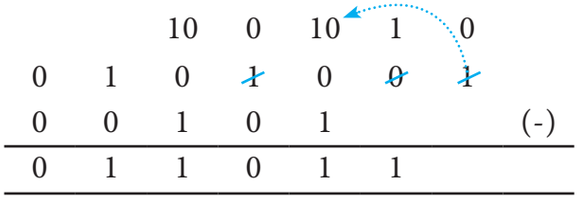
بائنری گھٹاؤ کے لیے مندرجہ ذیل ٹیبل ہے۔

A	B	تفریق (A-B)	ادھار
0	0	0	0
1	0	1	0
1	1	0	0
0	1	1	1

1 کو 0 سے گھٹاتے وقت، 1 کو اگلے سب سے اہم بٹ (MSB) سے ادھار لیں، جب اگلے سب سے اہم بٹ سے ادھار لیں ،اگر یہ 1 ہو تو اسے 0 سے بدل دیں۔ اگر اگلا سب سے زیادہ اہم بٹ 0 ہے تو آپ کو ایک زیادہ اہم بٹ سے ادھار لینا چاہیے۔

جس میں 1 ہو اور اسے 0 اور 0s سے بدل دیں یہاں تک کہ وہ 1s ہوجائے۔

مثال گھٹاؤ (Subtract) $1001010_2 - 10100_2$



ثال مندرجہ ذیل کے لیے بائنری جوڑ کو انجام دیں۔

$$(-21)_{10} + (5)_{10}$$

مرحلہ 1: 21- اور 5 کو بائنری شکل میں تبدیل کریں۔

21_{10}

2 کا پاور	16	8	4	2	1
بائنری نمبر	1	0	1	0	1

$00010101_2 = 21_{10}$

5_{10}

2 کا پاور	4	2	1
بائنری نمبر	1	0	1

$00000101_2 = 5_{10}$

مرحلہ نمبر 2

1	0	1	0	1	0	0	0	21_{10}
0	1	0	1	0	1	1	1	1's Compliment
1	1	0	1	0	1	1	1	2's Compliment

مرحلہ 3:

21- اور 5 کا بائنری جوڑ :

								کیری بٹ (Carry bit)
	1	1	1	1				
1	1	0	1	0	1	1	1	-21_{10}
1	0	1	0	0	0	0	0	5_{10}
0	0	0	0	1	1	1	1	-16_{10} (Result)

ورکشاپ

8. درج ذیل بائنری ریاضیات کو انجام دیں۔

$10_{10} + 15_{10}$ (A) $\quad -12_{10} + 5_{10}$ (B)

$14_{10} - 12_{10}$ (C) $\quad (-2_{10}) - (-6_{10})$ (D)

11th_Comp Science_UM_Ch 2.indd 24 18-04-2023 12:47:11

2.7 میموری میں کیریکٹرس کی نمائندگی
(Representing Characters in Memory)

جیسا کہ تعارف میں دکھایا گیا ہے، کمپیوٹر کو دیا گیا تمام ان پٹ ڈیٹا قابل فہم شکل میں ہونا چاہیے۔ عموما 26 بڑے حروف، 26 چھوٹے حروف حروف، 0 سے 9 ہندسوں اور خصوصی حروف کمپیوٹر میں استعمال ہوتے ہیں، جسے کیریکٹر سیٹ (Character set) کہا جاتا ہے۔ یہ تمام کریکٹر سیٹ صرف نمبروں کے ذریعے اشارہ کیا جاتا ہے۔ تمام کیریکٹر س جو کیریکٹر سیٹ میں ہیں ان کو ایک عام انکوڈنگ سسٹم (Encoding Systems) کی ضرورت پڑتی ہے۔ کمپیوٹر میں کئی طرح کے انکوڈنگ سسٹم کا استعمال ہوتا ہے جو درج ذیل ہیں۔

- BCD - بائنری کوڈیڈ ڈیسیمل (Binary Coded Decimal)
- EBCDIC - توسیعی بائنری کوڈیڈ ڈیسیمل انٹرچینج کوڈ (Extended Binary Coded Decimal Interchange Code)
- ASCII - امریکی معیاری کوڈ برائے تبادلۂ معلومات (American Standard Code for Information Interchange)
- یونیکوڈ (Unicode)
- ISCII - ہندوستانی معیاری کوڈ برائے تبادلۂ معلومات (Indian Standard Code for Information Interchange)

2.7.1 بائنری کوڈیڈ ڈیسیمل (BCD)

یہ انکوڈنگ سسٹم فی الحال عمل میں نہیں ہے۔ یہ 2^6 بٹ انکوڈنگ سسٹم ہے۔ یہ صرف $2^6 = 64$ حروف کو سنبھال سکتا ہے۔

2.7.2 امریکن اسٹینڈرڈ کوڈ فار انفارمیشن انٹرچینج (ASCII)

یہ سب سے زیادہ مقبول انکوڈنگ سسٹم ہے جو ریاست ہائے متحدہ امریکہ کی طرف سے تسلیم شدہ ہے۔ زیادہ تر کمپیوٹر اس سسٹم کو استعمال کرتے ہیں۔ یاد رکھیں کہ یہ انکوڈنگ سسٹم صرف انگریزی حروف سنبھال سکتا ہے۔ یہ کوڈنگ سسٹم 2^7 بٹ کو سنبھال سکتا ہے، جس کا مطلب 128 حروف ہیں۔

اس سسٹم میں ہر ایک حرف ایک انفرادی نمبر رکھتا ہے۔ (بحوالہ **ضمیمہ**)

نیا ایڈیشن (Version) ASCII -8، 2^8 بٹس ہیں اور 256 حروف کو سنبھال سکتے ہیں جو 0 سے 255 منفرد نمبروں کی نمائندگی کرتا ہے۔

ASCII کوڈ بڑے حرف 'A' کے مساوی 65 ہے۔ بائنری ASCII کی نمائندگی (7 بٹ) ہے جس کی قدر 1000001۔ نیز ASCII 01000001 میں 8 بٹ ہے۔

2.7.3 اکسٹینڈڈ بائنری کوڈیڈ ڈیسیمل انٹرچینج کوڈ (EBCDIC)

یہ ASCII کوڈ 8 بٹ کی نمائندگی کے ساتھ ملتا جلتا ہے ۔ یہ کوڈنگ سسٹم بین الاقوامی کاروبار (IBM) کی طرف سے تیار کیا گیا ہے۔ کوڈنگ سسٹم 256 حروف کو سنبھال سکتا ہے۔ ASCII میں ان پٹ کوڈ کو EBCDIC سسٹم میں تبدیل کیا جا سکتا ہے اور اس کے برعکس۔

2.7.4 انڈین اسٹینڈرڈ کوڈ فار انفارمیشن انٹرچینج (ISCII)

ISCII ہندوستان کے مقامی زبانوں کے حروف کو سنبھالنے کا ایک نظام ہے۔ یہ 8 بٹ کوڈ نگ سسٹم ہے۔ اس لیے یہ 256 (2^8) حروف کو سنبھال سکتا ہے۔ یہ نظام ہندوستان میں محکمۂ الیٹرانکس کی طرف سے سال 88 - 1986 میں تیار کیا گیا اور بیورو آف انڈین اسٹینڈرس (BIS) کے ذریعہ تسلیم کیا گیا ہے۔ اب یہ کوڈنگ سسٹم یونیکوڈ کے ساتھ مربوط ہے۔

2.7.5 یونیکوڈ (Unicode)

یہ کوڈنگ سسٹم زیادہ تر جدید کمپیوٹر میں استعمال ہوتا ہے۔ مقبول کوڈنگ اسکیم ASCII کے بعد یونیکوڈ ہے۔ ASCII صرف 256 حروف کی نمائندگی کر سکتا ہے ۔ اس لیے انگریزی اور یورپی زبانوں کو ASCII اکیلے سنبھال سکتا ہے۔ خاص طور پر ایسی صورت حال میں، جب تمل ،ملیالم، کنڑا اور تیلگو جیسی زبانیں کو ASCII کی طرف سے نمائندگی حاصل نہ ہو سکی۔ لہذا ، یونیکوڈ تمام عالمی زبانوں کے کوڈنگ سسٹم سنبھالنے لیے تیار کیا گیا ۔ یہ 16 بٹ کوڈ ہے اور 65536 حروف کو سنبھال سکتا ہے۔

یونیکوڈ اسکیم ہیکسا ڈیسیمل کی طرف سے متعارف کرایا گیا ہے۔ تامل، ملیالم، تیلگو اور کنڑا کے یونیکوڈ ٹیبل 2.6 میں دکھایا گیا ہے۔

11th_Comp Science_UM_Ch 2.indd 25 18-04-2023 12:47:11

ٹیبل 2.6

ملیالم کا یونیکوڈ ٹیبل

	0D0	0D1	0D2	0D3	0D4	0D5	0D6	0D7
0	ം 0D00	ഐ 0D10	ഠ 0D20	ര 0D30	ീ 0D40		ൠ 0D60	൰ 0D70
1	ഁ 0D01		ഡ 0D21	റ 0D31	ു 0D41		ൡ 0D61	൱ 0D71
2	ം 0D02	ഒ 0D12	ഢ 0D22	ല 0D32	ൂ 0D42		ൢ 0D62	൲ 0D72
3	ഃ 0D03	ഓ 0D13	ണ 0D23	ള 0D33	ൃ 0D43		ൣ 0D63	൳ 0D73
4		ഔ 0D14	ത 0D24	ഴ 0D34	ൄ 0D44	ഄ 0D54		൴ 0D74
5	അ 0D05	ക 0D15	ഥ 0D25	വ 0D35		യ 0D55		൵ 0D75
6	ആ 0D06	ഖ 0D16	ദ 0D26	ശ 0D36	ൊ 0D46	ഴ 0D56	൦ 0D66	൶ 0D76
7	ഇ 0D07	ഗ 0D17	ധ 0D27	ഷ 0D37	ോ 0D47	ൗ 0D57	൧ 0D67	൷ 0D77
8	ഈ 0D08	ഘ 0D18	ന 0D28	സ 0D38	ൌ 0D48	ഩ 0D58	൨ 0D68	൸ 0D78
9	ഉ 0D09	ങ 0D19	ഩ 0D29	ഹ 0D39		ഺ 0D59	൩ 0D69	൹ 0D79
A	ഊ 0D0A	ച 0D1A	പ 0D2A	ഺ 0D3A	ൎ 0D4A	഻ 0D5A	൪ 0D6A	ൺ 0D7A
B	ഋ 0D0B	ഛ 0D1B	ഫ 0D2B	഻ 0D3B	൏ 0D4B	഼ 0D5B	൫ 0D6B	ൻ 0D7B
C	ഌ 0D0C	ജ 0D1C	ബ 0D2C	഼ 0D3C	൐ 0D4C	ഽ 0D5C	൬ 0D6C	ർ 0D7C
D		ഝ 0D1D	ഭ 0D2D	ഽ 0D3D	൑ 0D4D	ാ 0D5D	൭ 0D6D	ൽ 0D7D
E	എ 0D0E	ഞ 0D1E	മ 0D2E	ാ 0D3E	൒ 0D4E	ി 0D5E	൮ 0D6E	ൾ 0D7E
F	ഏ 0D0F	ട 0D1F	യ 0D2F	ി 0D3F	൓ 0D4F	ീ 0D5F	൯ 0D6F	ൿ 0D7F

تمل کا یونیکوڈ ٹیبل

	0B8	0B9	0BA	0BB	0BC	0BD	0BE	0BF
0		ஐ 0B90		ர 0BB0	ீ 0BC0	ௐ 0BD0		௰ 0BF0
1				ற 0BB1	ு 0BC1			௱ 0BF1
2	ஂ 0B82	ஒ 0B92		ல 0BB2	ூ 0BC2			௲ 0BF2
3	ஃ 0B83	ஓ 0B93	ண 0BA3	ள 0BB3				ெ 0BF3
4		ஔ 0B94	த 0BA4	ழ 0BB4				ே 0BF4
5	அ 0B85	க 0B95		வ 0BB5				ை 0BF5
6	ஆ 0B86			ஶ 0BB6	ொ 0BC6		൦ 0BE6	௦ 0BF6
7	இ 0B87			ஷ 0BB7	ோ 0BC7	ௗ 0BD7	க 0BE7	௧ 0BF7
8	ஈ 0B88		ந 0BA8	ஸ 0BB8	ௌ 0BC8		உ 0BE8	௨ 0BF8
9	உ 0B89	ங 0B99	ன 0BA9	ஹ 0BB9			ங 0BE9	௩ 0BF9
A	ஊ 0B8A	ச 0B9A	ப 0BAA		ொ 0BCA		ச 0BEA	௪ 0BFA
B					ோ 0BCB		ஞ 0BEB	
C		ஜ 0B9C			ௗ 0BCC		ட 0BEC	
D					் 0BCD		எ 0BED	
E	எ 0B8E	ஞ 0B9E	ம 0BAE	ா 0BBE			அ 0BEE	
F	ஏ 0B8F	ட 0B9F	ய 0BAF	ி 0BBF			கூ 0BEF	

11th_Comp Science_UM_Ch 2.indd 26 18-04-2023 12:47:12

ٹیبل 2.6

تیلگو کا یونیکوڈ ٹیبل

	0C0	0C1	0C2	0C3	0C4	0C5	0C6	0C7
0	ఀ 0C00	ఐ 0C10	ఠ 0C20	ర 0C30	ీ 0C40		ౠ 0C60	
1	ఁ 0C01		డ 0C21	ఱ 0C31	ు 0C41		ౡ 0C61	
2	ం 0C02	ఒ 0C12	ఢ 0C22	ల 0C32	ూ 0C42		ౢ 0C62	
3	ః 0C03	ఓ 0C13	ణ 0C23	ళ 0C33	ృ 0C43		ౣ 0C63	
4		ఔ 0C14	త 0C24	ఴ 0C34	ౄ 0C44			
5	అ 0C05	క 0C15	థ 0C25	వ 0C35		ౕ 0C55		
6	ఆ 0C06	ఖ 0C16	ద 0C26	శ 0C36	ె 0C46	ౖ 0C56	౦ 0C66	
7	ఇ 0C07	గ 0C17	ధ 0C27	ష 0C37	ే 0C47		౧ 0C67	
8	ఈ 0C08	ఘ 0C18	న 0C28	స 0C38	ై 0C48	ౘ 0C58	౨ 0C68	౸ 0C78
9	ఉ 0C09	ఙ 0C19		హ 0C39		ౙ 0C59	౩ 0C69	౹ 0C79
A	ఊ 0C0A	చ 0C1A	ప 0C2A		ొ 0C4A	ౚ 0C5A	౪ 0C6A	౺ 0C7A
B	ఋ 0C0B	ఛ 0C1B	ఫ 0C2B		ో 0C4B		౫ 0C6B	౻ 0C7B
C	ఌ 0C0C	జ 0C1C	బ 0C2C		ౌ 0C4C		౬ 0C6C	౼ 0C7C
D		ఝ 0C1D	భ 0C2D	ఽ 0C3D	్ 0C4D		౭ 0C6D	౽ 0C7D
E	ఎ 0C0E	ఞ 0C1E	మ 0C2E	ా 0C3E			౮ 0C6E	౾ 0C7E
F	ఏ 0C0F	ట 0C1F	య 0C2F	ి 0C3F			౯ 0C6F	౿ 0C7F

کنڑا کا یونیکوڈ ٹیبل

	0C8	0C9	0CA	0CB	0CC	0CD	0CE	0CF
0	ಀ 0C80	ಐ 0C90	ಠ 0CA0	ರ 0CB0	ೀ 0CC0		ೠ 0CE0	
1	ಁ 0C81		ಡ 0CA1	ಱ 0CB1	ು 0CC1		ೡ 0CE1	☒ 0CF1
2	ಂ 0C82	ಒ 0C92	ಢ 0CA2	ಲ 0CB2	ೂ 0CC2		ೢ 0CE2	○○ 0CF2
3	ಃ 0C83	ಓ 0C93	ಣ 0CA3	ಳ 0CB3	ೃ 0CC3		ೣ 0CE3	
4		ಔ 0C94	ತ 0CA4		ೄ 0CC4			
5	ಅ 0C85	ಕ 0C95	ಥ 0CA5	ವ 0CB5		ೕ 0CD5		
6	ಆ 0C86	ಖ 0C96	ದ 0CA6	ಶ 0CB6	ೆ 0CC6	ೖ 0CD6	೦ 0CE6	
7	ಇ 0C87	ಗ 0C97	ಧ 0CA7	ಷ 0CB7	ೇ 0CC7		೧ 0CE7	
8	ಈ 0C88	ಘ 0C98	ನ 0CA8	ಸ 0CB8	ೈ 0CC8		೨ 0CE8	
9	ಉ 0C89	ಙ 0C99		ಹ 0CB9			೩ 0CE9	
A	ಊ 0C8A	ಚ 0C9A	ಪ 0CAA		ೊ 0CCA		೪ 0CEA	
B	ಋ 0C8B	ಛ 0C9B	ಫ 0CAB		ೋ 0CCB		೫ 0CEB	
C	ಌ 0C8C	ಜ 0C9C	ಬ 0CAC	಼ 0CBC	ೌ 0CCC		೬ 0CEC	
D		ಝ 0C9D	ಭ 0CAD	ಽ 0CBD	್ 0CCD		೭ 0CED	
E	ಎ 0C8E	ಞ 0C9E	ಮ 0CAE	ಾ 0CBE		ೞ 0CDE	೮ 0CEE	
F	ಏ 0C8F	ಟ 0C9F	ಯ 0CAF	ಿ 0CBF			೯ 0CEF	

11th_Comp Science_UM_Ch 2.indd 27 18-04-2023 12:47:12

ضمیمہ (Appendix)

امریکن اسٹینڈرڈ کوڈ فار انفارمیشن انٹرچینج (ASCII)

(صرف چند مخصوص حروف)

حروف تہجی (Alphabets)

حروف تہجی	اعشاریہ نمبر	بائنری نمبر (8 bit)	آکٹل نمبر	ہیکسا ڈیسیمل نمبر
A	65	01000001	101	41
B	66	01000010	102	42
C	67	01000011	103	43
D	68	01000100	104	44
E	69	01000101	105	45
F	70	01000110	106	46
G	71	01000111	107	47
H	72	01001000	110	48
I	73	01001001	111	49
J	74	01001010	112	4A
K	75	01001011	113	4B
L	76	01001100	114	4C
M	77	01001101	115	4D
N	78	01001110	116	4E
O	79	01001111	117	4F
P	80	01010000	120	50
Q	81	01010001	121	51
R	82	01010010	122	52
S	83	01010011	123	53
T	84	01010100	124	54
U	85	01010101	125	55
V	86	01010110	126	56
W	87	01010111	127	57
X	88	01011000	130	58
Y	89	01011001	131	59
Z	90	01011010	132	5A
a	97	01100001	141	61
b	98	01100010	142	62
c	99	01100011	143	63
d	100	01100100	144	64
e	101	01100101	145	65

11th_Comp Science_UM_Ch 2.indd 28 18-04-2023 12:47:12

f	102	01100110	146	66
g	103	01100111	147	67
h	104	01101000	150	68
i	105	01101001	151	69
j	106	01101010	152	6A
k	107	01101011	153	6B
l	108	01101100	154	6C
m	109	01101101	155	6D
n	110	01101110	156	6E
o	111	01101111	157	6F
p	112	01110000	160	70
q	113	01110001	161	71
r	114	01110010	162	72
s	115	01110011	163	73
t	116	01110100	164	74
u	117	01110101	165	75
v	118	01110110	166	76
w	119	01110111	167	77
x	120	01111000	170	78
y	121	01111001	171	79
z	122	01111010	172	7A

اعداد (Numeral)

حروف تہجی	اعشاریہ نمبر	بائنری نمبر (8 bit)	آکٹل نمبر	ہیکسا ڈیسیمل نمبر
0	48	00110000	60	30
1	49	00110001	61	31
2	50	00110010	62	32
3	51	00110011	63	33
4	52	00110100	64	34
5	53	00110101	65	35
6	54	00110110	66	36
7	55	00110111	67	37
8	56	00111000	70	38
9	57	00111001	71	39

11th_Comp Science_UM_Ch 2.indd 29 18-04-2023 12:47:12

مخصوص حروف (Special Characters)

مخصوص علامات	اعشاریہ نمبر	بائنری نمبر (8 bit)	آکٹل نمبر	ہیکسا ڈیسیمل نمبر
Blank	32	00100000	40	20
!	33	00100001	41	21
"	34	00100010	42	22
#	35	00100011	43	23
$	36	00100100	44	24
%	37	00100101	45	25
&	38	00100110	46	26
'	39	00100111	47	27
(	40	00101000	50	28
)	41	00101001	51	29
*	42	00101010	52	2A
+	43	00101011	53	2B
,	44	00101100	54	2C
-	45	00101101	55	2D
.	46	00101110	56	2E
/	47	00101111	57	2F
:	58	00111010	72	3A
;	59	00111011	73	3B
<	60	00111100	74	3C
=	61	00111101	75	3D
>	62	00111110	76	3E
?	63	00111111	77	3F
@	64	01000000	100	40
[	91	01011011	133	5B
\	92	01011100	134	5C
]	93	01011101	135	5D
^	94	01011110	136	5E
_	95	01011111	137	5F
`	96	01100000	140	60
{	123	01111011	173	7B
|	124	01111100	174	7C
}	125	01111101	175	7D
~	126	01111110	176	7E

11th_Comp Science_UM_Ch 2.indd 30 18-04-2023 12:47:12





تشخیص

سیکشن – A

صحیح جواب کا انتخاب کریں۔

1. وہ کون سا نمبر ہے جو کمپیوٹر کے سی پی یو (CPU) کے ذریعے پروسیس شدہ بٹس کی تعداد ہے؟
   A) بائٹ B) نبل C) ورڈ لینتھ D) بٹ
2. 1 کلو بائٹ کتنے بائٹس پر مشتمل ہوتا ہے؟
   A) 1000 B) 8 C) 4 D) 1024
3. ASCII کی توضیح ہے۔
   A) American School Code for Information Interchange
   B) American Standard Code for Information Interchange
   C) All Standard Code for Information Interchange
   D) American Society Code for Information Interchange
4. 2^50 کہا جاتا ہے۔
   A) کلو (Kilo) B) ٹیرا (Tera) C) پیٹا (Peta) D) زیٹا (Zetta)
5. بائنری کوڈیڈ ڈیسیمل سسٹم میں کتنے حروف سنبھالے جاسکتے ہیں؟
   A) 64 B) 255 C) 256 D) 128
6. 1101_2 کے لیے مساوی ہیکساڈیسیمل ہے؟
   A) F B) E C) D D) B
7. 00100110 کے 1 کی تکمیل کیا ہے؟
   A) 00100110 B) 11011001 C) 11010001 D) 00101001
8. ان میں سے کون سا آکٹل نمبر نہیں ہے؟
   A) 645 B) 234 C) 876 D) 123

سیکشن – B

بہت ہی مختصر جواب دیں۔

1. ڈیٹا کیا ہے؟
2. 1 کا تکمیلی طریقہ کار لکھیں۔
3. $(46)_{10}$ کو بائنری نمبر میں تبدیل کریں۔
4. ہم 1 کی تکمیل $(28)_{10}$ میں نہیں پا سکتے ہیں؟ وجہ بیان کیجیے۔
5. انکوڈنگ سسٹمز کی فہرست بنائیں جو میموری میں حروف کی نمائندگی کرتا ہے۔

سیکشن – C

مختصر جواب دیں۔

1. نمبر سسٹم کا بنیادی عدد (Radix) کیا ہے؟ مثال دیں۔
2. بائنری نمبر سسٹم پر نوٹ لکھیں۔
3. $(150)_{10}$ کو بائنری میں تبدیل کریں پھر اس بائنری نمبر آکٹل میں تبدیل کریں۔
4. ISCII پر مختصر نوٹ لکھیں۔
5. جوڑیں a) $-22_{10}+15_{10}$ b) $20_{10}+25_{10}$

سیکشن – D

تفصیل سے وضاحت کریں۔

1. a) فرکشنل ڈیسیمل کو بائنری میں تبدیل کرنے کا طریقہ لکھیں۔
   b) $(98.46)_{10}$ کو بائنری میں تبدیل کریں۔
2. مندرجہ ذیل اعشاریہ نمبر کے لیے 1 کی تکمیل اور 2 کی تکمیل تلاش کریں۔
   a) -98 b) -135
3. a) جوڑیں $1101010_2+101101_2$ b) گھٹائیں $1101011_2-111010_2$

11th_Comp Science_UM_Ch 2.indd 31 18-04-2023 12:47:12

2.8 تعارف

بولین الجبرا (Boolean Algebra) ایک ریاضیاتی نظم وضبط ہے جو ڈیجیٹل کمپیوٹر میں ڈیجیٹل سرکٹس (Digital Circuits) ڈیزائن کرنے کے لیے استعمال ہوتا ہے۔ یہ ڈیجیٹل سرکٹ کے ان پٹ اور آؤٹ پٹ کے درمیان تعلق کو بیان کرتا ہے۔ بولین الجبرا کا نام ایک انگریز ریاضی دان جارج بولے (George Boole) کے اعزاز میں دیا گیا ہے جس نے الجبرا کے بنیادی اصول تجویز کیے تھے۔

کیا آپ جانتے ہیں؟

جارج بولے (1815 – 1864) ایک پسماندہ خاندان میں پیدا ہوئے۔اور صرف ایک ابتدائی اسکول کی تعلیم حاصل کی۔اس کے باوجود اس نے اپنے آپ کو بغیر کسی مدد کے نوعمری میں انتہائی جدید ریاضی اور مختلف زبانیں سیکھی۔ اس نے سولہ سال کی عمر میں پڑھانا شروع کیا، اور انیس سال کی عمر میں اپنا اسکول شروع کیا۔ اپنی بیسویں دہائی کے وسط تک، اس نے اپنے دور میں ریاضی کے اہم اصولوں میں مہارت حاصل کر لی تھی۔



2.8.1 بائنری قابل قدر مقداریں:

ہر روز ہمیں منطقی فیصلے کرنے ہوتے ہیں:

1. کیا مجھے ہر روز کمپیوٹر سائنس کی کتاب اپنے ساتھ رکھنی چاہیے؟ ہاں / نہیں
2. 10 = 10-8 کیا یہ جواب درست ہے؟ ہاں / نہیں
3. چنئی بھارت کا دارالحکومت ہے؟ ہاں / نہیں
4. میں نے کل کیا کہا؟

اوپر دیئے گئے پہلے تین سوالات کے جواب درست (ہاں) یا false (نہیں) ہو سکتا ہے۔ لیکن چوتھا، ہم صحیح یا false جواب نہیں دے سکتے۔ اس طرح، ایسے جملے جن کے صحیح یا false ہونے کا تعین کیا جا سکتا ہے، انہیں "لوجیکل اسٹیٹمنٹ (Logical Statement) " یا "ٹروتھ فنکشنز (Truth Functions)" کہا جاتا ہے۔ صحیح یا false نتائج کو "Truth Values" کہا جاتا ہے۔ لوجیکل کانسٹینٹ (Logical Constant) 1 اور 0 کے ذریعہ دکھائے گئے true ائی کی اقدار؛ 1 کا مطلب صحیح اور 0 کا مطلب false ہے۔ وہ متغیر جو ان true ائی کی اقدار کو محفوظ کر سکتا ہے اسے "Logical variable" یا "Binary valued variables" یا "Bolean Variables" کہا جاتا ہے کیونکہ یہ True یا False کی دو قدروں میں سے ایک کو محفوظ کر سکتے ہیں۔

2.8.2 لوجیکل آپریشنز:

بولین الجبرا متغیرات اور آپریشنز (فنکشنز) کا استعمال کرتا ہے۔ بنیادی لوجیکل کارروائیاں OR, AND اور NOT ہیں، جو علامتی طور پر ڈاٹ (.)، جمع (+) اور اوور بار (over bar) / سنگل اپوسٹروفی (Single apostrophe) کے ذریعے ظاہر کی جاتی ہیں۔ ان علامتوں کو "Logical Operators" بھی کہا جاتا ہے۔

2.8.3 ٹروتھ ٹیبل:

ٹروتھ ٹیبل (Truth Table) لوجیکل متغیر یا بیانات کی تمام ممکنہ اقدار کے ساتھ true ائی اقدار کے دیے گئے امتزاج کے تمام ممکنہ نتائج کی نمائندگی کرتی ہے۔

2.8.4 AND آپریٹر۔

AND آپریٹر کی تعریف بولین الجبرا میں ڈاٹ (.) آپریٹر کے استعمال سے کی گئی ہے۔ یہ عام الجبرا میں ضرب کی طرح ہے۔ AND آپریٹر دو یا دو سے زیادہ ان پٹ متغیرات کو یکجا کرتا ہے تاکہ آؤٹ پٹ صرف اس صورت میں درست ہو جب تمام ان پٹ درست ہوں۔ 2 - ان پٹ AND آپریٹر کے لیے ٹروتھ ٹیبل مندرجہ ذیل دکھایا گیا ہے:

11th_Comp Science_UM_Ch 2.indd 32 18-04-2023 12:47:12

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

مندرجہ بالا 2- ان پٹ AND آپریشن کو اس طرح ظاہر کیا گیا ہے : Y = A . B

2.8.5 OR آپریٹر۔



جمع کا نشان OR آپریٹر کو ظاہر کرنے کے لیے استعمال کیا جاتا ہے۔ OR آپریٹر دو یا زیادہ ان پٹ متغیرات کو یکجا کرتا ہے تاکہ آؤٹ پٹ درست ہو اگر کم از کم ایک ان پٹ true ہے. 2 - ان پٹ OR آپریٹر کے لیے ٹروتھ ٹیبل مندرجہ ذیل دکھایا گیا ہے:

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

مندرجہ بالا 2 - ان پٹ OR آپریشن کو اس طرح ظاہر کیا گیا ہے: Y = A + B

2.8.6 NOT آپریٹر

NOT آپریٹر کے پاس ایک ان پٹ اور ایک آؤٹ پٹ ہوتا ہے۔ ان پٹ یا تو صحیح یا false ہے، اور آؤٹ پٹ ہمیشہ اس کے برعکس ہوتا ہے، یعنی NOT آپریٹر ان پٹ کو الٹ دیتا ہے۔ NOT آپریٹر کے لیے ٹروتھ ٹیبل جہاں A ان پٹ متغیر ہے اور Y آؤٹ پٹ ہے ذیل میں دکھایا گیا ہے:

A	Y
0	1
1	0

NOT آپریٹر کو الجبری طور پر بولین ایکسپریشن سے ظاہر کیا جاتا ہے: Y = A

مثال:

بولین مساوات پر غور کریں:

D = A + (B . C)

D 1 کے برابر ہے (true) اگر 1A ہے یا اگر 1(B . C) ہے، یعنی B = 0 اور C = 1

بصورت دیگر 0D کے برابر ہے (false)۔

بنیادی لوجک فنکشنز OR، AND اور NOT کو دوسرے لوجک آپریٹرز جیسے NAND اور NOR بنانے کے لیے بھی ملایا جا سکتا ہے

2.8.7 NAND آپریٹر

NAND، NOT اور AND کا مجموعہ ہے۔ NAND ایک AND آپریٹر کے آؤٹ پٹ کو الٹا کر کے تیار کیا جاتا ہے۔ NAND فنکشن کا الجبری اظہار ہے:

$$Y = \overline{A \cdot B}$$

NAND فنکشن ٹروتھ ٹیبل ذیل میں دکھایا گیا ہے:

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

A NAND B = NOT (A AND B)

2.8.8 NOR آپریٹر

NOR، NOT اور OR کا مجموعہ ہے۔ NOR کسی OR آپریٹر کے آؤٹ پٹ کو الٹا کر کے تیار کیا جاتا ہے۔ NOR فنکشن کا الجبری اظہار ہے:

$$Y = \overline{A \cdot B}$$

11th_Comp Science_UM_Ch 2.indd 33 18-04-2023 12:47:12

NOR فنکشن ٹروتھ ٹیبل ذیل میں دکھایا گیا ہے:

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

A NOR B = NOT (A OR B)

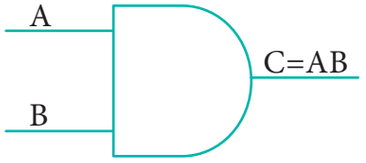
2.9 بیسک لوجک گیٹس:

گیٹ ایک بنیادی الیکٹرانک سرکٹ ہے جو آؤٹ پٹ سگنل تیار کرنے کے لیے ایک یا زیادہ سگنلز پر چلتا ہے۔ تین بنیادی گیٹس (Gates) ہیں یعنی OR، AND اور NOT۔ دیگر لوجک گیٹس جیسے XOR، NOR، NAND اور XNOR اخذ شدہ گیٹس ہیں جو بنیادی گیٹس سے اخذ کیے گئے ہیں۔ NAND اور NOR گیٹس کو یونیورسل گیٹس کہا جاتا ہے، کیونکہ ان کے ذریعے بنیادی لوجک گیٹس اخذ کیے جا سکتے ہیں۔

2.9.1 AND گیٹ

AND گیٹ میں دو یا زیادہ ان پٹ سگنل ہو سکتے ہیں اور آؤٹ پٹ سگنل تیار کر سکتے ہیں۔

آؤٹ پٹ صرف اس صورت میں "True" ہے جب دونوں ان پٹ "True" ہوں، بصورت دیگر، آؤٹ پٹ "False" ہے۔ دوسرے الفاظ میں آؤٹ پٹ 1 ہو گا اگر اور صرف اس صورت میں جب دونوں ان پٹ 1 ہوں؛ دوسری صورت میں آؤٹ پٹ 0 ہے۔ AND گیٹ کے آؤٹ پٹ کو variable کہنے والے C سے ظاہر کیا جاتا ہے، جہاں A اور B دو بولین متغیرات ہیں۔ بولین الجبرا میں، ایک متغیر '0' یا '1' میں سے کسی ایک کو لے سکتا ہے۔ AND گیٹ کی لوجیکل علامت ہے۔



تصویر 2.4 AND گیٹ کی لوجیکل علامت

AND گیٹ کے عمل کو ظاہر کرنے کا ایک طریقہ بولین فنکشن لکھنا ہے۔

C = A AND B

بولین الجبرا میں ضرب کا نشان AND آپریشن کا ہے۔ لہذا، AND گیٹ کا آؤٹ پٹ ہے۔

یا C = A . B

بس C = AB

"A AND B، C کے برابر ہے کے طور پر" پڑھیں۔ چونکہ یہاں دو ان پٹ متغیرات ہیں، ٹرو ٹیبل میں چار اندراجات ہیں، کیونکہ چار ممکنہ ان پٹ ہیں: 00، 01، 10 اور 11۔

مثال کے طور پر اگر دونوں ان پٹ 0 ہیں

C = A . B
= 0 . 0
= 0

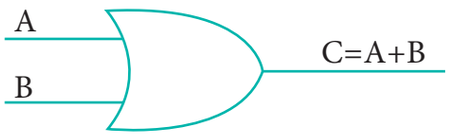
اینڈ گیٹ کے لیے ٹرو تھ ٹیبل ۔

Input		Output
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

ٹیبل 2.7 AND گیٹ کے لیے ٹرو تھ ٹیبل (Truth table)

2.9.2 OR گیٹ

OR گیٹ کو اس کا نام اس کے رویے سے ملتا ہے جیسے لوجیکل انکلوسیو (logical inclusive) "OR"۔ آؤٹ پٹ "True" ہے اگر ان پٹ میں سے ایک یا دونوں "True" ہیں۔ اگر دونوں ان پٹ "False" ہیں تو آؤٹ پٹ "false" ہے۔ دوسرے الفاظ میں آؤٹ پٹ 1 ہو گا اگر اور صرف اس صورت میں جب ایک یا دونوں ان پٹ 1 ہوں؛ دوسری صورت میں، آؤٹ پٹ 0 ہے۔ OR گیٹ کی لوجیکل علامت ہے۔



تصویر 2.5 OR گیٹ کی لوجیکل علامت

11th_Comp Science_UM_Ch 2.indd 34 18-04-2023 12:47:12

OR گیٹ آؤٹ پٹ ہے۔

$$C = A \text{ OR } B$$

ہم OR فنکشن کو ظاہر کرنے کے لیے + نشان استعمال کرتے ہیں۔ لہذا،

$$C = A + B$$

اسے "C برابر A OR B" کے طور پر پڑھیں۔

مثال کے طور پر، اگر دونوں ان پٹ 1 ہیں ،

$$C = A + B = 1 + 1 = 1$$

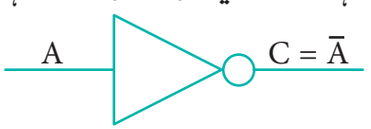
OR گیٹ کے لیے ٹرو تھ ٹیبل ہے۔

Input		Output
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

ٹیبل 2.8 OR گیٹ کے لیے ٹرو تھ ٹیبل

2.9.3 NOT گیٹ

NOT گیٹ، جسے لوجیکل انورٹر (Logical Inverter) کہا جاتا ہے، میں صرف ایک ان پٹ ہوتا ہے۔ یہ لوجیکل حالت کو الٹ دیتا ہے۔ دوسرے لفظوں میں آؤٹ پٹ C ہمیشہ ان پٹ کا تکمیلی ہوتا ہے۔ NOT گیٹ کی لوجیکل علامت ہے۔



تصویر 2.6 NOT گیٹ کی لوجیکل علامت

NOT گیٹ کا بولین فنکشن C = NOT A ہے۔

$$C = \text{NOT } A$$

بولین الجبرا میں، اوورہار (overbar) کا مطلب NOT آپریشن ہے۔ لہذا،

$$C = \overline{A}$$

اسے "C مساوی NOT A" یا "C مساوی A کی تکمیل" کے طور پر پڑھیں۔

اگر A، 0 ہے،

$$C = \overline{0} = 1$$

دوسری طرف، اگر A 1 ہے،

$$C = \overline{1} = 0$$

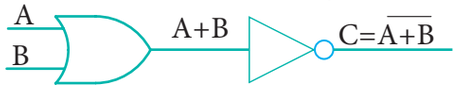
NOT گیٹ کے لیے ٹرو تھ ٹیبل ہے۔

Input	Output
A	C
1	0
0	1

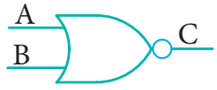
ٹیبل 2.9 NOT گیٹ کے لیے ٹرو تھ ٹیبل

2.9.4 NOR گیٹ

NOR گیٹ سرکٹ (circuit) ایک OR گیٹ ہے جس کے بعد ایک انورٹر (inverter) ہوتا ہے۔ اس کی آؤٹ پٹ "true" ہے اگر دونوں ان پٹ "false" ہیں بصورت دیگر، آؤٹ پٹ "false" ہے۔ دوسرے الفاظ میں، آؤٹ پٹ کے طور پر '1' حاصل کرنے کا واحد طریقہ یہ ہے کہ دونوں ان پٹ '0' ہوں۔ بصورت دیگر آؤٹ پٹ 0 ہے۔ NOR گیٹس کا لوجک سرکٹ (logic circuit) ہے۔



تصویر 2.7 NOR گیٹ کا لوجک سرکٹ



تصویر 2.8 NOR گیٹ کی لوجیکل علامت

NOR گیٹ کی آؤٹ پٹ ہے۔

$$C = (\overline{A + B})$$

اسے "C مساوی A NOT کے C A OR B کے مساوی ہے" یا "C مساوی A OR B کی تکمیل" کے طور پر پڑھیں۔

مثال کے طور پر اگر دونوں ان پٹ 0 ہیں،

$$C = (\overline{0 + 0}) = \overline{0} = 1$$

NOR گیٹ کے لیے ٹرو تھ ٹیبل ہے۔

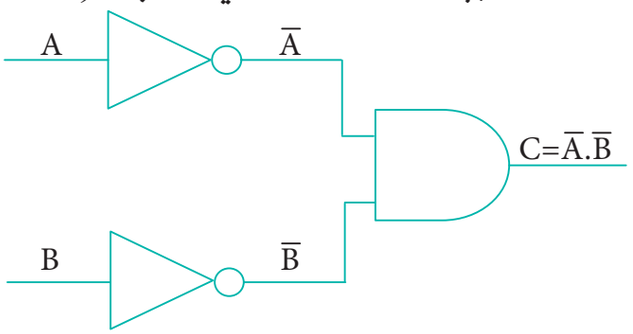
Input		Output
A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

ٹیبل 2.10 NOR گیٹ کے لیے ٹرو تھ ٹیبل

11th_Comp Science_UM_Ch 2.indd 35 18-04-2023 12:47:13

2.9.5 بلبلڈ AND گیٹ

بلبلڈ (Bubbled) AND گیٹ کا لوجک سرکٹ

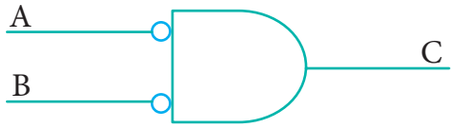


تصویر 2.9 بلبلڈ AND گیٹ کا لوجک سرکٹ

مندرجہ بالا سرکٹ میں، AND گیٹ کی ان پٹ لائنوں پر انورٹرز (Invertors) آؤٹ پٹ کو اس طرح دیتا ہے

$$C = (\overline{A} \cdot \overline{B})$$

اس سرکٹ کو ان پٹ پر بلبس (bubbles) کے طور پر دوبارہ تیار کیا جا سکتا ہے، جہاں بلبس الٹ کی نمائندگی کرتے ہیں۔



تصویر 2.10 بلبلڈ AND گیٹ کا لوجک علامت

ہم اسے بلبلڈ AND گیٹ کہتے ہیں۔ آئیے تمام ان پٹ امکانات کے لیے اس منطقی سرکٹ کا تجزیہ کریں۔

If A = 0 and B = 0 $C=(\overline{0}.\overline{0}) = 1.1 = 1$

If A = 0 and B = 1 $C=(\overline{0}.\overline{1}) = 1.0 = 0$

If A = 1 and B = 1 $C=(\overline{1}.\overline{0}) = 0.1 = 0$

If A = 1 and B = 1 $C=(\overline{1}.\overline{1}) = 0.0 = 0$

یہاں ٹرو تھ ٹیبل ہے

Input		Output
A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

آپ دیکھ سکتے ہیں کہ ایک بلبلڈ AND گیٹ NOR گیٹ کی طرح آؤٹ پٹ پیدا کرتا ہے۔ تو، آپ ہر NOR گیٹ کو بلبلڈ AND گیٹ سے بدل سکتے ہیں۔ دوسرے الفاظ میں سرکٹس قابل تبادلہ ہیں۔

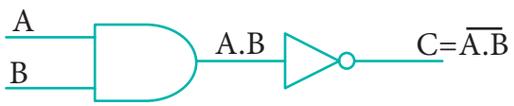
اس لیے

$$\overline{(A + B)} = \overline{A} \cdot \overline{B}$$

جو ڈی مورگن (De Morgans) کا پہلا نظریہ قائم کرتا ہے۔

2.9.6 NAND گیٹ

NAND گیٹ ایک AND گیٹ چلاتا ہے جس کے بعد NOT گیٹ ہوتا ہے۔ یہ لوجیکل آپریشن "AND" کے انداز میں کام کرتا ہے جس کے بعد الٹا ہوتا ہے۔ آؤٹ پٹ "false" ہے اگر دونوں ان پٹ "true" ہیں، بصورت دیگر، آؤٹ پٹ "true" ہے۔ دوسرے الفاظ میں NAND گیٹ کا آؤٹ پٹ 0 ہے اگر اور صرف اس صورت میں جب دونوں ان پٹ 1 ہوں، ورنہ آؤٹ پٹ 1 ہے۔ NAND گیٹ کا لوجک سرکٹ ہے۔



تصویر 2.11 NAND گیٹ کا لوجک سرکٹ

NAND گیٹ کی لوجیکل علامت ہے۔



تصویر 2.12 NAND گیٹ کا لوجیکل علامت

NAND گیٹ کا آؤٹ پٹ ہے

$$C = \overline{(A \cdot B)}$$

اسے "C" یا "A OR B کے NOT کے مساوی ہے" C مساوی A OR B کی تکمیل" کے طور پر پڑھیں۔

مثال کے طور پر اگر دونوں ان پٹ1 ہیں،

11th_Comp Science_UM_Ch 2.indd 36 18-04-2023 12:47:13

$$C = (\overline{1 . 1}) = \overline{1} = 0$$

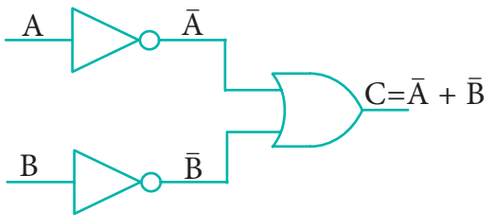
NAND گیٹ کے لیے ٹروتھ ٹیبل ہے۔

Input		Output
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

ٹیبل 2.11 NAND گیٹ کے لیے ٹروتھ ٹیبل

2.9.7 بلڈ OR گیٹ

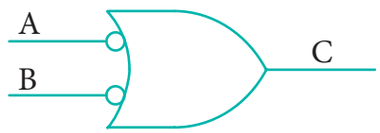
بلڈ OR گیٹ کا لوجک سرکٹ



تصویر 2.13 بلڈ OR گیٹ کا لوجک سرکٹ

اس سرکٹ کا آؤٹ پٹ $C = \overline{A} + \overline{B}$ لکھا جا سکتا ہے۔

مندرجہ بالا سرکٹ کے طور پر دوبارہ تیار کیا جا سکتا ہے ان پٹ پر.بلس، جہاں .بلس الٹ کی نمائندگی کرتے ہیں۔



تصویر 2.14 بلڈ OR گیٹ کا لوجک علامت

ہم اسے بلڈ OR گیٹ کہتے ہیں۔ بلڈ OR کے لیے ٹروتھ ٹیبل یہ ہے۔

Input		Output
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

ٹیبل 2.12 بلڈ OR گیٹ کے لیے ٹروتھ ٹیبل

اگر ہم بلڈ OR گیٹ کی ٹروتھ ٹیبل کا NAND گیٹ سے موازنہ کریں تو وہ ایک جیسے ہیں۔ لہذا سرکٹس قابل تبادلہ ہیں۔

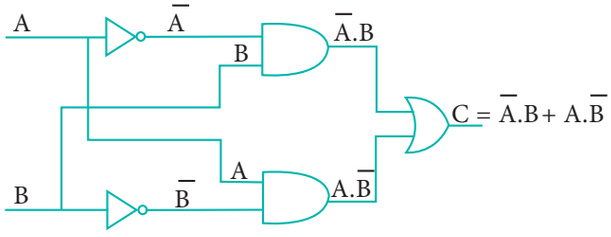
لہذا،

$$(\overline{A . B}) = \overline{A} + \overline{B}$$

جو ڈی مورگن کا دوسرا نظریہ قائم کرتا ہے۔

2.9.8 XOR گیٹ

XOR (خصوصی - OR) گیٹ اسی طرح کام کرتا ہے جیسے لوجیکل "either / or"۔ آؤٹ پٹ "true" ہے اگر ان پٹ میں سے کوئی ایک، لیکن دونوں نہیں، ان پٹس "true" ہیں۔ آؤٹ پٹ "false" ہے اگر دونوں ان پٹ "false" ہیں یا اگر دونوں ان پٹ "true" ہیں۔ اس سرکٹ کو دیکھنے کا ایک اور طریقہ یہ ہے کہ یہ مشاہدہ کیا جائے کہ اگر ان پٹ مختلف ہیں تو آؤٹ پٹ 1 ہے، لیکن اگر ان پٹ ایک جیسے ہیں تو 0۔ XOR گیٹ کا لوجک سرکٹ ہے۔



تصویر 2.15 XOR گیٹ کا لوجک سرکٹ

XOR گیٹ کا آؤٹ پٹ XOR گیٹ کے لیےٹروتھ ٹیبل ہے۔

Input		Output
A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

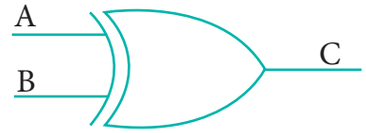
ٹیبل 2.13 XOR گیٹ کے لیے ٹروتھ ٹیبل

بولین الجبرا میں۔ خصوصی - OR آپریٹر $\oplus$ یا "گھیرے ہوئے پلس (encircled plus)"۔

لہذا $C = A \oplus B$

XOR گیٹ کی لوجیکل علامت ہے۔

11th_Comp Science_UM_Ch 2.indd 37 18-04-2023 12:47:13



تصویر 2.16 XOR گیٹ کی لوجیکل علامت

2.9.9 XNOR گیٹ

XNOR (خصوصی - NOR) گیٹ ایک مجموعہ XOR گیٹ ہے جس کے بعد ایک انورٹر ہوتا ہے۔ اگر ان پٹ ایک جیسے ہوں تو اس کا آؤٹ پٹ "true" ہے، اور اگر ان پٹ مختلف ہیں تو "false" ہے۔ آسان الفاظ میں، اگر ان پٹ ایک جیسے ہیں تو آؤٹ پٹ 1 ہے، ورنہ آؤٹ پٹ 0 ہے۔ XNOR گیٹ کا لوجک سرکٹ ہے۔



تصویر 2.17 XNOR گیٹ کا لوجک سرکٹ

XNOR کا اوٹ پٹ XOR کا NOT ہے۔

$$C = \overline{A \oplus B}$$

$$= \overline{A} \cdot B + A \cdot \overline{B}$$

$$= AB + \overline{A}\ \overline{B}$$

(ڈی مورگن کی تھیوریم کا استعمال کرتے ہوئے)

بولین الجبرا میں، $\odot$ یا "included dot" کا مطلب XNOR ہے۔

لہذا، $C = A \odot B$

لوجیکل علامت ہے۔



تصویر 2.18 XNOR گیٹ کا لوجک علامت

XNOR گیٹ کے لیے ٹروتھ ٹیبل ہے۔

Input		Output
A	B	C
0	0	1
0	1	0
1	0	0
1	1	1

ٹیبل 2.14 XNOR گیٹ کے لیے ٹروتھ ٹیبل

لوجک گیٹس کے امتزاج کا استعمال کرتے ہوئے، پیچیدہ آپریشن کیے جا سکتے ہیں۔ نظریہ میں، گیٹس کی تعداد کی کوئی حد نہیں ہے جو ایک ہی ڈیوائس میں ایک ساتھ ترتیب دیے جا سکتے ہیں۔ لیکن عملی طور پر، گیٹس کی تعداد کی ایک حد ہوتی ہے جو ایک دی گئی جسمانی جگہ میں پیک کیے جا سکتے ہیں۔ لوجک گیٹس کی صفیں ڈیجیٹل انٹیگریٹڈ سرکٹس (digital integrated circuits) میں پائی جاتی ہیں۔

بولین الجبری کی تھیوریم

Identity
$A + 0 = A$
$A \cdot 1 = A$

Complement
$A + \overline{A} = 1$
$A \cdot \overline{A} = 0$

Commutative
$A + B = B + A$
$A \cdot B = B \cdot A$

Associative
$A + (B + C) = (A + B) + C$
$A \cdot (B \cdot C) = (A \cdot B) \cdot C$

Distributive
$A \cdot (B + C) = A \cdot B + A \cdot C$
$A + (B \cdot C) = (A + B) \cdot (A + C)$

Null Element
$A + 1 = 1$
$A \cdot 0 = 0$

Involution
$\overline{(\overline{A})} = A$

Idempotence
$A + A = A$
$A \cdot A = A$

Absorption
$A + (A \cdot B) = A$
$A \cdot (A + B) = A$

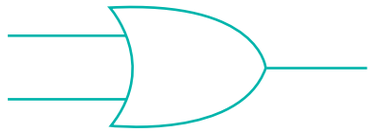
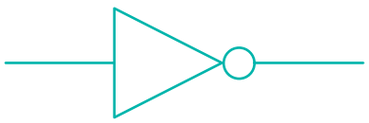
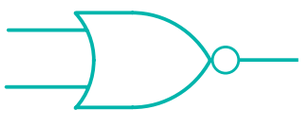
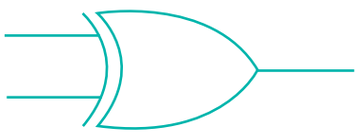
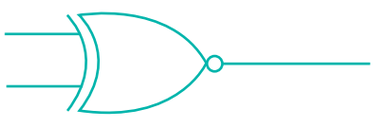
3rd Distributive
$A + \overline{A} \cdot B = A + B$

De Morgan's
$\overline{A + B} = \overline{A} \cdot \overline{B}$
$\overline{(A \cdot B)} = \overline{A} + \overline{B}$

11th_Comp Science_UM_Ch 2.indd 38 18-04-2023 12:47:13

ٹیبل 2.15

لاجک گیٹس اور ان سے متعلقہ ٹروتھ ٹیبلس

Logical Gates	Symbol	Truth Table
AND		A | B | AB 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1
OR		A | B | A + B 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1
NOT		A | $\overline{A}$ 0 | 1 1 | 0
NAND		A | B | $\overline{A\,B}$ 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0
NOR		A | B | $\overline{A + B}$ 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 0
XOR		A | B | $A \oplus B$ 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0
XNOR		A | B | $\overline{A \oplus B}$ 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1

11th_Comp Science_UM_Ch 2.indd 39 18-04-2023 12:47:13

تشخیص

سیکشن – A

صحیح جواب کا انتخاب کریں۔

1. کون سا بنیادی الیکٹرانک سرکٹ ہے جو ایک یا زیادہ سگنلز پر چلتا ہے؟

(A) بولین الجبرا (B) گیٹ

(C) فنڈامینٹل گیٹس (D) ڈرائیوڈ گیٹس

2. کس گیٹ کو لوجیکل انورٹر کہا جاتا ہے؟

AND (A) OR (B)

NOT (C) XNOR (D)

3. A + A = ?

A (A) O (B)

1 (C) A (D)

4. NOR کا مجموعہ کیا ہے؟

NOT(OR) (A) NOT(AND) (B)

NOT(NOT) (C) NOT(NOR) (D)

5. NAND کو .....گیٹ کہا جاتا ہے۔

(A) فنڈامینٹل گیٹ (B) ڈرائیوڈ گیٹ

(C) لوجیکل گیٹ (D) یونیورسل گیٹ

سیکشن – B

بہت مختصر جوابات دیں۔

1. بولین الجبرا کیا ہے؟
2. NAND گیٹ پر ایک مختصر نوٹ لکھیں۔
3. XOR گیٹ کے لیے ٹروتھ ٹیبل بنائیں۔
4. اسوسی ایٹیو قوانین لکھیں؟
5. ڈرائیوڈ گیٹس کیا ہیں؟

سیکشن – C

مختصر جوابات دیں۔

1. فنڈامینٹل گیٹس کی ٹروتھ ٹیبل کے بارے میں لکھیے۔
2. XNOR گیٹ پر ایک مختصر نوٹ لکھیں۔
3. وجہ بتائیں کہ NAND اور NOR کو یونیورسل گیٹس کیوں کہا جاتا ہے؟
4. XOR گیٹ کا ٹروتھ ٹیبل بتائیں۔
5. ڈی مورگن کا قانون لکھیں۔

سیکشن – D

تفصیل سے بیان کریں۔

1. اکپریشن اور ٹروتھ ٹیبل کے ساتھ فنڈامینٹل گیٹس کی وضاحت کریں۔
2. NAND اور NOR گیٹ کا استعمال کرتے ہوئے AND اور OR کو کیسے محسوس کیا جا سکتا ہے۔
3. اکپریشن اور ٹروتھ ٹیبل کے ساتھ ڈرائیوڈ گیٹس کی وضاحت کریں۔

11th_Comp Science_UM_Ch 2.indd 40 18-04-2023 12:47:13

باب 3

کمپیوٹر تنظیم

سیکھنے کے مقاصد

- کمپیوٹر کے اجزاء اور ان کے باہمی روابط کی تنظیم کو جاننا۔
- پروسیسرس (Processors) اور ان کی خصوصیات کو جاننا۔
- میموری آلات کی اہمیت اور کمپیوٹر میں ان کے کردار کو جاننا۔
- ROM، RAM کو دریافت کرنے اور ان میں سے ہر ایک کو الگ کرنے کے بارے میں جاننا۔
- کیش میموری (Cache Memory) کے بارے میں جاننا اور یہ کمپیوٹر کی کارکردگی کو کس طرح بہتر بناتا ہے۔
- ثانوی آلات اور ان کے استعمال کو جاننا۔
- پورٹس اور انٹرفیس کے بارے میں جاننا تاکہ بیرونی آلات کو منسلک کیا جا سکے۔

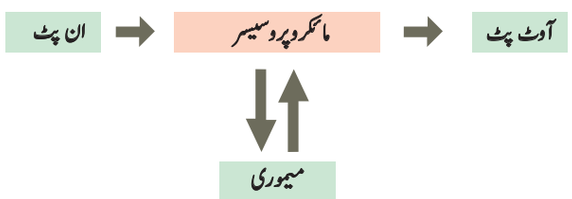
3.1 تعارف

کمپیوٹر تنظیم کمپیوٹر سسٹم کے ہارڈویئر اجزاء سے مل کر اپنے کام کو انجام دیتی ہے۔ اس میں ان پٹ (Input)/ آؤٹ پٹ (Output) ڈیوائسز، سینٹرل پروسیسنگ یونٹ (CPU)، اسٹوریج ڈیوائسز اور بنیادی میموری شامل ہیں۔ اس کا تعلق اس بات سے ہے کہ کمپیوٹر ہارڈویئر کے مختلف اجزاء کیسے کام کرتے ہیں۔ یہ اس بات سے بھی متعلق ہے کہ وہ کس طرح ایک آرکیٹیکچرل تصریح کو نافذ کرنے کے لیے آپس میں جڑے ہوئے ہیں۔ کمپیوٹر تنظیم کی اصطلاح کمپیوٹر فن تعمیر کی اصطلاح سے ملتی جلتی ہے۔ لیکن، کمپیوٹر فن تعمیر ،کمپیوٹر ڈیزائن کرنے میں انجینئرنگ کے تحفظات سے متعلق ہے۔ دوسری طرف، کمپیوٹر آرگنائزیشن ہارڈ ویئر کے اجزاء سے متعلق ہے جو پروگرامر کے لیے واضح ہے۔

3.2 مائیکرو پروسیسرس (Microprocessors) کی بنیادی باتیں

سی پی یو (CPU) کمپیوٹر کا اہم جزء ہے، جو تمام کام انجام دیتا ہے۔ اس کا احساس مائکرو پروسیسر سے ہوتا ہے جو ایک انٹیگریٹڈ سرکٹ (Integrated Circuit) ہے۔ مائیکرو پروسیسر پہلی بار 1970 کی دہائی کے اوائل میں متعارف کرایا گیا تھا۔ پہلا عمومی مقصد مائکرو پروسیسر 4004 جسے Intel Inc نے تیار کیا تھا۔

مائکرو پروسیسر ایک قابل پروگرام کثیر مقصدی سلیکان چپ (Silicon Chip) ہے۔ یہ گھڑی کی نبضوں سے چلتی ہے۔ یہ بائنری ڈیٹا کے طور پر ان پٹ کو قبول کرتا ہے اور پروسیسنگ کے بعد یہ میموری میں محفوظ ہدایات کے مطابق آؤٹ پٹ ڈیٹا فراہم کرتا ہے۔ مائیکرو پروسیسر پر مبنی نظام کا بلاک ڈایاگرام تصویر 3.1 میں دکھایا گیا ہے۔



تصویر 3.1 مائکروپروسیسر - بیسڈ سسٹم

مائکرو پروسیسر 3 اہم اکائیوں پر مشتمل ہے۔ وہ درج ذیل ہیں:

- **ریاضی اور منطقی یونٹ (ALU):** کمپیوٹر ہدایات پر مبنی ریاضی اور منطقی ہدایات کو انجام دینے کے لیے۔

11th_Comp Science_UM_Ch 3.indd 41 18-04-2023 13:07:35

- **کنٹرول یونٹ (CU):** سگنلز کے ذریعے کمپیوٹر کے مجموعی آپریشنز کو کنٹرول کرنا۔
- **رجسٹرس (Registers) (اندرونی میموری):** ان کا استعمال پروسیسر کے عمل کے لیے ہدایات اور ڈیٹا کو محفوظ رکھنے کے لیے کیا جاتا ہے۔

مائکرو پروسیسرس کی خصوصیات

مائیکرو پروسیسر کی کارکردگی درج ذیل خصوصیات پر منحصر ہے:

a) گھڑی کی رفتار (Clock Speed)

b) انسٹرکشن سیٹ (Instruction set)

c) لفظ کا سائز (Word Size)

a) گھڑی کی رفتار

ہر مائکرو پروسیسر کی ایک اندرونی گھڑی ہوتی ہے جو اس رفتار کو کنٹرول کرتی ہے جس پر وہ ہدایات کے مطابق عمل کرتا ہے۔ مائکرو پروسیسر جس رفتار سے ہدایات پر عمل کرتا ہے اسے گھڑی کی رفتار کہا جاتا ہے۔ گھڑی کی رفتار میگا ہرٹز MHz (Mega Hertz) یا گیگا ہرٹز GHz (Giga Hertz) میں ناپا جاتا ہے۔

b) انسٹرکشن سیٹ

ایک کمانڈ ہے جو کمپیوٹر کو ڈیٹا پر آپریشن کرنے کے لیے دی جاتی ہے، اسے ہدایت (Instruction) کہتے ہیں۔ مشین کی سطح کی ہدایت کا بنیادی سیٹ جسے ایک مائیکرو پروسیسر کو عمل میں لانے کے لیے ڈیزائن کیا گیا ہے اسے انسٹرکشن سیٹ (Instruction set) کہا جاتا ہے۔ یہ انسٹرکشن سیٹ مندرجہ ذیل آپریشن کے اقسام کی کارروائیوں کو انجام دیتا ہے:

- مواد کی منتقلی (Data Transfer)
- ریاضی کے آپریشنز (Arithmetic Operations)
- منطقی آپریشنز (Logical Operations)
- کنٹرول بہاؤ (Control Flow)
- ان پٹ / آؤٹ پٹ (Input / Output)

c) لفظ کا سائز

- بٹس کی تعداد جو ایک پروسیسر کے ذریعہ ایک ہدایت میں کارروائی کی جا سکتی ہے اسے اس کے لفظ کا سائز کہا جاتا ہے۔

لفظ کا سائز RAM کی مقدار کا تعین کرتا ہے جس کے ذریعہ مائکرو پروسیسر تک رسائی حاصل کی جاسکتی ہے۔

رفتار کی پیمائش

کیا آپ جانتے ہیں؟

ہرٹز (HERTZ) - مختصر Hz پیمائش کی معیاری اکائی ہے جو تعداد کی پیمائش کے لیے استعمال ہوتی ہے۔ چونکہ تعداد کو سائیکلس میں فی سیکنڈ ناپا جاتا ہے، اس لیے ایک ہرٹز ایک سائیکل فی سیکنڈ (Cycle per second) کے برابر ہے۔

ہرٹز عام طور پر لہروں کی تعداد کی پیمائش کرنے کے لیے استعمال ہوتا ہے، جیسے آواز کی لہریں، روشنی کی لہریں، اور ریڈیو کی لہریں۔ مثال کے طور پر، اوسطاً انسانی کان 20 سے 20،000 Hz ہرٹز کے درمیان آواز کی لہروں کا پتہ لگا سکتا ہے۔ 20 ہرٹز کے قریب صوتی لہروں کی پچ (Pitch) کم ہوتی ہے اور انہیں "باس (Bass)" تعداد کہا جاتا ہے۔ 5,000 ہرٹز سے اوپر کی آواز کی لہروں کی اونچی پچ ہوتی ہے اور انہیں "ٹریبل (Treble)" تعداد کہا جاتا ہے۔

جبکہ ہرٹز کو لہر کی تعداد کی پیمائش کے لیے استعمال کیا جا سکتا ہے، یہ کمپیوٹر پروسیسرس کی رفتار کی پیمائش کے لیے بھی استعمال ہوتا ہے۔ مثال کے طور پر، ہر سی پی یو (CPU) کو مخصوص گھڑی کی رفتار سے درجہ بندی کیا جاتا ہے۔ یہ نمبر بتاتا ہے کہ پروسیسر ہر سیکنڈ میں کتنے انسٹرکشن سائیکل انجام دے سکتا ہے۔ چونکہ جدید پروسیسر فی سیکنڈ میں لاکھوں یا اربوں ہدایات بھی انجام دے سکتے ہیں، اس لیے گھڑی کی رفتار کو عام طور پر میگاہرٹز یا گیگاہرٹز میں ناپا جاتا ہے۔

3.3 CPU اور میموری کے درمیان ڈیٹا مواصلات

سینٹرل پروسیسنگ یونٹ (CPU) کے پاس میموری ڈیٹا رجسٹر (MDR) اور میموری ایڈریس رجسٹر (MAR) ہے۔ میموری ڈیٹا رجسٹر (MDR) اس ڈیٹا کو رکھتا ہے جو میموری اور CPU کے درمیان منتقل ہوتا ہے۔ پروگرام کاؤنٹر (PC) میں ایک خصوصی رجسٹر ہے جو ہمیشہ اگلی ہدایات کو انجام دینے کا پتہ رکھتا ہے۔ CPU

11th_Comp Science_UM_Ch 3.indd 42 18-04-2023 13:07:35

کا ریاضی اور منطقی یونٹ میموری ایڈریس رجسٹر میں میموری کا پتہ رکھتا ہے تاکہ اسے دوبارہ حاصل کیا جائے۔

بس (BUS) تاروں کا ایک مجموعہ ہے جو کمپیوٹر کے اندرونی اجزاء کے درمیان رابطے کے لیے استعمال ہوتا ہے۔

RAM میں موجود لفظ کا سائز (بٹس کی تعداد) میموری ڈیٹا رجسٹر (MDR) جیسا ہے۔ اگر پروسیسر ایک 8 بٹ پروسیسر ہے جیسا کہ Intel 8085، اس کا MDR اور RAM میں لفظ دونوں 8 بٹس ہوتے ہیں۔

ریڈ (تقریری) آپریشن ، ڈیٹا (بٹس) کو لفظ سے میموری ڈیٹا رجسٹر میں منتقل کرتا ہے۔ رائٹ (تحریری) آپریشن میموری ڈیٹا رجسٹر سے ڈیٹا (بٹس) کو لفظ میں منتقل کرتا ہے۔

کیا آپ جانتے ہیں؟

اگر تار کے ایک سرے پر 5 V لگائی جاتی ہے تو دوسرے سرے پر بھی V حاصل ہوسکتا ہے۔ اسی طرح، بسیس (Buses) تاریں ہیں، اور بائنری ڈیٹا وولٹیجز (Voltages) ہیں (5V بطور 1 اور 0V بطور 0)، اور یہ بسیس ڈیٹا کو وولٹیج کے طور پر ایک سرے سے دوسرے سرے تک منتقل کر سکتی ہیں۔

3.4 مائیکرو پروسیسر کے اقسام

مائیکرو پروسیسر کو درج ذیل معیارات کی بنیاد پر درجہ بندی کیا جا سکتا ہے۔

- ڈیٹا کی چوڑائی جس پر کارروائی کی جا سکتی ہے۔
- انسٹرکشن سیٹ

3.4.1 ڈیٹا کی چوڑائی کی بنیاد پر مائیکرو پروسیسرس (Mircroprocessors) کی درجہ بندی

مائکرو پروسیسرس (Mircroprocessors) ہدایت کی کارروائی ڈیٹا کی چوڑائی پر منحصر ہے۔ مائیکرو پروسیسرز کو ذیل کے مطابق درجہ بندی کی گئی ہے:

- 8 بٹ مائکرو پروسیسر
- 16 بٹ مائکرو پروسیسر
- 32 بٹ مائکرو پروسیسر
- 64 بٹ مائکرو پروسیسر

3.4.2 انسٹرکشن سیٹ (Instruction Set) کی بنیاد پر مائیکرو پروسیسرس کی درجہ بندی

مائیکرو پروسیسرس کی درجہ بندی کرتے وقت انسٹرکشن سیٹ کے سائز پر غور کرنا اہم ہے۔انسٹرکشن سیٹس کی بنیاد پر مائکرو پروسیسرس دو قسمیں ہیں۔

- کم کردہ انسٹرکشن سیٹ کمپیوٹرز (RISC) (Reduced Instruction set Computers)
- کمپلیکس انسٹرکشن سیٹ کمپیوٹرز (CISC) (Complex Instruction set Computers)

RISC پروسیسرس کی مثالیں Pentium IV، AMD K6، Intel P6 اور K7 ہیں۔

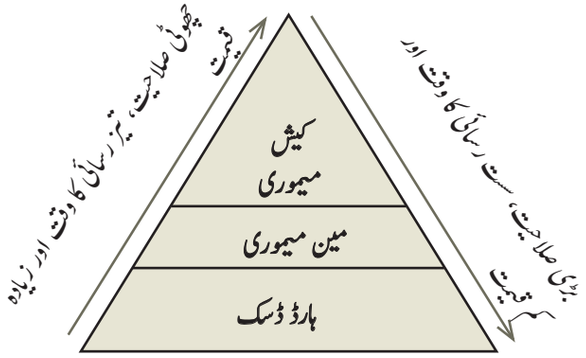
CISC پروسیسرز کی مثالیں، Intel 386 & 486 Pentium، Pentium II اور III، اور Motorola 68000 ہیں۔

3.5 میموری ڈیوائسس (Memory Devices)

میموری بالکل انسانی دماغ کی طرح ہوتی ہے۔ یہ ڈیٹا اور ہدایات کو ذخیرہ کرنے کے لیے استعمال کیاجاتا ہے۔ کمپیوٹر میموری کمپیوٹر میں ذخیرہ کرنے کی جگہ ہے، جہاں ڈیٹا اور ہدایات محفوظ کی جاتی ہیں۔ میموری تک رسائی (پڑھنے یا لکھنے) کے دو طریقے ہیں۔ وہ ترتیب وار رسائی (Sequential Access) اور بے ترتیب رسائی (Random Access) ہیں۔ ترتیب وار رسائی میں شروع سے آخر تک میموری کی رسائی منظم طریقے سے ہوتی

11th_Comp Science_UM_Ch 3.indd 43 18-04-2023 13:07:35

ہے۔ لیکن، بے ترتیب رسائی میں، میموری کے کسی بھی بائٹ تک پچھلے بائٹس کے ذریعے پہونچنے کے لیے بغیر رہنمائی کے براہ راست رسائی حاصل کی جا سکتی ہے۔ مختلف میموری ڈیوائسس کو صلاحیت، رفتار اور قیمت کے مطابق ترتیب دیا گیا ہے جیسا کہ تصویر 3.6 میں دکھایا گیا ہے۔



تصویر 3.6 میموری درجہ بندی (Hierarchy)

3.5.1 ریڈم ایکسیس میموری (RAM)

مین میموری کو دوسری صورت میں رینڈم ایکسیس میموری (Random Access Memory) کہا جاتا ہے۔ یہ انٹیگریٹڈ سرکٹس (ICs) کی شکل میں کمپیوٹرز میں دستیاب ہے۔ یہ کمپیوٹر میں وہ جگہ ہے جہاں آپریٹنگ سسٹم، ایپلیکیشن پروگرامز اور موجودہ استعمال میں موجود ڈیٹا کو عارضی طور پر رکھا جاتا ہے، تاکہ کمپیوٹر کے پروسیسر کے ذریعے ان تک رسائی حاصل کی جا سکے۔ معلومات کی سب سے چھوٹی اکائی جو میموری میں محفوظ کی جا سکتی ہے اسے بٹ (Bit) کہتے ہیں۔ میموری تک 8 بٹس کے مجموعہ سے رسائی حاصل کی جاسکتی ہے جسے بائٹ (Byte) کہتے ہیں۔

RAM ایک غیر مستحکم میموری (Volatile) ہے، جس کا مطلب ہے کہ اس میں ذخیرہ شدہ معلومات مستقل نہیں ہوتی ہیں۔ جیسے ہی پاور آف ہو جاتی ہے، جو بھی ڈیٹا RAM میں رہتا ہے وہ ضائع ہو جاتا ہے۔ یہ پڑھنے اور لکھنے دونوں کاموں کی اجازت دیتا ہے۔

3.5.2 RAM کی اقسام

RAM کی دو بنیادی اقسام ہیں۔

- متحرک ریم Dynamic RAM (DRAM)
- جامد ریم Static RAM (SRAM)

ڈیٹا کو محفوظ رکھنے کے لیے جو ٹیکنالوجی استعمال کی جاتی ہے، وہی ان دونوں اقسام میں فرق ہے۔ متحرک RAM ایک عام قسم کے ہونے کی وجہ سے اسے کثرت سے تازہ کرنے کی ضرورت ہے۔ جامد RAM کو بہت کم تازہ کرنے کی ضرورت ہے، جو اسے تیز تر بناتی ہے۔ لہذا، جامد RAM متحرک RAM سے زیادہ مہنگی ہے۔

3.5.3 ریڈ اونلی میموری (ROM)

ریڈ اونلی میموری (Read only Memory) سے مراد کمپیوٹر میں خاص میموری ہے جسے بناتے وقت پہلے سے ریکارڈ شدہ ڈیٹا ہوتا ہے جس میں ترمیم نہیں کی جاسکتی ہے۔ ذخیرہ شدہ پروگرام جو کمپیوٹر کو شروع کرتے ہیں اور تشخیص کرتے ہیں ROMs میں دستیاب ہیں۔ ROM اہم پروگراموں کو محفوظ کرتا ہے جیسے کہ وہ پروگرام جو کمپیوٹر کو بوٹ کرتا ہے۔ ایک بار جب ڈیٹا ROM چپ پر لکھا جاتا ہے، تو اسے تبدیل یا ہٹایا نہیں جا سکتا اور اسے صرف پڑھا جا سکتا ہے۔ ROM کمپیوٹر کے بند ہونے پر بھی اپنے مواد کو برقرار رکھتا ہے۔ لہذا، ROM کو غیر مستحکم میموری (Non Volatile) کہا جاتا ہے۔

3.5.3.1 پروگرامیبل ریڈ اونلی میموری (PROM)

پروگرامیبل ریڈ اونلی میموری (Programmable Read Only Memory) بھی ایک غیر مستحکم میموری ہے جس پر ڈیٹا صرف ایک بار لکھا جا سکتا ہے۔ ایک بار جب ایک پروگرام PROM پر لکھا جاتا ہے، تو یہ ہمیشہ کے لیے وہاں رہتا ہے۔ مین میموری کے برعکس، PROMs اپنے مواد کو کمپیوٹر بند ہونے کے بعد بھی برقرار رکھتے ہیں ۔

ROM، PROM سے مختلف ہے۔ PROM کو خالی میموری کے طور پر تیار کیا جاتا ہے، جبکہ ROM کو بناوٹ کے دوران ہی پروگرام کیا جاتا ہے۔ PROM پروگرامر یا PROM برنر کو PROM چپ پر ڈیٹا لکھنے کے لیے استعمال کیا جاتا ہے۔ PROM کو پروگرام کرنے کے عمل کو برننگ PROM کہا جاتا ہے۔

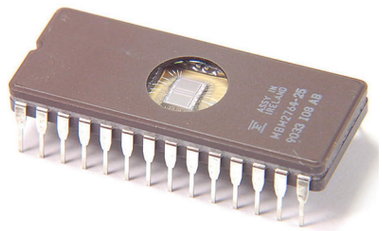
3.5.3.2 ایریزیبل پروگرامیبل ریڈ اونلی میموری (EPROM)

ایریزیبل پروگرامیبل ریڈ اونلی میموری (Erasable Programmable Read Only Memory) کی ایک خاص قسم ہے۔ایک PROM کے طور پر کام کرتا ہے، لیکن مواد کو بالائے بنفشی شعاعوں (Ultraviolet Rays) کا استعمال کرتے ہوئے مٹایا

11th_Comp Science_UM_Ch 3.indd 44 18-04-2023 13:07:35

جا سکتا ہے۔ EPROM اپنے مواد کو اس وقت تک برقرار رکھتا ہے جب تک کہ یہ بنفشی شعاعیں (Ultraviolet Light) کھل کر سامنے نہ آجائے۔ بنفشی شعاعیں اس کے مواد کو صاف کرتی ہے، جس سے میموری کو دوبارہ پروگرام کرنا ممکن ہو جاتا ہے۔

ایک EPROM ایک PROM سے مختلف ہے، PROM صرف ایک بار لکھا جا سکتا ہے اور اسے مٹایا نہیں جا سکتا۔ EPROMs کو پرسنل کمپیوٹرز میں بڑے پیمانے پر استعمال کیا جاتا ہے کیونکہ وہ کارخانہ دار (Manufacturer) کو PROM کے مواد کو جدید شدہ ورژن کے ساتھ تبدیل کرنے یا کمپیوٹر کی فراہمی سے پہلے مواد کو مٹانے کے قابل بناتے ہیں۔



تصویر 3.7 ایریز ایبل پروگرامیبل ریڈ اونلی میموری

کیا آپ جانتے ہیں؟

زیادہ تر EPROM چپس کی اوپری سطح پر ایک شفاف علاقہ ہوتا ہے جو اسٹیکرز سے ڈھکا ہوتا ہے۔ اگر اسے ہٹا دیا جاتا ہے، تو سورج کی روشنی میں بنفشی شعاعوں کی روشنی مواد کو مٹا سکتی ہے۔

3.5.3.3 الیکٹریکلی ایریز ایبل پروگرامیبل ریڈ اونلی میموری (EEPROM)

الیکٹریکلی ایریز ایبل پروگرام ایبل ریڈ اونلی میموری (Electrically Erasable Programmable Read Only Memory) ایک خاص قسم کی PROM ہے جسے برقی چارج سے بے نقاب کرکے مٹایا جاسکتا ہے۔ PROM کی دیگر اقسام کی طرح، EEPROM پاور آف ہونے پر بھی اپنے مواد کو برقرار رکھتا ہے۔ ROM کی دیگر تمام اقسام کے ساتھ موازنہ کرتے ہوئے، EEPROM کارکردگی میں سست ہے۔

3.5.4 کیش میموری (Cache Memory)

کیش میموری ایک بہت تیز رفتار اور مہنگی میموری ہے، جو میموری کی بازیافت کے عمل کو تیز کرنے کے لیے استعمال کی جاتی ہے۔ اس کی زیادہ قیمت کی وجہ سے، CPU مرکزی میموری کے سائز کے مقابلے میں کیش میموری کے چھوٹے سائز کے ساتھ آتا ہے۔ کیش میموری کے بغیر، جب بھی CPU ڈیٹا کی درخواست کرتا ہے، اسے مین میموری سے لانا پڑتا ہے جس میں زیادہ وقت لگتا ہے۔ کیش متعارف کرانے کا خیال یہ ہے کہ، یہ انتہائی تیز میموری ڈیٹا کو ذخیرہ کرے گی جس تک کثرت سے رسائی حاصل کی جاتی ہے اور اگر ممکن ہو تو، وہ ڈیٹا جو اس کے قریب ہے، اس سے تیز رسپانس ٹائم حاصل کرنے میں مدد ملتی ہے، جہاں رسپانس ٹائم، (Access Time) سے مراد یہ ہے کہ میموری پڑھنے/لکھنے کی درخواست کا کتنی جلدی جواب دے سکتی ہے۔ شکل 3.8 سی پی او اور مین میموری کے درمیان کیش میموری کی ترتیب کو ظاہر کرتا ہے۔



تصویر 3.8 کیش میموری اریج مینٹ

3.6 سیکنڈری اسٹوریج ڈیوائسز (Secondary Storage Devices)

ایک کمپیوٹر میں عام طور پر مرکزی میموری کی محدود مقدار ہوتی ہے جو کہ مہنگی اور غیر مستحکم (Non Volatile) ہوتی ہے۔ ڈیٹا اور پروگراموں کو مستقل طور پر ذخیرہ کرنے کے لیے، ثانوی سٹوریج کے آلات استعمال کیے جاتے ہیں۔ ثانوی سٹوریج ڈیوائسس مین میموری کے لیے معاون سٹوریج کے طور پر کام کرتی ہیں اور وہ فطرتاً غیر مستحکم ہوتے ہیں، سیکنڈری اسٹوریج کو بیک اپ اسٹوریج (Backup Storage) بھی کہا جاتا ہے۔

3.6.1 ہارڈ ڈسک

ہارڈ ڈسک (Hard Disk) ایک مقناطیسی ڈسک (Magnetic disk) ہے جس پر آپ ڈیٹا محفوظ کر سکتے ہیں۔ ہارڈ ڈسک میں ہر ایک ڈسک کے لیے سروں کے جوڑے کے ذریعے رسائی حاصل کرنے والی ڈسکوں کا سجا ہوا انتظام ہوتا ہے۔ ہارڈ ڈسک سنگل یا ڈبل سائیڈ ڈسک کے ساتھ آتی ہیں۔

3.6.2 کامپیکٹ ڈسک (CD)

ایک CD یا CD-ROM 1.2 ملی میٹر موٹی، پولی کاربونیٹ پلاسٹک (Polycarbonate Plastic) مواد سے بنایا گیا ہے۔ سطح پر ایلومینیم یا سونے کی ایک پتلی تہہ لگائی جاتی ہے۔ سی ڈی ڈیٹا کو چھوٹے انڈینٹیشنز کے طور پر پیش کیا جاتا ہے جسے "پٹس (Pits)" کہا جاتا ہے، جو پولی کاربونیٹ کی تہہ کے اوپری حصے میں ڈھلے ہوئے سرپل ٹریک میں انکوڈ کیا ہوا ہے۔ پٹس کے درمیان

11th_Comp Science_UM_Ch 3.indd 45 18-04-2023 13:07:36

والے علاقوں کو "زمین" (Lands) کہا جاتا ہے۔ سی ڈی پلیئر کے اندر ایک موٹر ڈسک کو گھماتی ہے۔ ایک عام CD-ROM میں 700MB کی گنجائش ہے۔



تصویر 3.9 کامپیکٹ ڈسک

3.6.3 ڈیجیٹل ورسٹائل ڈسک (DVD)

DVD (ڈیجیٹل ورسٹائل ڈسک یا ڈیجیٹل ویڈیو ڈسک) ایک آپٹیکل ڈسک (optical disk) ہے جو 4.7 جی بی (GB) تک، سی ڈی (CD) سے چھ گنا زیادہ ڈیٹا کو محفوظ کرنے کی گنجائش ہے ۔ ڈی وی ڈیز (DVDs) اکثر فلموں کو بہتر کوالٹی میں محفوظ کرنے کے لیے استعمال ہوتی ہیں۔ سی ڈیز (CDs) کی طرح ڈی وی ڈیز (DVDs) کو لیزر سے پڑھا جاتا ہے۔

ڈسک میں ایک یا دو اطراف اور ڈیٹا کی ایک یا دو پرتیں ہو سکتی ہیں؛ اطراف اور تہوں کی تعداد اس بات کا تعین کرتی ہے کہ یہ کتنا سنبھال سکتا ہے۔ دونوں اطراف کی پرتیں (Double – Layered sides) عام طور پر سنہری رنگ کے ہوتے ہیں، جب کہ ایک طرف کی پرت(Single – Layered Sides) عام طور پر چاندی کے رنگ کے ہوتے ہیں، جیسے کہ سی ڈی (CD)۔



تصویر 3.10 ڈیجیٹل ورسٹائل (Digital Versatile Disc)

3.6.4 فلیش میموری ڈیوائسز (Flash Memory Devices)

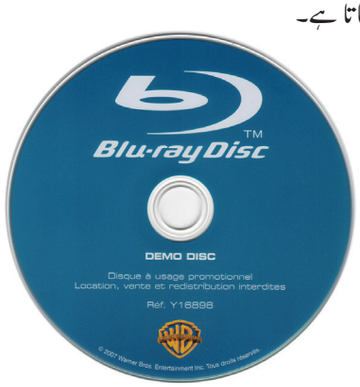
فلیش میموری (Flash Memory) ایک برقی (ٹھوس حالت) غیر مستحکم کمپیوٹر اسٹوریج میڈیم ہے، جسے برقی طور پر مٹایا اور دوبارہ پروگرام کیاجا سکتا ہے۔ وہ یا تو EEPROM ہیں، یا EPROM۔ فلیش میموریز کی مثالیں پین ڈرائیوس (Pendrives)، میموری کارڈس (Memory Cards)وغیرہ ہیں۔ فلیش میموریز ذاتی کمپیوٹرز ، ذاتی ڈیجیٹل اسسٹنٹ (PDA)، ڈیجیٹل آڈیو پلیئر س (Digital Audio players)، ڈیجیٹل کیمرے (Digital Camera) اور موبائل فون (Mobile Phones) میں استعمال کیا جا سکتا ہے۔فلیش میموری تیز رفتاراوقات فراہم کرتا ہے۔ جو وقت میموری میں ایک حرف کو پڑھنے یا لکھنے میں لگتا ہے اسے ایکسیس ٹائم (Access Time) کہتے ہیں۔ فلیش میموری میں 1 جی بی (GB) سے 2 ٹی بی (TB) کے درمیان گنجائش ہوتی ہے۔ فلیش میموری کا نمونہ تصویر 3.11 میں دکھایا گیا ہے۔



تصویر 3.11 فلیش میموری

3.6.5 بلو رے ڈسک

بلو رے ڈسک (Blu – Ray Disc) ایک اعلی کثافت آپٹیکل ڈسک (high-density optica disk) ہے جو DVD کی طرح ہوتا ہے۔ بلو رے ڈسک کی وہ قسم ہے جو پلے اسٹیشن گیمز (Play Station Games) اور ہائی ڈیفی نیشن (HD) فلمیں چلانے کے لیے استعمال ہوتی ہے۔ ایک ڈبل لیئر بلو رے ڈسک 50 GB (Giga Bytes) تک ڈیٹا محفوظ کر سکتی ہے۔ ڈی وی ڈی (DVD) ڈیٹا کو پڑھنے اور لکھنے کے لیے سرخ لیزر کا استعمال کرتی ہے۔ لیکن، بلو رے لکھنے کے لیے بلیو وائلٹ لیزر کا استعمال کرتا ہے۔ اس لیے اسے بلو رے کہا جاتا ہے۔



تصویر 3.12 بلو رے ڈسک

3.7 پورٹس اور انٹرفیس (Ports and Interface)

کمپیوٹر کے مدر بورڈ میں بہت سے I/O ساکٹ ہوتے ہیں جو کمپیوٹر کے پچھلے حصے میں پائے جانے والے پورٹس اور انٹرفیس سے جڑے ہوتے ہیں (تصویر 3.13)۔ بیرونی آلات کو پورٹس (Ports) اور انٹرفیس (Interface) سے منسلک کیا جا سکتا ہے۔ پورٹس کی مختلف اقسام ذیل میں دی گئی ہیں۔

11th_Comp Science_UM_Ch 3.indd 46 18-04-2023 13:07:36

سیریل پورٹ: پرانے کمپیوٹرز میں پائے جانے والے بیرونی آلات کو جوڑنے کے لیے۔

متوازی پورٹ: پرانے کمپیوٹرز میں پائے جانے والے پرنٹرز کو جوڑنے کے لیے۔

USB پورٹس: بیرونی آلات جیسے کیمرے، سکینر، موبائل فون، بیرونی ہارڈ ڈسک اور پرنٹرز کو کمپیوٹر سے جوڑنے کے لیے۔

USB 3.0 یونیورسل سیریل بس (USB) معیار کا تیسرا بڑا ورژن ہے جو کمپیوٹر کو دوسرے الیکٹرانک گیجٹس(ایک قسم کا آلہ) سے جوڑتا ہے جیسا کہ شکل 3.13 میں دکھایا گیا ہے۔ USB 3.0 5 گیگا بائٹ (GB) پر سیکنڈ تک ڈیٹا منتقل کر سکتا ہے۔ USB 3.1 اور USB 3.2 بھی جاری کیے گئے ہیں۔



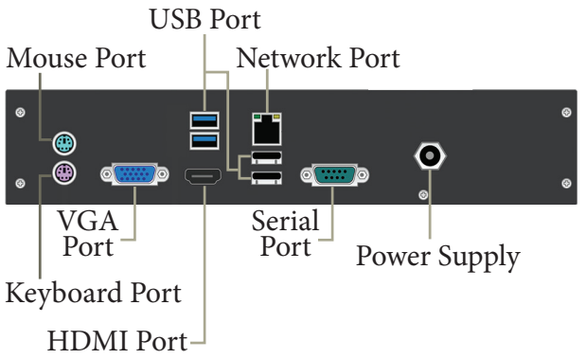
تصویر 3.13 USB 3.0 پورٹس

VGA کنیکٹر: کوئی بھی ڈسپلے ڈیوائس جیسے LCD پروجیکٹر یا مانیٹر کو جوڑنے کے لیے ہے۔

آڈیو پلگ: ساؤنڈ سپیکر، مائیکروفون اور ہیڈ فون کو جوڑنے کے لیے۔

PS/2 پورٹ: ماؤس اور کی بورڈ کو پی سی سے جوڑنے کے لیے۔

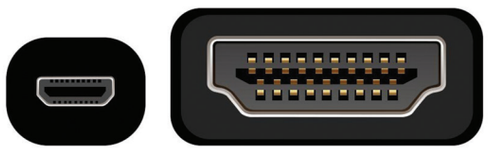
SCSI پورٹ: ہارڈ ڈسک ڈرائیوز اور نیٹ ورک کنیکٹرز کو جوڑنے کے لیے۔



تصویر3.14 پورٹس و انٹرفیس

ہائی ڈیفینیشن ملٹی میڈیا انٹرفیس (HDMI)

ہائی ڈیفینیشن ملٹی میڈیا انٹرفیس (High-Definition Multimedia Interface) ایک آڈیو/ویڈیو انٹرفیس ہے جو غیر کمپریسڈ ویڈیو اور آڈیو ڈیٹا کو ویڈیو کنٹرولر سے ایک ہم آہنگ کمپیوٹر مانیٹر، ایل سی ڈی (LCD) پروجیکٹر، ڈیجیٹل ٹیلی ویژن وغیرہ میں منتقل کرتا ہے۔



تصویر 3.15 ایچ ڈی ایم آئی (HDMI) پورٹس

سرگرمی

طلبہ کی سرگرمی

- کمپیوٹر کے اجزاء کی شناخت کریں۔
- بیرونی آلات جیسے پرنٹر/LCD پروجیکٹر کو جوڑنا۔

اساتذہ کی سرگرمی

- کمپیوٹر کے اجزاء دکھائیں۔
- مختلف ROM ICs دکھائیں۔
- فلیش میموری دکھائیں۔
- مختلف پورٹس اور ان کے استعمال کا مظاہرہ کریں۔

11th_Comp Science_UM_Ch 3.indd 47 18-04-2023 13:07:37

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں۔

1. مندرجہ ذیل میں سے کسے کمپیوٹر کا دماغ کہا جاتا ہے؟
   (a) ان پٹ ڈیوائسز
   (b) آؤٹ پٹ ڈیوائسز
   (c) میموری ڈیوائس
   (d) مائکرو پروسیسر

2. مندرجہ ذیل میں سے کون سا مائکرو پروسیسر یونٹ کا حصہ نہیں ہے؟
   (a) ALU (b) کنٹرول یونٹ (CU)
   (c) کیش میموری (Cache Memory)
   (d) رجسٹر (Register)

3. کتنے بٹس ایک لفظ کی تشکیل کرتے ہیں؟
   (a) 8
   (b) 16
   (c) 32
   (d) استعمال شدہ پروسیسر کے ذریعہ طے کیا جاتا ہے۔

4. جب میموری ایڈریس رجسٹر میں ایڈریس رکھا جاتا ہے تو درج ذیل میں سے کون سا آلہ مقام کی نشاندہی کرتا ہے؟
   (a) لوکیٹر (Locator)
   (b) انکوڈر (Encoder)
   (c) ڈی کوڈر (Decoder)
   (d) ملٹی پلیکسر (Multiplexer)

5. درج ذیل میں سے کون سا CISC پروسیسر ہے؟
   Intel P6 (a) AMD K6 (b)
   Pentium III (d) Pentium IV (c)

6. سب سے تیز میموری کون سی ہے؟
   (a) ہارڈ ڈسک
   (b) مین میموری
   (c) کیش میموری
   (d) بلیو رے ڈسک

7. ایک وقت میں 8 بٹس ایڈریس بس کے ساتھ ایک پروسیسر کی طرف سے کتنے میموری مقامات کی نشاندہی کی جاتی ہے؟
   28 (a) 1024 (b)
   256 (c) 8000 (d)

8. سنگل سائیڈڈ اور سنگل لیئر کے ساتھ 12cm قطر کی DVD کی گنجائش کیا ہے؟
   4.7 GB (a) 5.5 GB (b)
   7.8 GB (c) 2.2 GB (d)

9. سی ڈی میں دکھائے جانے والے ڈیٹا کا سب سے چھوٹا سائز کیا ہے؟
   (a) بلاکس (Blocks) (b) سیکٹرز (Sectors)
   (c) پٹس (Pits) (d) ٹریک (Tracks)

10. ڈسپلے ڈیوائسز کے ذریعے کمپیوٹر سے منسلک ہوتے ہیں۔
    USB port (a) Ps/2 port (b)
    SCSI port (c) VGA connector (d)

سیکشن - B

بہت مختصر جواب دیں۔

(1) وہ کون سے پیرامیٹرز ہیں جو مائکرو پروسیسر کی خصوصیات کو متاثر کرتے ہیں؟

(2) انسٹرکشن کیا ہے؟

(3) پروگرام کاؤنٹر کیا ہے؟

(4) HDMI کیا ہے؟

(5) EPROM کے مواد کو مٹانے کے لیے کون سا ذریعہ استعمال کیا جاتا ہے؟

سیکشن - C

مختصر جوابات دیں۔

(1) کمپیوٹر آرگنائزیشن کو کمپیوٹر آرکیٹیکچر سے تفریق کریں۔

11th_Comp Science_UM_Ch 3.indd 48 18-04-2023 13:07:37

(2) ڈیٹا کے سائز کی بنیاد پر مائکرو پروسیسر کی درجہ بندی کریں۔

(3) انسٹرکشن سیٹ کی بنیاد پر مائکرو پروسیسرز کی درجہ بندی لکھیں۔

(4) PROM اور EPROM میں تفریق کو واضح کریں۔

(5) کمپیوٹر میں دستیاب انٹرفیس اور پورٹس لکھیں۔

(6) سی ڈی اور ڈی وی ڈی میں تفریق کریں۔

(7) فلیش میموری اور EEPROM میں تفریق کس طرح کریں گے؟

سیکشن – D

تفصیل سے بیان کریں۔

(1) مائکرو پروسیسر کی خصوصیات کی وضاحت کریں۔

(2) پروسیسر کے ذریعے پڑھنے اور لکھنے کے کام کیسے انجام دیئے جاتے ہیں؟ وضاحت کریں۔

(3) ایکسیس ٹائم کی بنیاد پر میموری آلات کو صعودی (Ascending) ترتیب میں ترتیب دیں۔

(4) ROM کے اقسام کی وضاحت کریں۔

PTOHR

لغات

کمپیوٹر ہارڈ ویئر (Computer Hardware)	کمپیوٹر کے جسمانی حصے یا اجزاء۔ جیسے سی پی یو، مدر بورڈ، مانیٹر، کی بورڈ وغیرہ۔
انٹیل (Intel)	انٹیل کارپوریشن ایک امریکی ملٹی نیشنل کارپوریشن اور ٹیکنالوجی کمپنی ہے جو ہارڈویئر مینوفیکچرنگ، خاص طور پر مدر بورڈ اور پروسیسرز شامل ہے۔
سلیکون چپ (Silicon chip)	سلیکون چپ ایک مربوط، سیمی کنڈکٹر مواد کے ایک چھوٹے فلیٹ ٹکڑے پر الیکٹرانک سرکٹس کا سیٹ ہے۔
کثیر مقصدی (Multipurpose)	کثیر مقصدی کے کئی مقاصد ہیں۔
ایڈریس بس (Address Bus)	ایڈریس بس تاروں کا ایک مجموعہ ہے جو ایڈریس کو بٹس کے طور پر لے جاتی ہے۔
ڈیٹا بس (Data Bus)	ڈیٹا بس بٹس میں ڈیٹا لے جانے کے لیے تاروں کا ایک مجموعہ ہے۔
کنٹرول بس (Control Bus)	کنٹرول بس آپریشنز/فنکشنز کو کنٹرول کرنے کے لیے ایک کنٹرول لائن/ تاروں کا مجموعہ ہے۔
ریاضی کے آپریشنز (Arithmatic operations)	ریاضی کی کارروائیاں اعداد و شمار پر ریاضی کی کارروائیاں ہیں ۔ جیسے جوڑ، گھٹاؤ وغیرہ
مواد کی منتقلی (Data Transfer)	ڈیٹا ٹرانسفر کا مطلب ہے ڈیٹا کو ایک حصہ سے دوسرے حصہ میں منتقل کرنا
منطقی آپریشنز (Logical Operations)	منطقی آپریشنز بائنری/بولین ڈیٹا پر آپریشنز ہیں جیسے AND, OR, NOT
دو طرفہ (Bidirectional)	دو طرفہ کا مطلب ہے دونوں سمتیں/ راستے
یک طرفہ (Unidirectional)	یک سمت کا مطلب صرف ایک سمت ہے۔
رسائی کا وقت (Access Time)	رسائی کا وقت الیکٹرانک سسٹم کی درخواست، اور رسائی کے مکمل ہونے یا درخواست کردہ ڈیٹا کی واپسی کے درمیان وقت کی تاخیر ہے۔

11th_Comp Science_UM_Ch 3.indd 49 18-04-2023 13:07:37

باب 4

آپریٹنگ سسٹم کے نظریاتی تصورات

سیکھنے کے مقاصد

- ✓ آپریٹنگ سسٹمز کے تصور اور ان کے اقسام کو جاننا۔
- ✓ آپریٹنگ سسٹمز اور اس کے عمل کا بنیادی علم حاصل کرنا۔

4.1 سافٹ ویئر کا تعارف

ایک سافٹ ویئر ہدایات کا سیٹ ہے جو مخصوص کام انجام دیتا ہے۔ یہ مطلوبہ آؤٹ پٹ پیدا کرنے کے لیے بنیادی طور پر ہارڈ ویئر کے ساتھ بات چیت کرتا ہے۔

4.1.1 سافٹ ویئر کے اقسام

سافٹ ویئر کو دو اقسام میں تقسیم کیا گیا ہے:

1) ایپلیکیشن سافٹ ویئر (Application Software)
2) سسٹم سافٹ ویئر (System Software)

ایپلیکیشن سافٹ ویئر (Application Software) :

ایپلیکیشن سافٹ ویئر مخصوص کام انجام دینے کے لیے پروگراموں کا ایک مجموعہ ہے۔ مثال کے طور پر MS-word ٹیکسٹ دستاویز بنانے کے لیے ایک ایپلیکیشن سافٹ ویئر ہے اور VLC پلیئر آڈیو، ویڈیو فائلز اور بہت کچھ چلانے کے لیے جانا پہچانا ایپلی کیشن سافٹ ویئر ہے۔

سسٹم سوفٹ ویئر (System Software) :

سسٹم سافٹ ویئر کمپیوٹر پروگرام کی ایک قسم ہے جو کمپیوٹر کے ہارڈویئر اور ایپلیکیشن پروگراموں کو چلانے کے لیے ڈیزائن کیا گیا ہے۔ مثال آپریٹنگ سسٹم اور لینگویج پروسیسر۔

4.2 آپریٹنگ سسٹم (OS) کا تعارف

ایک آپریٹنگ سسٹم (OS) ایک سسٹم سافٹ ویئر ہے جو صارف اور کمپیوٹر کے درمیان ایک انٹرفیس کا کام کرتا ہے۔

یہ ان پٹ، آؤٹ پٹ اور دیگر پیریفیرل (کمپیوٹر سے منسلک آلات) آلات جیسے ڈسک ڈرائیوز، پرنٹرز اور الیکٹرانک گیجٹس (Gadgets) کو کنٹرول کرتا ہے۔ آپریٹنگ سسٹم کے افعال میں فائل مینجمنٹ، میموری مینجمنٹ، پروسیس مینجمنٹ اور ڈیوائس مینجمنٹ اور بہت کچھ شامل ہے۔

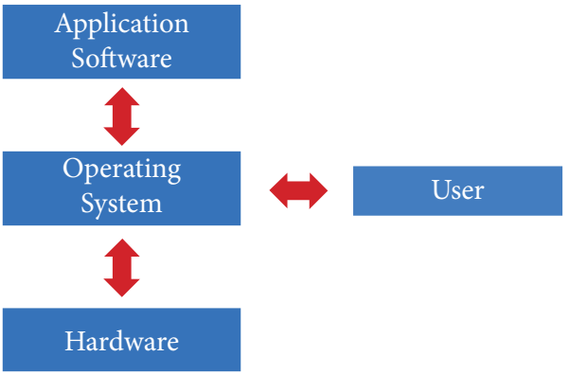


تصویر 4.1 آپریٹنگ سسٹم

آپریٹنگ سسٹم کے بغیر، کمپیوٹر تمام وسائل کو مؤثر طریقے سے منظم نہیں کر سکتا۔ جب کمپیوٹر کو آن کیا جاتا ہے تو آپریٹنگ سسٹم خود بخود میموری میں لوڈ ہو جاتا ہے۔

پرسنل کمپیوٹرز اور لیپ ٹاپس میں استعمال ہونے والے کچھ مشہور آپریٹنگ سسٹمس ونڈوز (Windows)، یونیکس (UNIX) اور لینکس (Linux) ہیں۔ موبائل آلات زیادہ تر اینڈرائیڈ اور ioS کو بطور موبائل OS استعمال کرتے ہیں۔

11th_Comp Science_UM_Ch 4.indd 50 18-04-2023 13:19:49



تصویر 4.2 آپریٹنگ سسٹم اور صارف کا تعامل

آپریٹنگ سسٹمز کا استعمال

آپریٹنگ سسٹم کے چند استعمالات درج ذیل ہیں۔

آپریٹنگ سسٹم کا بنیادی استعمال ہے۔

- اس بات کو یقینی بنانا کہ کمپیوٹر کو استعمال کرنے کے لیے صارف کو کیا کرنا چاہتا ہے۔
- صارفین اور کمپیوٹرز کے درمیان آسان تعامل۔
- پاور آن ہونے پر کمپیوٹر کا آپریشن خود بخود شروع ہو جانا۔ (Booting)
- ان پٹ اور آؤٹ پٹ ڈیوائسز کو کنٹرول کرنا۔
- مین میموری کے استعمال کا انتظام کرنا۔
- صارف کے پروگراموں کو سیکیورٹی فراہم کرنا۔

4.3 آپریٹنگ سسٹم (Operating System) کے اقسام

آپریٹنگ سسٹم کو ان کی پروسیسنگ کی صلاحیتوں کے لحاظ سے درج ذیل اقسام میں درجہ بندی کیا گیا ہے۔

4.3.1 سنگل یوزر آپریٹنگ سسٹم (Single User Operating System)

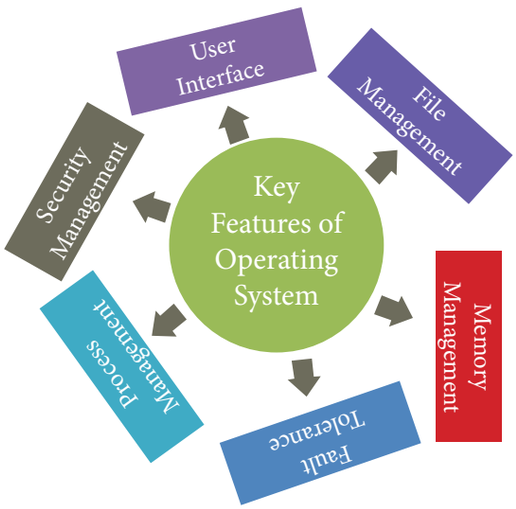
ایک آپریٹنگ سسٹم صرف ایک صارف کو ایک وقت میں ایک کام انجام دینے کی اجازت دیتا ہے۔ اسے سنگل یوزر اور سنگل ٹاسک آپریٹنگ سسٹم کہا جاتا ہے۔ MS-DOS سنگل یوزر اور سنگل ٹاسک آپریٹنگ سسٹم کے لیے ایک مثال ہے۔

4.3.2 ملٹی یوزر آپریٹنگ سسٹم (Multi User Operating System)

یہ کمپیوٹر اور لیپ ٹاپ میں استعمال ہوتا ہے جو ایک ہی وقت میں متعدد صارفین کے ذریعہ ایک ہی ڈیٹا اور ایپلی کیشنز تک رسائی حاصل کرنے کی اجازت دیتا ہے۔ صارفین ایک دوسرے سے رابطہ بھی کر سکتے ہیں۔ ونڈوز، لینکس اور یونیکس ملٹی یوزر آپریٹنگ سسٹم کی مثالیں ہیں۔

4.4 آپریٹنگ سسٹم کی اہم خصوصیات

مختلف کلیدی خصوصیات ذیل میں دی گئی ہیں۔



تصویر 4.3 آپریٹنگ سسٹم کے اہم خصوصیات

4.4.1 یوزر انٹرفیس (UI- User interface)

یوزر انٹرفیس آپریٹنگ سسٹم کی ایک اہم خصوصیت ہے۔ واحد طریقہ جس سے صارف کمپیوٹر کے ساتھ تعامل کر سکتا ہے۔ یہ GUI (Graphical User Interface) پر مبنی آپریٹنگ سسٹم کی کلیدی کامیابی کی ایک بڑی وجہ ہے۔ GUI ایک ونڈو پر مبنی سسٹم ہے جس میں I/O کو ڈائریکٹ کرنے کے لیے ایک پوائنٹنگ ڈیوائس (Pointing Devices) ہے، مینو میں سے انتخاب کرنے کے لیے منتخب کریں اور متن داخل کرنے کے لیے ایک کی بورڈ (Keyboard) کی ضرورت ہے۔ اس کے متحرک رنگ بہت آسانی سے صارف کو اپنی طرف متوجہ کرتا ہے۔

11th_Comp Science_UM_Ch 4.indd 51 18-04-2023 13:19:49

اب لینکس (Linux) کی تقسیم بھی GUI پر مبنی آپریٹنگ سسٹم کے طور پر دستیاب ہے۔ جب یوزر انٹرفیس کسی ایپلیکیشن کے لیے ڈیزائن کیا جاتا ہے تو درج ذیل نکات پر غور کرتا ہے۔

1. یوزر انٹرفیس کو چاہیے کہ وہ صارف کو اس مہارت کے ساتھ زیادہ دیر تک برقرار رکھنے کے قابل بنائے۔
2. یوزر انٹرفیس کو چاہیے کہ گاہک کو ان کی ضروریات کی بنیاد پر بھی مطمئن کرے۔
3. یوزر انٹرفیس کو چاہیے کہ صارف کا قیمتی وقت بچائے ۔
4. کسی بھی پروڈکٹ کا حتمی مقصد گاہک کو مطمئن کرنا ہوتا ہے۔ یوزر انٹرفیس گاہک کو مطمئن کرنے کے لیے بھی ہے۔
5. یوزر انٹرفیس کو چاہیے کہ صارف کی طرف سے کی گئی غلطیوں کی تعداد کم کرے۔

4.4.2 میموری مینجمنٹ (Memory Management)

میموری مینجمنٹ کمپیوٹر کی مرکزی میموری کو کنٹرول کرنے اور مربوط کرنے اور کمپیوٹر کی مجموعی کارکردگی کو بہتر بنانے کے لیے مختلف چلنے والے پروگراموں کو میموری بلاک (اسپیس) تفویض کرنے کا عمل ہے۔ میموری مینجمنٹ میں صارف کے مطالبات کی بنیاد پر انفرادی پروگراموں کے لیے مخصوص میموری بلاکس کو مختص کرنا شامل ہے۔

میموری مینجمنٹ کے عمل کا مقصد CPU کے استعمال اور مین میموری کے ذریعے اپنے صارفین کو کمپیوٹر کے ردعمل کی رفتار دونوں کو بہتر بنانا ہے۔ ان وجوہات کی بناء پر کمپیوٹرز کے کئی پروگراموں کو مین میموری میں رکھنا چاہیے جو میموری مینجمنٹ کی بہت سی مختلف اسکیموں کے ساتھ منسلک ہوتے ہیں۔

آپریٹنگ سسٹم میموری مینجمنٹ کے سلسلے میں درج ذیل سرگرمیوں کے لیے ذمہ دار ہے۔

- اس بات پر نظر رکھنا کہ میموری کا کون سا حصہ اس وقت استعمال ہو رہا ہے اور کون استعمال کر رہا ہے۔
- اس بات کا تعین کرنا کہ کون سے عمل (یا عمل کے حصے) اور ڈیٹا کو میموری میں اور باہر منتقل کرنا ہے۔
- مین میموری میں پروگرام کی ضرورت کے مطابق میموری بلاکس کی ایلوکیشن (Allocation) اور ڈی ایلوکیشن (de-allocation)۔ (Garbage Collection)

4.4.3 پروسیس مینجمنٹ (Process Management)

پروسیس مینجمنٹ ایک فنکشن ہے جس میں پروسیس (پروگرام) بنانا اور حذف کرنا اور ایک دوسرے کے ساتھ بات چیت اور ہم آہنگی کے لیے طریقہ کار فراہم کرنا شامل ہے۔

ایک سسٹم ٹاسک، جیسے پرنٹر یا اسکرین پر آؤٹ پٹ بھیجنے کو بھی پروسیس کہا جا سکتا ہے۔

ایک کمپیوٹر عمل کے مجموعے پر مشتمل ہوتا ہے، انہیں دو زمروں میں درجہ بندی کیا جاتا ہے:

- آپریٹنگ سسٹم کے عمل جو سسٹم کوڈ کے ذریعے انجام پاتے ہیں۔
- صارف کے عمل جو صارف کوڈ کے ذریعے انجام پاتے ہیں۔

یہ تمام عمل ممکنہ طور پر ایک ہی سی پی یو پر بیک وقت انجام دے سکتے ہیں۔

مندرجہ ذیل الگورتھمس (Algorithms) بنیادی طور پر پروسیسر کے کام کو مختص کرنے کے لیے استعمال کیا جاتا ہے۔

1. فیفو (FIFO) 2. ایس جے ایف (SJF)
3. راؤنڈ ربن (Round Ribbon)
4. ترجیح کی بنیاد پر (Based on priority)

FIFO (First in First out) شیڈولنگ (Scheduling)

یہ الگورتھم قطار لگانے کی تکنیک پر مبنی ہے۔ فرض کریں کہ ایک طالب علم ہے اپنے استاد سے گریڈ شیٹ لینے کے لیے قطار

11th_Comp Science_UM_Ch 4.indd 52 18-04-2023 13:19:50

میں کھڑا ہے۔ دوسرا طالب علم جو قطار میں سب سے پہلے کھڑا ہوتا ہے اسے پہلے اس کی گریڈ شیٹ ملتی ہے اور قطار سے نکل جاتا ہے۔ اس کے بعد قطار میں اگلا طالب علم اسے درست کرتا ہے وغیرہ۔ یہ FIFO الگورتھم کی بنیادی منطق ہے۔

تکنیکی طور پر، وہ عمل جو قطار میں پہلے داخل ہوتا ہے پہلے CPU کے ذریعے عمل میں لایا جاتا ہے، اس کے بعد اگلا وغیرہ۔ عمل کو قطار کی ترتیب میں انجام دیا جاتا ہے۔

SJF (پہلے مختصر ترین کام) (Shortest Job first) شیڈولنگ (Scheduling)

یہ الگورتھم کام کے سائز کی بنیاد پر کام کرتا ہے جو CPU کے ذریعے انجام دیا جا رہا ہے۔

دو کاموں A اور B پر غور کریں۔

1) A = 6 کلو بائٹس 2) B = 9 کلو بائٹس

پہلے نوکری (Job) "A" کو تفویض کیا جائے گا اور پھر نوکری "B" کی باری آئے گی۔

راؤنڈ رابن شیڈولنگ (Round Robin Scheduling)

راؤنڈ رابن (RR) شیڈولنگ الگورتھم خاص طور پر ٹائم شیئرنگ (Time Sharing) سسٹم کے لیے ڈیزائن کیا گیا ہے۔ نوکریاں (Jobs) ایک سرکلر طریقہ میں تفویض اور پروسیسر کا وقت ہیں۔ مثال کے طور پر تین نوکریاں A,B,C لیں۔ پہلے A کو CPU کو تفویض کیا جاتا ہے پھر B اور جاب C کو اور پھر A، B اور C وغیرہ کو۔

ترجیح کی بنیاد پر (Based On Priority)

دی گئی ملازمت (عمل) کو ترجیح کی بنیاد پر تفویض کیا جاتا ہے۔ جس کام کو ترجیح دی جائے وہ دوسری ملازمتوں سے زیادہ اہم ہے۔ دو ملازمتیں A اور B لیں۔ A کی ترجیح 5 اور B کی ترجیح 7 ہونے دیں۔

نوکری A سے پہلے نوکری B پروسیسر کو تفویض کی گئی ہے۔

4.4.4 سیکیورٹی مینجمنٹ (Security Management)

کمپیوٹر اور سافٹ ویئر انڈسٹری میں سب سے بڑا چیلنج صارف کے جائز ڈیٹا کو ہیکرز (Hackers) (کمپیوٹر کا ڈاٹا چرانے والے) سے بچانا ہے۔ آپریٹنگ سسٹم صارف کو تین درجے کی سیکیورٹیز فراہم کرتا ہے۔ وہ یہ ہیں

(1) فائل تک رسائی کی سطح (File access level)

(2) نظام کی سطح (System Level)

(3) نیٹ ورک کی سطح (Network Level)

دوسرے لوگوں کی بنائی ہوئی فائلوں تک رسائی حاصل کرنے کے لیے، آپ کو رسائی کی اجازت ہونی چاہیے۔ اجازتیں یا تو فائل کے تخلیق کار یا سسٹم کے منتظم کی طرف سے دی جا سکتی ہیں۔

سسٹم لیول سیکیورٹی ایک کثیر صارف ماحول میں پاس ورڈ کے ذریعہ پیش کی جاتی ہے۔

ونڈوز اور لینکس دونوں پاس ورڈ کی سہولت پیش کرتے ہیں۔

نیٹ ورک سیکیورٹی ایک ناقابل بیان ہے۔ اس لیے پوری دنیا کے لوگ ایسی سیکیورٹی فراہم کرنے کی کوشش کرتے ہیں۔

حفاظتی خصوصیات کی اوپر کی تمام سطحیں صرف آپریٹنگ سسٹم کے ذریعہ فراہم کی جاتی ہیں۔

4.4.5 غلطی برداشت کرنا (Fault Tolerance)

آپریٹنگ سسٹمز کو مضبوط ہونا چاہیے۔ جب کوئی خرابی ہوتی ہے تو آپریٹنگ سسٹم کو کریش (Crash) نہیں ہونا چاہیے، اس کے بجائے آپریٹنگ سسٹم میں غلطی کو برداشت کرنے کی صلاحیت ہوتی ہے اور سسٹم کی موجودہ حالت کو برقرار رکھا جاتا ہے۔

11th_Comp Science_UM_Ch 4.indd 53 18-04-2023 13:19:50

4.4.6 فائل مینجمنٹ (File Management)

فائل مینجمنٹ OS کا ایک اہم کام ہے جو ڈیٹا اسٹوریج کی تکنیک کو سنبھالتا ہے۔ آپریٹنگ سسٹم کمپیوٹر پر فائلوں، فولڈرز اور ڈائرکٹری سسٹم کا انتظام کرتا ہے۔ FAT (File Allocation Table) فائلوں کے بارے میں عام معلومات جیسے فائل کا نام، قسم (ٹیکسٹ یا بائنری)، سائز، ابتدائی پتہ اور رسائی موڈ کو محفوظ کرتا ہے۔ فائل مینیجر آپریٹنگ سسٹم فائلوں کو بنانے، ترمیم کرنے، کاپی کرنے، میموری کو مختص کرنے اور FAT کو اپ ڈیٹ کرنے میں مدد کرتا ہے۔ فائل مینجمنٹ کی چند دوسری تکنیکیں دستیاب ہیں. جیسے آنے والی پیڑھی کا فائل سسٹم (NTFS- Next Generation File System) اور ext2 (Linux)۔

4.4.7 ملٹی پروسیسنگ (Multi - Processing)

یہ آپریٹنگ سسٹم کی خصوصیات میں سے ایک ہے۔ اس میں ایک ہی چلنے والے عمل (Job) کے لیے دو یا زیادہ پروسیسرز ہیں۔ پروسیسنگ متوازی میں ہوتی ہے اسے متوازی پروسیسنگ(Parallel Processing) کہا جاتا ہے۔ چونکہ عمل متوازی طور پر ہوتا ہے، اس لیے یہ فیچر تیز رفتاری سے عمل درآمد کے لیے استعمال ہوتا ہے جس سے کمپیوٹنگ کی طاقت بڑھ جاتی ہے۔

4.4.8 ٹائم شیئرنگ (Time - Sharing)

یہ آپریٹنگ سسٹمز کی خصوصیات میں سے ایک ہے۔ یہ متعدد کاموں یا عمل کو بیک وقت انجام دینے کی اجازت دیتا ہے۔ ہر کام کے لیے ایک مقررہ وقت مختص کیا جاتا ہے۔ وقت کی اس تقسیم کو ٹائم شیئرنگ کہتے ہیں۔ ایک وقت گزرنے یا عمل مکمل ہونے کے بعد پروسیسر مختلف عملوں کے درمیان تیزی سے تبدیل کرتا ہے۔

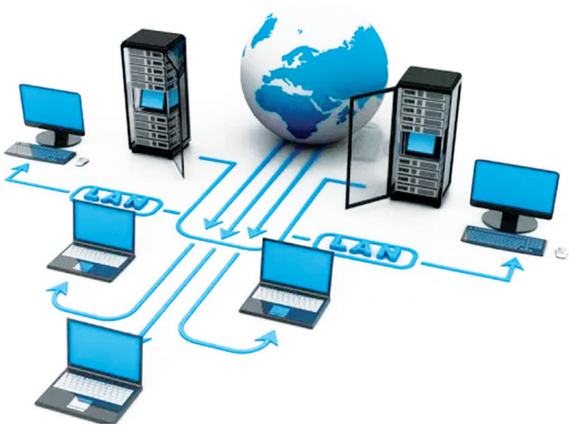
مثال کے طور پر فرض کریں کہ P1، P2، P3 نامی تین عمل ہیں اور ہر عمل کے لیے بالترتیب 30، 40، 50 منٹ مختص کیے گئے ہیں۔ اگر پروسیس P1، 20 منٹ کے اندر مکمل ہو جاتا ہے تو پروسیسر عملدرآمد کے لیے اگلا پروسیس P2 لیتا ہے۔ اگر پروسیس P2، 40 منٹ کے اندر مکمل نہیں ہو سکا، تو موجودہ پروسیس P2 کو روک دیا جائے گا اور اگلے پروسیس P3 پر تبدیل کر دیا جائے گا۔

4.4.9 تقسیم شدہ آپریٹنگ سسٹم (Distributed Operating System)

تقسیم شدہ آپریٹنگ سسٹم کا استعمال مشترکہ ڈیٹا اور فائلوں تک رسائی کے لیے کیا جاتا ہے جو انٹرنیٹ (Internet)/انٹرانیٹ (Intranet) کا استعمال کرتے ہوئے دنیا بھر میں کسی بھی مشین میں پائی جاتی ہیں۔ صارفین اس طرح رسائی حاصل کر سکتے ہیں جیسے یہ ان کے اپنے کمپیوٹر پر دستیاب ہو۔

تقسیم شدہ آپریٹنگ سسٹم کے فوائد درج ذیل ہیں:

- ایک مقام پر صارف نیٹ ورک پر دوسرے مقام پر دستیاب تمام وسائل کا استعمال کر سکتا ہے۔
- نیٹ ورک میں کمپیوٹر کے بہت سے وسائل آسانی سے شامل کیے جا سکتے ہیں۔
- گاہکوں اور گاہکوں کے ساتھ بات چیت کو بہتر بناتا ہے۔
- میزبان کمپیوٹر پر بوجھ کو کم کرتا ہے۔



تصویر 4.4 تقسیم شدہ آپریٹنگ سسٹمس

11th_Comp Science_UM_Ch 4.indd 54 18-04-2023 13:19:50

4.5 پرامینینٹ آپریٹنگ سسٹمس

پرامینینٹ OS مندرجہ ذیل ہیں:

- یونکس (UNIX)
- مائیکروسافٹ ونڈوز (Micro Soft Windows)
- لینکس (Linux)
- iOS
- اینڈرائیڈ(Android)

جدید آپریٹنگ سسٹم گرافیکل یوزر انٹرفیس (GUI) کااستعمال کرتے ہیں ۔ ایک GUI آئکنس (Icons)، بٹنس (Buttons)، مینوس (Menus) اور تمام پر گرافکس اور ٹیسٹ عناصر کے مجموعہ کا استعمال کرتے ہوئے آپ کے ماؤس کو کلک کرنے کی اجازت دیتا ہے تاکہ سب کچھ واضح طور پر اسکرین پر ظاہر ہوجائے ۔

طالب علم کی سرگرمی

سرگرمی 1: آپریٹنگ سسٹم کے لوگو (Logo) اور درست تفصیل کے درمیان ایک لکیر کھینچیں۔

	کمانڈ لائن آپریٹنگ سسٹم اوپن سورس سافٹ ویئر ڈویلپمنٹ اور فری آپریٹنگ سسٹم کی ایک مثال ہے۔
	موبائل فون ٹیکنالوجی کے لیے ایک مقبول آپریٹنگ سسٹم جو ایپل کی مصنوعات سے منسلک نہیں ہے۔
	ایپل کمپیوٹرز کے ساتھ استعمال کیا جاتا ہے اور کلاؤڈ کمپیوٹنگ کے ساتھ اچھا کام کرتا ہے۔
	ایپل آئی فون کے لیے استعمال ہونے کے لیے ڈیزائن کیا گیا ہے۔
	ایک آپریٹنگ سسٹم ہے جو یونیورسٹیوں، کمپنیوں، بڑے اداروں وغیرہ میں بہت مشہور ہے۔
	پرسنل کمپیوٹرز کے لیے سب سے مشہور GUI آپریٹنگ سسٹم۔

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں۔

1) ایک آپریٹنگ سسٹم
   A) ایپلیکیشن سافٹ ویئر
   B) ہارڈ ویئر
   C) سسٹم سافٹ ویئر
   D) کومپونینٹ

2) آپریٹنگ سسٹمز کے استعمال کی شناخت کریں۔
   A) انسان اور کمپیوٹر کے درمیان آسان تعامل
   B) ان پٹ اور آؤٹ پٹ ڈیوائسس کو کنٹرول کرنا
   C) مین میموری کے استعمال کا انتظام کرنا
   D) مندرجہ بالا تمام

3) مندرجہ ذیل میں سے کس آپریٹنگ سسٹم کا کام نہیں ہے ؟
   A) پروسیس مینجمنٹ
   B) میموری مینجمنٹ
   C) سیکیورٹی مینجمنٹ
   D) کمپلائر انویرونمینٹ

4) مندرجہ ذیل میں سے کون سا OS تجارتی طور پر لائسنس یافتہ آپریٹنگ سسٹم ہے؟

   Windows (A UBUNTU (B
   FEDORA (C REDHAT (D

5) درج ذیل میں سے کون سا آپریٹنگ سسٹم موبائل ڈیوائسس کو سپورٹ کرتا ہے؟

   Windows 7 (A Linux (B
   BOSS (C iOS (D

6) فائل مینجمنٹ انتظام کرتی ہے
   A) فائلز
   B) فولڈرز
   C) ڈائکٹری سسٹم
   D) اوپر کے تمام

11th_Comp Science_UM_Ch 4.indd 55 18-04-2023 13:19:51

7) انٹرایکٹو آپریٹنگ سسٹم فراہم کرتا ہے۔

A) گرافکس یوزر انٹرفیس (GUI)

B) ڈیٹا کی تقسیم

C) سیکورٹی مینجمنٹ

D) ریل ٹائم پروسیسنگ

8) سنگل ٹاسک آپریٹنگ سسٹم کی ایک مثال ہے۔

A) Linux

B) Windows

C) MS-DOS

D) Unix

9) لینکس کے ذریعہ استعمال ہونے والا فائل مینجمنٹ سسٹم ہے۔

A) ext2

B) NTFS

C) FAT

D) NFTS

سیکشن - B

بہت مختصر جواب دیں۔

1) آپریٹنگ سسٹم کے دو استعمالات کی فہرست بنائیں؟

2) ملٹی یوزر آپریٹنگ سسٹم کیا ہے؟

3) GUI کیا ہے؟

4) آپریٹنگ سسٹم میں سیکیورٹی مینجمنٹ کی کون سی خصوصیات دستیاب ہیں؟

5) ملٹی پروسیسنگ کیا ہے؟

6) کمپیوٹر میں استعمال ہونے والے مختلف آپریٹنگ سسٹم کیا ہیں؟

سیکشن - C

مختصر جواب دیں۔

1) ٹائم شیئرنگ کی خصوصیات کے فوائد اور نقصانات کیا ہیں؟

2) آپریٹنگ سسٹم کی اہم خصوصیات کی فہرست بنائیں

3) ملٹی پروسیسنگ پر ایک نوٹ لکھیں۔

سیکشن - D

تفصیل سے بیان کریں۔

1) تقسیم شدہ آپریٹنگ سسٹم کے تصور کے ساتھ اس کے فوائد کی وضاحت کریں۔

2) آپریٹنگ سسٹم کے لیے یوزر انٹرفیس بناتے وقت نوٹ کیے جانے والے نکات کی فہرست بنائیں۔

3) آپریٹنگ سسٹم میں پروسیس مینیجمنٹ الگورتھم کی وضاحت کریں۔

حوالہ جات

1) Silberschatz, galvin gagne, Operating System concepts – john wiley&sons,inc
2) Andrew s. Tanenbaum, modern Operating Systems – pearson publication
3) Andrew s. Tanenbaum , Operating Systems design and implementation, prentice hall publication
4) Tom anderson, Operating Systems: principles and practice, recursive books
5) Thomas w. Doeppner, Operating Systems in depth: design and programming, john wiley & sons, inc

11th_Comp Science_UM_Ch 4.indd 56 18-04-2023 13:19:52

باب 5

ونڈوز آپریٹنگ سسٹم کے ساتھ کام کرنا

سیکھنے کے مقاصد

اس باب میں تصورات کو سیکھنے کے بعد طلباء اس قابل ہوجائیں گے :

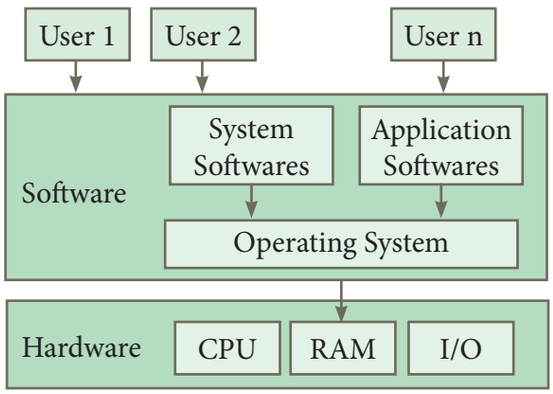
- آپریٹنگ سسٹم کے تصورات کو جاننا۔
- ونڈوز آپریٹنگ سسٹم کے ورژن کو جاننا۔
- ڈیسک ٹاپ اور ونڈو کے عناصر جیسے تصورات کو جاننا۔
- دستاویز ونڈو کو دریافت کرنا۔
- مختلف قسم کے آئکن کا موازنہ کرنا۔
- ونڈوز ڈائرکٹری کے ڈھانچے کو دریافت کرنا۔
- مخصوص ڈرائیوز میں فائلز اور فولڈرز بنانے کی مشق کرنا۔
- فائلوں اور فولڈرز کا نظم کرنا۔
- کمپیوٹر کو شروع کرنے اور بند کرنے کا طریقہ کار جاننا۔

5.1 آپریٹنگ سسٹم کا تعارف

ایک آپریٹنگ سسٹم (OS) سافٹ ویئر ہے (تصویر 5.1) جو ہارڈویئر کو دوسرے سافٹ ویئر کے ساتھ بات چیت اور کام کرنے کے قابل بناتا ہے۔ یہ صارف اور ہارڈ ویئر کے درمیان ایک انٹرفیس کے طور پر بھی کام کرتا ہے اور کمپیوٹر کے مجموعی عمل کو کنٹرول کرتا ہے۔

آپریٹنگ سسٹم کے کچھ اہم افعال درج ذیل ہیں جیسا کہ پچھلے باب میں زیر بحث آیا:

- میموری مینجمنٹ (Memory Management)
- پروسیس مینجمنٹ (Process Management)
- ڈیوائس مینجمنٹ (Process Management)
- فائل مینجمنٹ (File Management)
- سیکیورٹی مینجمنٹ (Security Management)
- سسٹم کی مجموعی کارکردگی کو کنٹرول کرنا۔



تصویر 5.1 آپریٹنگ سسٹم کا جائزہ

5.2 ونڈوز آپریٹنگ سسٹم کا تعارف

ہر کمپیوٹر کو کام کرنے کے لیے آپریٹنگ سسٹم کی ضرورت ہوتی ہے۔ مائیکروسافٹ ونڈوز سب سے مشہور گرافیکل یوزر انٹرفیس (GUI) میں سے ایک ہے۔ونڈوز میں ایک سے زیادہ ایپلی کیشنز ایک ساتھ کام کو انجام دے سکتے ہیں، اور اسے "ملٹی ٹاسکنگ" (Multitasking) کے نام سے جانا جاتا ہے۔

ونڈوز آپریٹنگ سسٹم کی بورڈ (Keyboard) اور ماؤس (Mouse) دونوں کو ان پٹ ڈیوائسز کے طور پر استعمال کرتا ہے۔ ماؤس کو ونڈوز کے آئیکنز پر کلک کرکے اس کے ساتھ بات چیت کرنے کے لیے استعمال کیا جاتا ہے۔ کی بورڈ کا استعمال حروف تہجی، ہندسوں اور خصوصی حروف کو داخل کرنے کے لیے کیا جاتا ہے۔

11th_Comp Science_UM_Ch 5.indd 57 18-04-2023 14:05:06

ونڈوز آپریٹنگ سسٹم کے کچھ کام یہ ہیں:

- ایپلی کیشنز (پروگرام) کی کمپیوٹر میں رسائی (ورڈ پروسیسنگ، گیمز، اسپریڈ شیٹس، کیلکولیٹر وغیرہ)۔
- کمپیوٹر پر کوئی نیا پروگرام ڈالنا۔
- ہارڈ ویئر کا انتظام جیسے پرنٹرز، سکینر، ماؤس، ڈیجیٹل کیمرے وغیرہ۔
- فائل مینجمنٹ کی سرگرمیاں (مثال کے طور پر فائلوں اور فولڈرز کو بنانا، ترمیم کرنا، محفوظ کرنا، حذف کرنا)۔
- کمپیوٹر کی ترتیبات کو تبدیل کرنا۔ جیسے کہ رنگ سکیم، اپنے مانیٹر کے اسکرین سیور وغیرہ۔

ٹیبل 5.1 کے حوالے سے، آئیے ہم ونڈوز آپریٹنگ سسٹم کے ورژنس (Versions) کو دیکھتے ہیں۔

5.3 ونڈوز کے مختلف ورژنس

ورژنس	لوگو	سال	مخصوص خصوصیات
Windows 1.x	MICROSOFT WINDOWS	1985	• 16 بٹ پروسیسر میں GUI کا تعارف • ماؤس کو ایک ان پٹ ڈیوائس کے طور پر متعارف کرایا گیا تھا۔
Windows 2.x	MICROSOFT WINDOWS	1987	• ونڈوز کو کم سے کم یا زیادہ سے زیادہ کرنے کی حمایت کرتا ہے۔ • کنٹرول پینل کی خصوصیت مختلف نظام کی ترتیبات اور حسب ضرورت کے اختیارات کے ساتھ متعارف کرائی گئی تھی۔
Windows 3.x	MICROSOFT WINDOWS	1992	• ملٹی ٹاسکنگ کا تصور متعارف کرایا۔ • 256 رنگوں کی حمایت کرتا ہے جس کی وجہ سے انٹرفیس میں زیادہ سے زیادہ جدید رنگین تصاویر نظر آتے ہیں۔
Windows NT	MICROSOFT WINDOWS	1993	• اسٹارٹ بٹن، ٹاسک بار، ونڈوز ایکسپلورر اور اسٹارٹ مینو کو متعارف کرایا گیا۔ • 32 بٹ پروسیسر متعارف کرایا اور ملٹی ٹاسکنگ پر زیادہ توجہ دی گئی۔
Windows 95	MICROSOFT WINDOWS	1995	• نیٹ ورک میں سرورز (Servers) کے طور پر کام کرنے کے لیے ڈیزائن کیا گیا ہے۔
Windows 98	MICROSOFT WINDOWS	1998	• آپریٹنگ سسٹم کے ساتھ ویب براؤزر (انٹرنیٹ ایکسپلورر) کا انضمام۔ • ونڈوز پر مبنی گیمز میں بہتری کے ساتھ ہی DOS گیمنگ غائب ہونا شروع ہو گئی۔ • پلگ اینڈ پلے فیچر متعارف کرایا گیا۔
Windows Me	Microsoft Windows Me Millennium Edition	2000	• اس نے خودکار نظام کی تشخیص اور بحالی کے اوزار کو متعارف کرایا

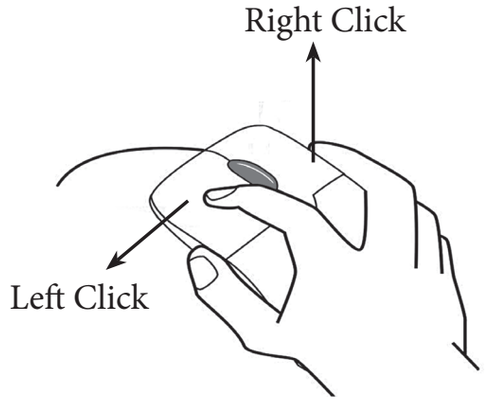
11th_Comp Science_UM_Ch 5.indd 58 18-04-2023 14:05:09

Windows 2000		2000	• کاروباری ڈیسک ٹاپ اور لیپ ٹاپ سسٹمز کے لیے آپریٹنگ سسٹم کے طور پر کام کیا جاتا ہے۔ • ونڈوز 2000 کے چار ورژن جاری کیے گئے: پروفیشنل (کاروباری ڈیسک ٹاپ اور لیپ ٹاپ سسٹمز کے لیے)، سرور (ویب سرور اور آفس سرور دونوں)، ایڈوانسڈ سرور (لائن آف بزنس ایپلی کیشنز کے لیے) اور ڈیٹا سینٹر سرور۔(ہائی ٹریفک کمپیوٹر نیٹ ورکس کے لیے)۔
Windows XP		2001	• 64 بٹ پروسیسر متعارف کرایا گیا۔ • تھیمز کے ساتھ ونڈوز کی ظاہری تصویر کو بہتر بنایا اور ایک مستحکم ورژن پیش کیا گیا۔
Windows Vista		2006	• ونڈوز کی تصویر و صورت کو اپ ڈیٹ کیا گیا۔
Windows 7		2009	• بوٹنگ ٹائم کو بہتر کیا گیا، نیا یوزر انٹرفیس متعارف کرایا جیسے ایرو پیک (Aero Peek)، پروگراموں کو ٹاسک بار میں پن کرنا، ہینڈ رائٹنگ کی شناخت وغیرہ اور انٹرنیٹ ایکسپلورر 8۔
Windows 8		2012	• ونڈوز 8 ونڈوز کے پچھلے ورژنز سے تیز ہے۔ • اسٹارٹ بٹن ہٹا دیا گیا تھا۔ • Windows 8 ملٹی کور پروسیسنگ، سالڈ اسٹیٹ ڈرائیوز (SSD)، ٹچ اسکرینز اور دیگر متبادل ان پٹ طریقوں کا بہتر فائدہ اٹھاتا ہے۔ • موبائل اور کمپیوٹر کے لیے مشترکہ پلیٹ فارم کے طور پر کام کیا جاتا ہے۔
Windows 10		2015	• اسٹارٹ بٹن دوبارہ شامل کیا گیا۔ • ایک سے زیادہ ڈیسک ٹاپ۔ • ایپ کی اطلاعات اور فوری کارروائیوں کے لیے مرکزی اطلاعاتی مرکز۔ • کورٹاناوائس (Cortana Voice) نے پرسنل اسسٹنٹ کو جاری کیا۔

ٹیبل 5.1 ونڈوز آپریٹنگ سسٹم کے ورژن۔

5.4 ماؤس کو سنبھالنا

ونڈو آپریٹنگ سسٹم سیکھنے سے پہلے آپ کو ماؤس اور اس کے افعال کے بارے میں مزید جاننا چاہیے۔



تصویر 5.2 ماؤس کے اعمال

ماؤس کے اعمال درج ذیل ہیں:

عمل	رد عمل
کسی شے کی طرف اشارہ کرنا	ماؤس پوائنٹر کو آئٹم پر منتقل کریں۔
کلک کرنا	اسکرین پر موجود آئٹم کی طرف اشارہ کریں، بائیں ماؤس کے بٹن کو دبائیں اور چھوڑ دیں۔
دایاں کلک	اسکرین پر موجود آئٹم کی طرف اشارہ کریں، ماؤس کے دائیں بٹن کو دبائیں اور چھوڑ دیں۔ دائیں ماؤس پر کلک کریں، بٹن مختلف اختیارات کے ساتھ ایک پاپ اپ مینو دکھاتا ہے۔
ڈبل کلک	اسکرین پر آئٹم کی طرف اشارہ کریں، تیزی سے بائیں ماؤس سے دو بار ہ بٹن دبائیں۔

11th_Comp Science_UM_Ch 5.indd 59 18-04-2023 14:05:10

ڈریگ اور ڈراپ	کسی آئٹم کی طرف اشارہ کریں پھر بائیں ماؤس کے بٹن کو دبائے رکھیں جب آپ پوائنٹر پریس کو حرکت دیتے ہیں اور آپ مطلوبہ پوزیشن پر پہنچ جاتے ہیں تو ماؤس بٹن کو چھوڑ دیں۔

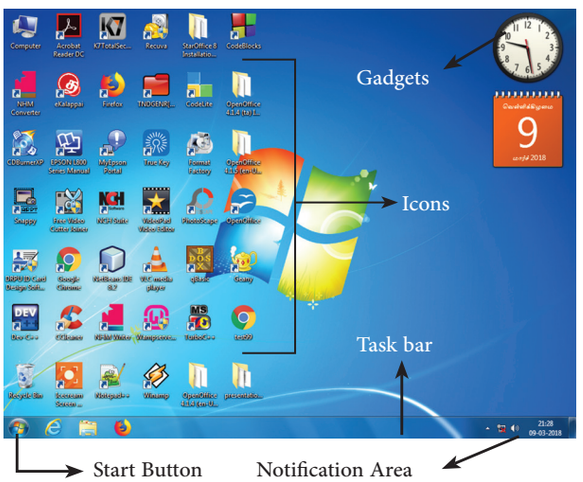
5.5 ونڈوز ڈیسک ٹاپ

ونڈوز کی اوپننگ اسکرین کو "ڈیسک ٹاپ" (Desktop) کہا جاتا ہے۔

آپ کے کمپیوٹر کا ڈیسک ٹاپ تصویر 5.3 میں نظر آنے والے سے مختلف نظر آ سکتا ہے۔

اس کی وجہ یہ ہے کہ ونڈوز آپ کو ڈیسک ٹاپ کی ظاہری تصویر کو تبدیل کرنے کی اجازت دیتا ہے۔

تصویر 5.3 میں، ڈیسک ٹاپ اسٹارٹ بٹن، ٹاسک بار، نوٹیفکیشن ایریا اور تاریخ اور وقت دکھاتا ہے۔



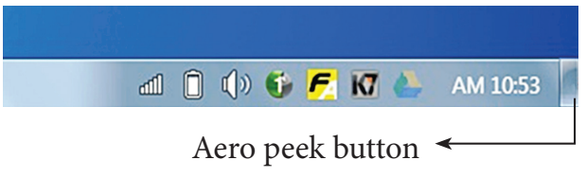
تصویر 5.3 مائیکروسافٹ ونڈوز 7 ڈیسک ٹاپ

5.5.1 آئکنس (Icons)

آئیکن ایک گرافک علامت ہے جو ونڈو کے عناصر جیسے فائلز، فولڈرز، شارٹ کٹس وغیرہ کی نمائندگی کرتی ہے، آئیکنز GUI پر مبنی ایپلی کیشنز میں اہم کردار ادا کرتے ہیں۔

5.5.1.1 معیاری آئکن (Standard Icons)

ونڈوز OS کو انسٹال کرتے وقت جو آئیکنس ڈیسک ٹاپ پر بطور ڈیفالٹ دستیاب ہوتے ہیں انہیں معیاری آئیکنس کہا جاتا ہے۔ تمام Windows OS میں دستیاب معیاری آئکنس، مائی کمپیوٹر (My Computer) ڈاکومنٹس (Documents) اور ری سائکل بین (Recycle Bin) ہیں۔



تصویر 5.4 ایرو پیک بٹن

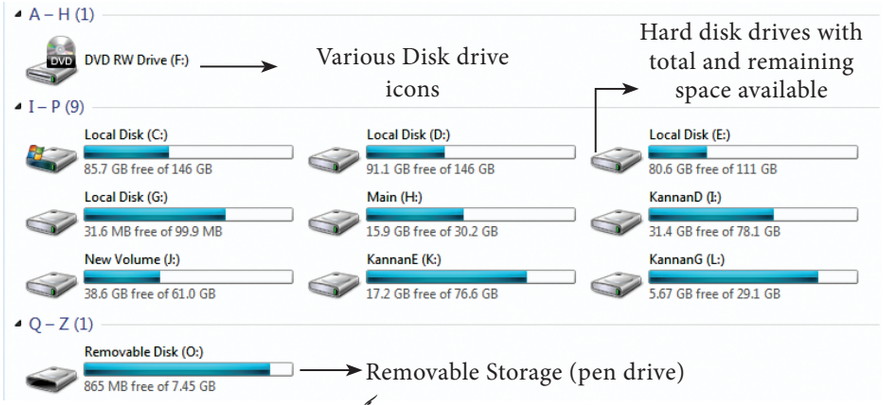
5.5.1.2 شارٹ کٹ آئکنس (Shortcut Icons)

شارٹ کٹ آئکنس کسی بھی ایپلیکیشن یا فائل یا فولڈر کے لیے بنائی جا سکتی ہیں۔ آئیکن پر ڈبل کلک کرنے سے متعلقہ ایپلیکیشن یا فائل یا فولڈر کھل جائے گا۔

(تصویر 5.5)



تصویر 5.5 آئکنس کے اقسام



تصویر 5.6 ڈسک ڈرائیو آئکنس

11th_Comp Science_UM_Ch 5.indd 60 18-04-2023 14:05:11

5.5.1.3 ڈسک ڈرائیو آئکنس (Disk drive icons)

ڈسک ڈرائیو آئکنس گرافک طور پر پانچ ڈسک ڈرائیو اختیارات کی نمائندگی کرتی ہیں۔ (i) ہارڈ ڈسک (ii) CD- ROM/ DVD ڈرائیو (iii) پین ڈرائیو (iv) دیگر ہٹائی جانے والی اسٹوریج جیسے موبائل، سمارٹ فون، ٹیبلٹ وغیرہ، (v) نیٹ ورک ڈرائیوز اگر آپ کا سسٹم دوسرے سسٹم سے منسلک ہے۔

کیا آپ جانتے ہیں؟

آپ Winkey +D یا ایرو پیک کو دبا کر کسی بھی وقت ڈیسک ٹاپ پر جا سکتے ہیں۔ ٹاسک بار میں ایرو پیک کہاں ہے یہ جاننے کے لیے آپ تصویر 5.4 کو دیکھ سکتے ہیں

5.6 ونڈو (The Window)

ونڈو کسی درخواست یا دستاویز میں ایک عام مستطیل مقام ہے۔ یہ اسکرین پر ایک ایسا مقام ہے جو کسی مخصوص پروگرام کے لیے معلومات دکھاتا ہے۔

5.7 ایپلی کیشن ونڈو (Application Window)

یہ کمپیوٹر اسکرین پر ایک ایسا مقام ہے جس میں متعین حدود ہیں، اور جس کے اندر معلومات ظاہر ہوتی ہیں۔ ایسی ونڈوز کا سائز تبدیل کیا جا سکتا ہے، بڑا کیا جاسکتا ہے، چھوٹا کیا جا سکتا ہے، برابر رکھا جا سکتا ہے اور ایک کے اوپر ایک رکھا جا سکتا ہے، وغیرہ۔

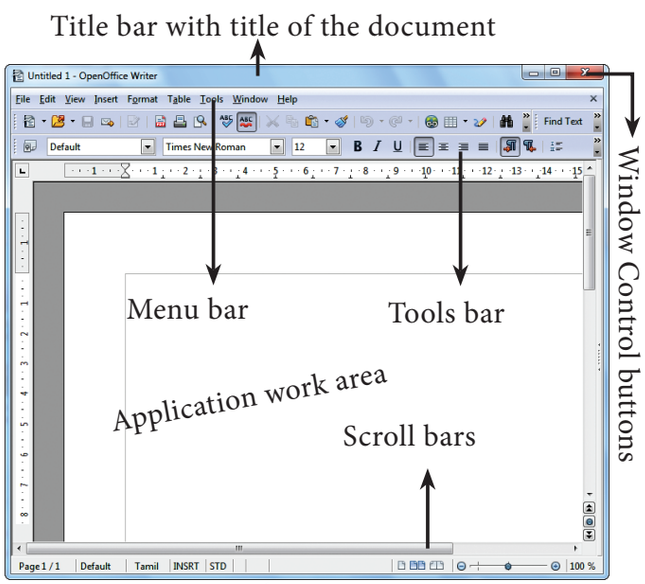
ایک ایپلیکیشن ونڈو میں ایک کھلی ایپلیکیشن ہوتی ہے یعنی موجودہ ایپلیکیشن جیسے ورڈ یا پینٹ۔ جب دو یا دو سے زیادہ ونڈوز کھولی جاتی ہیں تو ان میں سے صرف ایک فعال ہوتی ہے اور باقی غیر فعال ہوتی ہیں۔ تصویر 5.6 اور 5.7 اوپن آفس رائٹر کی ایپلیکیشن ونڈو اور ڈیسک ٹاپ میں کھلی ہوئی ایک سے زیادہ ونڈوز کی ظاہری تصویر کو ظاہر کرتے ہیں۔

5.8 ڈاکومنٹ ونڈو (Document Window)

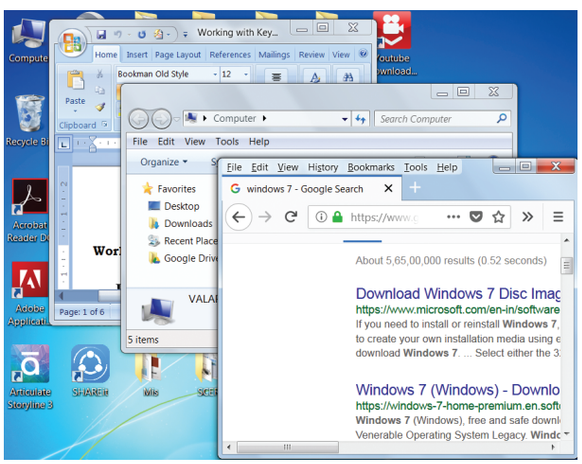
ایک دستاویز ونڈو اسکرین کا ایک حصہ ہے جو کسی دستاویز کے مواد کو ظاہر کرنے کے لیے استعمال ہوتا ہے۔ تصویر 5.8 دستاویز کی ونڈوز کی ایک مثال ہے۔

نوٹ

جب آپ کوئی بھی ایپلی کیشن کھولتے ہیں، جیسے اوپن آفس رائٹر، اوپن آفس امپریس یا اوپن آفس کیلک وغیرہ، آپ کو اسکرین پر دو ونڈوز نظر آئیں گی۔ بڑی ونڈو کو ایپلی کیشن ونڈو (Application Window) کہا جاتا ہے۔ یہ ونڈو صارف کو ایپلیکیشن پروگرام کے ساتھ بات چیت کرنے میں مدد کرتی ہے۔ چھوٹی ونڈو، جو کہ ایپلی کیشن ونڈو کے اندر ہوتی ہے، اسے ڈاکومنٹ ونڈو(Document Window) کہا جاتا ہے۔ یہ ونڈو ٹائپنگ، ایڈیٹنگ، ڈرائنگ، اور ٹیکسٹ اور گرافکس کو فارمیٹ کرنے کے لیے استعمال ہوتی ہے۔

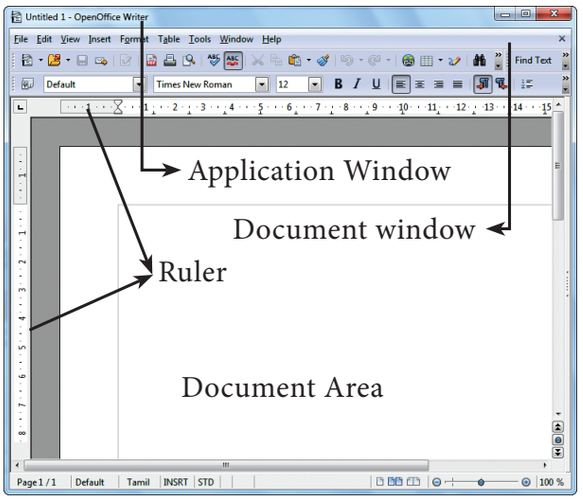


تصویر 5.6 ایپلی کیشن ونڈو



تصویر 5.7 ڈیسک ٹاپ میں ایک سے زیادہ ونڈوز

11th_Comp Science_UM_Ch 5.indd 61 18-04-2023 14:05:11

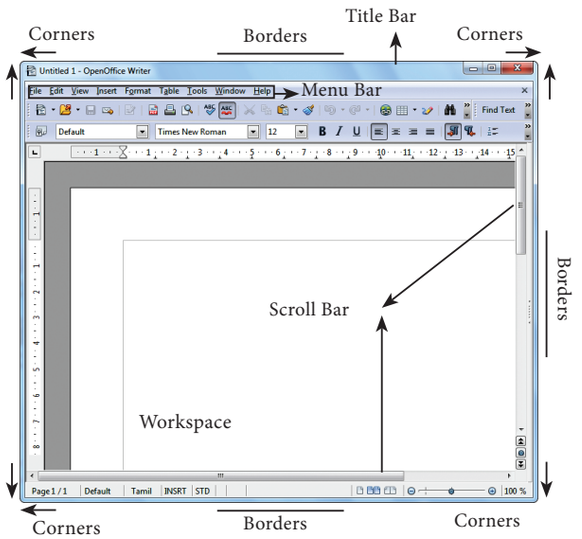


تصویر 5.8 ڈاکومنٹ ونڈو

5.9 ونڈو کے عناصر (Elements of a window)

تصویر 5.9 ونڈو کے عناصر کو سمجھنے میں مدد کرتا ہے۔

5.9.1 ٹائٹل بار (Title Bar) – ٹائٹل بار ایپلیکیشن کا نام اور کھولی گئی دستاویز کا نام ظاہر کرے گا۔ اس میں منی مائز (Minimize)، میکسی مائز (maximize) اور کلوز (close) بٹن بھی شامل ہوں گے۔

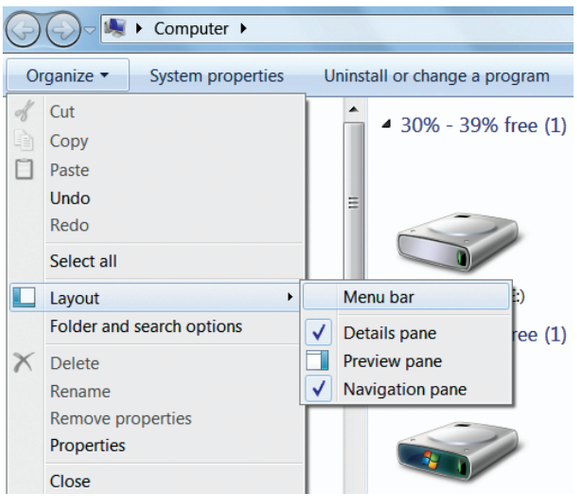


تصویر 5.9 ونڈوکے عناصر

5.9.2 مینو بار (Menu Bar)

مینو بار ٹائٹل بار کے نیچے نظر آتا ہے۔ مینو بار میں موجود مینوس تک Alt بٹن دبا کر اور مینو کے ٹائٹل میں انڈر لائن والے لفظ تک رسائی حاصل کی جا سکتی ہے۔ مزید برآں، Alt یا F10 دبانے سے مینو بار کے پہلے مینو پر توجہ مرکوز ہو جاتی ہے۔

ونڈوز 7 میں، مینو بار کی غیر موجودگی میں، آرگنائز (organise) پر کلک کریں اور ڈراپ ڈاؤن مینو سے لے آؤٹ (Layout) آپشن پر کلک کریں اور اس فہرست سے مطلوبہ آئٹم کو منتخب کریں۔



تصویر 5.10 مینو بار کو ظاہر کرنے کے لیے

تصویر 5.10 یہ سمجھنے میں مدد کرتا ہے کہ مینو بار کو اس کی غیر موجودگی میں کیسے نمایاں بنایا جائے۔

5.9.3 ورک اسپیس (The Workspace)

ورک اسپیس دستاویز کی ونڈو میں آپ کے دستاویز کا متن درج کرنے یا ٹائپ کرنے کا مقام ہے۔ تصویر 5.10 دستاویز کی ونڈو میں ورک اسپیس کا مقام دکھاتا ہے۔

5.9.4 اسکرول بارز (Scroll bars) – اسکرول بارز ، ورک اسپیس کو افقی یا عمودی طور پر اسکرول کرنے کے لیے استعمال کیا جاتا ہے۔ تصویر 5.9 اسکرول بارز کو دکھاتا ہے۔

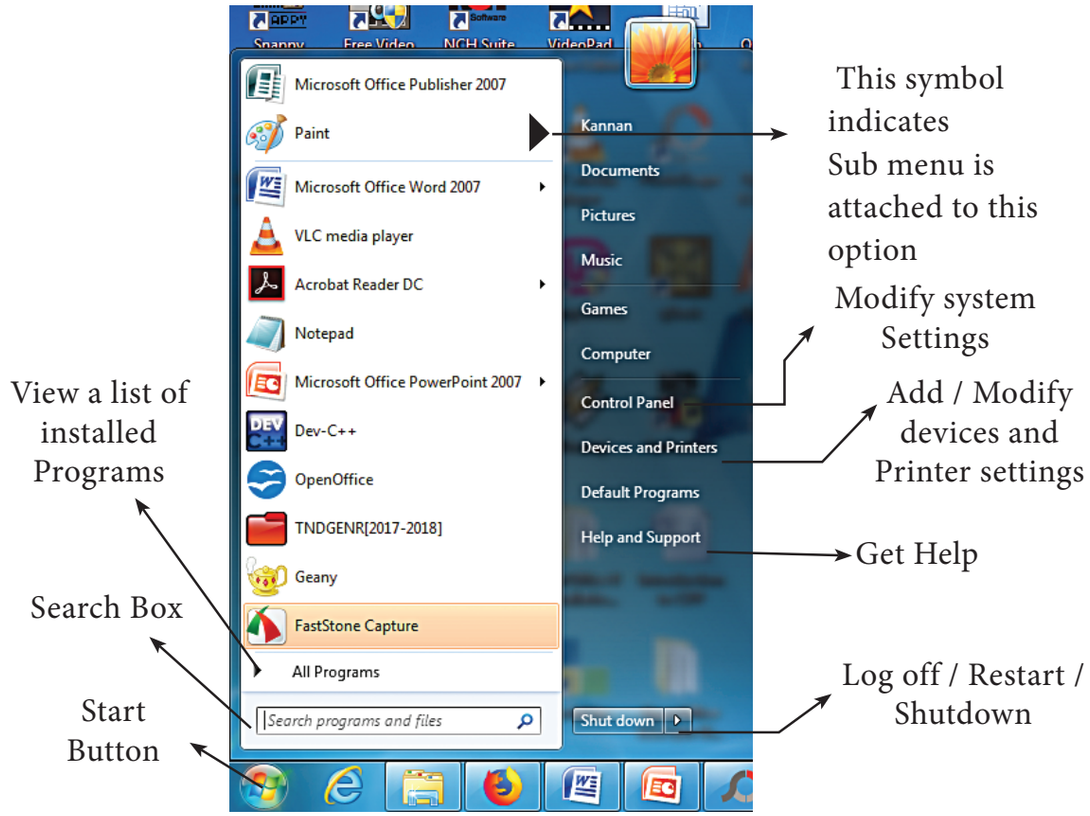
5.9.5 کارنراور بورڈرس (Corners and borders)

ونڈوکے کارنر اور بارڈرس، ونڈورز کو گھسیٹنے اور سائز تبدیل کرنے میں مدد کرتے ہیں۔ جب بارڈر یا کارنر پر رکھا جائے تو ماؤس پوائنٹر ڈبل ہیڈیڈ ایرو (Double Headed Arrow) میں بدل جاتا ہے۔ بارڈر یا کارنر کو اس سمت میں گھسیٹیں جس کی طرف ڈبل ہیڈیڈ ایرو مطلوبہ سائز تک اشارہ کرتا ہے، جیسا کہ تصویر 5.9 میں دکھایا گیا ہے۔ کارنرس کو ترچھی اسکرین پر گھسیٹ کر ونڈوز کا سائز تبدیل کیا جا سکتا ہے۔

11th_Comp Science_UM_Ch 5.indd 62 18-04-2023 14:05:11

5.10.1 اسٹارٹ مینو (Start Menu)

ونڈوز اسکرین کے نچلے بائیں کونے میں اسٹارٹ بٹن ہے۔ جب آپ بٹن پر کلک کریں گے تو اسٹارٹ مینو ظاہر ہوگا۔ اسٹارٹ مینو کا استعمال کرتے ہوئے، آپ کوئی بھی ایپلیکیشن شروع کر سکتے ہیں۔



تصویر 5.11 – اسٹارٹ مینو

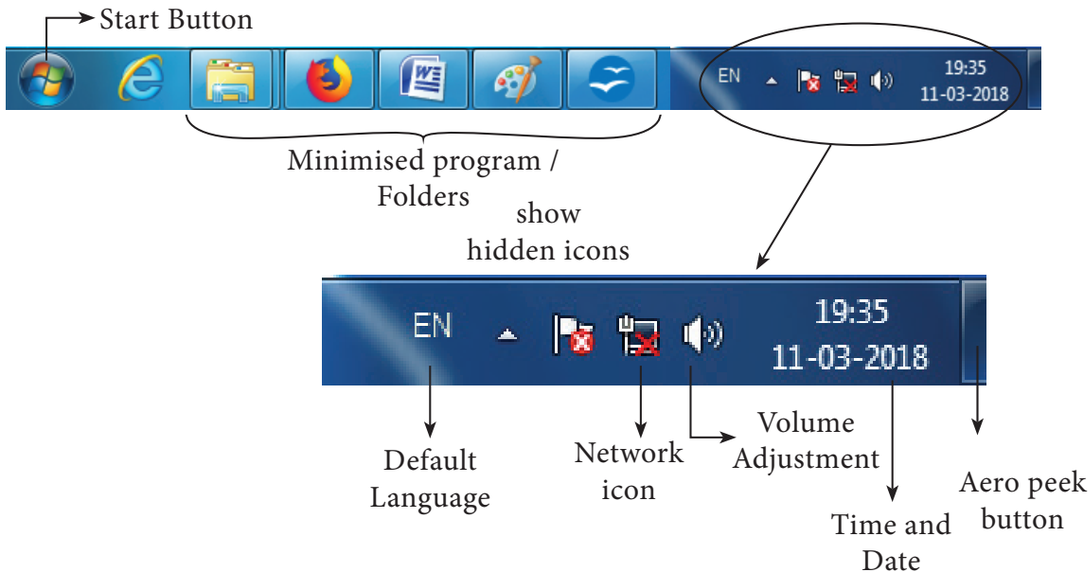
ٹاسک بار (Task Bar)

اسکرین کے نیچے ایک افقی بار (Horizontal bar) ہے جسے ٹاسک بار کہتے ہیں۔ اس بار میں (بائیں سے دائیں) اسٹارٹ بٹن، مختلف پروگراموں کے شارٹ کٹس، منی مائز پروگرامز اور انتہائی دائیں کونے میں آپ سسٹم ٹرے system tray دیکھ سکتے ہیں جس میں والیوم کنٹرول، نیٹ ورک، تاریخ اور وقت وغیرہ شامل ہیں۔ اسٹارٹ بٹن کے آگے فوری لانچ ٹول بار (Quick Launch tool bar) جس میں کثرت سے استعمال ہونے والی ایپلیکیشنز کے لیے کام ہوتا ہے۔

5.10.2 کمپیوٹر آئیکن (Computer Icon)

اس آئیکون پر کلک کرنے سے صارف سسٹم میں نصب ڈسک ڈرائیورز (Disc Drivers) کو دیکھ سکتا ہے۔ ونڈوز ایکس پی XP، وسٹا vista میں اس آئیکن کو ونڈوز 8 اور 10 میں " مائی کمپیوٹر" (My Computer) کہا جاتا ہے، اسے "دس پی سی" (This PC) کہا جاتا ہے۔ کمپیوٹر آئیکن کی فعالیت ونڈوز کے تمام ورژنز میں یکساں رہتی ہے جیسا کہ تصویر 5.13 میں دکھایا گیا ہے

11th_Comp Science_UM_Ch 5.indd 63 18-04-2023 14:05:12



تصویر 5.12 ٹاسک بار

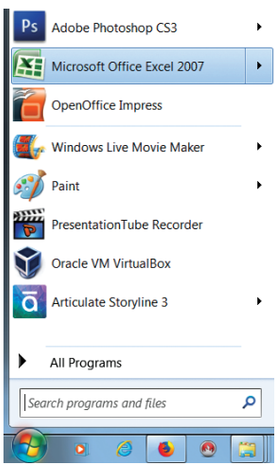


تصویر 5.13 ونڈوز (OS) کے ورژنس میں کمپیوٹر آئیکن

5.10.3 ایپلی کیشنز کو چالو اور بند کرنا۔

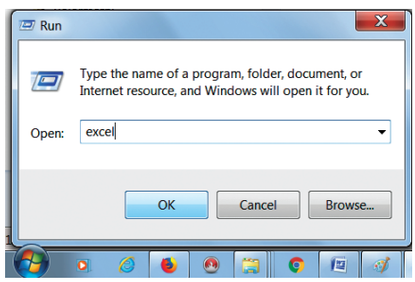
آپ کے کمپیوٹر پر نصب کردہ (Installed) زیادہ تر ایپلیکیشنز اسٹارٹ مینو کے ذریعے دستیاب ہیں۔ سسٹم سیٹ اپ پر منحصر اسٹارٹ مینو میں ایپلیکیشنز مختلف ہوتی ہیں۔ ایپلیکیشن شروع کرنے کے لیے:

1. اسٹارٹ بٹن پر کلک کریں اور پھر آل پروگرامز کی طرف اشارہ کریں۔ پروگرام کا مینو ظاہر ہوتا ہے۔ (تصویر5.14)
2. اس گروپ کی طرف اشارہ کریں جس میں وہ ایپلیکیشن ہے جسے آپ شروع کرنا چاہتے ہیں، اور ایپلیکیشن کے نام پر کلک کریں۔



تصویر 5.14 اسٹارٹ مینو کا استعمال کرتے ہوئے ایپلی کیشن چالو کرنا

3. آپ سٹارٹ مینو پر رن (Run) پر کلک کر کے اور ایپلیکیشن کا نام لکھ کر بھی کھول سکتے ہیں۔ (تصویر 5.15)



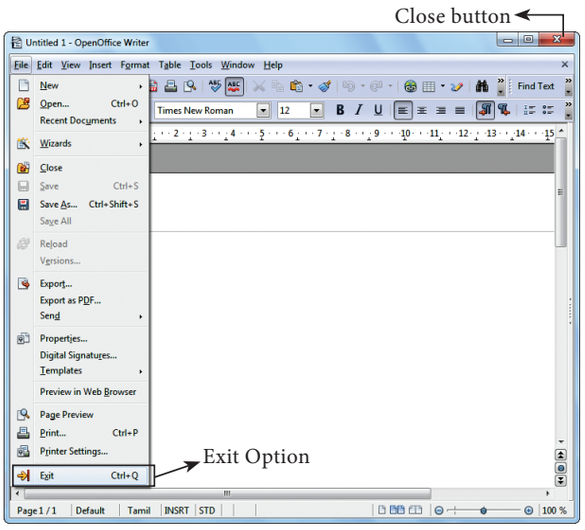
تصویر 5.15 رن (Run) کا استعمال کرتے ہوئے ایک پروگرام چالو کرنا۔

4. کسی ایپلی کیشن کو بند کرنے کے لیے، ایپلی کیشن ونڈو کے اوپر دائیں کونے میں Close بٹن پر کلک کریں۔ (تصویر 5.16)

ورکشاپ

1. ◆ اسٹارٹ مینو اور رن آپشن کا استعمال کرتے ہوئے ایپلیکیشن ورڈ پیڈ کو چالو کریں۔
   ◆ فائل مینو کا استعمال کرتے ہوئے ورڈ پیڈ ایپلیکیشن کو بند کریں۔

11th_Comp Science_UM_Ch 5.indd 64 18-04-2023 14:05:12



تصویر 5.16 Close بٹن اور Exit آپشن کا استعمال کرتے ہوئے ایپلیکیشن کو بند کرنا

5. آپ ونڈوز 7 میں فائل ← ایگزٹ اور فائل ← کلوز آپشن پر کلک کرکے بھی کسی ایپلیکیشن کو بند کرسکتے ہیں۔ (تصویر 5.16)

5.11 فائلوں اور فولڈرز کا انتظام

ونڈوز 7 (Windows 7) میں، آپ اپنے دستاویزات اور پروگراموں کو فائلوں اور فولڈرز کی تصویر میں ترتیب دے سکتے ہیں۔ آپ فائلوں اور فولڈرز کو منتقل، کاپی، نام تبدیل، حذف اور تلاش کر سکتے ہیں۔

5.11.1 فائلیں اور فولڈرس بنانا

5.11.1.1 فولڈرس بنانا

آپ اپنی فائلوں کو ہارڈ ڈسک پر یا دیگر آلات میں کئی جگہوں پر محفوظ کر سکتے ہیں۔ اپنی فائلوں کو بہتر طریقے سے ترتیب دینے کے لیے، آپ انہیں فولڈرس میں محفوظ کر سکتے ہیں۔

دو طریقے ہیں جن سے آپ نیا فولڈر بنا سکتے ہیں:

طریقہ I:

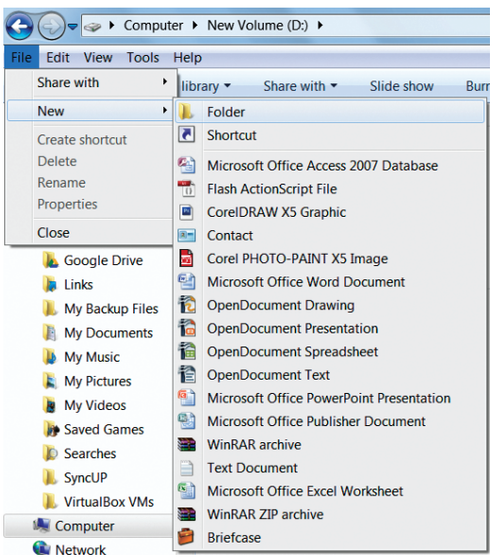
مرحلہ 1: کمپیوٹر آئیکن کھولیں۔

مرحلہ 2: کوئی بھی ڈرائیو کھولیں جہاں آپ نیا فولڈر بنانا چاہتے ہیں۔ (مثال کے طور پر D منتخب کریں:)

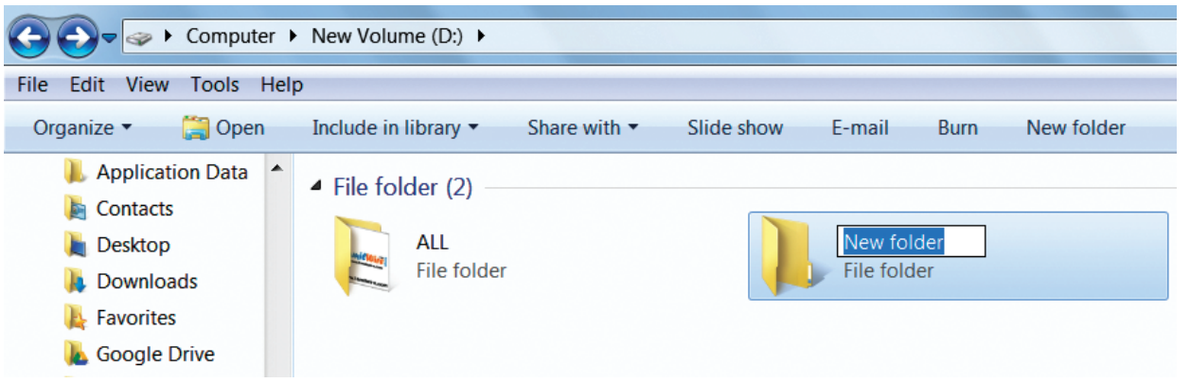
مرحلہ 3: فائل ← نیو ← فولڈر پر کلک کریں۔

مرحلہ 4: پہلے سے طے شدہ نام "New Folder" کے ساتھ ایک نیا فولڈر بنایا گیا ہے۔ (تصویر 5.19)

مرحلہ 5: فولڈر کا نام ٹائپ کریں اور انٹر بٹن دبائیں۔ (تصویر 5.20 میں "ٹیسٹ فولڈر" کے نام سے نئے بنائے گئے فولڈر کو دکھایا گیا ہے)۔

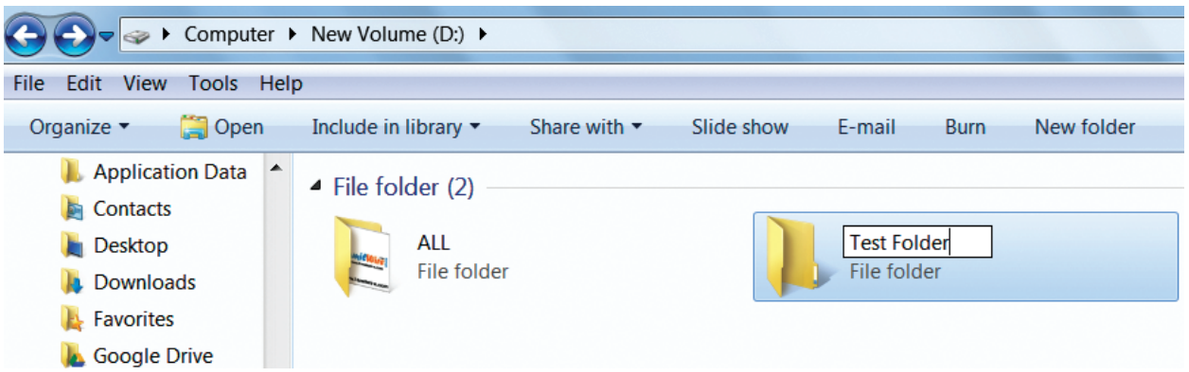


تصویر 5.17 فائل مینو کا استعمال کرتے ہوئے فولڈر بنانا



تصویر 5.18 ڈیفالٹ نام کے ساتھ نیا فولڈر بنایا گیا۔

11th_Comp Science_UM_Ch 5.indd 65 18-04-2023 14:05:12



تصویر 5.19 نئے فولڈر کا نام تبدیل کرنا

طریقہ II:

ڈیسکٹ ٹاپ میں فولڈر بنانے کے لیے:

مرحلہ 1: ڈیسک ٹاپ میں، دایاں کلک کریں ← نیو ← فولڈر۔ (تصویر 5.20 طریقہ کار کو دکھایا گیا ہے)

مرحلہ 2: ایک فولڈر پہلے سے طے شدہ نام "New Folder" کے ساتھ ظاہر ہوتا ہے اور اسے نمایاں کیا جائے گا جیسا کہ تصویر 5.22 میں دکھایا گیا ہے۔

مرحلہ 3: اپنا مطلوبہ نام ٹائپ کریں اور انٹر کی دبائیں۔

مرحلہ 4: فولڈر کا نام بدل جائے گا۔

ورکشاپ

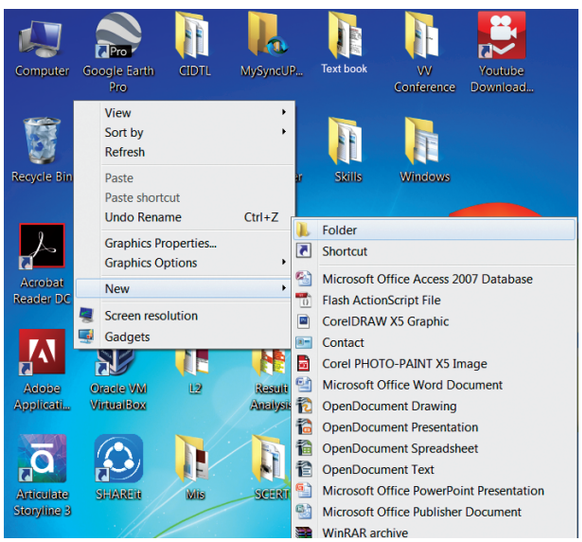
2. زیر بحث طریقوں میں سے کسی ایک کو استعمال کرتے ہوئے اپنے نام کے ساتھ مائی ڈاکومنٹ میں ایک فولڈر بنائیں۔

5.11.1.2 فائلیں بنانا (ورڈ پیڈ)

ورڈ پیڈ ونڈوز OS میں متنی دستاویزات بنانے اور ان میں ہیرا پھیری کرنے کے لیے ایک اندرونِ تعمیر ورڈ پروسیسر ایپلی کیشن ہے۔

ورڈ پیڈ میں فائلیں بنانے کے لیے آپ کو نیچے دیے گئے مراحل پر عمل کرنا ہوگا۔

1. اسٹارٹ پر کلک کریں ← آل پروگرامز ← اسیسریز ← ورڈپیڈ یا رن ← ورڈ پیڈ ٹائپ کریں،اوکے پر کلک کریں۔ ورڈ پیڈ ونڈو کھل جائے گی جیسا کہ تصویر 5.22 میں دکھایا گیا ہے۔
2. ورک اسپیس (Work Space) میں مواد ٹائپ کریں اور فائل کو محفوظ کریں یا Ctrl + S.کا استعمال کرتے ہوئے فائل کو محفوظ کریں۔
3. Save As ڈائیلاگ باکس کھل جائے گا۔
4. ڈائیلاگ باکس میں، وہ مقام منتخب کریں جہاں آپ فائل کو محفوظ کرنا چاہتے ہیں،لک ان ڈراپ ڈاؤن فہرست باکس (Look in Drop down list box) کا استعمال کرتے ہوئے ۔
5. فائل کے نام کے ٹیکسٹ باکس میں فائل کا نام ٹائپ کریں۔
6. سیو بٹن پر کلک کریں۔

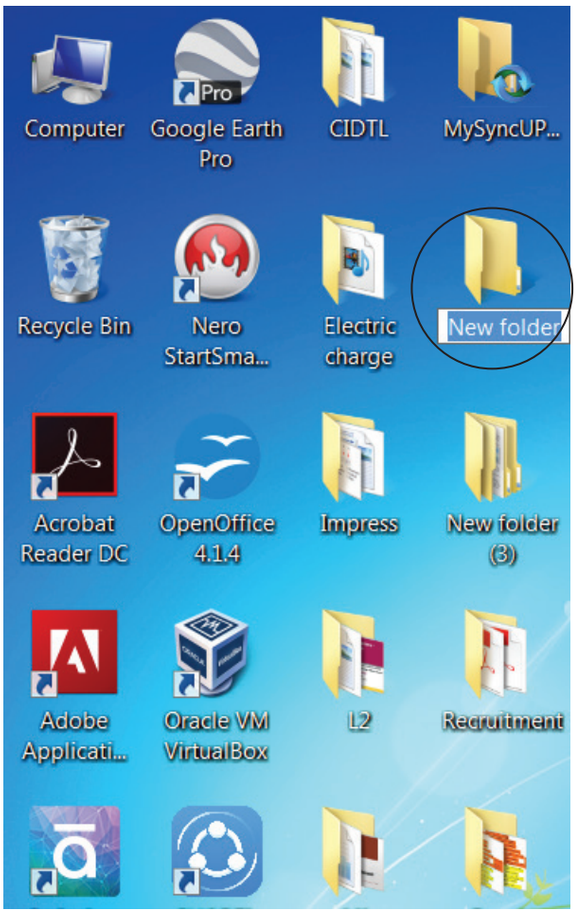


تصویر 5.20 ڈیسک ٹاپ میں فولڈر بنانا

ورکشاپ

3. ورڈ پیڈ ایپلیکیشن کھولیں اور اسے مائی ڈاکو منٹ میں اپنے نام کے ساتھ بنائے گئے فولڈر میں محفوظ کریں۔

11th_Comp Science_UM_Ch 5.indd 66 18-04-2023 14:05:12



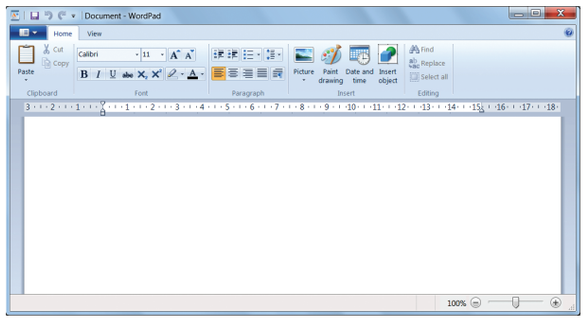
تصویر 5.21 ڈیسک ٹاپ پر نئے فولڈر کا آئیکن

5.11.2 فائلیں اور فولڈرس تلاش کرنا

آپ اسٹارٹ مینو پر موجود سرچ باکس کو کمپیوٹر یا کسی مخصوص ڈرائیو میں کسی خاص فولڈر یا فائل کو تیزی سے تلاش کرنے کے لیے استعمال کر سکتے ہیں.

ایک فولڈر تلاش کرنے کے لئے:

1. اسٹارٹ بٹن پر کلک کریں، سرچ باکس اسٹارٹ مینو کے نیچے ظاہر ہوتا ہے۔
2. فائل یا فولڈر کا نام ٹائپ کریں جسے آپ تلاش کرنا چاہتے ہیں۔ یہاں تک کہ اگر آپ فائل یا فولڈر کا نام دیتے ہیں، تو یہ مخصوص نام سے شروع ہونے والی فائلوں یا فولڈرز کی فہرست دکھائے گا۔ (تصویر 5.23)
3. مخصوص ناموں والی فائلیں یا فولڈرز ظاہر ہوں گے، اگر آپ اس فائل پر کلک کریں گے، تو یہ براہ راست اس فائل یا فولڈر کو کھول دے گا۔

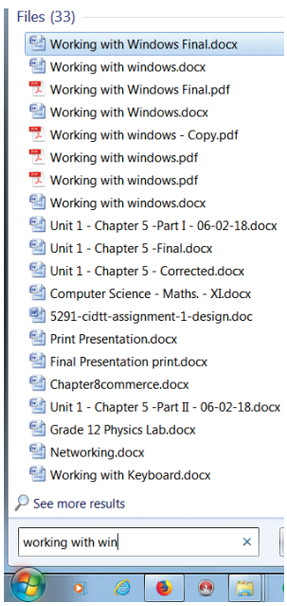


تصویر 5.22 ورڈ پیڈ – ورڈ پروسیسر ایپلی کیشن

4. ایک اور اختیار ہے جسے "مزید نتائج دیکھیں" (See more results) کہا جاتا ہے جو سرچ باکس کے اوپر ظاہر ہوتا ہے۔
5. اگر آپ اس پر کلک کرتے ہیں، تو یہ آپ کو تلاش کے نتائج کے ڈائیلاگ باکس میں لے جائے گا جہاں آپ اس فائل یا فولڈر کو کلک کر کے کھول سکتے ہیں۔

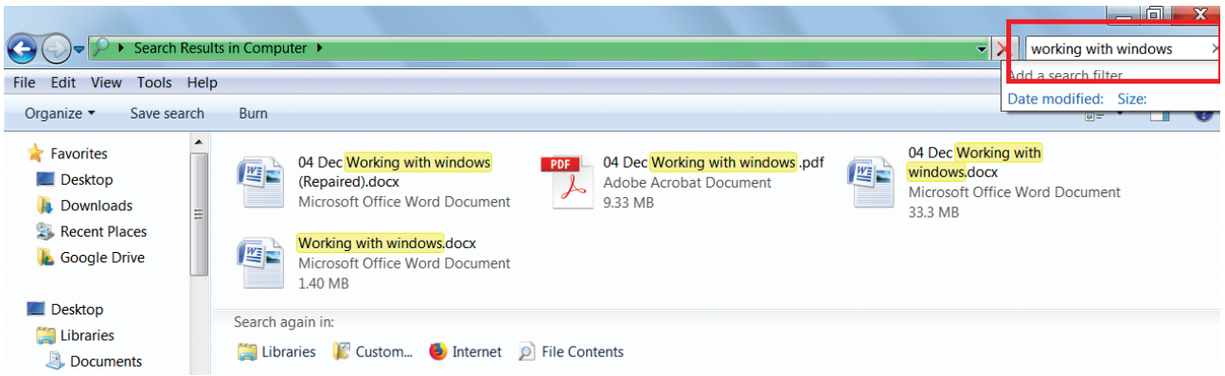
کمپیوٹر آئیکن کا استعمال کرتے ہوئے فائلوں یا فولڈرز کو تلاش کرنا

1. ڈیسک ٹاپ یا اسٹارٹ مینو سے کمپیوٹر آئیکن پر کلک کریں۔
2. کمپیوٹر ڈسک ڈرائیو اسکرین ظاہر ہوگی اور اس پر اس اسکرین کے اوپری دائیں کونے میں سرچ باکس کا آپشن (Option) موجود ہے۔ (تصویر 5.24)
3. فائل یا فولڈر کا نام ٹائپ کریں جسے آپ تلاش کرنا چاہتے ہیں۔ یہاں تک کہ اگر آپ فائل یا فولڈر کا نام دیتے ہیں، تو یہ مخصوص نام کے ساتھ فائلوں یا فولڈرس کی فہرست دکھائے گا۔



تصویر 5.23 اسٹارٹ بٹن کا استعمال کرتے ہوئے فائل/فولڈر تلاش کرنا

11th_Comp Science_UM_Ch 5.indd 67 18-04-2023 14:05:13



تصویر 5.24 کمپیوٹر آپریٹنگ اسکرین میں فائل/فولڈر تلاش کرنا

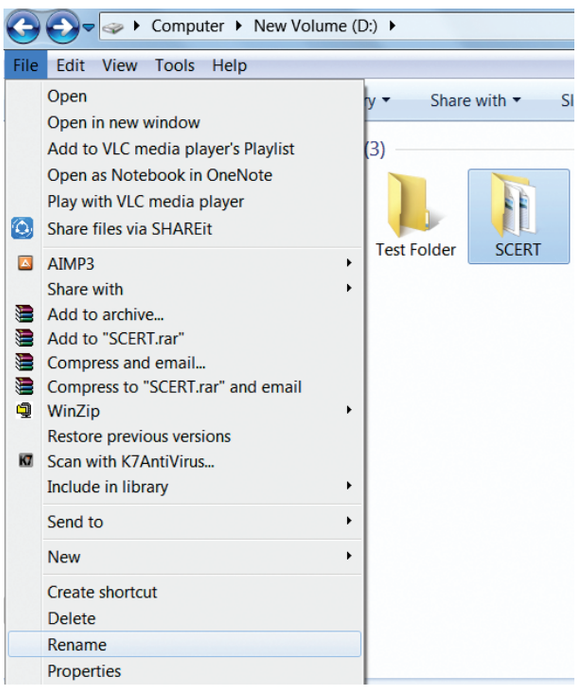
4. صرف کلک کریں اور اس فائل یا فولڈر کو کھولیں۔

ورکشاپ

4. مندرجہ بالا طریقہ کار کا استعمال کرتے ہوئے ورکشاپ- 3 میں بنائی گئی فائل کو تلاش کریں۔

5.11.3 موجودہ فائلوں یا فولڈرس کو کھولنا (Opening existing Files or Folders)

فائل یا فولڈر کو کھولنے کا سب سے عام طریقہ اس پر ڈبل کلک کرنا ہے۔



تصویر 5.25 استعمال کرتے ہوئے فائل/فولڈر کا نام تبدیل کرنا

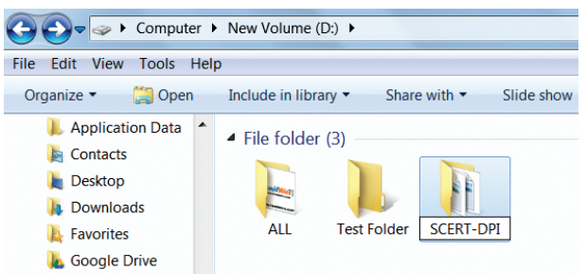
5.11.4 فائلوں یا فولڈرس کا نام تبدیل کرنا

فائلوں یا فولڈرس کا نام تبدیل کرنے کے کئی طریقے ہیں۔ آپ فائل مینو، بائیں ماؤس بٹن یا دائیں ماؤس بٹن کا استعمال کرتے ہوئے فائل کا نام بدل سکتے ہیں۔

طریقہ 1

فائل مینو کا استعمال کرتے ہوئے

1. جس فائل یا فولڈر کا آپ نام تبدیل کرنا چاہتے ہیں اسے منتخب کریں۔
2. فائل پر کلک کریں ← نام تبدیل کریں۔
3. نیا نام ٹائپ کریں۔
4. نام تبدیل کرنے کے عمل کو حتمی صورت دینے کے لیے، تصویر 5.25 کی طرح انٹر دبائیں۔



تصویر 5.26 فولڈر کا نام تبدیل کر دیا گیا۔

تصویر 5.26 آپ دیکھ سکتے ہیں کہ فولڈر کا نام تبدیل کر کے SCERT سے SCERT -DPI رکھا گیا ہے۔

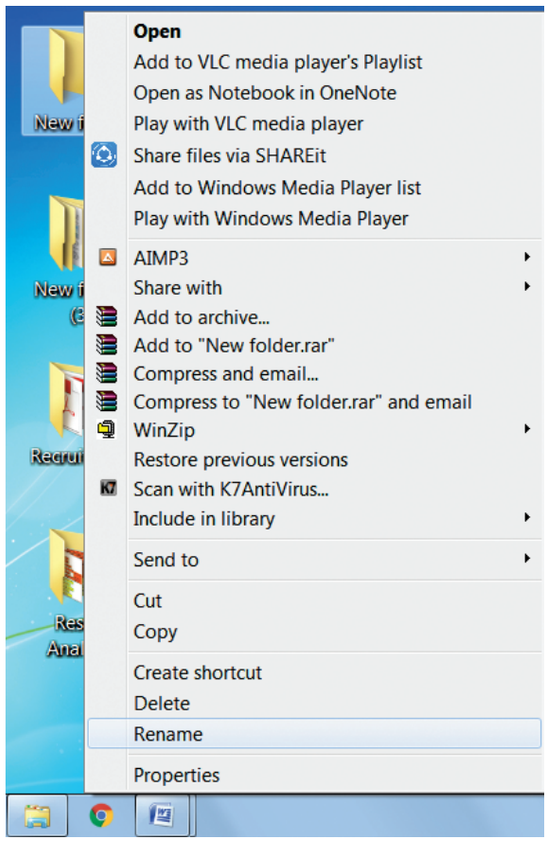
طریقہ 2

دائیں ماؤس بٹن کا استعمال کرتے ہوئے

1. جس فائل یا فولڈر کا آپ نام تبدیل کرنا چاہتے ہیں اسے منتخب کریں۔
2. فائل یا فولڈر پر ماؤس کے دائیں بٹن پر کلک کریں۔ (تصویر 5.27)
3. پاپ اپ مینو سے Rename کو منتخب کریں

11th_Comp Science_UM_Ch 5.indd 68 18-04-2023 14:05:13

4. نیا نام ٹائپ کریں۔
5. نام تبدیل کرنے کے عمل کو حتمی صورت دینے کے لیے، انٹر دبائیں۔
6. تصویر 5.29 ظاہر کرتا ہے کہ "New Folder" کا نام C++ کے نام سے بدل دیا گیا ہے۔



تصویر 5.27 دائیں ماؤس بٹن کا استعمال کرتے ہوئے فائل/فولڈر کا نام تبدیل کرنا



تصویر 5.28 نئے فولڈر کا نام C++ کے نام سے بدل دیا گیا ہے۔

طریقہ 3

بائیں ماؤس بٹن کا استعمال کرتے ہوئے

1. جس فائل یا فولڈر کا آپ نام تبدیل کرنا چاہتے ہیں اسے منتخب کریں۔
2. F2 دبائیں یا فائل یا فولڈر پر کلک کریں۔ نام کے ارد گرد ایک مستطیل (rectangle) ظاہر ہو گا۔
3. نیا نام ٹائپ کریں۔
4. نام تبدیل کرنے کے عمل کو حتمی صورت دینے کے لیے انٹر دبائیں۔

ورکشاپ

5. فائل مینو، بائیں ماؤس بٹن یا دائیں ماؤس بٹن کا استعمال کرتے ہوئے آپ کی بنائی ہوئی فائل کا نام تبدیل کریں۔

5.11.5 فائلوں اور فولڈرس کو منتقل/کاپی کرنا

آپ مختلف طریقوں سے اپنی فائلوں یا فولڈرز کو دوسری جگہ منتقل کر سکتے ہیں۔

فائلوں اور فولڈرز کو منتقل کرنا

طریقہ I – کٹ اور پیسٹ (Cut and Paste)

کسی فائل یا فولڈر کو منتقل کرنے کے لیے، پہلے فائل یا فولڈر کو منتخب کریں اور پھر درج ذیل میں سے ایک کو منتخب کریں:

- **ایڈٹ ← کٹ پر کلک کریں یا** Ctrl + X دبائیں۔ یا پھر پاپ مینو سے رائٹ کلک ← کٹ پر کلک کریں۔
- فائل (فائلوں) یا فولڈر (فولڈرس) کو نئے مقام پر منتقل کرنے کے لیے، نئے مقام پر جائیں اور ایڈٹ مینو سے ایڈٹ ← پیسٹ پر کلک کریں ۔یاکی بورڈ سے **Ctrl + V دبائیں۔**
- یا پاپ اپ مینو سے دائیں کلک ← پیسٹ پر کلک کریں۔ فائل کو نئی جگہ پر چسپاں کیا جائے گا۔

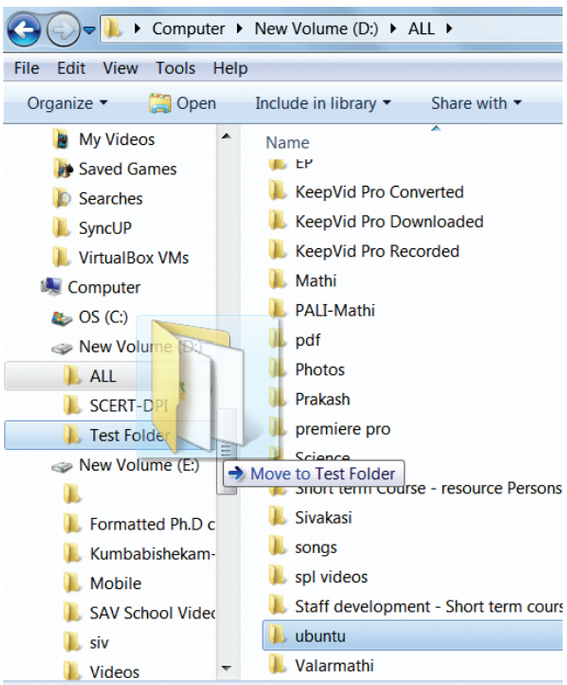
طریقہ II – ڈریگ و ڈراپ (Drag and Drop)

ڈسک ڈرائیو ونڈو میں، ہمارے پاس دو پینس (Panes) ہیں جنہیں بائیں اور دائیں پینس کہتے ہیں۔ بائیں پین میں، فائلیں یا فولڈرز درخت کے ڈھانچے کی طرح دکھائے جاتے ہیں۔ دائیں پین میں مخصوص فولڈرس بائیں پین میں موجود فائلیں مختلف آپشنز کے ساتھ ظاہر ہوتی ہیں۔

11th_Comp Science_UM_Ch 5.indd 69 18-04-2023 14:05:13

- ڈسک ڈرائیو ونڈو کے دائیں پین میں، وہ فائل یا فولڈر منتخب کریں جسے آپ منتقل کرنا چاہتے ہیں۔
- دائیں پین سے منتخب فائل یا فولڈر پر کلک کریں اور بائیں پین پر فولڈر کی فہرست میں گھسیٹیں۔
- ٹارگٹ فولڈر (Target Folder) ظاہر ہونے پر ماؤس بٹن کو چھوڑ دیں (فعال)۔
- آپ کی فائل یا فولڈر اب نئے مقام میں ظاہر ہوگا۔

تصویر 5.29 دکھاتا ہے کہ ڈریگ اینڈ ڈراپ طریقہ استعمال کرکے فائلوں یا فولڈرس کو کیسے منتقل کیا جائے۔



تصویر 5.29 ڈریگ اینڈ ڈراپ کا استعمال کرتے ہوئے فائل / فولڈر کو منتقل کرنا

فائلوں اور فولڈرس کو کاپی کرنا

فائلوں اور فولڈرس کو کاپی کرنے کے مختلف طریقے ہیں:

طریقہ I – کاپی اور پیسٹ کریں۔

کسی فائل یا فولڈر کو کاپی کرنے کے لیے، پہلے فائل یا فولڈر کو منتخب کریں اور پھر درج ذیل میں سے ایک کو منتخب کریں:

- پاپ اپ مینو سے ایڈٹ ← کاپی یا CTRL + C پر کلک کریں یا دائیں کلک ← کاپی پر کلک کریں۔
- نئی جگہ پر فائل یا فولڈر پیسٹ کرنے کے لیے، ہدف والے مقام پر جائیں پھر درج ذیل میں سے کوئی ایک کریں:
- ایڈٹ ← پیسٹ یا CTRL + V پر کلک کریں۔
- یا پاپ اپ مینو سے دائیں کلک ← پیسٹ پر کلک کریں۔

طریقہ II – ڈریگ و ڈراپ۔

- دائیں پین میں، وہ فائل یا فولڈر منتخب کریں جسے آپ کاپی کرنا چاہتے ہیں۔
- منتخب فائل / یا فولڈر کو بائیں طرف فولڈر کی فہرست میں کلک کریں اور گھسیٹیں، اور جہاں آپ فائل / یا فولڈر کاپی کرنا چاہتے ہیں وہاں چھوڑ دیں۔
- آپ کی فائل (فائلیں) اور فولڈر اب نئے مقام میں ظاہر ہوں گے۔

نوٹ

اگر آپ متعدد فائلوں یا فولڈرز کو منتخب کرنا چاہتے ہیں تو **Ctrl + Click** کا استعمال کریں۔

5.11.6 فائلوں اور فولڈرز کو ہٹانے کے قابل ڈسک پر کاپی کرنا

فائلوں کو ہٹانے کے قابل ڈسک میں منتقل کرنے کے کئی طریقے ہیں۔

- کاپی اور پیسٹ (Copy and Paste)
- سینڈ ٹو (Send to)

طریقہ I – کاپی اور پیسٹ کریں۔

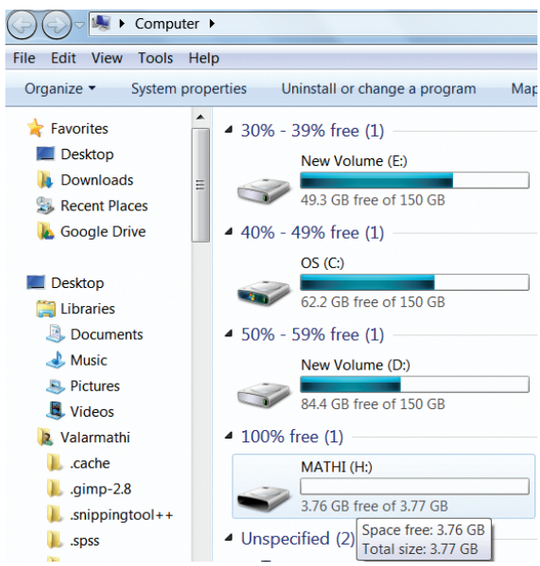
- USB فلیش ڈرائیو کو براہ راست دستیاب USB پورٹ میں لگائیں۔
- اگر USB فلیش ڈرائیو یا بیرونی ڈرائیو فولڈر خود بخود نہیں کھلتا ہے، تو ان اقدامات پر عمل کریں:
- اسٹارٹ ← کمپیوٹر پر کلک کریں۔ (تصویر 5.30)

11th_Comp Science_UM_Ch 5.indd 70 18-04-2023 14:05:13



تصویر 5.30 اسٹارٹ مینو سے کمپیوٹر آپشن کا انتخاب کریں۔

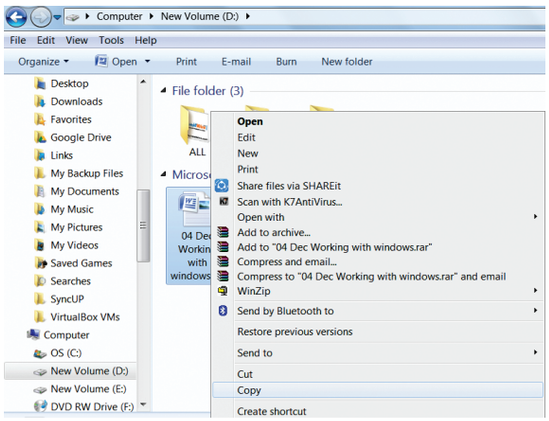
- USB فلیش ڈرائیو سے منسلک نکالی جانے والی ڈسک (Removeable Disk) پر ڈبل کلک کریں۔ (تصویر 5.31)



تصویر 5.31 نکالی جانے والی ڈسک پر ڈبل کلک کرنا

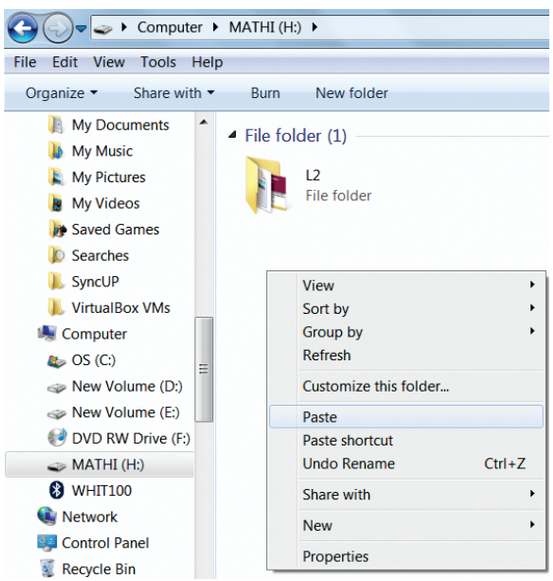
- اپنے کمپیوٹر میں ان فولڈرس پر جائیں جن میں وہ فائلیں ہیں جنہیں آپ منتقل کرنا چاہتے ہیں۔

جس فائل کو آپ کاپی کرنا چاہتے ہیں اس پر دائیں کلک کریں، پھر کاپی کو منتخب کریں۔ (تصویر 5.32)



تصویر 5.32 دائیں کلک کا استعمال کرتے ہوئے فائل کاپی کرنا

- نکالی جانے والی ڈسک ونڈو پر واپس جائیں، ونڈو کے اندر دائیں کلک کریں، پھر پیسٹ کو منتخب کریں۔ (تصویر 5.33)

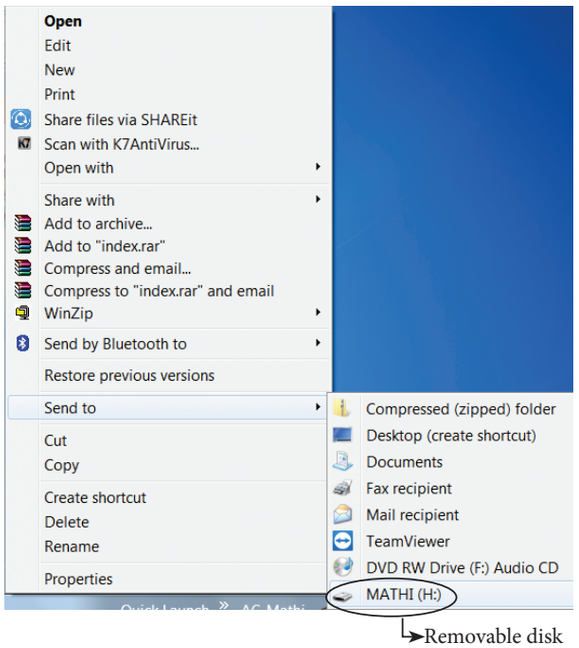


تصویر 5.33 دائیں کلک کا استعمال کرتے ہوئے فائل کو پیسٹ کرنا

طریقہ II – سینڈ ٹو (Send To)

- USB فلیش ڈرائیو کو براہ راست دستیاب USB پورٹ میں لگائیں۔
- اپنے کمپیوٹر میں ان فولڈرز پر جائیں جن میں وہ فائلیں ہیں جنہیں آپ منتقل کرنا چاہتے ہیں۔
- اس فائل پر دائیں کلک کریں جسے آپ اپنی نکالی جانے والی ڈسک میں منتقل کرنا چاہتے ہیں۔
- سینڈ ٹو پر کلک کریں اور USB فلیش ڈرائیو سے وابستہ نکالی جانے والی ڈسک کو منتخب کریں۔ (تصویر 5.34)

11th_Comp Science_UM_Ch 5.indd 71 18-04-2023 14:05:13



تصویر 5.34 سینڈ ٹو کا استعمال کرتے ہوئے فائل کاپی کرنا

5.11.7 فائلوں اور فولڈرس کو حذف کرنا (Deleting Files and Folders)

- جب آپ کسی فائل یا فولڈر کو حذف کرتے ہیں، تو یہ ری سائیکل بین میں چلا جاتا ہے۔

فائل یا فولڈر کو حذف کرنے کے لیے:

وہ فائل یا فولڈر منتخب کریں جسے آپ حذف کرنا چاہتے ہیں۔

1. فائل یا فولڈر پر رائٹ کلک کریں، پاپ اپ مینو سے ڈلیٹ آپشن کو منتخب کریں یا فائل ← ڈلیٹ پر کلک کریں۔ یا کی بورڈ سے ڈلیٹ کی کو دبائیں۔
2. فائل ڈلیٹ ہوجائے گا اور ری سائکل بین میں منتقل ہوجائے گا۔

ورکشاپ

6. ◆ مائی ڈاکو منٹ میں آپ کی بنائی ہوئی فائل کو D ڈرائیو میں منتقل کریں۔
   ◆ D ڈرائیو سے آپ کی بنائی ہوئی فائل کو نکالی جانے والی ڈسک میں کاپی کریں۔

ورکشاپ

7. مائی ڈاکومنٹس کے تحت آپ کی بنائی ہوئی فائل کی نقل بنانے کے بعد ڈیلیٹ کریں۔

نوٹ

کسی فائل یا فولڈر کو مستقل طور پر حذف کرنے کے لیے (یعنی کسی فائل یا فولڈر کو ری سائیکل بن میں بھیجنے سے بچنے کے لیے)، شفٹ بٹن کو دبائے رکھیں، اور کی بورڈ پر ڈیلیٹ دبائیں۔

ریسائیکل بن (Recycle Bin)

ریسائیکل بن ایک خاص فولڈر ہے جو صارف کے ذریعے حذف شدہ فائلوں یا فولڈرز کو رکھنے کے لیے ہے، جس کا مطلب ہے کہ آپ کے پاس اب بھی انہیں بازیافت کرنے کا موقع ہے۔ صارف ری سائیکل بن میں موجود فائلوں یا فولڈرز کو بحال کیے بغیر رسائی حاصل نہیں کر سکتا۔ ری سائیکل بن سے فائل یا فولڈر کو بحال کرنے کے لیے

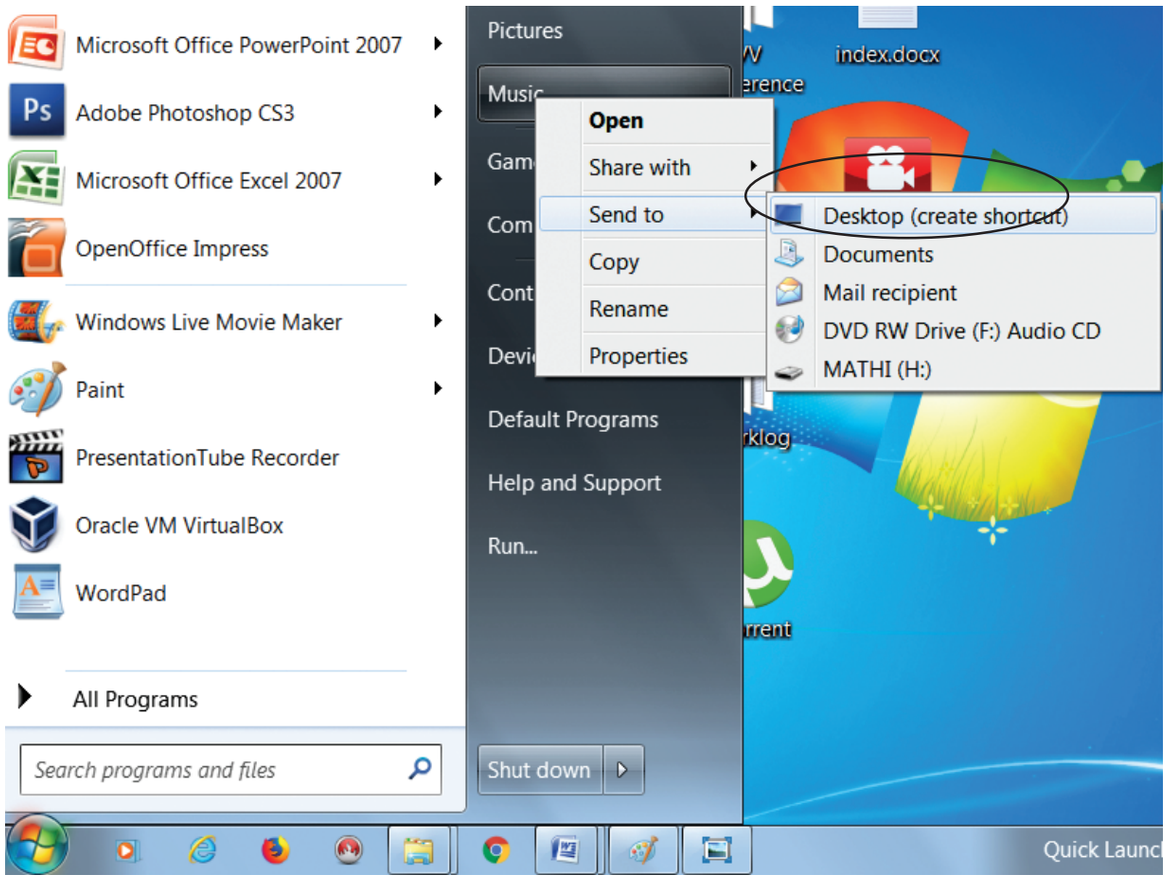
- ری سائیکل بن کھولیں۔
- بحال ہونے والی فائل یا فولڈر پر دائیں کلک کریں اور پاپ اپ مینو سے ری اسٹور (Restore) کو منتخب کریں۔
- متعدد فائلوں یا فولڈرز کو بحال کرنے کے لیے ری اسٹور آل آئیٹمس (Restore All Items) کو منتخب کریں۔
- ری سائیکل بن میں موجود تمام فائلوں کو حذف کرنے کے لیے ایمپٹی دی ری سائیکل بن (Empty the Recycle Bin) کو منتخب کریں۔

5.12 ڈیسک ٹاپ پر شارٹ کٹ بنانا

آپ کے اکثر استعمال ہونے والے فولڈرز اور فائلوں کے شارٹ کٹ آپ کے کام کو خودکار بنانے میں معاون ہوسکتے ہیں اور ڈیسک ٹاپ پر رکھے جا سکتے ہیں۔

- وہ فائل یا فولڈر منتخب کریں جسے آپ ڈیسک ٹاپ پر شارٹ کٹ کے طور پر رکھنا چاہتے ہیں۔
- فائل یا فولڈر پر دائیں کلک کریں۔
- شارٹ کٹ مینو سے Send to کو منتخب کریں، پھر ذیلی مینو سے ڈیسک ٹاپ (Create Shortcut) کو منتخب کریں۔
- فائل یا فولڈر کے لیے ایک شارٹ کٹ اب آپ کے ڈیسک ٹاپ پر ظاہر ہوگا اور آپ اسے ڈیسک ٹاپ سے اسی طرح کھول سکتے ہیں جیسے کسی دوسرے آئیکن کو۔ تصویر 5.35۔

11th_Comp Science_UM_Ch 5.indd 72 18-04-2023 14:05:14



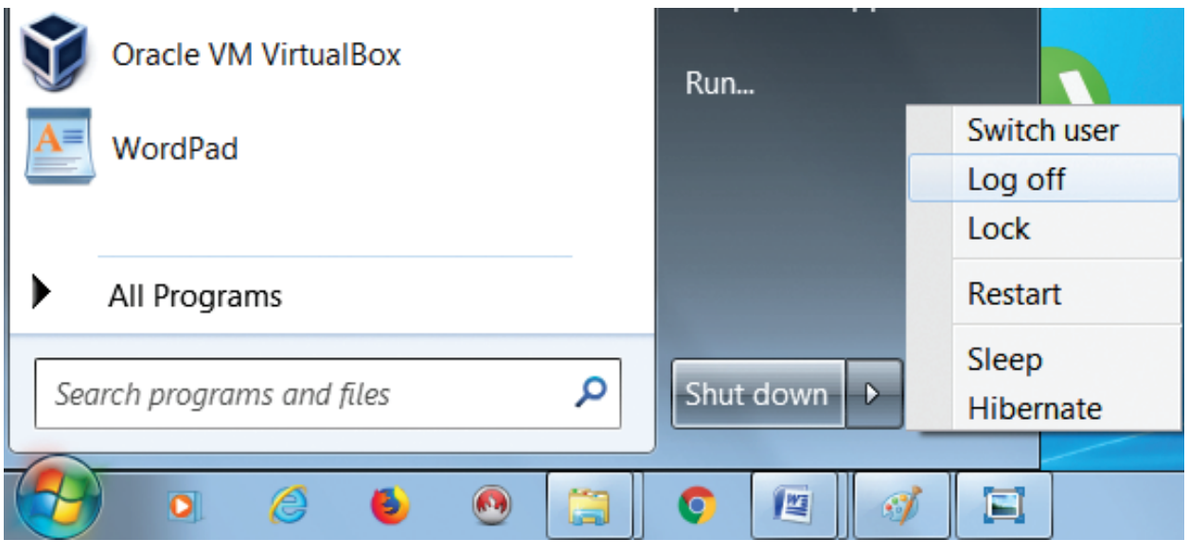
تصویر 5.35 شارٹ کٹ بنانا

5.13 کمپیوٹر کو بند کرنا یا لاگ آف کرنا

ایک بار جب آپ تمام کھلی ایپلی کیشنز کو بند کر دیتے ہیں، تو آپ یا تو اپنے کمپیوٹر کو لاگ آف کر سکتے ہیں یا کمپیوٹر کو بند کر سکتے ہیں۔

لاگ آف (Log off)

کمپیوٹر کو لاگ آف / شٹ ڈاؤن کے لیے



تصویر 5.36 لاگ آف آپشن

11th_Comp Science_UM_Ch 5.indd 73 18-04-2023 14:05:14

- اسٹارٹ ← لاگ آف پر کلک کریں (شٹ ڈاؤن کے آگے تیر پر کلک کریں) یا اسٹارٹ ← شٹ ڈاؤن پر کلک کریں۔ (تصویر 5.37)
- اگر آپ کے پاس کوئی کھلا پروگرام ہے، تو آپ سے انہیں بند کرنے کے لیے کہا جائے گا یا ونڈوز زبردستی بند ہو جائیں گی، اگر آپ ایسا کرتے ہیں تو آپ کوئی بھی غیر محفوظ شدہ معلومات کھو دیں گے۔
- **سوئچ یوزر (Switch User) :** اپنے کھلے پروگراموں اورونڈوز کے عمل کو بند کیے بغیر کمپیوٹر پر دوسرے صارف اکاؤنٹ پر جائیں۔
- **لاگ آف (Log off) :** اپنے تمام کھلے پروگراموں اور ونڈوز کے عمل کو بند کرنے کے بعد کمپیوٹر پر دوسرے صارف کے اکاؤنٹ پر جائیں۔
- **لاک (Lock) :** جب آپ کمپیوٹر سے دور ہوں تو اسے لاک کریں۔
- **ری اسٹارٹ (Restart) :** کمپیوٹر کو دوبارہ شروع کریں (Reboot)۔ (یہ اختیار اکثر نئے سافٹ ویئر یا ونڈوز اپ ڈیٹ کو انسٹال کرنے کے حصے کے طور پر درکار ہوتا ہے۔)
- **سلیپ (Sleep) :** کمپیوٹر کو کم پاور موڈ میں رکھتا ہے جو تمام چلنے والے پروگراموں کو برقرار رکھتا ہے اور تیز رفتار دوبارہ شروع کرنے کے لیے کمپیوٹر میموری میں ونڈوز کھولتا ہے۔
- **ہائبرنیٹ (Hibernate) (صرف لیپ ٹاپ کمپیوٹرز پر پایا جاتا ہے):** تمام چلنے والے پروگراموں کو محفوظ کرنے کے بعد کمپیوٹر کو کم پاور موڈ میں رکھتا ہے اور فوری دوبارہ شروع کرنے کے لیے مشین کی ہارڈ ڈرائیو پر ونڈوز کو کھولتا ہے۔

سرگرمی

طالب علم کی سرگرمی

1. ونڈوز (Windows) اور اوبنٹو (Ubuntu) کا استعمال کرتے ہوئے فائلیں اور فولڈرز بنائیں اور ان کا موازنہ کریں۔
2. ونڈوز 7، ونڈوز 8 اور ونڈوز 10 میں ایک فائل/فولڈر بنائیں۔ اسے بناتے وقت آپ کو جن اختلافات کا سامنا کرنا پڑتا ہے اس پر رپورٹ تیار کریں۔

تشخیص

سیکشن - A

RHWOP

صحیح جواب کا انتخاب کریں۔

1. نیچے دیے گئے اختیارات میں سے، آپریٹنگ سسٹم کے زیر انتظام آپریشنز کا انتخاب کریں۔
   a. میموری
   b. پروسیسیز
   c. ڈسک اور I/O ڈیوائسز
   d. اوپر کے تمام
2. آپ کی فائل کو محفوظ کرنے کے لیے بہت سی ونڈوز ایپلی کیشنز کا ڈیفالٹ فولڈر کون سا ہے؟
   a. مائی ڈاکومنٹ
   b. مائی پکچر
   c. ڈاکومنٹ اور سیٹنگس
   d. مائی کمپیوٹر
3. درج ذیل میں سے کس OS کے تحت Shift + Delete کا آپشن - کسی فائل یا فولڈر کو مستقل طور پر ڈیلیٹ کر دیتا ہے؟

   a. Windows 7 b. MS-DOS
   c. Linux d. Android OS
4. ونڈوز ایکس پی اور ونڈوز 7 میں Hibernate کا کیا مطلب ہوتا ہے؟
   a. کمپیوٹر کو محفوظ طریقے سے دوبارہ اسٹارٹ کرنا۔
   b. ہائبرنیٹ موڈ میں کمپیوٹر کو دوبارہ اسٹارٹ کرنا۔
   c. کمپیوٹر کو بند کر دیں اور چل رہے تمام کام بند کر دیں۔
   d. چل رہی ایپلیکیشنز کو بند کیے بغیر کمپیوٹر کو بند کریں۔
5. ونڈوز میں فائل کا نام تبدیل کرنے کے لیے استعمال ہونے والی شارٹ کٹ کلید

   a. F2 b. F4
   c. F5 d. F6

11th_Comp Science_UM_Ch 5.indd 74 18-04-2023 14:05:14

سیکشن – B

بہت مختصر جوابات دیں۔

1. ملٹی ٹاسکنگ کے نام سے کیا جانا جاتا ہے؟
2. اسٹینڈر آئکنس کسے کہتے ہیں؟
3. فائلوں اور فولڈرس میں فرق کریں؟
4. سیو اور سیوایز آپشن میں کیا فرق ہے؟
5. آپ فائل کا نام کیسے بدلیں گے؟

سیکشن – C

مختصر جوابات دیں۔

1. ونڈوز آپریٹنگ سسٹم کے کام کیا ہیں؟
2. ری سائیکل بن پر ایک نوٹ لکھیں۔
3. ونڈوز کے عناصر پر ایک نوٹ لکھیں۔
4. نیا فولڈر بنانے کے دو طریقے لکھیں۔
5. کاپی اور منتقل میں فرق کریں۔

سیکشن – D

تفصیل سے بیان کریں۔

1. ونڈوز آپریٹنگ سسٹم کے ورژن کی وضاحت کریں۔
2. فائل یا فولڈر تلاش کرنے کے مختلف طریقوں کی وضاحت کریں۔
3. ونڈوز OS میں شارٹ کٹ بنانے کا طریقہ کار لکھیں۔

RSVOM

A-Z
لغت

آپریٹنگ سسٹم Operating System (OS)	سسٹم سافٹ ویئر جو ہارڈویئر کو دوسرے سافٹ ویئر کے ساتھ بات چیت اور کام کرنے کے قابل بناتا ہے۔
ماؤس Mouse	مٹھی میں سماجانے والا ہارڈویئر (Handheld Hardware) ان پٹ ڈیوائس جو GUI میں کرسر (Cursor) کو کنٹرول کرتا ہے اور متن، آئکنس، فائلز اور فولڈرز کو منتقل اور منتخب کرتا ہے۔
ونڈوز Windows	مائیکروسافٹ کارپوریشن کے ذریعہ تیار کردہ مشہور ومعروف آپریٹنگ سسٹم۔
ڈیسک ٹاپ Desktop	ونڈوز آپریٹنگ سسٹم کا کھلا ہوا اسکرین۔
آئیکن Icon	چھوٹی تصویر جو ایک کمانڈ کی نمائندگی کرتی ہے۔
فولڈر Folder	فائلوں کا مجموعہ
لینکس Linux	ایک آپریٹنگ سسٹم۔

11th_Comp Science_UM_Ch 5.indd 75 18-04-2023 14:05:14

تفصیلات اور خلاصہ

سیکھنے کے مقاصد۔

کے قابل ہو جائیں گے۔ اس باب میں تصورات کو سیکھنے کے بعد، طلباء

- الگورتھمک مسئلہ حل کرنے کے تصور کو سمجھنے
- مسئلہ حل کرنے میں الگورتھمک تکنیک کے علم کا اطلاق کرنے

ہماری روزمرہ کی زندگی میں انجام پانے والے متعدد عمل ہدایات کی ترتیب کے مرحلہ وار عمل پر عمل پیرا ہوتے ہیں۔ صبح اسکول کے لیے تیار ہونا، "کولمس (kolams)" بنانا، ڈش بنانا، دو نمبر شامل کرنا عمل کی مثالیں ہیں۔ پروسیسز الگورتھم (Algorithm) کو عمل میں لا کر تیار کیے جاتے ہیں۔ اس باب میں، ہم دیکھیں گے کہ الگورتھم کی وضاحت کیسے کی جاتی ہے اور کس طرح کسی عمل کے عناصر کو الگورتھم میں خلاصہ کیا جاتا ہے۔

6.1 الگورتھمس

ایک الگورتھم (Algorithm) کسی کام کو پورا کرنے یا کسی مسئلے کو حل کرنے کے لیے ہدایات کا ایک سلسلہ ہے۔ ایک ہدایت ایک عمل کی وضاحت کرتی ہے۔ جب ہدایات پر عمل کیا جاتا ہے، ایک عمل تیار ہوتا ہے، جو مطلوبہ کام کو پورا کرتا ہے، یا دیئے گئے مسئلے کو حل کرتا ہے۔ ہم ایک الگورتھم کا تقابل کسی نسخے سے کر سکتے ہیں، اور اس کے نتیجے میں ہونے والے عمل کا پکانے سے ہوتا ہے۔

ہم کمپیوٹر میں اپنے الگورتھم کو چلانے میں دلچسپی رکھتے ہیں۔ کمپیوٹر صرف پروگرامنگ لینگویج میں ہدایات پر عمل کر سکتا ہے۔ کمپیوٹر کی ہدایات کو بیانات کے نام سے بھی جانا جاتا ہے۔ لہٰذا، بالآخر، الگورتھم کا اظہار پروگرامنگ زبان کے بیانات کے ذریعے کیا جانا چاہیے۔

ایک مسئلہ دیے گئے ان پٹ ڈیٹا، مطلوبہ آؤٹ پٹ ڈیٹا اور ان پٹ اور آؤٹ پٹ ڈیٹا کے درمیان تعلق سے بیان کیا جاتا ہے۔ ایک الگورتھم ان پٹ ڈیٹا کے ساتھ عمل درآمد شروع کرتا ہے، بیانات کو انجام دیتا ہے، اور آؤٹ پٹ ڈیٹا کے ساتھ عملدرآمد کو ختم کرتا ہے۔ جب یہ عمل مکمل کر لیتا ہے، تو ان پٹ ڈیٹا اور آؤٹ پٹ ڈیٹا کے درمیان مخصوص تعلق کو مطمئن کیا جانا چاہیے۔ تب ہی الگورتھم نے دیئے گئے مسئلے کو حل کیا ہے۔

کیا آپ جانتے ہیں؟

جی پولیا (G. Polya) ہنگری (Hungarian) کا ایک ریاضی دان تھا۔ اس نے امتزاج، نمبر تھیوری، عددی تجزیہ اور امکانی نظریہ میں بنیادی شراکت کی۔ وہ ہیورس ٹکس (Heuristics) اور ریاضی کی تعلیم میں اپنے کام کے لیے بھی جانا جاتا ہے، جو طلبہ اور اساتذہ کے لیے ریاضی میں مزید دریافت اور ایجاد کے لیے مسائل کے حل کے منظم طریقوں کی نشاندہی کرتا ہے۔

میں، وہ ریاضی "(How to solve it) اس کو کیسے حل کریں" کے مسئلے کو حل کرتے وقت درج ذیل اقدامات تجویز کرتا ہے:



1. مسئلے کو سمجھیں۔
2. ایک منصوبہ تیار کریں۔
3. منصوبے کو انجام دیں۔
4. اپنے کام کا جائزہ لیں۔

ایک الگورتھم بیانات کی ایک مرحلہ وار ترتیب ہے جس کا مقصد کسی مسئلے کو حل کرنا ہے۔ جب ان پٹ ڈیٹا کے ساتھ عمل کیا جاتا ہے، تو یہ ایک کمپیوٹیشنل (Computational) عمل پیدا کرتا ہے، اور آؤٹ پٹ ڈیٹا کے ساتھ ختم ہوتا ہے، ان پٹ ڈیٹا اور آؤٹ پٹ ڈیٹا کے درمیان مخصوص تعلق کو پورا کرتا ہے۔

11th_Comp Science_UM_Ch 6.indd 76 18-04-2023 14:23:54

مثال دو نمبروں کو شامل کرنا: دو نمبروں کو شامل کرنے کے لیے، ہم دو نمبروں کے سب سے زیادہ دائیں ہندسوں کو جوڑ کر آگے بڑھتے ہیں، پھر اگلے دائیں سب سے زیادہ ہندسوں کو ساتھ لے کر آگے بڑھاتے ہیں جو پچھلی (دائیں) پوزیشن سے نکلے ہیں، وغیرہ۔ 2586 اور 9237 کو شامل کرنے کا کمپیوٹیشنل (Computational) عمل ٹیبل 6.1 میں دکھایا گیا ہے۔

Step	5	4	3	2	1
Carry	1	0	1	1	-
Number	1	2	5	8	6
Number	2	9	2	3	7
Sum	1	1	8	2	3

ٹیبل 6.1: دو نمبروں کو شامل کرنے کا عمل

6.2 الگورتھمک مسائل (Algorithmic Problems)

الگورتھم کی تعمیر کے لیے کچھ اصول اور تکنیکیں ہیں۔ ہم عام طور پر کہتے ہیں کہ ایک مسئلہ فطرت میں الگورتھم ہے جب اس کے حل میں الگورتھم کی تعمیر شامل ہو۔ کچھ قسم کے مسائل کو فوری طور پر الگورتھم کے طور پر تسلیم کیا جا سکتا ہے۔

مثال 6.2 بکری، گھاس اور بھیڑیے کے معروف مسئلے پر غور کریں: ایک کسان ایک بکری، ایک گھاس کا بنڈل اور ایک بھیڑیا کو دریا کے پار لے جانا چاہتا ہے۔ تاہم، اس کی کشتی ایک وقت میں ان میں سے صرف ایک لے سکتی ہے۔ اس لیے دریا کے اس پار جانے کے لیے کئی سفر ضروری ہیں۔ مزید یہ کہ بکری کو گھاس کے ساتھ اکیلا نہیں چھوڑنا چاہیے (ورنہ بکری گھاس کھا جائے گی) اور بھیڑیے کو بکری کے ساتھ اکیلا نہ چھوڑا جائے (ورنہ بھیڑیا بکری کو کھا جائے گا)۔ کسان اس کام کو کیسے پورا کر سکتا ہے؟ ابتدائی طور پر، ہم فرض کرتے ہیں کہ چاروں دریا کے ایک ہی طرف ہیں، اور آخر میں، چاروں کو مخالف سمت میں ہونا ضروری ہے۔ دریا پار کرتے وقت کسان کا کشتی میں ہونا ضروری ہے۔ ایک حل ہدایات کی ایک ترتیب پر مشتمل ہوتا ہے جو بتاتا ہے کہ کون یا کیا پار کرنا چاہئے؟ لہذا، یہ ایک الگورتھمک مسئلہ ہے۔ ہدایات اس طرح ہو سکتی ہیں۔

کسان کو بھیڑیے کے ساتھ پار کرنے دو۔

یا

کسان کو اکیلا کراس کرنے دو۔



تاہم، کچھ الگورتھمک مسائل ہمیں الگورتھم بنانے کی ضرورت نہیں رکھتے۔ اس کے بجائے، ایک الگورتھم فراہم کیا گیا ہے اور ہمیں اس کی کچھ خصوصیات ثابت کرنے کی ضرورت ہے۔

مثال 6.3 کروم لینڈ (Chromeland) کے گرگٹ (chameleons) کے مسئلے پر غور کریں: کروم لینڈ کے جزیرے پر گرگٹ کی تین مختلف اقسام ہیں: سرخ گرگٹ، سبز گرگٹ اور نیلے گرگٹ۔ جب بھی دو مختلف رنگوں کے گرگٹ ملتے ہیں تو دونوں کا رنگ بدل کر تیسرے رنگ میں آجاتا ہے۔ سرخ، سبز اور نیلے رنگ کے گرگٹوں کی کس تعداد کے لیے ملاقاتوں کا ایک سلسلہ ترتیب دینا ممکن ہے جس کے نتیجے میں تمام گرگٹ ایک ہی رنگ کا مظاہرہ کریں؟ یہ ایک الگورتھمک مسئلہ ہے، کیونکہ گرگٹ کے درمیان ملاقاتوں کا انتظام کرنے کے لیے الگورتھم موجود ہے۔ الگورتھم کی کچھ خصوصیات کا استعمال کرتے ہوئے، ہم یہ معلوم کر سکتے ہیں کہ گرگٹ کی ابتدائی تعداد کس مقصد کے لیے ممکن ہے۔

6.3 الگورتھم کے بلاکس کی تعمیر

ہم بنیادی بلڈنگ بلاکس کا استعمال کرتے ہوئے الگورتھم بناتے ہیں جیسے

- ڈیٹا (Data)
- متغیرات (Variables)
- کنٹرول فلو (Control flow)
- فنکشنز (Functions)

6.3.1 ڈیٹا

الگورتھم ان پٹ ڈیٹا لیتے ہیں، ڈیٹا پر کارروائی کرتے ہیں، اور آؤٹ پٹ ڈیٹا تیار کرتے ہیں۔ کمپیوٹر ڈیٹا پر آپریشن کرنے کے

11th_Comp Science_UM_Ch 6.indd 77 18-04-2023 14:23:54

لیے ہدایات فراہم کرتے ہیں۔ مثال کے طور پر، وہاں عدد پر ریاضی کا عمل کرنے کے لیے ہدایات ہیں ، جیسے جوڑ، گھٹاؤ، ضرب اور تقسیم۔ اعداد و شمار اور متن جیسے مختلف قسم کے ڈیٹا ہیں۔

6.3.2 متغیرات (Variables)

متغیرات کو ڈیٹا ذخیرہ کرنے کے لیے بکس کا نام دیا گیا ہے۔ جب ہم ڈیٹا پر آپریشن کرتے ہیں تو ہمیں نتائج کو متغیرات میں ذخیرہ کرنے کی ضرورت ہوتی ہے۔ متغیر میں ذخیرہ شدہ ڈیٹا کو متغیر کی قدر کے نام سے بھی جانا جاتا ہے۔ ہم اسائنمنٹ اسٹیٹمنٹ کا استعمال کرتے ہوئے ایک ویلیو کو متغیر میں محفوظ کر سکتے ہیں یا متغیر کی قدر کو تبدیل کر سکتے ہیں۔

حقیقی دنیا میں کمپیوٹیشنل عمل کی حالت ہوتی ہے۔ جیسے جیسے ایک عمل تیار ہوتا ہے، حالت بدل جاتی ہے۔ ہم الگورتھم میں کسی عمل کی حالت اور حالت کی تبدیلی کی نمائندگی کیسے کرتے ہیں؟ کسی عمل کی حالت کو الگورتھم میں متغیرات کے سیٹ سے ظاہر کیا جا سکتا ہے۔ عمل کے کسی بھی مقام پر حالت صرف اس مقام پر متغیرات کی قدر ہے۔ جیسے جیسے متغیرات کی قدریں بدل جاتی ہیں، حالت بدل جاتی ہے۔

مثال 6.4۔ حالت: ٹریفک سگنل تین حالتوں میں سے کسی ایک میں ہو سکتا ہے: سبز، گہرا پیلا، یا سرخ۔ ٹریفک کے ہموار بہاؤ کی اجازت دینے کے لیے حالت کو تبدیل کیا گیا ہے۔ حالت کی نمائندگی ایک واحد متغیر سگنل کے ذریعے کی جا سکتی ہے جس میں تین اقدار میں سے ایک ہو سکتی ہے: سبز، گہرا پیلا، یا سرخ۔

6.3.3 کنٹرول فلو

الگورتھم بیانات کی ایک ترتیب ہے۔ تاہم، ایک بیان پر عمل کرنے کے بعد، اگلے بیان پر عمل درآمد کرنے کی ضرورت نہیں ہے کہ الگورتھم میں اگلا بیان ہو۔ اس کے بعد جو بیان دیا جائے گا اس کا انحصار عمل کی حالت پر ہو سکتا ہے۔ اس طرح، بیانات پر عمل درآمد کی ترتیب اس ترتیب سے مختلف ہو سکتی ہے جس میں وہ لکھے گئے ہیں۔

الگورتھم بیانات پر عمل درآمد کا یہ حکم کنٹرول فلو کے نام سے جانا جاتا ہے۔ حالت کے لحاظ سے کنٹرول کے بہاؤ کو تبدیل کرنے کے لیے تین اہم کنٹرول بہاؤ بیانات ہیں۔

- ترتیب وار کنٹرول فلو میں، بیانات کی ایک ترتیب ایک کے بعد ایک اسی ترتیب سے عمل میں لائی جاتی ہے جس طرح وہ لکھے جاتے ہیں۔
- متبادل کنٹرول فلو میں، حالت کی ایک کیفیت کی جانچ کی جاتی ہے، اور اگر شرط درست ہے، تو ایک بیان پر عمل کیا جاتا ہے۔ اگر شرط غلط ہے تو، ایک متبادل بیان عمل میں لایا جاتا ہے۔
- آئی ٹریٹیو کنٹرول فلو (iterative control flow) میں، حالت کی ایک کیفیت کی جانچ کی جاتی ہے ، اور اگر شرط درست ہے، تو ایک بیان پر عمل درآمد کیا جاتا ہے۔ شرط کو جانچنے اور بیان پر عمل درآمد کے دو مراحل اس وقت تک دہرائے جاتے ہیں جب تک کہ شرط غلط نہ ہو جائے۔

6.3.4 فنکشنز (Functions)

الگورتھم بہت پیچیدہ ہو سکتے ہیں۔ الگورتھم کے متغیرات اور متغیر کے درمیان انحصار بہت زیادہ ہو سکتا ہے۔ پھر، الگورتھم کو صحیح طریقے سے بنانا مشکل ہے۔ ایسے حالات میں، ہم الگورتھم کو حصوں میں توڑ دیتے ہیں، ہر حصے کو الگ الگ بناتے ہیں، اور پھر حصوں کو مکمل الگورتھم میں ضم کرتے ہیں۔

الگورتھم کے حصوں کو فنکشنز (Functions) کے نام سے جانا جاتا ہے۔ ایک فنکشن ایک ذیلی الگورتھم کی طرح ہے۔ یہ ایک ان پٹ لیتا ہے، اور ایک آؤٹ پٹ تیار کرتا ہے، جس سے مطلوبہ ان پٹ آؤٹ پٹ تعلق کو پورا کیا جاتا ہے۔

مثال 6.5 فرض کریں کہ ہم ریڈیس r (Radius) اور اونچائی h کے سلنڈر (cylinder) کی سطح کے رقبے کا حساب لگانا چاہتے ہیں۔

$$A = 2\pi r^2 + 2\pi rh$$

11th_Comp Science_UM_Ch 6.indd 78 18-04-2023 14:23:54

ہم ایک دائرے کے رقبے کا حساب لگانے کے لیے اور دوسرا دائرے کے فریم کے لیے دو فنکشنز کی شناخت کر سکتے ہیں، ۔ اگر ہم دو فنکشنز کو دائرہ_ایریا (r) (area) اور دائرہ_ دائرہ (r) کے بطور خلاصہ کرتے ہیں، تو سلنڈر_ایریا (r,h) کو بطور حل کیا جاسکتا ہے۔

cylinder_area (r,h) = 2 X circle_area (r) + circle_circumference (r) X h

6.4 الگورتھم ڈیزائن ٹیکنکس

الگورتھم ڈیزائن (algorithm design) کرنے کے چند بنیادی اصول اور ٹیکنکس ہیں۔

1. **فصیلات:** مسئلہ کو حل کرنے کا پہلا قدم مسئلہ کو واضح طور پر بیان کرنا ہے۔ دیئے گئے ان پٹ اور مطلوبہ آؤٹ پٹ کے لحاظ سے ایک مسئلہ بیان کیا گیا ہے۔ تصریح میں دیے گئے ان پٹ کی خصوصیات اور ان پٹ اور آؤٹ پٹ کے درمیان تعلق کو بھی بیان کرنا چاہیے۔

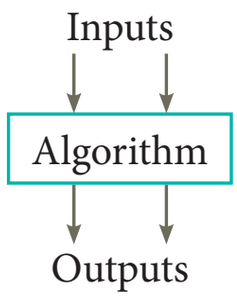
2. **ابسٹریکشن (Abstraction) (علیحدگی):** ایک مسئلہ بہت ساری تفصیلات پر مشتمل ہو سکتا ہے۔ ان میں سے کئی تفصیلات مسئلے کو حل کرنے کے لیے غیر ضروری ہیں۔ صرف چند تفصیلات ضروری ہیں۔ غیر ضروری تفصیلات کو نظر انداز کرنا یا چھپانا اور کسی ہستی کو صرف اس کی ضروری خصوصیات سے ماڈل بنانا ابسٹریکشن کہلاتا ہے۔ مثال کے طور پر، جب ہم کسی عمل کی حالت کی نمائندگی کرتے ہیں، تو ہم مسئلہ کے لیے ضروری متغیرات کو ہی منتخب کرتے ہیں اور غیر ضروری تفصیلات کو نظر انداز کرتے ہیں۔

3. **کمپوزیشن (composition):** ایک الگورتھم تفویض اور کنٹرول فلو کے اسٹیٹمنٹ پر مشتمل ہوتا ہے۔ ایک کنٹرول فلو سٹیٹمنٹ حالت کی کیفیت کی جانچ کرتا ہے اور شرط کی قدر پر منحصر ہے، اگلے بیان پر عمل درآمد کا فیصلہ کرتا ہے۔

4. **ڈی کمپوزیشن (Decomposition):** ہم مرکزی الگورتھم کو فنکشنز میں تقسیم کرتے ہیں۔ ہم ہر فنکشن کو مرکزی الگورتھم اور دیگر فنکشنز سے آزادانہ طور پر بناتے ہیں۔ آخر میں، ہم فنکشنز کا استعمال کرتے ہوئے مین الگورتھم بناتے ہیں۔ جب ہم فنکشنز استعمال کرتے ہیں تو فنکشن کی تفصیلات جاننا کافی ہوتا ہے۔ یہ جاننا ضروری نہیں ہے کہ فنکشن کو کیسے لاگو کیا جاتا ہے۔

6.5 تفصیلات (Specification)

کسی مسئلے کو حل کرنے کے لیے، سب سے پہلے ہمیں اس مسئلے کو واضح طور پر بیان کرنا چاہیے۔ دیئے گئے ان پٹ اور مطلوبہ آؤٹ پٹ کے ذریعہ ایک مسئلہ کی وضاحت کی گئی ہے۔ کسی مسئلے کو حل کرنے کے لیے الگورتھم ڈیزائن کرنے کے لیے، ہمیں دیے گئے ان پٹ کی خصوصیات اور مطلوبہ آؤٹ پٹ کی خصوصیات کو جاننا چاہیے۔ الگورتھم کا مقصد ان پٹ اور مطلوبہ آؤٹ پٹ کے درمیان تعلق قائم کرنا ہے۔



تصویر 6.1: ان پٹ آؤٹ پٹ ریلیشن

ایک الگورتھم دیے گئے ان پٹ کی خصوصیات اور ان پٹ اور مطلوبہ آؤٹ پٹ کے درمیان تعلق سے بیان کیا جاتا ہے۔ آسان الفاظ میں، الگورتھم کی تفصیلات مطلوبہ ان پٹ آؤٹ پٹ تعلق ہے۔

ان پٹ اور آؤٹ پٹس الگورتھم اور صارف کے درمیان متغیرات کے ذریعے منتقل ہوتے ہیں۔ جب الگورتھم شروع ہوتا ہے تو متغیر کی اقدار کو ابتدائی حالت کے طور پر جانا جاتا ہے، اور جب الگورتھم ختم ہوتا ہے تو متغیر کی قدروں کو حتمی حالت کے طور پر جانا جاتا ہے۔

11th_Comp Science_UM_Ch 6.indd 79 18-04-2023 14:23:54

P کو ان پٹس کی مطلوبہ پراپرٹی اور Q کو مطلوبہ آؤٹ پٹ کی خاصیت مانیں۔ پھر الگورتھم S کے طور پر بیان کیا گیا ہے۔

1. الگورتھم_نام (ان پٹس)
2. -- ان پٹس: P
3. -- آؤٹ پٹس: Q

اس تشریح کا مطلب ہے کہ اگر الگورتھم P کو مطمئن کرنے والے ان پٹ کے ساتھ شروع ہوتا ہے، تو یہ Q کو مطمئن کرنے والے آؤٹ پٹ کے ساتھ ختم ہو جائے گا۔

ایک ڈبل ڈیش (double dash) -- اشارہ کرتا ہے کہ باقی لائن ایک تبصرہ ہے۔ تبصرے وہ بیانات ہیں جو انسانی قارئین کے لیے کسی پروگرام کی تشریح کے لیے استعمال ہوتے ہیں اور کمپیوٹر کے ذریعے نہیں کیے جاتے۔ الگورتھم کو سمجھنے کے لیے بہاؤ کے اہم مقامات پر تبصرے مفید، اور ضروری بھی ہیں۔ ہماری الگورتھمک اشارے میں، ہم ایک تبصرہ لائن شروع کرنے کے لیے ڈبل ڈیش (--) استعمال کرتے ہیں۔ (C++ میں، ایک ڈبل سلیش // اشارہ کرتا ہے کہ باقی لائن ایک تبصرہ ہے)۔

مثال 6.6 ۔ایک عدد A کو دوسرے عدد B سے تقسیم کرنے کے بعد اقتباس اور بقیہ کی گنتی کرنے کے لیے الگورتھم کی تفصیلات لکھیں۔ مثال کے طور پر،

تقسیم (22 ,5) = 4 , 2

تقسیم (15 ,3) = 5 , 0

A اور B کو ان پٹ متغیر ہونے دیں۔ ہم حصص کو متغیر q میں اور بقیہ کو متغیر r میں محفوظ کریں گے۔ تو q اور r آؤٹ پٹ متغیر ہیں۔

ان پٹ A اور B کی خصوصیات کیا ہیں؟

1. A ایک عدد صحیح ہونا چاہیے۔ باقی صرف عددی تقسیم کے لیے معنی خیز ہے، اور
2. B کو 0 نہیں ہونا چاہئے، کیونکہ 0 سے تقسیم کی اجازت نہیں ہے۔

ہم ان پٹ کی خصوصیات کو بطور بیان کریں گے۔

-- ان پٹس: A ایک عدد اور $B \neq 0$ ہے۔

ان پٹ A اور B، اور آؤٹ پٹ q اور r کے درمیان مطلوبہ تعلق کیا ہے؟

1. دو آؤٹ پٹ q (قطعہ) اور r (بقیہ) کو پراپرٹی کو مطمئن کرنا چاہئے

$A = q \times B + r$، اور

2. بقیہ r تقسیم کرنے والے B سے کم ہونا چاہیے،

$0 \leq r < B$

ان تقاضوں کو یکجا کرتے ہوئے، ہم مطلوبہ ان پٹ آؤٹ پٹ تعلق کو بطور بیان کریں گے۔

-- آؤٹ پٹس: $A = q \times B + r$ اور $r<B>0$

تبصرہ جو شروع ہوتا ہے -- ان پٹس: درحقیقت دی گئی ان پٹ کی خاصیت ہے۔ تبصرہ جو شروع ہوتا ہے -- آؤٹ پٹ: ان پٹ اور آؤٹ پٹس کے درمیان مطلوبہ تعلق ہے۔ الگورتھم کی تصریح ہے۔

1. **تقسیم (A , B)**
2. **-- ان پٹس: A ایک عدد اور $B \neq 0$ ہے۔**
3. **-- آؤٹ پٹس: $A = q \times B + r$ اور $r \leq B > 0$**

تفصیلات کی شکل: ہم تصریح کو معیاری تین حصوں کی شکل میں لکھ سکتے ہیں:

- الگورتھم اور ان پٹ کا نام۔
- ان پٹ: ان پٹ کی خاصیت۔
- آؤٹ پٹ: مطلوبہ ان پٹ آؤٹ پٹ رشتہ۔

پہلا حصہ الگورتھم اور ان پٹ کا نام ہے۔ دوسرا حصہ ان پٹ کی پراپرٹی ہے۔ یہ ایک تبصرے کے طور پر لکھا گیا ہے جس کا آغاز -- ان پٹس سے ہوتا ہے: تیسرا حصہ مطلوبہ ان پٹ آؤٹ پٹ تعلق ہے۔ یہ ایک تبصرے کے طور پر لکھا گیا ہے جس کا آغاز -- outputs: سے ہوتا ہے۔ ان پٹس اور آؤٹ پٹس کو انگریزی اور ریاضیاتی اشارے کا استعمال کرتے ہوئے لکھا جا سکتا ہے۔

مثال 6.7 ۔ کسی عدد کے اسکوائر روٹ (Square root) کی گنتی کے لیے الگورتھم کی تفصیلات لکھیں۔

11th_Comp Science_UM_Ch 6.indd 80 18-04-2023 14:23:54

1. آئیے الگورتھم کا نام square_root دیں۔
2. یہ نمبر کو ان پٹ کے طور پر لیتا ہے۔ آئیے ان پٹ کا نام n دیں۔ n منفی نہیں ہونا چاہئے۔
3. یہ آؤٹ پٹ کے طور پر n کا اسکوائر- روٹ پیدا کرتا ہے۔ آئیے آؤٹ پٹ کو y کا نام دیں۔ پھر n کو y کا مربع ہونا چاہیے۔

اب الگورتھم کی تصریح ہے۔

اسکوائر_روٹ (n)

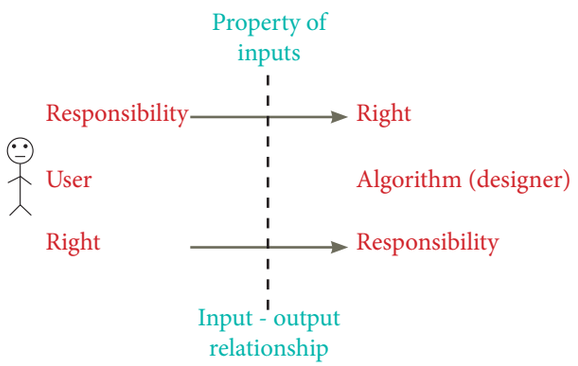
-- ان پٹس: n ایک حقیقی نمبر ہے، $n \geq 0$۔

-- آؤٹ پٹس : y ایک حقیقی نمبر ہے جیسے کہ $n = 2 y$۔

6.5.1 معاہدے کے طور پر تفصیلات (Specification as contract)

الگورتھم کی تفصیلات الگورتھم کے ڈیزائنر اور الگورتھم کے صارفین کے درمیان ایک معاہدے کے طور پر کام کرتی ہیں، کیونکہ یہ ڈیزائنر اور صارف کے حقوق اور ذمہ داریوں کی وضاحت کرتی ہے۔

اس بات کو یقینی بنانا کہ ان پٹ مطلوبہ خصوصیات کو پورا کرتے ہیں صارف کی ذمہ داری ہے، لیکن ڈیزائنر کا حق ہے۔ مطلوبہ ان پٹ آؤٹ پٹ تعلق ڈیزائنر کی ذمہ داری اور صارف کا حق ہے۔ اہم بات یہ ہے کہ اگر صارف ان پٹ کی خصوصیات کو پورا کرنے میں ناکام رہتا ہے، تو ڈیزائنر مطلوبہ ان پٹ -آؤٹ پٹ (input-output) تعلق کو پورا کرنے کی اپنی ذمہ داری سے آزاد ہے۔



تصویر 6.2 : ان پٹ پراپرٹی اور ان پٹ آؤٹ پٹ کا تعلق بطور حقوق اور ذمہ داریاں

مثال 6.8 ۔ الگورتھم اسکوئر_روٹ کی تفصیلات پر غور کریں۔

مربع_روٹ (n)

-- ان پٹس: n ایک حقیقی نمبر ہے، $n \geq 0$۔

-- آؤٹ پٹس : y ایک حقیقی نمبر ہے جیسا کہ $n = y^2$۔

الگورتھم ڈیزائنر یہ فرض کر سکتا ہے کہ دیا گیا نمبر غیر منفی ہے، اور الگورتھم بناتا ہے۔ صارف توقع کر سکتا ہے کہ آؤٹ پٹ دیے گئے نمبر کا اسکوئر_روٹ ہو گا۔

آؤٹ پٹ دیے گئے نمبر کا منفی اسکوئر_روٹ ہو سکتا ہے۔ تفصیلات نے اس بات کا ارتکاب نہیں کیا کہ آؤٹ پٹ مثبت اسکوئر_روٹ ہے۔ اگر صارف ان پٹ کے بطور منفی نمبر پاس کرتا ہے، تو آؤٹ پٹ کو نمبر کا اسکوئر_روٹ ہونا ضروری نہیں ہے۔

6.6 ابسٹریکشن (علیحدگی / نظر انداز) (Abstraction)

سائیکل (bicycle) چلانے کے لیے پیڈل (pedal)، ہینڈل بار، بریک اور بیل کے کام کو سمجھنا کافی ہے۔ ایک سوار کے طور پر، ہم ان چار خصوصیات کے ذریعے سائیکل کا ماڈل بناتے ہیں۔ ایک سائیکل میں بہت ساری تفصیلات ہوتی ہیں، جنہیں سوار نظر انداز کر سکتا ہے۔ وہ تفصیلات سائیکل چلانے کے مقصد سے غیر متعلق ہیں۔

ایک مسئلہ میں بہت ساری تفصیلات شامل ہوسکتی ہیں۔ ان میں سے کئی تفصیلات مسئلے کو حل کرنے کے لیے غیر متعلقہ ہیں۔ صرف چند تفصیلات ضروری ہیں۔ابسٹریکشن غیر متعلقہ تفصیلات کو نظر انداز کرنے یا چھپانے کا عمل ہے اور کسی مسئلے کو صرف اس کی ضروری خصوصیات سے ماڈلنگ کرتا ہے۔ ہماری روزمرہ کی زندگی میں، ہم پیچیدگی کو سنبھالنے کے لیے لاشعوری طور پر علیحدگی کا استعمال کرتے ہیں۔ابسٹریکشن پیچیدگی کے انتظام کے لیے استعمال ہونے والا سب سے مؤثر ذہنی ٹول ہے۔ اگر ہم کسی مسئلے کو مناسب طریقے سے ابسٹریکشن نہیں کرتے ہیں، تو ہم غیر ضروری تفصیلات سے نمٹ سکتے ہیں اور حل کو پیچیدہ بنا سکتے ہیں۔

11th_Comp Science_UM_Ch 6.indd 81 18-04-2023 14:23:55

مثال 6.9 نقشہ (Map) ان چیزوں کا ابسٹریکشن ہے جو ہم زمین پر پاتے ہیں۔ ہم زمین پر ہر تفصیل کی نمائندگی نہیں کرتے ہیں۔ نقشہ بنانے والا ان تفصیلات کو چنتا ہے جن کے بارے میں ہمیں جاننے کی ضرورت ہے۔ مختلف نقشے مختلف مقاصد کے لیے بنائے جاتے ہیں اور اس لیے مختلف ابسٹریکشنس (Abstractions) کا استعمال کرتے ہیں، یعنی وہ مختلف خصوصیات کو چھپاتے ہیں یا ان کی نمائندگی کرتے ہیں۔ سڑک کا نقشہ ڈرائیوروں کے لیے بنایا گیا ہے۔ وہ عام طور پر پہاڑیوں کے بارے میں فکر نہیں کرتے ہیں لہذا زیادہ تر پہاڑیوں کو سڑک کے نقشے پر نظر انداز کیا جاتا ہے۔ ایک واکر کا نقشہ اس بات میں دلچسپی نہیں رکھتا کہ آیا سڑک ایک طرفہ گ لی ہے، اس لیے اس طرح کی تفصیلات کو نظر انداز کر دیا جاتا ہے۔

مثال 6.10 طب میں، مختلف ماہرین انسانی جسم کے مختلف ابسٹریکشنس کے ساتھ کام کرتے ہیں۔ ایک آرتھوپیڈیشن (Orthopaedician) کنکال کے نظام (Skeletal system) کے ابسٹریکشن کے ساتھ کام کرتا ہے، جبکہ ایک معدے کا ماہر (Gastroenterologist) نظام انہضام (Digestive system) کے ساتھ کام کرتا ہے۔ ایک فزیو تھراپسٹ (Physiotherapist) انسانی جسم کو اس کے عضلاتی نظام (Muscular system) سے ابسٹریکشن کرتا ہے۔

ہم الگورتھم کی تعمیر کے دوران مختلف طریقوں سے ابسٹریکشن کا استعمال کرتے ہیں — مسائل کی تصریح میں، متغیرات کے ذریعے حالت کی نمائندگی کرتے ہوئے، اور ایک الگورتھم کو فنکشنز کے لیے تحلیل کرتے ہیں۔ ایک الگورتھم ڈیزائنر کو یہ پہچاننے کے لیے تربیت دی جانی چاہیے کہ مسئلہ کو حل کرنے کے لیے کون سی خصوصیات ضروری ہیں، اور کون سی تفصیلات غیر ضروری ہیں۔ اگر ہم غیر ضروری تفصیلات شامل کرتے ہیں، تو یہ مسئلہ اور اس کے حل کو زیادہ پیچیدہ بنا دیتا ہے۔

تصریح ان پٹ کی ابسٹریک اور مطلوبہ ان پٹ آؤٹ پٹ تعلق سے مسئلہ کو ختم کرتی ہے۔ ہم مسئلے کو حل کرنے کے لیے ضروری خصوصیات کو پہچانتے ہیں، اور غیر ضروری تفصیلات کو نظر انداز کرتے ہیں۔

حالت (State): الگورتھم میں، حساب کی حالت کو متغیرات کے سیٹ سے ابسٹریکٹ (Abstract) کیا جاتا ہے۔

فنکشنز (Functions) : جب ایک الگورتھم بہت پیچیدہ ہوتا ہے، تو ہم اسے فنکشنز میں تحلیل کر سکتے ہیں اور ہر فنکشن کو اس کی تفصیلات کے مطابق ابسٹریک کر سکتے ہیں۔

6.6.1 اسٹیٹ (State)

اسٹیٹ ایک بنیادی اور اہم ابسٹریکشن ہے۔ کمپیوٹیشنل عمل کی حالت ہوتی ہے۔ ایک کمپیوٹیشنل عمل ابتدائی حالت سے شروع ہوتا ہے۔ جیسے جیسے اعمال ہوتے ہیں، اس کی حالت بدل جاتی ہے۔ یہ ایک حتمی حالت کے ساتھ ختم ہوتا ہے۔ الگورتھم میں متغیرات کے سیٹ کے ذریعے عمل کی حالت کوابسٹریک کیا جاتا ہے۔ عمل کے کسی بھی مقام پراسٹیٹ صرف اس مقام پر متغیرات کی قدر ہے۔

مثال 6.11 چوکلیٹ بارس (Chocolate bars): ایک مستطیل چاکلیٹ بار کو افقی اور عمودی نالیوں سے مربع میں تقسیم کیا جاتا ہے۔ ہم بار کو انفرادی مربع میں توڑنا چاہتے ہیں۔

شروع کرنے کے لیے، ہمارے پاس پوری بار ایک ہی ٹکڑے کے طور پر ہے۔ ایک ٹکڑا منتخب کرکے اور اسے اس کی نالیوں میں سے ایک کے ساتھ توڑ کر کٹ بنایا جاتا ہے۔ اس طرح ایک کٹ ایک ٹکڑے کو دو ٹکڑوں میں تقسیم کرتی ہے۔ بار کو اس کے انفرادی مربع میں توڑنے کے لیے کتنے کٹس (cuts) کی ضرورت ہے؟

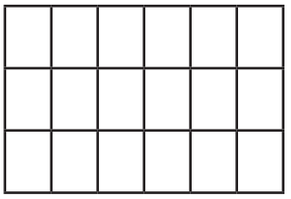
اس مثال میں، ہم مسئلہ کے ضروری متغیرات کا ابسٹریک کریں گے۔ ہم مثال 8.6 میں مسئلہ حل کرتے ہیں۔

ضروری متغیرات (Essential variables): ٹکڑوں کی تعداد اور کٹوتیوں کی تعداد مسئلے کے ضروری متغیرات ہیں۔ ہم ان کی نمائندگی بالترتیب دو متغیرات p اور c سے کریں گے۔ اس طرح، عمل کی حالت کو دو متغیرات p اور c سے ابسٹریک کیا جاتا ہے۔

غیر متعلقہ تفصیلات: (Irrelevant details)

1. مسئلہ چاکلیٹ بار کو انفرادی ٹکڑوں میں کاٹنا یا ڈاک ٹکٹوں کی ایک شیٹ کو انفرادی ڈاک ٹکٹوں میں کاٹنا ہو سکتا ہے۔ یہ غیر متعلقہ ہے۔ مسئلہ صرف مربع کے گرڈ (grid of squares) کو انفرادی مربع میں کاٹ رہا ہے۔

11th_Comp Science_UM_Ch 6.indd 82 18-04-2023 14:23:55



2. ٹوٹی کی ترتیب اور نتیجے میں آنے والے ٹکڑوں کی شکلیں اور سائز بھی غیر متعلق ہیں۔ p اور c سے، ہم انفرادی ٹکڑوں کے سائز کو دوبارہ نہیں بنا سکتے۔ لیکن، یہ مسئلہ حل کرنے سے متعلق نہیں ہے۔

مثال 6.12 مثال 6.2، بکری، گھاس اور بھیڑیے کے مسئلے پر غور کریں۔ اس مثال میں،ہم مسئلہ کی وضاحت لکھیں گے۔ ہم اسے مثال 7.1 میں حل کریں گے۔ اس مسئلے میں چار افراد شامل ہیں، اور ہر ایک دریا کے دونوں کناروں میں سے ایک پر ہے۔ اس کا مطلب ہے کہ ہم حالت کی نمائندگی چار متغیرات سے کر سکتے ہیں، اور ان میں سے ہر ایک کی دو قدروں میں سے ایک ہے۔ آئیے ہم متغیرات کو کسان، بکری، گھاس اور بھیڑیے کے نام دیتے ہیں، اور ان کی ممکنہ قدریں L اور R۔ L کی قدر کا مطلب ہے "بائیں طرف"۔ R کی قدر کا مطلب ہے "دائیں طرف"۔ چونکہ کشتی ہمیشہ کسان کے ساتھ ہوتی ہے، اس لیے اس کی پوزیشن کی نمائندگی کرنے کے لیے متغیر متعارف کرانا ضروری نہیں ہے۔

ابتدائی حالت میں، چاروں متغیرات کسان، بکری، گھاس، بھیڑیا کی قدر L ہے۔

کسان، بکری، گھاس، بھیڑیا = L , L, L, L

آخری حالت میں، چاروں متغیرات کی قدر R ہونی چاہیے۔

کسان، بکری، گھاس، بھیڑیا = R, R, R, R

مسئلہ کی تفصیلات cross_river ہے۔

-- ان پٹس: کسان، بکری، گھاس، بھیڑیا = L, L, L, L

-- آؤٹ پٹس: کسان، بکری، گھاس، بھیڑیا = R, R, R, R

دو پابندیوں کے تابع ہے کہ

1. بکری کو گھاس کے ساتھ تنہا نہیں چھوڑا جا سکتا:

اگر بکری = گھاس تو کسان = بکری

2. بکری کو بھیڑیے کے ساتھ تنہا نہیں چھوڑا جا سکتا:

اگر بکری = بھیڑیا تو کسان = بکری

6.6.2 اسائنمنٹ اسٹیٹمینٹ (Assignment Statement)

اقدار کو ذخیرہ کرنے کے لیے متغیرات کو بکس کا نام دیا گیا ہے۔ تفویض کا بیان متغیر میں کسی قدر کو ذخیرہ کرنے کے لیے استعمال کیا جاتا ہے۔ یہ اسائنمنٹ آپریٹر کے بائیں جانب متغیر اور دائیں جانب ایک قدر کے ساتھ لکھا جاتا ہے۔

variable := value

جب اس تفویض کو انجام دیا جاتا ہے، تو دائیں طرف کی قدر بائیں جانب متغیر میں محفوظ ہو جاتی ہے۔ اسائنمنٹ

m := 2

متغیر m میں ویلیو 2 اسٹور کر تا ہے

m

2

اگر متغیر میں پہلے سے ہی ایک قدر ذخیرہ ہے تو، اسائنمنٹ اپنی قدر کو دائیں طرف کی قدر میں تبدیل کر دیتی ہے۔ متغیر کی پرانی قدر ختم ہو گئی ہے۔

کسی تفویض کا دائیں جانب اکسپریشن (Expression) ہو سکتا ہے۔

variable := expression

اس صورت میں، اکسپریشن کا اندازہ لگایا جاتا ہے اور اکسپریشن کی قدر متغیر میں محفوظ کی جاتی ہے۔ اگر اکسپریشن میں متغیر موجود ہے تو، متغیر کی موجودہ قدر اکسپریشن کی تشخیص میں استعمال ہوتی ہے، اور پھر متغیر کو اپ ڈیٹ کیا جاتا ہے۔ مثال کے طور پر، تفویض

m := m + 3

11th_Comp Science_UM_Ch 6.indd 83 18-04-2023 14:23:55

m کی موجودہ قدر کا استعمال کرتے ہوئے اکسپریشن m + 3 کا اندازہ کرتا ہے۔

```
m + 3
= 2 + 3
= 5
```

اور قدر 5 کو متغیر m میں محفوظ کرتا ہے۔



تفویض بیان کے دونوں اطراف کو علامت کے ذریعہ الگ کیا جاتا ہے :=، جو اسائنمنٹ آپریٹر (Assignment operator) کے نام سے جانا جاتا ہے، اور " بیکمس – becomes " یا " اسائنڈ – assigned " کے طور پر پڑھا جاتا ہے۔ تفویض کا بیان

```
v := e
```

"بیکمس" v، e کے طور پر پڑھا جاتا ہے۔ نوٹ کریں کہ اسائنمنٹ آپریٹر مساوات آپریٹر 1 نہیں ہے۔ v = : e اور v = e کے معنی مختلف ہیں۔ تفویض ایک متغیر کی ریاضیاتی مساوات بیان نہیں کرتا، لیکن متغیر کی قدر کو تبدیل کرتا ہے۔ تفویض m := m + 3 یہ نہیں بتاتا کہ mm + 3 کے برابر ہے۔ بلکہ یہ متغیر m کی قدر کو m + 3 کی قدر میں تبدیل کرتا ہے۔

ایک تفویض بیان ایک ساتھ متعدد متغیرات کی قدروں کو تبدیل کر سکتا ہے۔ اس صورت میں، بائیں جانب متغیرات کی تعداد دائیں جانب کے اکسپریشن کی تعداد سے مماثل ہونی چاہیے۔ مثال کے طور پر، اگر ہم تین متغیر v1، v2 اور v3 کو تفویض کرنا چاہتے ہیں، تو ہمیں 3 اکسپریشن کی ضرورت ہے، کہیے، e1, e2, e3۔

```
vl, v2, v3 := el, e2 , e3
```

بائیں طرف متغیرات کی کوما (Comma) سے الگ کردہ فہرست ہے۔ دائیں طرف اکسپریشن کی ایک کوما سے الگ کردہ فہرست ہے۔ عمل کرنے کی ایک تفویض بیان، پہلے متغیرات کی موجودہ قدروں کا استعمال کرتے ہوئے دائیں طرف کے تمام اکسپریشن کا جائزہ لیں، اور پھر انہیں بائیں جانب متعلقہ متغیرات میں محفوظ کریں۔

مثال 6.13 میں تفویض کے بعد متغیر m اور n کی قدریں کیا ہیں؟ لائن (1) اور لائن (3)؟

```
1. m, n := 2 , 5
2. ? , ? = m, n --
3. m,n:=m+3,n-1
4. ? , ? = m, n --
```

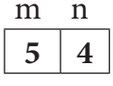
لائن میں تفویض (1) متغیر m میں 2 اور متغیر n میں 5 ذخیرہ کرتا ہے۔



لائن (3) میں تفویض m + 3 اور n – 1 کی موجودہ قدروں کو m اور n کے طور پر استعمال کرتے ہوئے اکسپریشن کی جانچ کرتا ہے۔

```
m + 3, n - 1
= 2 + 3 , 5 - 1
=5 ,4
```

اور قدر 5 اور 4 کو بالترتیب m اور n متغیر میں محفوظ کرتا ہے۔




```
1. m, n := 2, 5
2. m, n = 2 , 5 --
3. m, n := m + 3, n - 1
4. m, n = 2 + 3, 5-1 = 5, 4 --
```

دو اسائنمنٹس کے بعد متغیرات کی قدریں لائن (2) اور لائن (4) میں دکھائی جاتی ہیں۔

---

^بدقسمتی سے، کئی پروگرامنگ زبانیں، بشمول C++، علامت = کو بطور اسائنمنٹ آپریٹر استعمال کرتی ہیں، اور اس لیے، ایک اور علامت، == بطور اِکوالٹی آپریٹر(equality operator)۔

11th_Comp Science_UM_Ch 6.indd 84 18-04-2023 14:23:55

مثال 6.14 مثال 6.11 میں، ہم نے عمل کی حالت کو دو متغیروں p اور c سے خلاصہ کیا۔ اگلا مرحلہ چاکلیٹ بار کو کاٹنے کے عمل کو ماڈل بنانا ہے۔ جب ہم کسی ٹکڑے کا ایک ہی کٹ بناتے ہیں تو ٹکڑوں کی تعداد (p) اور کٹوں کی تعداد(c) دونوں میں 1 کا اضافہ ہوتا ہے۔

ہم اسے اسائنمنٹ اسٹیٹمنٹ (assignment statement) کے ذریعے ماڈل بنا سکتے ہیں۔

p, c := p + 1, c+1

جسے بالترتیب p اور p + 1 "become" c اور c + 1 کے طور پر پڑھا جاتا ہے۔

یاد رکھنے کے نکات:

- ایک پروگرامنگ زبان بنیادی بیانات اور مرکب بیانات کی تحریر کے لیے ایک اشارے فراہم کرتی ہے۔
- ایک الگورتھم کسی مسئلے کو حل کرنے کے لیے بیانات کی ایک مرحلہ وار ترتیب ہے۔
- جیسا کہ ایک الگورتھم کو عمل میں لایا جاتا ہے، ایک عمل تیار ہوتا ہے جو مسئلہ کو حل کرتا ہے۔
- الگورتھم کے مسئلے کو حل کرنے میں الگورتھم کی تعمیر کے ساتھ ساتھ الگورتھم کی خصوصیات کو ثابت کرنا شامل ہے۔
- ایک الگورتھم کی تصریح الگورتھم کے نام پر مشتمل ہوتی ہے (اس کے ان پٹ کے ساتھ)، ان پٹ پراپرٹی، اور مطلوبہ ان پٹ آؤٹ پٹ تعلق۔
- الگورتھم کی تفصیلات الگورتھم کے ڈیزائنر اور صارفین کے درمیان ایک معاہدہ ہے۔
- ابسٹریکشن (abstraction) سے غیر متعلقہ تفصیلات کو چھپانے یا نظر انداز کرنے کا عمل ہے تاکہ کسی مسئلے کو صرف اس کی ضروری خصوصیات سے ہی ماڈل بنایا جا سکے۔
- ابسٹریکس (abstracts) مسئلہ کے ضروری متغیرات کے ذریعہ مسئلہ کو خلاصہ کرتی ہے۔
- الگورتھم میں متغیرات کی قدریں عمل کی حالت کی وضاحت کرتی ہیں۔
- تفویض کا بیان متغیرات کی قدروں کو تبدیل کرتا ہے، اور اسی طرح حالت بھی تبدیل کرتا ہے۔





تشخیص

سیکشن – A

صحیح جواب کا انتخاب کریں

1. درج ذیل میں سے کون سی سرگرمیاں الگورتھمک نوعیت کی ہیں؟

(a) ایک سائیکل جمع کریں۔ (b) سائیکل کی وضاحت کریں۔
(c) سائیکل کے پرزوں پر لیبل لگائیں۔ (d) وضاحت کریں کہ سائیکل کیسے کام کرتی ہے۔

2. مندرجہ ذیل میں سے کون سی سرگرمیاں فطرت میں الگورتھم نہیں ہیں؟

(a) دو نمبروں کو ضرب دیں۔ (b) ایک کولم کھینچیں۔
(c) پارک میں چہل قدمی کریں۔ (d) دو نمبروں کا تبادلہ۔

3. کام کے لیے غیر ضروری تفصیلات کو چھوڑنا اور کام کی صرف ضروری خصوصیات کی نمائندگی کرنا کہلاتا ہے۔

(a) تفصیلات (b) ابسٹریکشن (c) کمپوزیشن (d) ڈی کمپوزیشن

4. ان پٹ پراپرٹی اور ان پٹ آؤٹ پٹ کے تعلق کو بتانے سے مسئلہ معلوم ہوتا ہے۔

(a) تفصیلات (b) بیان (c) الگورتھم (d) تعریف

11th_Comp Science_UM_Ch 6.indd 85 18-04-2023 14:23:55

5. ان پٹ آؤٹ پٹ کے تعلق کو یقینی بنانا

(a) الگورتھم کی ذمہ داری اور صارف کا حق۔

(b) صارف کی ذمہ داری اور الگورتھم کا حق۔

(c) الگورتھم کی ذمہ داری لیکن صارف کا حق نہیں۔

(d) صارف اور الگورتھم دونوں کی ذمہ داری۔

6. اگر اسائنمنٹ سے پہلے i = 5 i := i-1 اسائنمنٹ کے بعد، i کی قدر ہے

5 (a) 4 (b) 3 (c) 2 (d)

7. اگر اسائنمنٹ سے پہلے i>0 اسائنمنٹ کے بعد i := i-1 تو ہم یہ نتیجہ اخذ کر سکتے ہیں کہ

0 < i (a) 0 ≤ i (b) i = 0 (c) 0 ≥ i (d)

سیکشن – B

بہت ہی مختصر جوابات دیں۔

1. الگورتھم کی وضاحت کریں۔
2. الگورتھم اور عمل کے درمیان فرق کریں۔
3. ابتدائی طور پر،

L, L, L, L = کسان، بکری، گھاس، بھیڑیا

اور کسان بکری کے ساتھ دریا پار کرتا ہے۔ اسائنمنٹ سٹیٹمنٹ کے ساتھ کارروائی کا نمونہ بنائیں۔

4. کم از کم دو نمبروں کو تلاش کرنے کے لیے ایک فنکشن کی وضاحت کریں۔
5. اگر $\sqrt{2} = 1.414$;، اور square_root() فنکشن 1.414- لوٹاتا ہے، تو کیا یہ مندرجہ ذیل تفصیلات کی خلاف ورزی کرتا ہے؟

-- مربع_روٹ (x)

-- ان پٹس: x ایک حقیقی نمبر ہے، $x \geq 0$

-- آؤٹ پٹس: y ایک حقیقی نمبر ہے جیسا کہ $y^2=x$

سیکشن – C

مختصر جوابات دیں

1. آپ کب کہتے ہیں کہ ایک مسئلہ فطرت میں الگورتھم ہے؟
2. الگورتھم کی تفصیلات کا فارمیٹ کیا ہے؟
3. ابسٹریکشن کیا ہے؟
4. الگورتھم میں حالت کی نمائندگی کیسے کی جاتی ہے؟
5. تفویض بیان کی شکل اور معنی کیا ہے؟
6. اسائنمنٹ آپریٹر اور مساوات آپریٹر میں کیا فرق ہے؟

11th_Comp Science_UM_Ch 6.indd 86 18-04-2023 14:23:55

سیکشن – D

تفصیل سے بتائیں

1. ایک الگورتھم ہائپوٹینوز (hypotenuse) کی تفصیلات لکھیں جس کے ان پٹ دائیں زاویہ والے مثلث کے دو چھوٹے اطراف کی لمبائی ہیں، اور آؤٹ پٹ تیسری طرف کی لمبائی ہے۔

2. فرض کریں کہ آپ ایک الگورتھم کے ذریعے چوکور مساوات ax2 + bx + c = 0 کو حل کرنا چاہتے ہیں۔

quadratic_solve (a, b, c)

-- inputs : ?

-- outputs: ?

آپ فارمولہ استعمال کرنے کا ارادہ رکھتے ہیں اور آپ صرف حقیقی نمبر کی جڑوں کو ہینڈل کرنے کے لیے تیار ہیں۔ ایک مناسب تصریح لکھیں۔

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

3. مشمولات کا تبادلہ کریں: A اور B کے نشان والے دو گلاس دیئے جائیں۔ گلاس A ایپل ڈرنک سے بھرا ہوا ہے اور گلاس B انگور کے مشروب سے بھرا ہوا ہے۔ شیشے A اور B کے مواد کے تبادلے کے لیے، مناسب متغیرات کے ذریعے حالت کی نمائندگی کریں، اور الگورتھم کی تفصیلات لکھیں۔

کتابیں (Books)

1. Roland Backhouse, Algorithmic Problem Solving, John Wiley & Sons Ltd, 2011.
2. Krysia Broda, Susan Eisenbach, Hessam Khoshnevisan, Steve Vickers, Reasoned Programming, Prentice Hall, 1994

11th_Comp Science_UM_Ch 6.indd 87 18-04-2023 14:23:55

کمپوزیشن اور ڈی کمپوزیشن

سیکھنے کے مقاصد

اس باب میں تصورات کو سیکھنے کے بعد، طلباء کے قابل ہو جائیں گے۔

- الگورتھم تکنیکوں میں استعمال ہونے والے اشارے کو جاننے
- الگورتھمک تکنیکوں میں کمپوزیشن اور ڈی کمپوزیشن کو سمجھنے

باب 1 میں، ہم نے دیکھا کہ الگورتھم بیانات پر مشتمل ہیں۔ بیانات کو مرکب بیانات میں گروپ کیا جا سکتا ہے، جو الگورتھم کے درجہ بندی کی ساخت کو جنم دیتا ہے۔ ڈی کمپوزیشن مسئلہ حل کرنے کی ابتدائی تکنیکوں میں سے ایک ہے۔ ایک الگورتھم کو حصوں میں تقسیم کیا جا سکتا ہے، جس میں صرف اعلیٰ سطحی تفصیلات کا اظہار ہوتا ہے۔ پھر، ہر حصے کو چھوٹے حصوں میں بہتر کیا جا سکتا ہے، باریک تفصیلات کا اظہار کیا جا سکتا ہے، یا ہر حصے کو ایک فنکشن کے طور پر خلاصہ کیا جا سکتا ہے۔

7.1 الگورتھم کے لیے اشارے (Notations for Algorithms)

ہمیں الگورتھم کی نمائندگی کرنے کے لیے ایک اشارے کی ضرورت ہے۔ الگورتھم کی نمائندگی کے لیے بنیادی طور پر تین مختلف اشارے ہیں۔

- ایک پروگرامنگ لینگویج کمپیوٹرز کے ذریعے انجام دینے والے الگورتھم کے اظہار کے لیے ایک اشارہ ہے۔
- سیوڈو کوڈ (pseudo code) پروگرامنگ لینگویجز کی طرح ایک اشارے ہے۔ سیوڈو کوڈ میں ظاہر کیے گئے الگورتھم کا مقصد کمپیوٹر کے ذریعے نہیں بلکہ لوگوں کے درمیان رابطے کے لیے ہے۔
- فلو چارٹ (flow chart) الگورتھم کی نمائندگی کرنے کے لیے ایک خاکہ نما اشارہ ہے۔ وہ فلو چارٹ کی ایک بصری بصیرت دیتے ہیں، جب الگورتھم کو عمل میں لایا جاتا ہے۔

7.1.1 پروگرامنگ لینگویج (Programming Language)



ایک پروگرامنگ لینگویج الگورتھم کے اظہار کے لیے ایک اشارہ ہے تاکہ کمپیوٹر الگورتھم کو چلا سکے۔ ایک الگورتھم کا اظہار پروگرامنگ زبان میں پروگرام کہلاتا ہے۔ C، C++ اور Python پروگرامنگ زبانوں کی مثالیں ہیں۔ پروگرامنگ زبان رسمی ہے۔ پروگراموں کو پروگرامنگ لینگویج کے گرامر کو بالکل ماننا چاہیے۔یہاں تک کہ اوقاف کی علامتیں بھی درست ہونی چاہئیں۔ وہ انگریزی یا تامل جیسی قدرتی زبانوں کے غیر رسمی انداز کی اجازت نہیں دیتے۔ ایک مترجم ہے جو پروگرام کو کمپیوٹر کے ذریعے قابل عمل ہدایات میں ترجمہ کرتا ہے۔ اگر ہمارے پروگرام میں گرامر کی غلطیاں ہیں تو یہ مترجم ترجمہ نہیں کر سکے گا۔

7.1.2 سیوڈو کوڈ (Pseudo code)

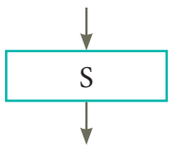
پروگرامنگ کا سیڈوکوڈ زبان جیسی تعمیرات اور سادہ انگریزی کا مجموعہ ہے۔ یہ اشارے نہ رسمی ہے اور نہ ہی درست۔ یہ وہی بلڈنگ بلاکس استعمال کرتا ہے جیسے پروگرام، جیسے متغیرات اور کنٹرول فلو۔ لیکن، یہ بیانات اور حالات کے لیے قدرتی انگریزی کے استعمال کی اجازت دیتا ہے۔ سیوڈو کوڈ کے طور پر ظاہر کردہ الگورتھم کمپیوٹرز کے لیے براہ راست عمل کرنے کے لیے نہیں ہے، بلکہ انسانی قارئین کے سمجھنے کے لیے ہے۔اس لیے کسی پروگرامنگ لینگویج کے گرامر کے اصولوں پر عمل کرنے کی ضرورت نہیں ہے۔ تاہم، سیڈوکوڈ بھی سخت اور درست ہونا چاہیے۔ سیوڈو کوڈ الگورتھم کی نمائندگی کے لیے سب سے زیادہ استعمال ہونے والا اشارہ ہے۔

11th_Comp Science_UM_Ch 7.indd 88 18-04-2023 14:42:09

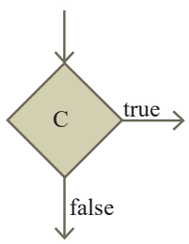
7.1.3 فلو چارٹس (Flow charts)

فلو چارٹ الگورتھم کی نمائندگی کرنے کے لیے ایک خاکہ نما اشارہ ہے۔ وہ بصری انداز میں ڈایاگرام کا استعمال کرتے ہوئے الگورتھم کے کنٹرول کے بہاؤ کو دکھاتے ہیں۔ فلو چارٹس میں، مستطیل خانے سادہ بیانات کی نمائندگی کرتے ہیں، ہیرے کے سائز کے خانے حالات کی نمائندگی کرتے ہیں، اور تیر (arrows) یہ بتاتے ہیں کہ الگورتھم کے عمل کے دوران کنٹرول کس طرح کام کرتا ہے۔ فلو چارٹ بکسوں کا ایک مجموعہ ہے جس میں بیانات اور شرائط شامل ہیں جو تیروں کے ذریعہ جڑے ہوئے ہیں جس ترتیب میں خانوں کو انجام دیا جانا ہے۔

1. ایک بیان ایک مستطیل خانے میں ایک واحد آؤٹ گوئنگ ایرو (out going arrow) کے ساتھ ہوتا ہے، جو اس باکس کی طرف اشارہ کرتا ہے جسے اگلا کیا جانا ہے۔



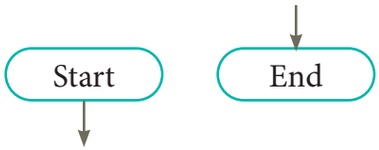
2. ایک شرط ہیرے (diamond) کی تصویر والے باکس میں دو باہر جانے والے تیروں کے ساتھ موجود ہے، جس پر سچ اور غلط کا لیبل لگا ہوا ہے۔ صحیح تیر اس باکس کی طرف اشارہ کرتا ہے جس پر عمل درآمد کیا جائے اگر شرط درست ہے، اور جھوٹا تیر اس باکس کی طرف اشارہ کرتا ہے جس پر عمل درآمد کیا جائے گا اگر شرط غلط ہے۔



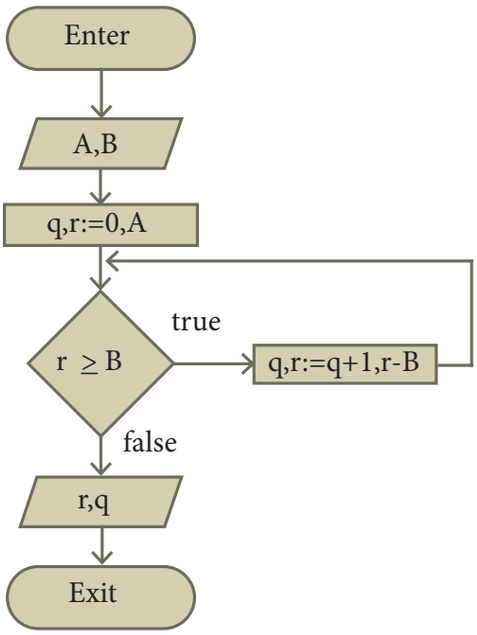
3. متوازی الاضلاع باکسز (Parallelogram boxes) دیے گئے ان پٹ اور تیار کردہ آؤٹ پٹ کی نمائندگی کرتے ہیں۔



4. آغاز اور اختتام کے نشان والے خصوصی خانوں کا استعمال کسی عمل کے آغاز اور اختتام کو ظاہر کرنے کے لیے کیا جاتا ہے:



عدد A کو دوسرے عدد B سے تقسیم کرنے کے بعد ایک الگورتھم کا فلو چارٹ تصویر 7.1 میں دکھایا گیا ہے، مختلف خانوں جیسے کہ ان پٹ، آؤٹ پٹ، حالت، اور تفویض، اور خانوں کے درمیان کنٹرول فلو کو ظاہر کرتا ہے۔ الگورتھم کی وضاحت مثال 7.4 میں کی گئی ہے۔



تصویر 7.1: عددی تقسیم کے لیے فلو چارٹ

فلو چارٹس کے بھی نقصانات ہیں۔ (1) فلو چارٹس پروگرامنگ لینگویج یا سیڈو کوڈ میں الگورتھم کی نمائندگی سے کم کمپیکٹ (compact) ہیں۔ (2) وہ الگورتھم کی بنیادی درجہ بندی کی ساخت کو غیر واضح کرتے ہیں۔ (3) متبادل بیانات اور لوپس (loops) نظم و ضبط کے کنٹرول فلو کے ڈھانچے ہیں۔ فلو چارٹس ہمیں نظم و ضبط کے کنٹرول فلو کے ڈھانچے تک محدود نہیں کرتے ہیں۔

11th_Comp Science_UM_Ch 7.indd 89 18-04-2023 14:42:09

7.2 کمپوزیشن
(Composition)

بیان ایک جملہ ہے جو کمپیوٹر کو ایک عمل کرنے کا حکم دیتا ہے۔ ہم پہلے ہی اسائنمنٹ کا بیان دیکھ چکے ہیں۔ یہ ایک سادہ سا بیان ہے، جو متغیرات کی قدروں کو تبدیل کرنے کے لیے استعمال ہوتا ہے۔ بیانات دوسرے بیانات پر مشتمل ہوسکتے ہیں، جو الگورتھم کی درجہ بندی کی ساخت کا باعث بنتے ہیں۔ دوسرے بیانات پر مشتمل بیانات کو مرکب بیانات (compound statements) کہا جاتا ہے۔

کنٹرول فلو کے بیانات مرکب بیانات ہیں۔ وہ عمل کی حالت کے لحاظ سے عمل کے کنٹرول فلو کو تبدیل کرنے کے لیے استعمال ہوتے ہیں۔ تین اہم کنٹرول فلو بیانات ہیں:

- ترتیب وار (sequential)
- متبادل (Alternative)
- اٹریٹیو (Iterative)

جب ایک کنٹرول فلو سٹیٹمنٹ پر عمل درآمد کیا جاتا ہے، تو عمل کی حالت کی جانچ کی جاتی ہے، اور نتیجہ پر منحصر ہے، ایک بیان کو عمل درآمد کے لیے منتخب کیا جاتا ہے۔

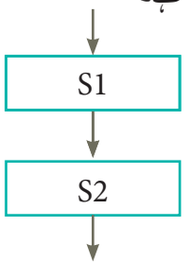
7.2.1 سلسلہ وار بیان
(sequential statement)

ایک ترتیب وار بیان بیانات کی ترتیب پر مشتمل ہے۔ ترتیب میں بیانات کو یکے بعد دیگرے عمل میں لایا جاتا ہے، اسی ترتیب میں جیسے وہ الگورتھم میں لکھے جاتے ہیں، اور کنٹرول فلو کو ترتیب وار کہا جاتا ہے۔ S1 اور S2 کو اسٹیٹمنٹس ہونے دیں۔ S1 اور S2 پر مشتمل ایک ترتیب وار بیان اس طرح لکھا جاتا ہے۔

S1
S2

ترتیب وار بیان کو انجام دینے کے لیے، پہلے S1 کریں اور پھر S2 کریں۔

اوپر دیئے گئے ترتیب وار بیان کو فلو چارٹ میں دکھایا جا سکتا ہے جیسا کہ تصویر 7.2 میں دکھایا گیا ہے۔ S1 سے S2 تک کا تیر اشارہ کرتا ہے کہ S1 کو عمل میں لایا گیا ہے، اور اس کے بعد، S2 کو عمل میں لایا گیا ہے۔



تصویر 7.2 : ترتیب وار کنٹرول فلو

کسی مسئلے کے لیے ان پٹ کی خاصیت کو P، اور ان پٹ آؤٹ پٹ کا تعلق Q ہونے دیں۔ اگر بیان S مسئلہ حل کرتا ہے، تو یہ لکھا جاتا ہے۔

1. P --
2. S
3. Q --

اگر ہم مسئلہ کو دو اجزاء میں تحلیل کرتے ہیں، تو ہمیں دو بیانات S1 اور S2 کی ترتیب کے طور پر S تحریر کرنے کی ضرورت ہے۔

S1 اور S2 کا ان پٹ - آؤٹ پٹ رشتہ، R کہئے، جو S2 کی ان پٹ پراپرٹی ہے۔

1. P --
2. S1
3. R --
4. S2
5. Q --

مثال 7.1۔ آیئے مثال کے طور پر کسان، بکری، گھاس اور بھیڑیا کا مسئلہ 6.2 حل کریں۔ ہم نے چار متغیر کسان، بکری، گھاس، اور بھیڑیے کے ذریعے عمل کی حالت کی نمائندگی کرنے کا فیصلہ کیا، جو بالترتیب کسان، بکری، گھاس اور بھیڑیا کے اطراف کی نمائندگی کرتے ہیں۔ ابتدائی حالت میں، چاروں متغیرات کی قدر L (بائیں طرف) ہوتی ہے۔ آخری حالت میں، چاروں متغیرات کی قدر R (دائیں طرف) ہونی چاہیے۔ مقصد یہ ہے کہ ایک اسٹیٹمنٹ S کو تشکیل دیا جائے تاکہ ابتدائی حالت سے آخری حالت میں جا سکے۔

11th_Comp Science_UM_Ch 7.indd 90 18-04-2023 14:42:09

1. -- کسان، بکری، گھاس، بھیڑیا = L, L, L, L

2. S

3. -- کسان، بکری، گھاس، بھیڑیا = R, R, R, R

ہمیں تفویض بیانات کی ترتیب کے طور پر S کو تحریر کرنا ہے جیسے کہ کسی بھی درمیانی حالت میں موجود نہ ہو۔

1. بکری اور بھیڑیے کی قدر ایک جیسی ہے لیکن کسان کی قدر برعکس ہے، یا
2. بکری اور گھاس کی قدر یکساں ہے لیکن کسان کی قدر برعکس ہے۔ ان رکاوٹوں کے تابع، اسائنمنٹ کا ایک سلسلہ اور ہر اسائنمنٹ کے بعد کی حالت کو تصویر 7.3 میں دکھایا گیا ہے۔

1. -- کسان، بکری، گھاس، بھیڑیا = L، L، L، L
2. کسان، بکری : = R, R
3. -- کسان، بکری، گھاس، بھیڑیا = L، L، R، R
4. کسان: = L
5. کسان، بکری، گھاس، بھیڑیا = L، L، R، L
6. کسان، گھاس : = R, R
7. -- کسان، بکری، گھاس، بھیڑیا = L، R، R، R
8. کسان، بکری : = L, L
9. -- کسان، بکری، گھاس، بھیڑیا = L، L، R، L
10. کسان، بھیڑیا: = R, R
11. -- کسان، بکری، گھاس، بھیڑیا = R، L، R، R
12. کسان : = L
13. -- کسان، بکری، گھاس، بھیڑیا = L، L، R، R
14. کسان، بکری : = R, R
15. -- کسان، بکری، گھاس، بھیڑیا = R، R، R، R

تصویر 7.3 : بکری، گھاس اور بھیڑیے کے مسئلے کے لیے تفویض کی ترتیب۔

لائنوں (1) اور (15) کے علاوہ، لائن (7) میں، بکری اور گھاس کی قدر ایک جیسی ہے، لیکن کسان کی بھی وہی قدر ہے۔ لائن (9) میں، بکری اور بھیڑیے کے پاس ہے۔ ایک ہی قدر، لیکن کسان کی بھی وہی قدر ہے جو ان کی ہے۔ اس طرح، تسلسل نے رکاوٹوں کی خلاف ورزی کیے بغیر، مقصد کی حالت حاصل کر لی ہے۔

7.2.2 متبادل بیان (Alternative statement)

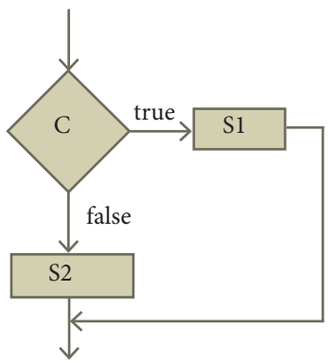
شرط ایک جملہ ہے جو حالت کے امتحان کو بیان کرتا ہے۔ اگر C شرط ہے اور دونوں S1 اور S2 بیانات ہیں، پھر

```
if   C
     S1
else
     S2
```

ایک بیان ہے، جسے متبادل بیان کہا جاتا ہے، جو درج ذیل عمل کو بیان کرتا ہے:

1. جانچ کریں کہ C صحیح ہے یا غلط۔
2. اگر C صحیح ہے، تو S1 کریں؛ دوسری صورت میں S2 کریں۔

سیڈو کوڈ (pseudo code) میں، دو متبادلات S1 اور S2 کو بالترتیب if اور else مطلوبہ الفاظ سے انڈینٹنگ (indenting) کرکے اشارہ کیا جاتا ہے۔متبادل کنٹرول فلو کو تصویر 2.4 کے فلو چارٹ میں دکھایا گیا ہے۔ کنڈیشن C میں دو سبکدوش ہونے والے تیر ہیں، درست اور غلط کا لیبل لگا ہوا ہے۔ حقیقی تیر S1 باکس کی طرف اشارہ کرتا ہے۔ غلط تیر S2 باکس کی طرف اشارہ کرتا ہے۔ S1 اور S2 کے تیر متبادل بیان کے بعد والا خانہ، ایک ہی باکس کی طرف اشارہ کرتے ہیں ۔



تصویر 7.4: متبادل کنٹرول فلو

11th_Comp Science_UM_Ch 7.indd 91 18-04-2023 14:42:09

مشروط بیان: (Conditional statement)

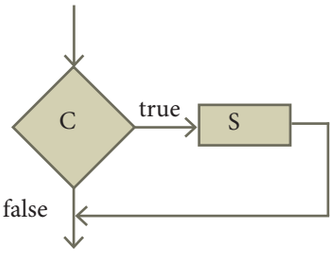
بعض اوقات ہمیں صرف اس صورت میں ایک بیان پر عمل کرنے کی ضرورت ہوتی ہے جب کوئی شرط درست ہو اور اگر شرط غلط ہو تو کچھ نہ کریں۔یہ اس متبادل بیان کے برابر ہے جس میں دوسری شق خالی ہے۔ متبادل بیان کی اس قسم کو مشروط بیان کہا جاتا ہے۔ اگر C شرط ہے اور S ایک بیان ہے، تو

if C
S

ایک بیان ہے، جسے مشروط بیان کہا جاتا ہے، جو درج ذیل عمل کو بیان کرتا ہے:

1. جانچ کریں کہ C صحیح ہے یا غلط۔
2. اگر C صحیح ہے تو S کریں؛ ورنہ کچھ نہ کریں۔

مشروط کنٹرول فلو کو تصویر 2.5 کے فلو چارٹ میں دکھایا گیا ہے۔



تصویر 7.5: مشروط کنٹرول فلو

مثال 7.2 دو نمبروں کی کم از کم: دو نمبرز a اور b کو دیکھتے ہوئے، ہم متبادل بیان کا استعمال کرتے ہوئے دو میں سے کم از کم تلاش کرنا چاہتے ہیں۔ آئیے کم از کم کو ایک متغیر نام کے نتیجے میں ذخیرہ کرتے ہیں۔ ایک a ↓ b کو کم از کم a اور b کی نشاندہی کرنے دیں (مثال کے طور پر (4 ↓ 2 = 2, -5 ↓ 6 = −5) ۔ پھر، الگورتھم منیمم (Algorithm minimum) کی تصریح ہے۔

```
minimum(a, b)
-- input s : a , b
-- outputs: result = a ↓ b
```

الگورتھم منیمم (Algorithm minimum) کے طور پر بیان کیا جا سکتا ہے۔

```
1.  minimum (a, b)
2.  -- a, b
3.  if a < b
4.  result : = a
5.  else
6.  result = b
7.  -- result = a ↓ b
```

7.2.3 کیس کا تجزیہ (Case Analysis)

متبادل بیان مسئلہ کو دو صورتوں میں تجزیہ کرتا ہے۔ کیس کے تجزیہ کا بیان اسے متعدد معاملات میں عام کرتا ہے۔ کیس کا تجزیہ مسئلہ کو الگ الگ کیسز کے ایک مکمل سیٹ میں تقسیم کرتا ہے۔ ہر معاملے میں، مسئلہ آزادانہ طور پر حل کیا جاتا ہے. اگر C1، C2 اور C3 شرائط ہیں، اور S1، S2، S3 اور S4 بیانات ہیں، تو 4 - کیس تجزیہ بیان کی تصویر ہے،

```
1.  case C1
2.  S1
3.  C2
4.  S2
5.  C3
6.  S3
7.  else
8.  S4
```

حالات C1، C2 اور C3 کا بدلے میں جائزہ لیا جاتا ہے۔ پہلی شرط کے لیے جو صحیح میں جانچتی ہے، متعلقہ بیان پر عمل درآمد کیا جاتا ہے، اور کیس کے تجزیہ کا بیان ختم ہو جاتا ہے۔ اگر کوئی بھی شرط درست نہیں ہوتی ہے، تو ڈیفالٹ کیس S4 پر عمل درآمد کیا جاتا ہے۔

1. مقدمات مکمل ہیں: مقدمات میں سے کم از کم ایک صحیح ہے۔ اگر تمام شرائط غلط ہیں، تو ڈیفالٹ کیس درست ہے۔
2. مقدمات متضاد ہیں: مقدمات میں سے صرف ایک درست ہے۔ اگرچہ ایک سے زیادہ شرطوں کا درست ہونا ممکن ہے،

11th_Comp Science_UM_Ch 7.indd 92 18-04-2023 14:42:09

لیکن کیس کا تجزیہ ہمیشہ صرف ایک کیس کو انجام دیتا ہے، پہلی صورت جو صحیح ہے۔ اگر تین شرائط متضاد ہیں، تو چار صورتیں ہیں (C1)1، (2) C2، (3)C3، (4) (C1 نہیں) اور (C2 نہیں) اور (C3 نہیں)۔

مثال 7.3 ہم ایک الگورتھم چاہتے ہیں جو دو نمبروں کا آپس میں موازنہ کرے اور نتیجہ اس طرح پیش کرے۔

$$\text{compare}(a, b) = \begin{cases} 1- & \text{if } a < b \\ 0 & \text{if } a = b \\ 1 & \text{if } a > b \end{cases}$$

ہم حالت کو 3 الگ الگ کیسوں کے ایک مکمل سیٹ میں تقسیم کر سکتے ہیں: a < b, a = b and a > b پھر ہم ایک کیس کے تجزیہ کا استعمال کرتے ہوئے ()compare کی وضاحت کر سکتے ہیں۔

```
1.  compare(a, b)
2.      case a < b
3.          result := -1
4.      case a = b
5.          result := 0
6.      else -- a > b
7.          result : = 1
```

7.2.4 آئی ٹریٹیو اسٹیٹمنٹ (اعادہ بیان) (Iterative statement)

ایک آئی ٹریٹیو عمل ایک ہی عمل کو بار بار شرط C کے ساتھ انجام دیتا ہے ۔ اگر C شرط ہے اور S ایک بیان ہے، تو پھر

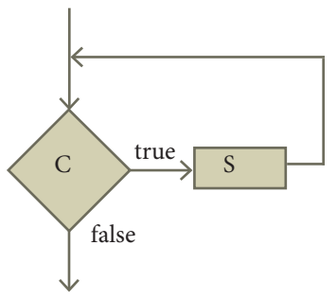
```
while C
    S
```

ایک بیان ہے، جسے آئی ٹریٹیو اسٹیٹمنٹ کہا جاتا ہے، جو درج ذیل عمل کو بیان کرتا ہے:

1. جانچ کریں کہ C صحیح ہے یا غلط۔
2. اگر C صحیح ہے، تو S کریں اور مرحلہ 1 پر واپس جائیں۔ ورنہ کچھ نہ کریں۔

آئی ٹریٹیو اسٹیٹمنٹ کو عام طور پر لوپ (loop) کے نام سے جانا جاتا ہے۔ یہ دو مراحل، جانچ C اور عمل کرنے والے S کو اس وقت تک دہرایا جاتا ہے جب تک C غلط نہ ہو جائے۔ جب C غلط ہو جاتا ہے، لوپ ختم ہو جاتا ہے، اور کنٹرول آئی ٹریٹیو اسٹیٹمنٹ کے آگے بیان کی طرف جاتا ہے۔ شرط C اور سٹیٹمنٹ S کو بالترتیب لوپ کنڈیشن (loop Condition) اور لوپ باڈی (loop body) کہا جاتا ہے۔ لوپ کی حالت کی جانچ کرنا اور لوپ باڈی کو ایک بار انجام دینے کو آئی ٹریشن (Iteration) کہا جاتا ہے۔ not C کو ٹرمینیشن کنڈیشن (Termination Condition) طور پر جانا جاتا ہے۔

آئی ٹریٹیو کنٹرول فلو کو تصویر 7.6 کے فلو چارٹ میں دکھایا گیا ہے۔ حالت C میں دو باہر جانے والے تیر ہیں، صحیح اور غلط۔ حقیقی تیر S باکس کی طرف اشارہ کرتا ہے۔ اگر C درست ہے تو، S باکس کو عمل میں لایا جائے گااور کنٹرول فلو C باکس میں واپس چلا جاتا ہے۔ غلط تیر (false arrow) آئی ٹریٹیو اسٹیٹمنٹ (نقطہ خانہ) کے بعد باکس کی طرف اشارہ کرتا ہے۔ اگر C غلط ہے، تو لوپ ختم ہو جاتا ہے اور کنٹرول لوپ کے بعد اگلے باکس میں چلا جاتا ہے۔



تصویر 7.6 : آئی ٹریٹیو کنٹرول فلو

مثال 7.4۔ ایک عدد A کو دوسرے عدد B سے تقسیم کرنے کے بعد اقتباس اور بقیہ کی گنتی کے لیے ای آئی ٹریٹیو الگورتھم بنائیں۔

ہم نے مثال کے طور پر 6.6 میں الگورتھم کی تفصیلات مرتب کی ہیں۔

```
divide (A , B)
-- inputs:  A is an integer and B ≠ 0
-- outputs :  q and r such that A = q X B
      + r and
--    0 ≤ r < B
```

اب ہم ایک آئی ٹریٹیو الگورتھم بنا سکتے ہیں جو تفصیلات کو پورا کرتا ہے۔

11th_Comp Science_UM_Ch 7.indd 93 18-04-2023 14:42:10

divide (A , B)

-- inputs: A is an integer and B ≠ 0

-- outputs : q and r such that A = q X B + r and

-- 0 < r < B

q, r : = 0, A

while r ≥ B

q, r := q + 1, r - B

الگورتھم کو تصویر 7.1 میں فلو چارٹ کے طور پر پیش کیا گیا ہے۔

ہم ٹیسٹ ان پٹ کے لیے مرحلہ وار الگورتھم کو عمل میں لا سکتے ہیں، بولیں، (A, B) = (22, 5)۔ ٹیبل 7.1 کی ہر قطار ایک آئی ٹریشن دکھاتی ہے۔ — آئی ٹریشن کے اختتام پر اظہارات اور متغیرات کی قدروں کی تشخیص۔ نوٹ کریں کہ اظہار کی تشخیص پچھلی قطار سے متغیر کی قدروں کا استعمال کرتی ہے۔ آؤٹ پٹ متغیر q اور r ہر آئی ٹریشن میں اپنی قدروں کو تبدیل کرتے ہیں۔ ان پٹ متغیرات A اور B اپنی اقدار کو تبدیل نہیں کرتے ہیں۔ تکرار 0 لوپ شروع ہونے سے ٹھیک پہلے اقدار کو ظاہر کرتا ہے۔ تکرار 4 کے آخر میں، حالت (B ≤ r) = (2 ≤ r) غلط ہے، اور اس لیے لوپ = (4,2) (q, r) کے ساتھ ختم ہوتا ہے۔

iteration	q	q+1	r	r-B	A	B
0	0		22		22	5
1	1	0+1	17	22-5		
2	2	1 + 1	12	17-5		
3	3	2+1	7	12-5		
4	4	3+1	2	7-5		

ٹیبل 7.1: تقسیم کا مرحلہ وار عمل

(22, 5)

مثال 7.5. مثال 1.3 کے کروم لینڈ (Chromeland) کے گرگٹ (Chameleons) کے مسئلے میں، فرض کریں کہ دو قسم کے گرگٹ تعداد میں برابر ہیں۔ ایک الگورتھم بنائیں جو ان دو اقسام کے درمیان ملاقاتوں کا اہتمام کرے تاکہ وہ اپنا رنگ تیسری قسم میں تبدیل کر لیں۔ آخر میں، سب کو ایک ہی رنگ دکھانا چاہیے۔

آیئے ہم ہر قسم کے گرگٹ کی تعداد کو متغیرات a ، b اور c کے ذریعے اور ان کی ابتدائی قدروں کو بالترتیب A ، B اور C سے ظاہر کرتے ہیں۔ a = b کو ان پٹ پراپرٹی (input property) بننے دیں۔ ان پٹ آؤٹ پٹ کا تعلق a = b = 0 اور c = A+B+C ہے۔ آیئے الگورتھم کو مونوکرومیٹائز (monochromatize) کا نام دیں۔ الگورتھم کے طور پر بیان کیا جا سکتا ہے۔

monochromatize(a, b, c)

-- inputs: a=A, b=B, c=C, a=b

-- outputs : a = b = 0 , c = A+B+C

ہر آئی ٹریٹیو مرحلے میں، دو قسم کے دو گرگٹ (تعداد میں برابر) ملتے ہیں اور اپنے رنگوں کو تیسرے میں تبدیل کرتے ہیں۔ مثال کے طور پر، اگر A, B, C = 4, 4, 6، تو ملاقاتوں کا سلسلہ نتیجہ خیز ہو گا۔

iteration	a	b	c
0	4	4	6
1	3	3	8
2	2	2	10
3	1	1	12
4	0	0	14

ٹیبل 7.2: گرگٹ کی دو اقسام کے درمیان ملاقاتوں کا سلسلہ برابر تعداد میں۔

ہر میٹنگ میں، a اور b ہر ایک میں 1 کی کمی ہوتی ہے، اور c میں 2 کا اضافہ ہوتا ہے۔ حل کو ایک آئی ٹریٹیو الگورتھم کے طور پر ظاہر کیا جا سکتا ہے۔

مونوکروموٹائز (a,b,c) (monochromatize)

-- inputs: a=A, b=B, c=C, a=b

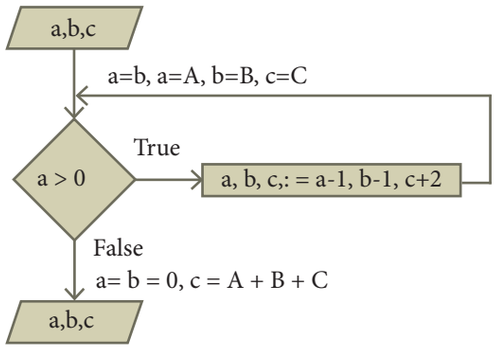
-- outputs: a = b = 0, c = A+B+C

while a > 0

a, b, c := a-1, b-1, c+2

الگورتھم کو تصویر 7.7 کے فلو چارٹ میں دکھایا گیا ہے۔

11th_Comp Science_UM_Ch 7.indd 94 18-04-2023 14:42:10



تصویر 7.7: الگورتھم مونوکروموٹائز (monochromatize)

7.3 ڈی کمپوزیشن (Decomposition)

ڈی کمپوزیشن مسئلہ حل کرنے کی ابتدائی تکنیکوں میں سے ایک ہے۔ اس میں کسی مسئلے کو چھوٹے اور زیادہ قابل انتظام مسائل میں تقسیم کرنا، اور چھوٹے مسائل کے حل کو جوڑ کر اصل مسئلہ کو حل کرنا شامل ہے۔ اکثر مسائل کی ساخت ہوتی ہے۔ ہم مسئلے کی ساخت کا استحصال کر سکتے ہیں اور اسے چھوٹے مسائل میں توڑ سکتے ہیں۔ پھر چھوٹے مسائل کو مزید توڑا جاسکتا ہے جب تک کہ وہ دوسرے آسان ذرائع سے حل کرنے کے لیے کافی چھوٹے نہ ہوجائیں۔ اس کے بعد ان کے حل کو ایک ساتھ ملا کر اصل مسئلہ کا حل نکالا جاتا ہے۔

7.3.1 ریفائنمنٹ (Refinement)

کسی مسئلے کو چھوٹے ذیلی مسائل میں تحلیل کرنے کے بعد، اگلا مرحلہ یا تو ذیلی مسئلے کو بہتر کرنا ہے یا ذیلی مسئلے کو خلاصہ کرنا ہے۔

1. ہر ذیلی مسئلے کو مزید تفصیلی مراحل میں بڑھایا جا سکتا ہے۔ ہر قدم کو مزید بہتر قدموں تک بڑھایا جا سکتا ہے، وغیرہ۔ یہ ریفائنمنٹ (تطہیر) کے طور پر جانا جاتا ہے۔
2. ہم ذیلی مسئلہ کو بھی خلاصہ کر سکتے ہیں۔ ہم ہر ذیلی مسئلے کو اس کی ان پٹ پراپرٹی اور ان پٹ آؤٹ پٹ کے تعلق سے بیان کرتے ہیں۔ اہم مسئلے کو حل کرتے وقت، ہمیں صرف ذیلی مسائل کی تفصیلات جاننے کی ضرورت ہے۔ ہمیں یہ جاننے کی ضرورت نہیں ہے کہ ذیلی مسائل کیسے حل ہوتے ہیں۔

مثال 7.6 صبح اسکول جانے والے کی کارروائی پر غور کریں۔ کارروائی کے طور پر لکھا جا سکتا ہے

1 اسکول کے لیے تیار ہو جائیں۔

ہم اس کارروائی کو چھوٹے، زیادہ قابل انتظام کارروائی کے اقدامات میں تحلیل کر سکتے ہیں جو وہ ترتیب میں لیتی ہے:

1 ناشتہ کھائیں (Eat breakfast)
2 کپڑے پہنیں (Put on clothes)
3 گھر سے نکلیں (Leave home)

ہم نے ایک عمل کو اعمال کی تفصیلی ترتیب میں بہتر کیا ہے۔ تاہم، ان میں سے ہر ایک عمل کو مزید تفصیلی سطح پر اعمال کی ترتیب میں بڑھایا جا سکتا ہے، اور اس توسیع کو دہرایا جا سکتا ہے۔ کارروائی "ناشتہ کھاؤ" کے طور پر توسیع کی جا سکتی ہے۔

1 -- ناشتہ کھائیں
2 اڈلی کھائیں
3 انڈے کھائیں
4 کیلے کھائیں

کارروائی " کپڑے پہنیں " کو اس طرح توسیع کی جا سکتی ہے۔

1 -- کپڑے پہنیں
2 نیلا لباس پہنیں
3 موزے اور جوتے پہنیں
4 شناختی کارڈ پہنیں

اور " گھر سے نکلیں " کو اس طرح توسیع کی جا سکتی ہے ۔

1 -- گھر سے نکلیں
2 سائیکل کو باہر نکالیں
3 سائیکل کو چلائیں

اس طرح، "اسکول کے لیے تیار ہوجائیں" کی پوری کارروائی کو اس طرح بہتر بنایا گیا ہے۔

1 -- ناشتہ کھائیں
2 اڈلی کھائیں
3 انڈے کھائیں
4 کیلے کھائیں

11th_Comp Science_UM_Ch 7.indd 95 18-04-2023 14:42:10

5

6 --کپڑے پہنیں

7 نیلا لباس پہنیں

8 موزے اور جوتے پہنیں

9 شناختی کارڈ پہنیں

10

11 -- گھر سے نکلیں

12 سائیکل کو باہر نکالیں

13 سائیکل کو چلائیں

ریفائنمنٹ ہمیشہ اعمال کی ترتیب نہیں ہوتی۔ طالب علم جو کچھ کرتا ہے اس کا انحصار ماحول پر ہو سکتا ہے۔ وہ ناشتہ کیسے کھاتی ہے اس کا انحصار اس بات پر ہے کہ اسے کتنی بھوک لگی ہے اور میز پر کیا ہے۔ وہ کون سے کپڑے پہنتی ہے اس کا انحصار ہفتے کے دن پر ہوتا ہے۔ ہم مشروط (conditional) اور آئی ٹریٹیو بیانات کا استعمال کرتے ہوئے اس طرز عمل کو بہتر کرسکتے ہیں جو ماحول پر منحصر ہے۔

1 -- ناشتہ کھائیں

2 اگر بھوک لگی ہو اور دسترخوان پر اڈلی ہو

3 اڈلی کھائیں

4 اگر بھوک لگی ہو اور میز پر انڈے ہوں

5 انڈے کھائیں

6 اگر بھوک لگی ہو اور میز پر کیلے ہوں

7 کیلے کھائیں

8

9 -- کپڑے پہنیں

10 اگر چہارشنبہ ہے

11 نیلا لباس پہنیں

12 ورنہ

13 سفید لباس پہنیں

14 موزے اور جوتے پہنیں

15 شناختی کارڈ پہنیں

16

17 -- گھر سے نکلیں

18 سائیکل کو باہر نکالیں

19 سائیکل کو چلائیں

عمل "اڈلیاں کھاؤ" کو ایک آئی ٹریٹیو عمل کے طور پر مزید بہتر کیا جا سکتا ہے:

1 -- اڈلیاں کھائیں

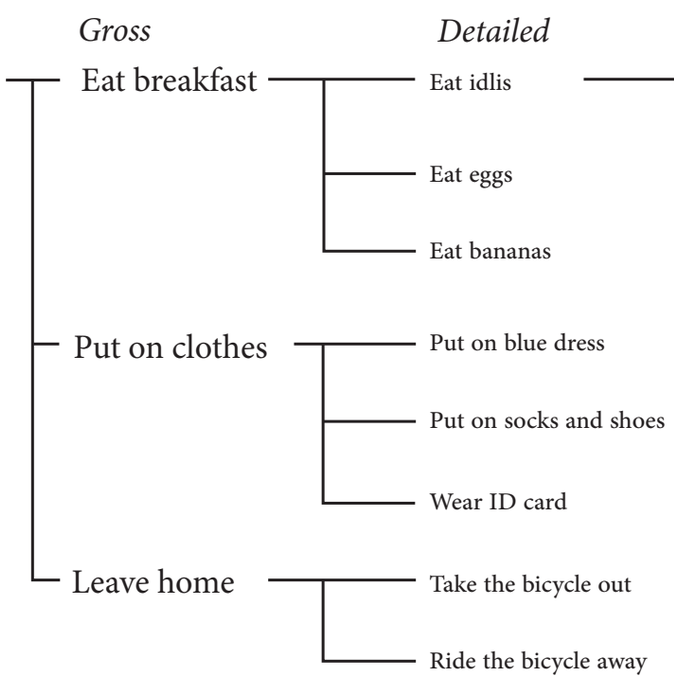
2 اڈلیوں کو پلیٹ میں رکھ دیں

3 چٹنی شامل کریں

4 جبکہ اڈلیاں پلیٹ میں ہوں

5 اڈلی کا ایک ٹکڑا کھائیں

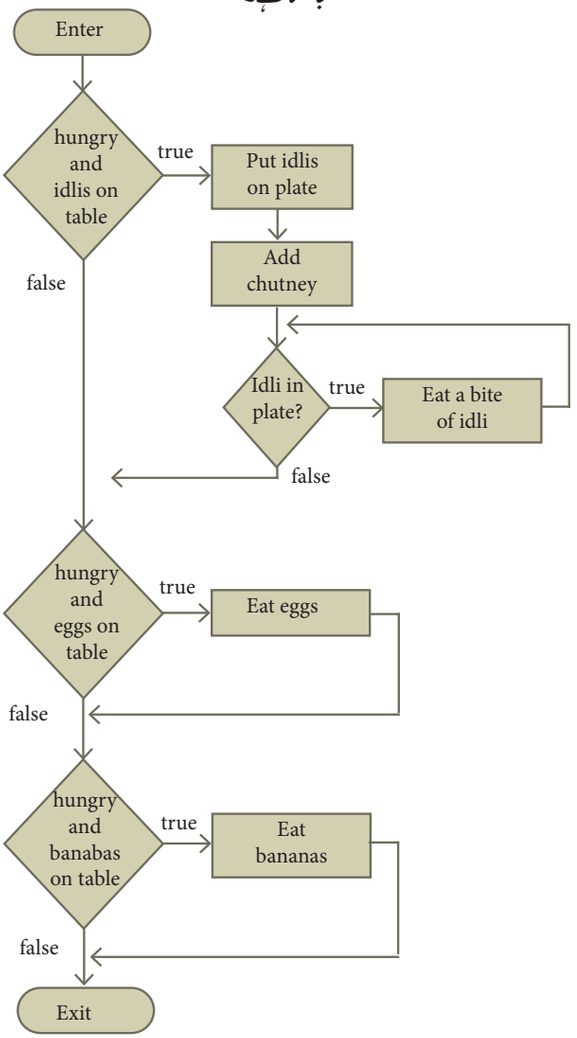
" اسکول کے لیے تیار ہو جائیں" کو پے در پے سطحوں میں کیسے بہتر کیا جاتا ہے، تصویر 2.8 میں دکھایا گیا ہے۔



شکل 7.8: تفصیلات کی مختلف سطحوں پر ریفائنمنٹ *(Refinement)*

11th_Comp Science_UM_Ch 7.indd 96 18-04-2023 14:42:10

عمل "ناشتہ کھائیں" کو تصویر 2.9 میں دکھایا گیا فلو چارٹ میں دیکھا جاسکتا ہے۔

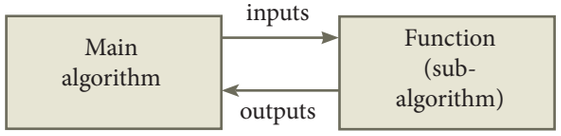


تصویر 7.9: ناشتہ کھانے کا فلو چارٹ

نوٹ کریں کہ فلو چارٹ ریفائنمنٹ کی درجہ بندی کی ساخت نہیں دکھاتا ہے۔

7.3.2 فنکشنس (Functions)

الگورتھمک مسئلہ ذیلی مسائل میں ڈی کمپوزڈ ہونے کے بعد، ہم ذیلی مسائل کو فنکشنز کے طور پر خلاصہ کر سکتے ہیں۔ ایک فنکشن ایک ذیلی الگورتھم کی طرح ہے۔ الگورتھم کی طرح، ایک فنکشن ان پٹ پراپرٹی، اور مطلوبہ ان پٹ آؤٹ پٹ تعلق کے ذریعہ بیان کیا جاتا ہے۔



تصویر 7.10: فنکشن کی تعریف

مرکزی الگورتھم میں کسی فنکشن کو استعمال کرنے کے لیے، صارف کو صرف فنکشن کی تفصیلات جیسے فنکشن کا نام، ان پٹ پراپرٹی، اور ان پٹ آؤٹ پٹ کا تعلق جاننے کی ضرورت ہوتی ہے ۔ صارف کو اس بات کو یقینی بنانا چاہیے کہ فنکشن کو بھیجے گئے ان پٹس مخصوص پراپرٹی کو پورا کریں گے اور یہ فرض کر سکتے ہیں کہ فنکشن کے آؤٹ پٹ ان پٹ آؤٹ پٹ کے تعلق کو پورا کرتے ہیں۔ اس طرح، فنکشن کے صارفین کو صرف یہ جاننے کی ضرورت ہوتی ہے کہ فنکشن کیا کرتا ہے، نہ کہ فنکشن کے ذریعے یہ کیسے کیا جاتا ہے۔ فنکشن کو دیگر مسائل کو حل کرنے میں ایک "بلیک باکس (black box) " استعمال کیا جا سکتا ہے۔

بالآخر، کوئی الگورتھم کا استعمال کرتے ہوئے فنکشن کو نافذ کرتا ہے۔ تاہم، فنکشن کے صارفین کو فنکشن کو لاگو کرنے کے لیے استعمال ہونے والے الگورتھم کے بارے میں جاننے کی ضرورت نہیں ہے۔ یہ صارفین سے پوشیدہ ہے۔ صارفین کو یہ جاننے کی ضرورت نہیں ہے کہ اسے استعمال کرنے کے لیے فنکشن کو کیسے لاگو کیا جاتا ہے۔

فنکشن کو نافذ کرنے کے لیے استعمال ہونے والا الگورتھم اپنے متغیرات کو برقرار رکھ سکتا ہے۔ یہ متغیرات فنکشن کے لیے مقامی ہیں اس لحاظ سے کہ وہ فنکشن کے صارف کو نظر نہیں آتے۔ نتیجتاً، صارف کے پاس مرکزی الگورتھم میں برقرار رکھنے کے لیے کم متغیرات ہوتے ہیں، جو مرکزی الگورتھم کی بے ترتیبی کو کم کرتے ہیں۔

مثال 7.7 جانچ کے مسئلے پر غور کریں کہ آیا ایک مثلث دائیں زاویہ ہے، اس کے تین اطراف a, b, c, جہاں c سب سے لمبا رخ ہے۔ مثلث دائیں زاویہ ہے، اگر

$$a^2 + b^2 = c^2$$

ہم عدد کو مربع کرنے کے ذیلی مسئلے کی نشاندہی کر سکتے ہیں۔ فرض کریں کہ ہمارے پاس ایک فنکشن اسکوائر () ہے، جس کی وضاحت کی گئی ہے۔

square(y)

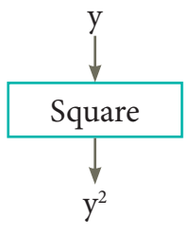
-- **inputs : y**

-- **outputs : y^2**

11th_Comp Science_UM_Ch 7.indd 97 18-04-2023 14:42:10

```
1   right_angled(a, b, c)
2   -- inputs: c ≥ a, c ≥ b
3   -- outputs: result = true if c² = a² + b²;
4   --          result = false , otherwise
5    if square (c) = square (a) + square (b)
6       result := true
7   else
8       result := false
```

ہم اس فنکشن کو تین بار یہ جانچنے کے لیے استعمال کر سکتے ہیں کہ آیا کوئی مثلث دائیں زاویہ ہے یا نہیں۔ square() ایک "بلیک باکس" ہے — ہمیں یہ نہیں معلوم کہ فنکشن مربع کی گنتی کیسے کرتا ہے۔ ہمیں صرف اس کی تفصیلات جاننے کی ضرورت ہے۔



تصویر 7.11: اسکوائر فنکشن

یاد رکھنے کے نکات

- مرکب بیانات (compound statement) ترتیب وار، متبادل اور آئی ٹریٹیو کنٹرول فلو کے بیانات پر مشتمل ہوتے ہیں۔
- کسی شرط کی قدر درست یا غلط ہے، متغیرات کی قدروں پر منحصر ہے۔
- متبادل بیان (Alternative statement) شرط کی قدر کے لحاظ سے، دو بیانات میں سے بالکل ایک کا انتخاب اور عمل درآمد کرتا ہے۔
- مشروط بیان (Conditional statement) صرف اس صورت میں عمل میں لایا جاتا ہے جب شرط درست ہو۔ ورنہ کچھ نہیں کیا جاتا۔
- آئی ٹریٹیو بیان (Iterative statement) بار بار کسی شرط کا جائزہ لیتا ہے اور جب تک شرط درست ہو تب تک بیان پر عمل درآمد کرتا ہے۔
- پروگرامنگ لینگویج، سیوڈو کوڈ، اور فلو چارٹ الگورتھم کے اظہار کے لیے اشارے ہیں۔
- ڈی کمپوزیشن (Decomposition) کسی مسئلے کو چھوٹے ذیلی مسائل میں تقسیم کرتا ہے اور اصل مسئلے کو حل کرنے کے لیے ان کے حل کو یکجا کرتا ہے۔
- ایک فنکشن ایک ذیلی مسئلہ کا خلاصہ ہے، اور اس کی ان پٹ خاصیت، اور اس کے ان پٹ آؤٹ پٹ تعلق سے مخصوص ہوتا ہے۔
- فنکشن کے صارفین کو صرف یہ جاننے کی ضرورت ہے کہ فنکشن کیا کرتا ہے، نہ کہ یہ کیسے کیا جاتا ہے۔
- ریفائنمنٹ (Refinement) میں، اعلیٰ سطح سے شروع ہو کر، ہر بیان کو بار بار بعد کی سطحوں میں مزید تفصیلی بیانات میں پھیلایا جاتا ہے۔

تشخیص

سیکشن – A

صحیح جواب کا انتخاب کریں

1. فرض کریں اسائنمنٹ سے پہلے u, v = 10,5 ہے ۔ اسائن منٹس کی ترتیب کے بعد u اور v کی قدریں کیا ہیں؟

u:=v 1

v:=u 2

11th_Comp Science_UM_Ch 7.indd 98 18-04-2023 14:42:10

(a) u, v = 5, 5 (c) u, v = 10, 5

(b) u, v = 5, 10 (d) u, v = 10, 10

2. مندرجہ ذیل خصوصیات میں سے کون سی اسائنمنٹ کے بعد درست ہے (لائن 3 پر؟)

```
1  --i, j = 0, 0
2  i, j := i+1, j-1
3  -- ?
```

(a) i+j >0 (b) i+j < 0 (c) i+j =0 (d) i = j

3. اگر C1 غلط ہے اور C2 صحیح ہے تو، مرکب بیان

```
1  if C1
2     S1
3  else
4  if  C2
5     S2
6  else
7     S3
```

executes

(a) S1 (b) S2 (c) S3 (d) none

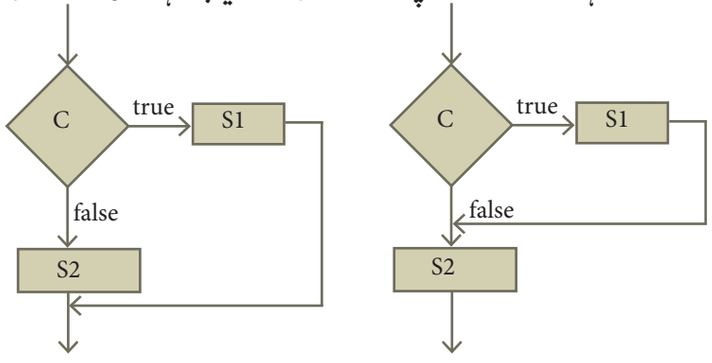
4. اگر C لوپ سے بالکل پہلے غلط ہے، تو کنٹرول گزرتا ہے بذریعہ

```
1  S1
2  while C
3  S2
4  S3
```

(a) S1 ; S3 (b) S1 ; S2 ; S3

(c) S1 ; S2 ; S2 ; S3 (d) S1 ; S2 ; S2 ; S2 ; S3

5. اگر C صحیح ہے، S1 کو دونوں فلو چارٹ میں عمل میں لایا جاتا ہے، لیکن S2 کو عمل میں لایا گیا ہے



(1) (2)

(a) (1) only (b) (2) only

(c) both (1) and (2) (d) neither (1) nor (2)

11th_Comp Science_UM_Ch 7.indd 99 18-04-2023 14:42:10

6. لوپ کو کتنی بار دوہرایا جاتا ہے؟

```
i := 0
while i ≠ 5
    i := i + 1
```

4 (a) 5 (b) 6 (c) 0 (d)

سیکشن – B

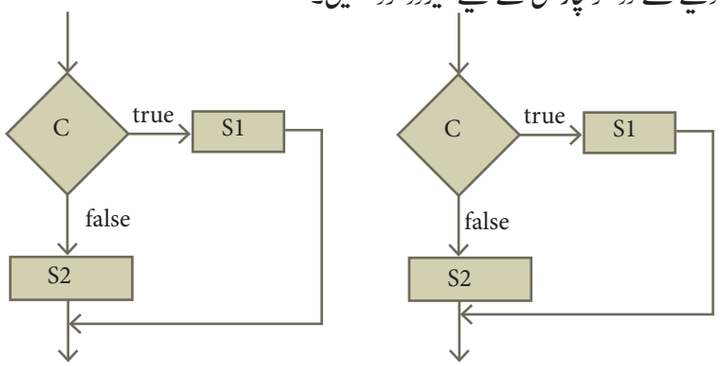
بہت مختصر جوابات دیں

1. شرط اور بیان کے درمیان فرق کریں۔
2. مشروط بیان کے لیے فلو چارٹ بنائیں۔
3. مشروط بیان اور آئی ٹریٹیو بیان دونوں کی ایک شرط اور ایک بیان ہے۔ وہ کیسے مختلف ہیں؟
4. الگورتھم اور پروگرام میں کیا فرق ہے؟
5. فنکشن ایک ابسٹریکشن کیوں ہے؟
6. ہم ایک بیان کو کیسے بہتر کرتے ہیں؟

سیکشن – C

مختصر جوابات دیں۔

1. دیے گئے دو فلو چارٹس کے لیے سیوڈو کوڈ لکھیں۔



2. اگر C لائن 2 میں غلط ہے، تو اس الگورتھم میں کنٹرول فلو کو ٹریس کریں۔

```
1  S1
2  -- C is false
3  if C
4  S2
5  else
6  S3
7  S4
```


11th_Comp Science_UM_Ch 7.indd 100 18-04-2023 14:42:10

3. کیس کا تجزیہ کیا ہے؟

4. متبادل بیانات کا استعمال کرتے ہوئے 3- کیس تجزیہ کے لیے فلو چارٹ بنائیں۔

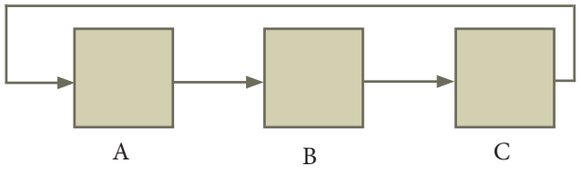
5. ایک عدد کو دو مختلف طریقوں سے دوگنا کرنے کے لیے فنکشن کی وضاحت کریں (1) 2xn, (2) n+n :

سیکشن – D

تفصیل سے بتائیں

1. مشمولات کا تبادلہ کریں: A اور B کے نشان والے دو گلاس دیئے گئے ہیں۔ گلاس A ایپل ڈرنک سے بھرا ہوا ہے اور گلاس B انگور کے مشروب سے بھرا ہوا ہے۔ گلاس A اور B کے مواد کے تبادلے کے لیے تصریح لکھیں، اور تصریح کو پورا کرنے کے لیے اسائنمنٹس کا ایک سلسلہ لکھیں۔

2. مندرجات کو پھیلائیں: تفصیلات لکھیں اور متغیرات A، B اور C کے مواد کو پھیلانے کے لیے الگورتھم بنائیں جیسا کہ ذیل میں دکھایا گیا ہے: تیر(arrows) بتاتے ہیں کہ B کو A کی قدر ملتی ہے، C کو B کی قدر ملتی ہے اور A کو C کی قدر حاصل ہوتی ہے۔



3. منتقل کرنے کا مسئلہ۔ آپ کو 5، 8 اور 3 لیٹر کی تین بوتلیں دی جاتی ہیں۔ 8L بوتل تیل سے بھری ہوئی ہے، جبکہ باقی دو خالی ہیں۔ 8L بوتل میں تیل کو دو برابر مقدار میں تقسیم کریں۔ مناسب متغیرات کے ذریعہ عمل کی حالت کی نمائندگی کریں۔ عمل کی ابتدائی اور آخری حالتیں کیا ہیں؟ اسائنمنٹ کے ذریعہ ایک بوتل سے دوسری بوتل میں تیل کی منتقلی کا نمونہ بنائیں۔ حتمی حالت کو حاصل کرنے کے لیے اسائنمنٹس کی ترتیب لکھیں۔

4. فیکٹوریل (4) (factorial) کے لیے الگورتھم کے مرحلہ وار عمل کو ٹریس کریں۔

```
factorial(n)
-- inputs : n is an integer , n ≥ 0
-- outputs : f = n!
f, i := 1 ,1
while i ≤ n
f, i := f × i, i+1
```


11th_Comp Science_UM_Ch 7.indd 101 18-04-2023 14:42:10

باب 8

آئی ٹریشن و ریکرشن (دہراؤ اور اعادہ)
(Iteration and Recursion)

سیکھنے کے مقاصد

اس باب میں تصورات کو سیکھنے کے بعد طلباء

- الگورتھم ٹیکنکس میں استعمال ہونے والے مختلف حالتوں اور غیر متزلزل (invariants) کے تصورات کو جاننے
- آئی ٹریشن و ریکرشن کے عمل میں الگورتھمک تکنیک کو نافذ کرنے کے قابل ہوجائیں گے۔

کئی مسائل ہیں جو ایک ہی عمل کو بار بار کرنے سے حل ہو سکتے ہیں۔ دوہراؤ اور اعادہ دونوں الگورتھم ڈیزائن کی تکنیک ہیں ایک ہی عمل کو بار بار انجام دینے کے لیے۔ ایک ہی عمل کو بار بار دہرانے کا کیا فائدہ؟ اگرچہ عمل ایک ہی ہے، لیکن جس حالت میں کارروائی کی جاتی ہے وہ ایک جیسی نہیں ہے۔ جب بھی ہم عمل کو انجام دیتے ہیں، حالت بدل جاتی ہے۔ لہذا، ایک ہی کارروائی کو بار بار انجام دیا جاتا ہے، لیکن مختلف حالتوں میں۔ کیفیت اس طرح بدلتی ہے کہ عمل مطلوبہ ان پٹ آؤٹ پٹ تعلق کو حاصل کرنے کے لیے آگے بڑھتا ہے۔

دوہراؤ (Iteration): آئی ٹریشن میں، لوپ باڈی (loop body) کو بار بار عمل میں لایا جاتا ہے جب تک کہ لوپ کی حالت درست ہو۔ ہر بار جب لوپ باڈی کو عمل میں لایا جاتا ہے، متغیرات کو اپ ڈیٹ کیا جاتا ہے۔ تاہم، متغیرات کی ایک خاصیت بھی ہے جو لوپ باڈی کے عمل میں کوئی تبدیلی نہیں رہتی۔ اس غیر تبدیل شدہ خاصیت کو لوپ انویرینٹ (loop invariant) کہا جاتا ہے۔ آئی ٹریٹیو الگورتھم کی تعمیر اور استدلال کی کلید لوپ انویرینٹ ہے۔

ریکرشن (Recursion): آئی ٹریٹیو الگورتھم ڈیزائن کی ایک اور تکنیک ہے، جو دوہراؤ سے قریبی تعلق رکھتی ہے، لیکن زیادہ طاقتور ہے۔ ری کرشن کا استعمال کرتے ہوئے، ہم ایک دیے گئے ان پٹ کے ساتھ ایک مسئلہ حل کرتے ہیں، اسی مسئلے کو ان پٹ کے ایک حصے کے ساتھ حل کرکے اور اصل مسئلے کا حل تیار کرکے جزوی ان پٹ تک مسئلہ حل کرتے ہیں۔

کیا آپ جانتے ہیں؟

ای ڈبلیو دِیکسٹرا (EW Dijkstra) کمپیوٹنگ سائنس کے سب سے باثر علمبرداروں میں سے ایک تھا۔ انہوں نے مختلف شعبوں میں بنیادی خدمات انجام دیں۔ پروگرامنگ لینگویج ڈیزائن، آپریٹنگ سسٹم، اور پروگرام ڈیزائن کے طور پر انہوں نے "سٹرکچرڈ پروگرامنگ (structured programming)" کا فقرہ تیار کیا جس نے سافٹ ویئر انجینئرنگ کے نظم و ضبط کی بنیاد رکھنے میں مدد کی۔ 1972 میں، انہیں ACM ٹورنگ ایوارڈ (Turing award) سے نوازا گیا، جو کمپیوٹر سائنس میں سب سے زیادہ امتیاز سمجھا جاتا ہے۔ Dijkstra سے منسوب ہے کہ "کمپیوٹر سائنس کمپیوٹر کے بارے میں اس سے زیادہ نہیں ہے جتنا کہ فلکیات دوربینوں کے بارے میں ہے۔"



8.1 غیر متزلزل (Invariants)

مثال 8.1 فرض کریں کہ درج ذیل اسائنمنٹ کو (u, v) = (20,15) کے ساتھ انجام دیا گیا ہے۔ ہم تفویض سے پہلے اور بعد میں تشریح کر سکتے ہیں۔

```
-- before: u, v = 20, 15
u, v :=u+5,v-5
-- after: u, v = 25, 10
```

تفویض کے بعد (u, v) = (25, 10)۔ لیکن آپ فنکشن u + v کی قدر کے بارے میں کیا مشاہدہ کرتے ہیں؟

11th_Comp Science_UM_Ch 8.indd 102 18-04-2023 14:50:22

before: u + v = 20 + 15 = 35

after: u + v = 25 + 10 = 35

اسائنمنٹ نے u + v کی قدر میں کوئی تبدیلی نہیں کی ہے۔ ہم کہتے ہیں کہ u + v اسائنمنٹ کا ایک غیر متزلزل ہے۔ ہم تفویض سے پہلے اور بعد میں متضاد اظہار کے ساتھ تشریح کر سکتے ہیں۔

```
-- before: u + v = 35
u, v : = u + 5, v - 5
-- after : u + v = 35
```

ہم کہہ سکتے ہیں، u + v ایک غیر متزلزل ہے: یہ 35 سے پہلے اور بعد میں ہے۔ یا ہم کہہ سکتے ہیں u + v = 35 ایک غیر متزلزل ہے: یہ پہلے اور بعد میں صحیح ہے۔

مثال 8.2 اگر ہم مندرجہ ذیل اسائنمنٹ کو اسائنمنٹ کے بعد (p, c = 10, 9) کے ساتھ انجام دیتے ہیں، (p, c) = (11, 10)۔

```
-- before : p, c = 10 , 9
p, c := p + 1, c+1
-- after: p, c = 11, 10
```

کیا آپ ایک غیر متزلزل دریافت کرسکتے ہیں؟ پہلے اور بعد میں p - c کی قدر کیا ہے؟

before: p — c = 10 — 9 = 1

after: p — c = 11 — 10 = 1

ہمیں معلوم ہوتا ہے کہ p – c = 1 ایک غیر متزلزل ہے۔

عام طور پر، اگر متغیرات کے اظہار کی کسی اسائنمنٹ سے پہلے اور بعد میں ایک ہی قدر ہوتی ہے، تو یہ اسائنمنٹ کا غیر متزلزل ہے۔ مانیں کہ P(u, v) ایک اظہار ہے جس میں u اور v متغیرات شامل ہیں ۔ P(u, v) سے u کو el اور v کو e2 سے بدل کر P(u, v)[u, v:= el, e2] حاصل کیا جاتا ہے۔ ایک ہی وقت میں P(u, v) ایک غیر متزلزل u, v := el, e2 if کا اسائنمنٹ ہے۔

P(u,v) [u,v := el, e2] = P(u,v)

مثال 8.3 دکھائیں کہ p – c تفویض کا ایک غیر متزلزل ہے۔

p, c := p + 1, c + 1

Let P(p, c) = p - c. Then

P (p, c) [p, c := p + 1, c + l]

= p — c [p, c := p + 1, c + l]

= (p + 1) — (c + 1)

= p — c

= P(P , c)

چونکہ (p – c)[p, c := p+l, c+l] = p – c, p – c تفویض p، c + 1، c := p + 1 کا ایک غیر متزلزل ہے۔

مثال 8.4 تفویض کے تحت دو متغیرات m اور n پر غور کریں۔

m, n := m + 3, n - 1

کیا اظہار m + 3n ایک غیر متزلزل ہے؟

Let P(m, n) = m + 3n. Then

P(m, n) [m, n := m + 3, n — l]

= m + 3n [m, n := m + 3, n — l]

= (m + 3) + 3(n — l)

= m + 3 + 3n — 3

= m + 3n

= P(m, n)

چونکہ (m + 3n)[m, n := m + 3, n – 1] = m + 3n, m + 3n تفویض m, n := m + 3, n – 1 کا ایک غیر متزلزل ہے۔

8.2 لوپ انویرینٹ (Loop invariant)

ایک لوپ میں، اگر L لوپ باڈی B کا انویرینٹ ہے، تو L کو لوپ انویرینٹ (loop invariant) کے طور پر جانا جاتا ہے۔

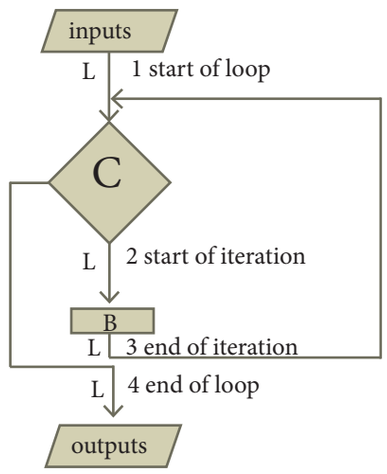
```
while C
-- L
B
-- L
```


11th_Comp Science_UM_Ch 8.indd 103 18-04-2023 14:50:22

لوپ انویرینٹ ہر بار لوپ باڈی سے پہلے اور لوپ باڈی کے بعد درست ہوتا ہے۔ چونکہ L پہلی اعادہ کے آغاز میں صحیح ہے، L لوپ کے شروع میں بھی صحیح ہے (لوپ سے بالکل پہلے)۔ چونکہ L آخری اعادہ کے آخر میں صحیح ہے، L صحیح ہے جب لوپ بھی ختم ہو جائے (صرف لوپ کے بعد)۔ اس طرح، اگر L ایک لوپ ویریئنٹ ہے، تو یہ الگورتھم کے چار اہم نکات پر صحیح ہے، جیسا کہ الگورتھم میں بیان کیا گیا ہے اور تصویر 3.1 میں دکھایا گیا ہے۔

1. لوپ کے شروع میں (لوپ سے بالکل پہلے)
2. ہر اعادہ کے شروع میں (لوپ باڈی سے پہلے)
3. ہر اعادہ کے آخر میں (لوپ باڈی کے بعد)
4. لوپ کے آخر میں (صرف لوپ کے بعد)

```
1. -- L, start of loop
   while
       C
2.     -- L, start of iteration
       B
3.     -- L, end of iteration
4. -- L, end of loop
```



تصویر 8.1: وہ پوائنٹس جہاں لوپ انویرینٹ صحیح ہے۔

ایک لوپ بنانے کے لیے،

1. لوپ کے شروع میں لوپ انویرینٹ قائم کریں۔
2. لوپ باڈی کو متغیرات کو اپ ڈیٹ کرنا چاہیے، تاکہ اختتام کی طرف بڑھے، اور اسی وقت لوپ انویرینٹ کو برقرار رکھے۔
3. جب لوپ ختم ہوتا ہے، تو ختم ہونے کی حالت اور لوپ انویرینٹ کو ان پٹ - آؤٹ پٹ تعلق قائم کرنا چاہیے۔

8.3 غیر متزلزل – مثالیں
(Invariants – examples)

لوپ انویرینٹ ایک لوپ میں چار اہم پوائنٹس میں درست ہے۔ لوپ انویرینٹ کا استعمال کرتے ہوئے، ہم ان پوائنٹس پر متغیر کی خصوصیات کے بارے میں لوپ اور وجہ بنا سکتے ہیں۔

مثال 8.5 a^n کا حساب کرنے کے لیے ایک آئی ٹریٹیو الگورتھم ڈیزائن کریں۔ آیئے ہم الگورتھم پاور (a, n) کا نام دیتے ہیں۔ مثال کے طور پر،

power(10, 4) = 10000
power (5, 3) = 125
power (2, 5) = 32

الگورتھم پاور(a, n) a^n کا حساب ایک کو مجموعی طور پر n مرتبہ ضرب دے کر شمار کرتا ہے۔

$$a^n = \underbrace{a \times a \times \ldots \times a}_{n \text{ times}}$$

تفصیلات اور لوپ انویرینٹ کو تبصرے کے طور پر دکھایا گیا ہے۔

```
power (a, n)
-- inputs: n is a positive integer
-- outputs: p = a^n
p, i := 1, 0
while i ≠ n
   -- loop invariant: p = a^i
   p, i :=p X a, i+1
```


11th_Comp Science_UM_Ch 8.indd 104 18-04-2023 14:50:22

پاور کا مرحلہ وار عمل (2، 5) ٹیبل 8.1 میں دکھایا گیا ہے۔ ہر قطاراعادہ کے آخر میں دو متغیر p اور i کی قدروں کو دکھاتی ہے، اور ان کا حساب کیسے لگایا جاتا ہے۔ ہم دیکھتے ہیں کہ $p = a^i$ لوپ کے شروع میں صحیح ہے، اور ہر قطار میں صحیح رہتا ہے۔ لہذا، یہ ایک لوپ invariant ہے۔

a^i	i+1	i	p x a	p	iteration
2^0		0		1	0
2^1	0 + 1	1	1x2	2	1
2^2	1 + 1	2	2x2	4	2
2^3	2 + 1	3	4x2	8	3
2^4	3 + 1	4	8x2	16	4
2^5	4+1	5	16x2	32	5

ٹیبل 8.1: پاور کا ٹریس (5,2)

جب لوپ ختم ہوتا ہے، $p = a^1$ پھر بھی صحیح ہے، لیکن $i = 5$۔ لہذا، $p = a^5$۔ عام طور پر، جب لوپ ختم ہوتا ہے، $p = a^n$۔ اس طرح، ہم نے تصدیق کی ہے کہ پاور (a,n) اس کی تفصیلات کو پورا کرتی ہے۔

مثال 8.6 مثال 6.11 کے چاکلیٹ بار (chocolate bar) کے مسئلے کو یاد کریں۔ بار کو اس کے انفرادی مربعوں (individual squares) میں توڑنے کے لیے کتنے ٹکڑوں کی ضرورت ہے؟

ہم نے بالترتیب p اور c متغیرات کے ٹکڑوں کی تعداد اور کٹوتیوں کی تعداد کی نمائندگی کرنے کا فیصلہ کیا۔ جب بھی کوئی ٹکڑا بنایا جاتا ہے تو ٹکڑوں کی تعداد میں ایک اضافہ ہوتا ہے اور ٹکڑوں کی تعداد بھی ایک سے بڑھ جاتی ہے۔ ہم نے اسے ایک اسائنمنٹ کے ذریعے ماڈل بنانے کا فیصلہ کیا۔

```
p, c := p + 1, c+1
```

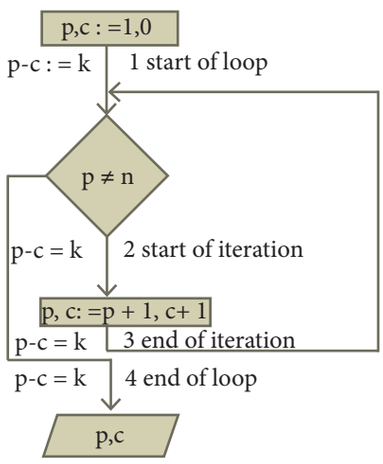
بار کو کاٹنے کے عمل کو لوپ کے ذریعہ ماڈل کیا جاسکتا ہے۔ ہم ایک ٹکڑا اور صفر ٹکڑے کے ساتھ شروع کرتے ہیں، $p = 1$ اور $c = 0$۔ n انفرادی مربعوں کی تعداد ہونے دیں۔ جب ٹکڑوں کی تعداد p انفرادی مربعوں n، کی تعداد کے برابر ہو جائے تو عمل ختم ہو جاتا ہے۔

```
p, c : = 1 , 0
while p ≠ n
    p, c := p + 1, c+1
```

ہم نے مشاہدہ کیا ہے (مثال 8.2 میں) کہ p - c تفویض p, c := p + 1, c + 1. Let p - c = k، کا ایک متغیر ہے۔ p – c = k، جہاں k ایک مستقل ہے۔الگورتھم کے پوائنٹس جہاں p – c = k درست ہیں ذیل کے الگورتھم میں اور تصویر 8.2 کے فلو چارٹ میں دکھائے گئے ہیں۔

```
p, c : = 1 , 0
1.  -- p - c = k
    while p ≠ n
2.      -- p - c = k
        p, c := p+1, c+1
3.      -- p - c = k
4.  --p-c=k,p=n
```

لوپ انویرینٹ p- c = k لوپ کے شروع میں درست ہے (لائن 1)۔ مزید برآں، لوپ کے آغاز پر، p– c = 1. لہذا، k = 1، اور لوپ کا انویرینٹ p – c = 1 ہے



تصویر 8.2: وہ پوائنٹس جہاں لوپ انویرینٹ صحیح ہے۔

جب لوپ ختم ہوتا ہے (لائن 4)، لوپ انویرینٹ اب بھی صحیح ہے (p – c = 1)۔ مزید یہ کہ لوپ کی حالت غلط ہے (p = n)۔ p – c = 1 اور p = n سے،

11th_Comp Science_UM_Ch 8.indd 105 18-04-2023 14:50:22

1. **p — c = 1 loop invariant**
2. **p = n end of the loop**
3. **n — c = 1 from 1, 2**
4. **c = n — 1 from 3**

جب عمل ختم ہوتا ہے، کٹوتیوں کی تعداد مربعوں کی تعداد سے ایک کم ہوتی ہے۔

مثال 8.7 ان درختوں پر 6 برابر فاصلے والے درخت اور 6 چڑیاں بیٹھی ہیں، ایک چڑیا ایک درخت پر۔ اگر ایک چڑیا ایک درخت سے دوسرے درخت کی طرف اڑتی ہے تو اسی وقت دوسری چڑیا اپنے درخت سے اتنے ہی فاصلے پر کسی دوسرے درخت کی طرف اڑتی ہے لیکن مخالف سمت میں۔ کیا تمام چڑیاں ایک درخت پر جمع ہو سکتی ہیں؟

آئیے 1 سے 6درختوں کا اشاریہ تیار کریں۔ ایک چڑیا کا اشاریہ اس کا اشاریہ ہے، جس درخت پر یہ اس وقت بیٹھی ہے۔ اڑتی ہوئی چڑیوں کی ایک جوڑی کو بطور ماڈل ایک لوپ کا تکراری مرحلہ بنایا جا سکتا ہے ۔ جب چڑیا درخت سے اڑتی ہے تو درخت i سے درخت i + d پر اڑتی ہے، دوسری چڑیا درخت j سے درخت j – d پر اڑتی ہے۔ اس طرح، ہر ایک کے بعد تکراری مرحلہ کے اشاریہ جات کا مجموعہ S چڑیاں متغیر رہتی ہیں۔ مزید یہ کہ ایک لوپ انویرینٹ شروع اور لوپ کے اختتام پر صحیح ہے۔

لوپ کے شروع میں، انویرینٹ کی قدر ہے

$$S = 1 + 2 + 3 + 4 + 5 + 6 = 21$$

جب لوپ ختم ہوتا ہے، لوپ انویرینٹ ایک ہی قدر ہے۔ البتہ، جب لوپ ختم ہوتا ہے، اگر تمام چڑیاں ایک ہی درخت پر ہوتیں، تو k، پھر S=6k

S = 21,	لوپ کے شروع میں لوپ انویرینٹ
S = 6k,	لوپ کے آخر میں لوپ انویرینٹ

6k= 21,	لوپ انویرینٹ شروع اور آخر میں ایک ہی قدر رکھتا ہے۔
21 6 کا ضرب ہے۔	

یہ ممکن نہیں ہے 21- کا ضرب 6 نہیں ہے۔ لوپ انویرینٹ کے ساتھ چڑیا کے اشاریوں کی مطلوبہ حتمی قدریں ممکن نہیں ہیں۔ اس لیے تمام چڑیاں ایک درخت پر جمع نہیں ہو سکتیں۔

مثال 8.8 مثال 6.3 کے کروم لینڈ (Chromeland) کے گرگٹ (chameleons) پر غور کریں۔ کروم لینڈ پر 13 سرخ، 15 سبز اور 17 نیلے گرگٹ ہیں۔ جب مختلف رنگوں کے دو گرگٹ آپس میں ملتے ہیں تو وہ دونوں اپنا رنگ بدل کر تیسرے رنگ میں تبدیل ہو جاتے ہیں (مثال کے طور پر اگر سرخ اور سبز ملیں تو دونوں نیلے ہو جاتے ہیں)۔ کیا ایسی ملاقاتوں کا اہتمام کرنا ممکن ہے جس کے نتیجے میں تمام گرگٹ نیلے رنگ کی نمائش کریں؟

r، g، اور b کو سرخ، سبز اور نیلے رنگ کے گرگٹ کے نمبر ہونے دیں۔ ہم ایک تکراری عمل کے طور پر دو قسم کی ملاقاتوں کو ماڈل کر سکتے ہیں۔ (r, g, b) کو (r−1, g−1, b+2) یا (r−1, g+2, b−1) یا (r+2, g−1, b) ملاقات میں تبدیل کرتی ہے۔ مثال کے طور پر، سرخ اور سبز گرگٹ کی ملاقات پر غور کریں۔

r, g, b := r-1, g-1, b+2

کسی بھی دو قسم کے اعداد میں فرق یا تو تبدیل نہیں ہوتا یا 3 سے تبدیل ہوتا ہے۔ یہ ایک انویرینٹ ہے۔

r - 1 - (g - 1) = r - g

r - 1 - (b + 2) = (r - b) - 3

g - 1 - (b + 2) = (g - b) - 3

یہ تینوں صورتوں کے لیے درست ہے۔ اگر تکراری عمل کے آغاز میں کوئی بھی دو اقسام تعداد میں 3 کے ضرب سے مختلف ہوتی ہیں، تو تکراری عمل کے ختم ہونے پر فرق کو 3 کے مراحل میں 0 تک کم کیا جا سکتا ہے۔ تاہم، شروع میں،

r - g = 13 - 15 = -2

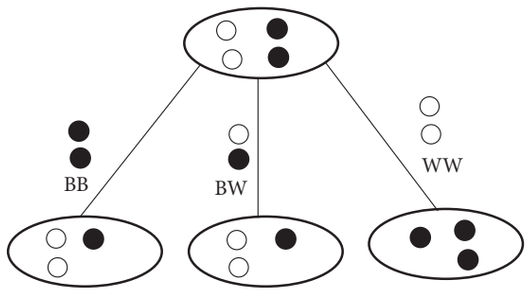
11th_Comp Science_UM_Ch 8.indd 106 18-04-2023 14:50:22

$$g - b = 15 - 17 = -2$$

$$b - r = 17 - 13 = 4$$

کوئی دو رنگ تعداد میں 3 کے ضرب سے مختلف نہیں ہوتے۔ اس لیے تمام گرگٹ کو ایک رنگ میں تبدیل نہیں کیا جا سکتا۔

مثال 8.9 سنگ مرمر کا جار (Jar of marbles): آپ کو دو قسم کے سنگ مرمروں سے بھرا ہوا جار دیا جاتا ہے، سفید اور سیاہ، اور اس کھیل کو کھیلنے کو کہا جاتا ہے۔ جار (Jar) سے تصادفی طور پر دو ماربل منتخب کریں۔ اگر وہ ایک ہی رنگ کے ہیں، تو انہیں باہر پھینک دیں، لیکن ایک اور کالا ماربل ڈال دیں (آپ فرض کر سکتے ہیں کہ آپ کے پاس فالتو سنگ مرمر کی لامتناہی فراہمی ہے)۔ اگر وہ مختلف رنگوں کے ہیں، تو سفید کو واپس جار میں رکھیں اور سیاہ کو پھینک دیں۔ اگر آپ کو سفید اور سیاہ سنگ مرمر کی اصل تعداد معلوم ہوتی تو جار میں آخری سنگ مرمر کا رنگ کیا ہے؟



تصویر 8.3: جار کے ماربلز میں حالت کی تبدیلی

جار میں سفید اور سیاہ سنگ مرمر کی تعداد کو دو متغیرات w اور b سے ظاہر کیا جا سکتا ہے۔ ہر تکراری قدم میں، b اور w تبدیل ہوتے ہیں جو کہ نکالے گئے دو ماربلز کے رنگوں پر منحصر ہوتے ہیں: سیاہ سیاہ، سیاہ سفید یا سفید سفید۔ یہ تصویر 8.3 میں دکھایا گیا ہے اور ذیل میں الگورتھم میں بیان کیا گیا ہے۔

1 جبکہ جار میں کم از کم دو سنگ مرمر
2 -- b , w
3 کوئی بھی دو ماربل نکال لیں۔
4 کیس دونوں سیاہ ہیں -- BB
5 دونوں سنگ مرمر پھینک دیں
6 ایک سیاہ سنگ مرمر واپس رکھیں
7 -- b = b '−1، w = w'، b+w = b'+w' −1
8 کیس دونوں سفید ہیں --WW
9 دونوں ماربل پھینک دو
10 ایک سیاہ سنگ مرمر واپس رکھیں
11 --b = b'+1، w = w'−2، --b+w = b'+w'−1
12 else -- BW
13 کالے کو پھینک دو
14 سفید کو واپس رکھیں
15 --b = b'−1، w = w'، b+w = b'+w'−1

ہر کیس کے لیے، الگورتھم میں b، w اور b+w تبدیلی کیسے دکھائی جاتی ہے، جہاں b' اور w' دو ماربلز نکالنے سے پہلے متغیر کی قدریں ہیں۔ تبدیلیوں کا طریقہ دیکھیں۔ یا تو یہ تبدیل نہیں ہوتا، یا 2 تک کم ہوتا ہے۔اس کا مطلب ہے کہ w کی برابری، چاہے وہ طاق ہو یا جفت، تبدیل نہیں ہوتی۔ w کی برابری غیر متغیر ہے۔

فرض کریں، کھیل کے آغاز پر، w برابر ہے۔ جب کھیل ختم ہوتا ہے، w اب بھی برابر ہے۔ مزید یہ کہ، صرف ایک سنگ مرمر باقی ہے، w+b = 1

1	w + b = 1	لوپ کا اختتام
2	w = 0 or w = 1	سے 1
3	w جفت ہے	لوپ انویرینٹ ہے۔
4	w = 0	سے 2,3
5	b = 1	سے 1,4

آخری سنگ مرمر کالا ہونا چاہیے۔ اسی طرح، اگر کھیل کے آغاز میں، سفیدوں کی ایک طاق تعداد ہے، آخری ماربل سفید ہونا ضروری ہے۔

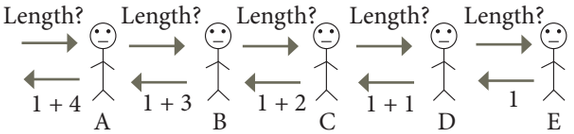
ایک آخری سوال: کیا ہم کبھی ایسی حالت میں پہنچتے ہیں جس میں صرف ایک سنگ مرمر ہو؟ جی ہاں، کیونکہ ماربلز کی کل تعداد b+w ہر قدم پر ہمیشہ ایک سے کم ہوتی ہے، آخرکار یہ 1 ہو جائے گی۔

11th_Comp Science_UM_Ch 8.indd 107 18-04-2023 14:50:23

8.4 ریکرشن
(Recursion)

Recursion ایک الگورتھم ڈیزائن تکنیک ہے، جس کا تعلق انڈکشن (induction) سے ہے۔ یہ آئی ٹریشن (Iteration) کی طرح ہے، لیکن زیادہ طاقتور ہے ریکرشن کا استعمال کرتے ہوئے، ہم ان پٹ کے ایک حصے کے ساتھ مسئلے کی مثالوں کو حل کرکے، دیئے گئے ان پٹ کے ساتھ ایک مسئلہ حل کرسکتے ہیں۔

مثال 8.10 گاہک ایک کاؤنٹر پر قطار میں کھڑے ہیں۔ کاؤنٹر پر موجود آدمی جاننا چاہتا ہے کہ لائن میں کتنے گاہک انتظار کر رہے ہیں۔



تصویر 8.4: ایک لکیر کی لمبائی

خود لمبائی کو گننے کے بجائے، وہ گاہک A سے سر کے ساتھ لائن کی لمبائی پوچھتا ہے، گاہک A کسٹمر B سے گاہک B کے ساتھ لائن کی لمبائی کے بارے میں پوچھتا ہے، وغیرہ۔ جب سوال لائن میں آخری صارف تک پہنچتا ہے، E، چونکہ اس کے پیچھے کوئی نہیں ہے، اس لیے وہ 1 سے D کو جواب دیتا ہے جس نے اس سے پوچھا تھا۔ D، 2 = 1+1، C کو جواب دیتا ہے، C، 3 = 2+1، B کو جواب دیتا ہے، B، 4=3+1، A کو جواب دیتا ہے اور A، 5 = 4+1 جو اب دیتا ہے اس شخص کو جو کاؤنٹر میں ہے۔

8.4.1 ری کرسیو عمل (Recursive process)

مثال 8.10 ایک ری کرسیو عمل کی وضاحت کرتی ہے۔ آیئے ہم 5 صارفین کی ترتیب کو A، B، C، D اور E کے طور پر پیش کرتے ہیں۔

[A,B,C,D,E]

مسئلہ ترتیب کی لمبائی کا حساب لگانا ہے [A,B,C,D,E]۔ آیئے اپنے سولور (solver) کی لمبائی کا نام دیں۔ اگر ہم کسی ترتیب کو بطور ان پٹ پاس کرتے ہیں، تو سولور کی لمبائی کو ترتیب کی لمبائی کو آؤٹ پٹ کرنا چاہیے۔

length [A,B,C,D,E] = 5

سولور کی لمبائی ترتیب [A,B,C,D,E] کو اپنے پہلے گاہک اور باقی ترتیب کو توڑ دیتی ہے۔

first [A ,B,C,D,E] = A

rest [A ,B,C,D,E] = [B ,C,D,E]

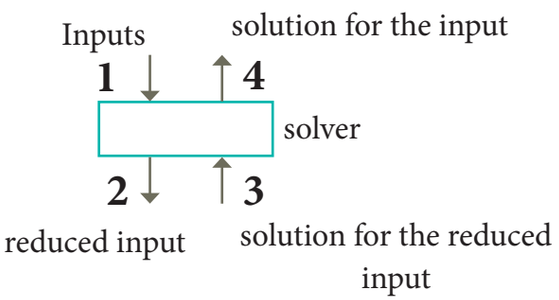
بار بار کسی مسئلے کو حل کرنے کے لیے، سولور کی لمبائی کم ہوئی ترتیب [B,C,D,E] کو بطور ذیلی حل کرنے والے کے پاس کرتی ہے، جو کہ لمبائی کی ایک اور مثال ہے۔ حل کرنے والا فرض کرتا ہے کہ ذیلی حل کرنے والا [B,C,D,E] کی لمبائی نکالتا ہے، 1 کا اضافہ کرتا ہے، اور اسے [A,B,C,D,E] کی لمبائی کے طور پر آؤٹ پٹ کرتا ہے۔

length [A,B,C,D,E] = 1 + length [B,C,D,E]

Each solver

1. ایک ان پٹ وصول کرتا ہے،
2. کم سائز کا ایک ان پٹ ذیلی حل کرنے والے کو دیتا ہے،
3. ذیلی حل کرنے والے سے کم ان پٹ کا حل حاصل کرتا ہے، اور دیے گئے ان پٹ کے لیے حل تیار کرتا ہے

جیسا کہ تصویر 8.5 میں دکھایا گیا ہے۔



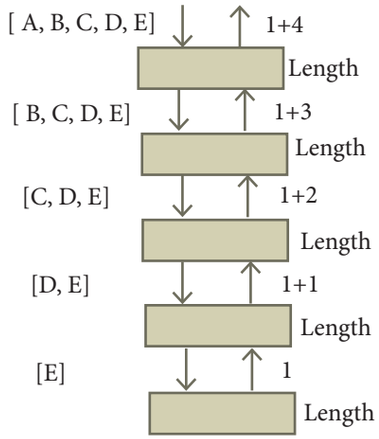
تصویر 8.5: ری کرسیو عمل میں حل کرنے کی ایک مثال

تصویر 8.6 موصولہ ان پٹ اور ہر ایک کے ذریعہ تیار کردہ حل کو ظاہر کرتا ہے۔

مثال کے طور پر حل کرنے والا 8.10۔ ہر حل کرنے والا ان پٹ کے سائز کو ایک سے کم کرتا ہے اور اسے ذیلی حل کرنے والے کو منتقل کرتا ہے، جس کے نتیجے میں 5 حل کرنے والے ہوتے ہیں۔ یہ اس وقت تک جاری رہتا ہے جب تک کہ حل کرنے والے کو موصول ہونے والا ان پٹ اتنا چھوٹا نہ ہو کہ براہ راست حل نکال سکے۔ آخری سولور کو ان پٹ کے طور پر [E] موصول ہوا۔

11th_Comp Science_UM_Ch 8.indd 108 18-04-2023 14:50:23

چونکہ [E] کافی چھوٹا ہے، حل کرنے والا [E] کی لمبائی کو فوراً 1 کے طور پر نکالتا ہے، اور ری کرشن (recursion) رک جاتا ہے۔



تصویر 8.6: حل کرنے والوں اور ذیلی حل کرنے والوں کے ساتھ ری کرسیو عمل

لمبائی [A,B,C,D,E] کے لیے ری کرشن عمل (recursion process) کو تصویر 8.7 میں دکھایا گیا ہے

1 length [A,B,C,D,E]
2 = 1 + length [B,C,D,E]
3 = 1 +1 + length [C,D,E]
4 = 1 + 1+ 1 + length [D,E]
5 = 1 + 1+ 1 + 1 + length [E]
6 = 1 + 1 + 1 + 1 +1
7 = 1 +1 +1 + 2
8 = 1 + 1 + 3
9 = 1 + 4
10 = 5

تصویر 8.7: ایک ترتیب کی لمبائی کے حساب کے لیے ری کرسیو عمل

8.4.2 ری کرسیو مسئلہ حل کرنا (Recursive problem solving)

ہر حل کرنے والے کو ان پٹ کے سائز کی جانچ کرنی چاہئے۔ اگر سائز کافی چھوٹا ہے تو، حل کرنے والے کو براہ راست مسئلے کا حل نکالنا چاہیے۔ اگر سائز کافی چھوٹا نہیں ہے، تو حل کرنے والے کو ان پٹ کا سائز کم کرنا چاہیے اور کم ان پٹ کے ساتھ مسئلہ حل کرنے کے لیے ذیلی حل کرنے والے کو بلانا چاہیے۔ مثال کے طور پر 8.10، حل کرنے والے کے الگورتھم کو بطور اظہار کیا جا سکتا ہے۔

$$\text{length of sequence} = \begin{cases} 1 & \text{if sequence has only one customer} \\ 1 + \text{length of tail,} & \text{otherwise} \end{cases}$$

بار بار کسی مسئلے کو حل کرنے کے لیے، حل کرنے والا مسئلہ کو ذیلی مسائل میں کم کر دیتا ہے، اور ذیلی مسئلے کو حل کرنے کے لیے حل کرنے والے کی ایک اور مثال کو بلاتا ہے ، جسے ذیلی حل (sub – solver) کے نام سے جانا جاتا ہے۔ ذیلی مسئلے کا ان پٹ سائز اصل مسئلے کے ان پٹ سائز سے چھوٹا ہے۔ جب حل کرنے والا ذیلی حل کرنے والے کو بلاتا ہے، تو اسے ری کرسیو کال (recursive call) کے طور پر جانا جاتا ہے۔ ری کرسیو کا جادو حل کرنے والے کو یہ فرض کرنے کی اجازت دیتا ہے کہ ذیلی حل کرنے والا (بار بار آنے والی کال) ذیلی مسئلے کا حل نکالتا ہے۔ پھر، ذیلی مسئلے کے حل سے، حل کرنے والا دیے گئے مسئلے کا حل بناتا ہے۔

جیسا کہ ذیلی حل کرنے والے مسئلے کو چھوٹے سائز کے ذیلی مسائل میں کم کرتے چلے جاتے ہیں، آخر کار ذیلی مسئلہ اتنا چھوٹا ہو جاتا ہے کہ بغیر ری کرشن کے براہ راست حل کیا جا سکتا ہے۔ لہذا، ایک بار بار حل کرنے والے کی دو صورتیں ہیں:

1. **بیس کیس (Base Case):** مسئلہ کا سائز اتنا چھوٹا ہے کہ براہ راست حل کیا جا سکتا ہے۔ حل نکالیں۔ کم از کم ایک بیس کیس ہونا چاہیے۔

2. **ری کرشن مرحلہ (Recursion step):** مسئلہ کا سائز کافی چھوٹا نہیں ہے۔مسئلے کو ایک ذیلی مسئلہ میں تبدیل کریں، دیے گئے مسئلے سے سائز میں سختی سے چھوٹاکریں۔ ذیلی مسئلے کو حل کرنے کے لیے ذیلی حل کنندہ کو بلائیں۔فرض کریں کہ ذیلی حل کرنے والا ذیلی مسئلے کا حل نکالتا ہے۔ دیئے گئے مسئلے کا حل تیار کریں۔

11th_Comp Science_UM_Ch 8.indd 109 18-04-2023 14:50:23

ری کرسیو مسئلہ حل کرنے کی تکنیک کا یہ خاکہ ذیل میں دکھایا گیا ہے۔

solver (input)

if input is small enough

construct solution

else

find sub_problems of reduced input

solutions to sub_problems = solver for each sub_problem

construct solution to the problem from

solutions to the sub_problems

جب بھی ہم ری کرشن کا استعمال کرتے ہوئے کوئی مسئلہ حل کرتے ہیں، تو ہمیں ان دو صورتوں کو یقینی بنانا ہوتا ہے: ری کرشن کے مرحلے میں، ری کرسیو کال کے ان پٹ کا سائز دیے گئے ان پٹ کے سائز سے سختی سے چھوٹا ہوتا ہے، اور کم از کم ایک بنیادی کیس ہوتا ہے۔

8.4.3 ری کرشن - مثالیں (Recursion -Examples)

مثال 8.11 ایک ترتیب کی لمبائی کے لیے ری کرسیو الگورتھم (Recursive Algorithm) کو بطور لکھا جا سکتا ہے۔

```
length (s)
-- inputs : s
-- outputs : length of s
  if s has one customer -- base case
    1
  else
    1 + length(tail(s)) -- recursion step
```

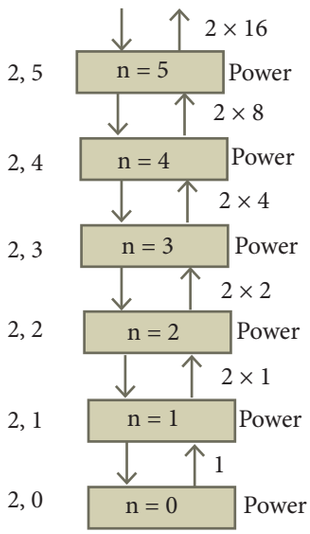
مثال 8.12 a^n کی گنتی کرنے کے لیے ایک ری کرسیو الگورتھم (Recursive Algorithm) ڈیزائن کریں۔ ہم نے مثال 8.5 میں شمار کرنے کے لیے آئی ٹریٹیو الگورتھم (Iterative Algorithm) بنایا۔ جو a^n کے طور پر بار بار بیان کیا جا سکتا ہے۔

$$a^n = \begin{cases} 1 & \text{if } n = 0 \\ a \times a^{n-1} & \text{otherwise} \end{cases}$$

ری کرسیو تعریف (recursive definition) کو کمپیوٹنگ پاور (a, n) کے لیے ایک بار بار حل کرنے والے کے طور پر ظاہر کیا جا سکتا ہے۔

```
power (a, n)
-- inputs: n is an integer , n ≥ 0
-- outputs : a^n
if n = 0 -- base case
  1
else    --recursion step
        a × power (a, n-1)
```

پاور (2,5) کا حساب لگانے کے لیے محلول کے ساتھ ری کرسیو عمل کو تصویر 8.8 میں دکھایا گیا ہے۔



تصویر 8.8: پاور کا حساب لگانے کے لیے حل کرنے والوں کے ساتھ ری کرسیو عمل (2,5)

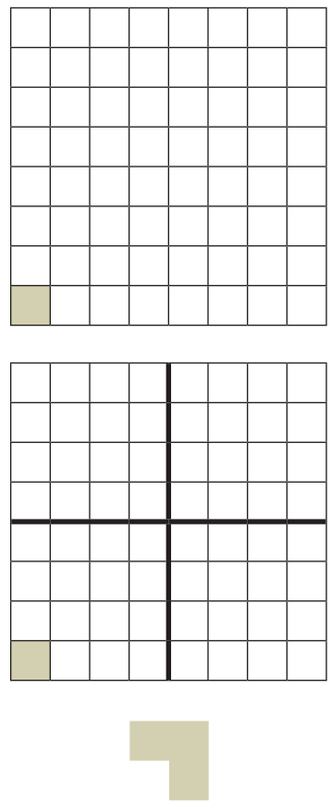
پاور (2,5) کے نتیجے میں ری کرسیو عمل کو تصویر 8.9 میں دکھایا گیا ہے۔

```
power (2,5)
= 2 × power (2,4)
= 2 × 2 × power(2,3)
= 2 × 2 × 2 × power(2, 2)
= 2 × 2 × 2 × 2 × power (2,1)
= 2 × 2 × 2 × 2 × 2 × power (2,0)
= 2 × 2 × 2 × 2 × 2 × 1
= 2 × 2 × 2 × 2 × 2
= 2 × 2 × 2 × 4
= 2 × 2 × 8
= 2 × 16
= 32
```

تصویر 8.9: پاور کے لیے ری کرسیو عمل (2,5)

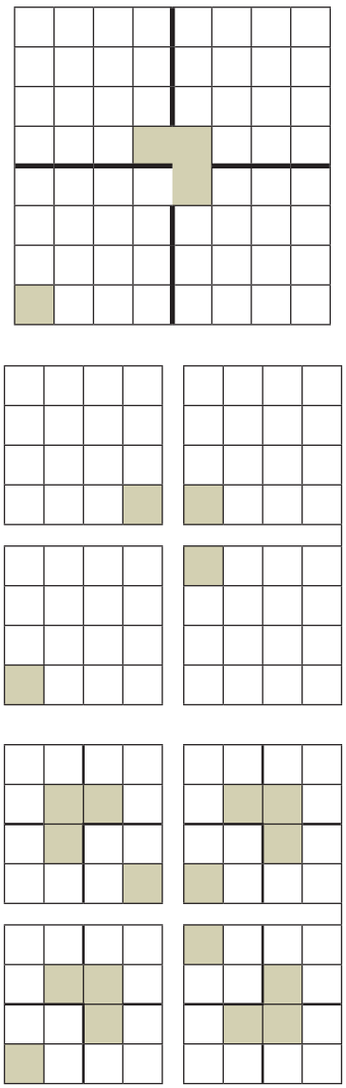
11th_Comp Science_UM_Ch 8.indd 110 18-04-2023 14:50:23

مثال 8.13 کونے سے ڈھکنے والا بورڈ (corner-covered board) $2^n \times 2^n$ مربعوں کا بورڈ ہوتا ہے جس میں ایک کونے کا مربع ایک مربع ٹائل سے ڈھکا ہوتا ہے۔ تین ملحقہ مربعوں کے ساتھ بنی ہوئی ایک L کی شکل والی ٹائل ہے (تصویر 8.10 دیکھیں)۔ کونے سے ڈھکنے والے بورڈ کو بغیر اوورلیپ کے L کی شکل والے ٹرائیومینوز (Triominoes) سے ڈھانپیں۔ ٹرائیومینوز کو ضرورت کے مطابق گھمایا جا سکتا ہے۔

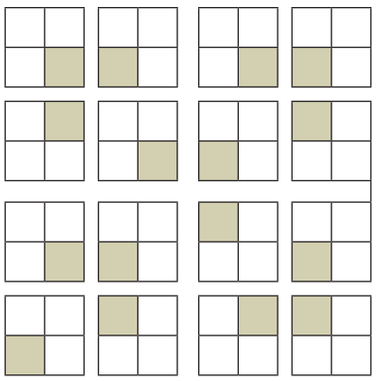


تصویر 8.10: کونے سے ڈھکا بورڈ اور ٹرائیومینو (Triominoe)

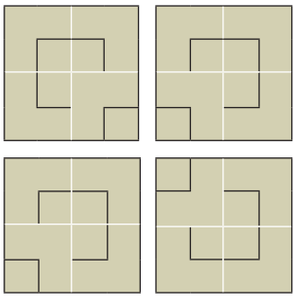
مسئلہ کا سائز n (سائز $2^n \times 2^n$ کا بورڈ) ہے۔ ہم ری کرشن سے مسئلہ حل کر سکتے ہیں۔ بیس کیس $n = 1$ ہے۔ یہ 2×2 کونے سے ڈھکا بورڈ ہے۔ ہم اسے ایک ٹرائیومینو سے ڈھانپ سکتے ہیں اور مسئلہ حل کر سکتے ہیں۔ ری کرشن کے مرحلے میں، بورڈ کے بیچ میں افقی اور عمودی لکیریں کھینچ کر، $2^n \times 2^n$ سائز کے کونے سے ڈھکے ہوئے بورڈ کو 4 ذیلی بورڈز میں تقسیم کریں، جن میں سے ہر ایک کا سائز $2^{n-1} \times 2^{n-1}$ ہے۔ پورے بورڈ کے بیچ میں ایک ٹرائیومینو رکھیں تاکہ کونے سے ڈھکے ذیلی بورڈ کا احاطہ نہ ہو، جیسا کہ تصویر 8.11 کا سب سے بائیں طرف والا بورڈ میں دکھایا گیا ہے۔ اب، ہمارے پاس چار کونے سے ڈھکے ہوئے بورڈز ہیں، جن میں سے ہر ایک کا سائز $2^{n-1} \times 2^{n-1}$ ہے۔



تصویر 8.11: 2×2^3 سائز کے کونے سے ڈھکے بورڈ کو ڈھانپنے کا بار بار چلنے والا عمل



11th_Comp Science_UM_Ch 8.indd 111 18-04-2023 14:50:23



ہمارے پاس 4 ذیلی مسائل ہیں جن کا سائز دیئے گئے مسئلے کے سائز سے سختی سے چھوٹا ہے۔ ہم ہر ذیلی مسائل کو بار بار حل کر سکتے ہیں۔

```
tile corner_covered board of size n

if n = 1 -- base case

    cover the 3 squares with one
    triominoe

else    -- recursion step

    divide board into 4 sub_boards of
    size n-1

    place a triominoe at centre of board ,

    leaving out the corner_covered sub
    -board

    tile each sub_board of size n-1
```

$2^3 \times 2^3$ کونے سے ڈھکے ہوئے بورڈ کو ڈھانپنے کے نتیجے میں ری کرسیو عمل کو تصویر 8.11 میں دکھایا گیا ہے۔

یاد رکھنے کے لیے نکات

- آئی ٹریشن دو مراحل کو دہراتا ہے شرط کا اندازہ لگانا اور بیان پر عمل کرنا، جب تک کہ شرط درست ہو۔
- متغیرات پر مشتمل ایک ایکسپریشن ہے ، جو ان متغیرات میں سے کسی تفویض کے ذریعے تبدیل نہیں ہوتا، وہ اسائنمنٹ کا انویرینٹ (invariant) کہلاتا ہے۔
- لوپ باڈی (loop body) کے لیے ایک انویرینٹ کو لوپ انویرینٹ (loop invariant) کے نام سے جانا جاتا ہے۔
- ایک لوپ انویرینٹ صحیح ہے۔
- (a) لوپ کے شروع میں (لوپ سے بالکل پہلے)
- (b) ہر آئی ٹریشن کے آغاز پر (لوپ باڈی سے پہلے)
- (c) ہر آئی ٹریشن کے آخر میں (لوپ باڈی کے بعد)
- (d) لوپ کے آخر میں (صرف لوپ کے بعد)
- جب ایک لوپ ختم ہوتا ہے، لوپ انویرینٹ درست ہوتا ہے۔ اس کے علاوہ برطرفی کی شرط بھی درست ہے۔
- ری کرشن میں کم از کم ایک بیس کیس ہونا چاہیے۔
- ری کرشن مرحلہ مسئلے کو چھوٹے سائز کے ذیلی مسائل میں توڑ دیتا ہے، فرض کرتا ہے کہ ذیلی مسائل کا حل ری کرسیو کالوں کے ذریعے دیا جاتا ہے، اور دیے گئے مسئلے کا حل تیار کرتا ہے۔
- ری کرشن میں، ذیلی مسئلے میں ان پٹ کا سائز دیے گئے ان پٹ کے سائز سے سختی سے چھوٹا ہونا چاہیے۔

11th_Comp Science_UM_Ch 8.indd 112 18-04-2023 14:50:23

WG0U8

تشخیص

سیکشن – A

صحیح جواب کا انتخاب کریں

1. ایک لوپ انویرینٹ درست ہونے کی ضرورت نہیں ہے۔

(a) لوپ کے آغاز میں (b) ہر آئی ٹریشن کے شروع میں

(c) ہر آئی ٹریشن کے آخر میں (d) الگورتھم کے آغاز پر

2. ہم ایک بساط کو ڈومینوز کے ساتھ ڈھانپنا چاہتے ہیں، ☐☐ بالترتیب سیاہ مربعوں کی تعداد اور ڈومینوز کے ذریعے احاطہ کیے گئے سفید مربعوں کی تعداد، ڈومینو رکھ کر ماڈلنگ کی جا سکتی ہے۔

b := b + 2 (a) w := w + 2 (b) b, w := b+1, w+1 (c) b := w (d)

3. اگر a, b : = a + 8, b + 7 تفویض n x b + m x a کے لئے ایک متغیر ہے تو m اور n کی قدریں ہیں:

m = 8, n = 7 (a) m = 7, n = -8 (b) m = 7, n = 8 (c) m = 8, n = -7 (d)

4. مندرجہ ذیل میں سے کون سا اسائنمنٹ کا متغیر نہیں ہے؟

m, n := m+2, n+3

m mod 2 (a) n mod 3 (b) 3 X m - 2 X n (c) 2 X m - 3 X n (d)

5. اگر فبونیکی نمبر (Fibonacci Number) کو ری کرسیو طور پر بیان کیا جاتا ہے۔

$$F(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F(n-1) + F(n-2) & \text{otherwise} \end{cases}$$

F(4) کا اندازہ لگانے کے لیے، کتنی بار F() کا اطلاق ہوتا ہے؟

3 (a) 4 (b) 8 (c) 9 (d)

6. اس ری کرسیو تعریف کا استعمال کرتے ہوئے

$$a^n = \begin{cases} 1 & \text{if } n = 0 \\ a \times a^{n-1} & \text{otherwise} \end{cases}$$

a^{10} کو شمار کرنے کے لیے کتنے ضربوں کی ضرورت ہے؟

11 (a) 10 (b) 9 (c) 8 (d)

سیکشن – B

بہت ہی مختصر جوابات دیں

1. غیر متغیر کیا ہے؟
2. ایک لوپ انویرینٹ کی وضاحت کریں۔
3. کیا لوپ کی حالت کی جانچ لوپ انویرینٹ کو متاثر کرتی ہے؟ کیوں؟
4. لوپ انویرینٹ، لوپ کنڈیشن اور بار بار ان پٹ آؤٹ پٹ کے درمیان کیا تعلق ہے؟

11th_Comp Science_UM_Ch 8.indd 113 18-04-2023 14:50:23

5. ری کرسیو مسئلہ کا حل کیا ہے؟

6. قدرتی نمبر کے فیکٹوریل کی ری کرسیو طور پر وضاحت کریں۔

سیکشن – C

مختصر جوابات

1. ایک میز پر 7 ٹمبلر ہیں، سب الٹے کھڑے ہیں۔ آپ کو ایک ہی حرکت میں بیک وقت کوئی بھی 2 ٹمبلر موڑنے کی اجازت ہے۔ کیا ایسی صورت حال تک پہنچنا ممکن ہے جب تمام ٹمبلر دائیں طرف ہوں؟ (اشارہ: الٹا ٹمبلر کی تعداد کی برابری غیر متغیر ہے۔)

2. ناک آؤٹ ٹورنامنٹ کھیلوں کا ایک سلسلہ ہے۔ ہر کھیل میں دو کھلاڑی مقابلہ کرتے ہیں۔ ہارنے والا ناک آؤٹ ہو جاتا ہے (یعنی مزید نہیں کھیلتا)، جیتنے والا جاری رہتا ہے۔ ٹورنامنٹ کا فاتح وہ کھلاڑی ہوتا ہے جو باقی تمام کھلاڑیوں کے ناک آؤٹ ہونے کے بعد رہ جاتا ہے۔ فرض کریں کہ ایک ٹورنامنٹ میں 1234 کھلاڑی ہیں۔ ٹورنامنٹ کے فاتح کا فیصلہ ہونے سے پہلے کتنے کھیل کھیلے جاتے ہیں؟

3. بادشاہ وکرمادتیہ (Vikramaditya) کے پاس دو جادوئی تلواریں ہیں۔ ایک کے ساتھ، وہ ایک ڈریگن کے 19 سر کاٹ سکتا ہے، لیکن اس کے بعد ڈریگن کے 13 سر بڑھ جاتے ہیں۔ دوسری تلوار سے وہ 7 سر کاٹ سکتا ہے لیکن 22 نئے سر اگتے ہیں۔ اگر تمام سر کاٹ دیئے جائیں تو ڈریگن مر جائے گا۔ اگر ڈریگن کے اصل میں 1000 سر ہیں تو کیا یہ کبھی مر سکتا ہے؟ (اشارہ: ہیڈز موڈ 3 کی تعداد غیر متغیر ہے۔)

سیکشن – D

تفصیل سے بتائیں

1. معمول کے رنگ کے ساتھ 8 × 8 کی بساط فرض کریں۔ "دوبارہ رنگنے" آپریشن قطار یا کالم کے تمام مربعوں کا رنگ بدل دیتا ہے۔ آپ بار بار دوبارہ رنگ کر سکتے ہیں۔ مقصد صرف ایک سیاہ مربع کو حاصل کرنا ہے۔ دکھائیں کہ آپ مقصد حاصل نہیں کر سکتے۔ (اشارہ: اگر کسی قطار یا کالم میں b سیاہ مربع ہے تو یہ (|8-b) -b|) تبدیل ہو جاتا ہے۔

2. پاور کو بار بار کے طور پر بھی بیان کیا جا سکتا ہے۔

$$a^n = \begin{cases} 1 & \text{if } n = 0 \\ a \times a^{n-1} & \text{if } n \text{ is odd} \\ a^{n/2} \times a^{n/2} & \text{if } n \text{ is even} \end{cases}$$

اس تعریف کو استعمال کرتے ہوئے ایک ری کرسیو الگورتھم بنائیں۔ a^{10} کا حساب لگانے کے لیے کتنے ضربوں کی ضرورت ہے؟

3. سنگل مربع سے ڈھکنے والا بورڈ $2^n \times 2^n$ مربعوں کا بورڈ ہوتا ہے جس میں ایک مربع کو ایک مربع ٹائل سے ڈھانپا جاتا ہے۔ دکھائیں کہ اس بورڈ کو بغیر اوورلیپ (overlap) کے ٹرائیومینوز (triominoes) سے ڈھانپنا ممکن ہے۔

11th_Comp Science_UM_Ch 8.indd 114 18-04-2023 14:50:23

باب 9

C++ کا تعارف

9.1 تعارف

C++ مقبول ترین پروگرامنگ زبان میں سے ایک ہے جو طریقہ کار اور آبجیکٹ اورینٹڈ پروگرامنگ پیراڈائمز (Object Oriented programming paradigms) دونوں کی حمایت کرتی ہے۔ اس طرح، C++ کو ہائبرڈ زبان (**hybrid language**) کہا جاتا ہے۔ C++ اپنی پیشرو C زبان کا ایک سپر سیٹ (توسیع) ہے۔ بیارنے سٹروسٹرپ (Bjrane Stroustrup) نے اپنی نئی زبان کا نام "C with Classes" رکھا ہے ۔ C++ نام رِک میسٹی (Rick Mascitti) نے تیار کیا تھا جہاں C++ لینگویج انکریمنٹ آپریٹر (language increment operator) ہے۔

سیکھنے کے مقاصد

اس باب کی تکمیل کے بعد طلباء کے قابل ہوجائیں گے

- C++ پروگرامنگ لینگویج کے بنیادی بلڈنگ بلاکس کو سمجھنے
- سادہ C++ پروگرام بنانے
- C++ پروگراموں پر عمل درآمد اور ڈیبگ (debug) کرنے

بیارنے سٹروسٹرپ (Bjarne Stroustrup)

C++ پروگرامنگ زبان کا موجد



ایک ڈنمارک کے کمپیوٹر سائنسدان ہیں جو 30 دسمبر 1950 کو پیدا ہوئے۔ انہوں نے 1975 میں آرہس یونیورسٹی (Aarhus University)، ڈنمارک سے ریاضی اور کمپیوٹر سائنس میں ماسٹر ڈگری حاصل کی اور 1979 میں کیمبرج یونیورسٹی، انگلینڈ سے کمپیوٹر سائنس میں پی ایچ ڈی کی۔

C++ کی تاریخ

C++ کو AT&T بیل لیبارٹری (Bell Laboratory) میں بیارنے سٹروسٹرپ نے 1979 کے دوران تیار کیا تھا۔ C++ اصل میں C زبان سے ماخوذ ہے اور بہت سی زبانوں جیسے Simula, BCPL, Ada, ML, CLU اور ALGOL 68 سے متاثر ہے۔ 1983 تک اسے "New C" کہا جاتا تھا اور "C with Classes"۔ 1983 میں، رِک میسٹی (Rick Mascitti) نے نام بدل کر C++ رکھا۔

C++ سیکھنے کے فوائد

- C++ ایک انتہائی پورٹیبل زبان ہے اور اکثر ملٹی ڈیوائس (multi -device)، ملٹی پلیٹ فارم ایپ ڈیولپمنٹ (multi – platform App development) کے لیے انتخاب کی زبان ہوتی ہے۔
- C++ ایک آبجیکٹ پر مبنی پروگرامنگ لینگویج ہے اور اس میں کلاسز (classes)، ان ہیری ٹینس (inheritance)، پولیمورفزم (polymorphism)، ڈیٹا ایبسٹریکشن (Data abstraction) اور انکیپ سولیشن(encapsulation) شامل ہیں۔
- C++ میں ایک بھرپور فنکشن لائبریری ہے۔
- C++ رعایت کی ہینڈلنگ ، وراثت اور فنکشن اوور لوڈنگ (function overloading) کی اجازت دیتا ہے جو C میں ممکن نہیں ہے۔
- C++ ایک طاقتور، موثر اور تیز زبان ہے۔ یہ ایپلی کیشنز کی ایک وسیع رینج - GUI ایپلی کیشنز (Applications) سے لے کر گیمز کے لیے 3D گرافکس (Graphics) تک ریئل ٹائم (real –time) ریاضی کے نقوش تلاش کرتا ہے ۔

11th_Comp Science_UM_Ch 9.indd 115 18-04-2023 15:04:15

9.2 کریکٹر سیٹ
(Character set)

کریکٹر سیٹ حروف کا ایک مجموعہ ہے جو C++ پروگرام لکھنے کے لیے استعمال ہوتا ہے۔ ایک حرف کسی بھی حروف تہجی، نمبر یا کسی دوسری علامت (خصوصی حروف) کی نمائندگی کرتا ہے جو زیادہ تر کی بورڈ میں دستیاب ہے۔ C++ درج ذیل حروف کو قبول کرتا ہے۔

A …. Z, a ….. z	Alphabets
0 …. 9	Numeric
+ - * / ~ ! @ # $ % ^& [ ] ( ) { } = >< _ \ | ? . , : ‘ “ ;	Special Characters
Blank space, Horizontal tab (→), Carriage return (↵), Newline, Form feed	White space
C++ can process any of the 256 ASCII characters as data.	Other characters

9.3 لغوی اکائیاں (ٹوکنس):
(Lexical Units (Tokens))

C++ پروگرام کے بیانات بہت سے مختلف چھوٹے عناصر جیسے کمانڈز، متغیرات، مستقل اور بہت سی علامتوں کے ذریعے بنائے جاتے ہیں جنہیں آپریٹرز (operators) اور پنکچوئیٹرس (Punctuators) کہا جاتا ہے۔ان انفرادی عناصر کو اجتماعی طور پر لغوی اکائیاں (lexical units) یا لغوی عناصر (Lexical elements) یا ٹوکنس (Tokens) کہا جاتا ہے۔ C++ میں درج ذیل ٹوکنس ہیں:

- کلیدی الفاظ (Keywords)
- آپریٹرز (Operators)
- شناخت کنندگان (identifiers)
- پنکچوئیٹر (punctuators)
- لٹرلس (literals)

ٹوکن (TOKEN):

پروگرام میں سب سے چھوٹی انفرادی اکائی کو ٹوکن یا لغوی یونٹ کہا جاتا ہے۔

9.3.1 کلیدی الفاظ (keywords)

کلیدی الفاظ وہ محفوظ الفاظ ہیں جو C++ مرتب کرنے والے کو مخصوص معنی دیتے ہیں۔ وہ C++ پروگراموں کی تعمیر کے لیے ضروری عناصر ہیں۔ زیادہ تر کلیدی الفاظ C، C++ اور Java میں عام ہیں۔

C++ ایک کیس حساس پروگرامنگ لینگویج ہے اس لیے تمام مطلوبہ الفاظ چھوٹے حروف میں ہونے چاہئیں۔

ٹیبل 9.1 C++ کلیدی الفاظ

catch	case	break	auto	asm
default	continue	const	class	char
enum	else	double	do	delete
goto	friend	for	float	extern
new	long	int	inline	if
register	public	protected	private	operator
static	sizeof	signed	short	return
throw	this	template	switch	struct
virtual	unsigned	union	typedef	try
		while	volatile	void

- نظر ثانی اور اضافے کے ساتھ، مطلوبہ الفاظ کی حالیہ فہرست میں یہ بھی شامل ہیں:

using, namespace, bal, static_cast, const_cast, dynamic_cast, true, false

- ڈبل انڈر سکور پر مشتمل شناخت کنندگان C++ نفاذ اور معیاری لائبریریوں کے استعمال کے لیے محفوظ ہیں اور صارفین کو ان سے بچنا چاہیے۔

9.3.2 شناخت کنندگان (Identifiers)

شناخت کنندگان صارف کے متعین کردہ نام ہیں جو C++ پروگرام کے مختلف حصوں کو دیئے گئے ہیں۔ متغیرات، افعال، صفیں، کلاسز وغیرہ، یہ ایک پروگرام کے بنیادی تعمیراتی بلاکس ہیں۔ ہر زبان میں شناخت کنندگان کے نام کے لیے مخصوص اصول ہوتے ہیں۔

شناخت کنندہ کا نام دینے کے اصول:

- شناخت کنندہ کا پہلا حرف حروف تہجی یا انڈر سکور (underscore) (_) ہونا چاہیے۔
- صرف حروف تہجی، ہندسے اور انڈر سکور کی اجازت ہے۔ شناخت کنندہ کے حصے کے طور پر دیگر خصوصی حروف کی اجازت نہیں ہے۔
- C++ کیس حساس ہے کیونکہ یہ اپر (upper) اور لوئر کیس (Lower case) کے حروف کو مختلف طریقے سے دیکھتا ہے۔
- محفوظ الفاظ یا مطلوبہ الفاظ کو شناخت کنندہ کے نام کے طور پر استعمال نہیں کیا جا سکتا۔

11th_Comp Science_UM_Ch 9.indd 116 18-04-2023 15:04:15

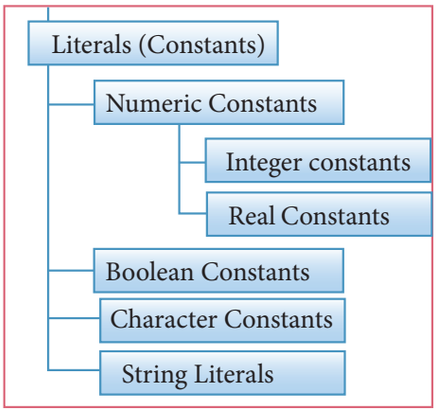
ANSI معیارات کے مطابق، C++ اس کی لمبائی کی کوئی حد نہیں رکھتا اور اس لیے تمام حروف اہم ہیں۔

Reason for invalid	Valid / Invalid	Identifiers
	Valid	Num
	Valid	NUM
	Valid	_add
	Valid	total_sales
	Valid	tamilMark
Contains special character (-)	Invalid	num-add
This is one of the keyword. Keyword cannot be used as identifier names.	Invalid	this
Name must start begins with an alphabet or an underscore	Invalid	2myfile

- آپ نام کے مختلف حصوں کو الگ کرنے کے لیے متغیر ناموں میں ایک انڈر سکور استعمال کر سکتے ہیں (مثال کے طور پر: total_sales ایک درست شناخت کنندہ ہے جہاں **total sales** نامی متغیر ایک غلط شناخت کنندہ ہے)۔
- آپ کیپٹل سٹائل اشارے استعمال کر سکتے ہیں، جیسے کہ **tamilMark** یعنی دوسرے لفظ کے پہلے حرف کو بڑا کرنا۔

9.3.3 لٹریلس (کانس ٹینس) (Literals (Constants))

لٹریلس ڈیٹا آئٹمز ہیں جن کی قدریں کسی پروگرام کے عمل کے دوران تبدیل نہیں ہوتی ہیں۔ اس لیے لٹریلس کو کانس ٹینس (Constants) کہا جاتا ہے۔ C++ میں کئی قسم کے لٹریلس ہیں:



تصویر 9.1 کانس ٹینس کی اقسام

عددی کانس ٹینس (Numeric Constants)

جیسا کہ نام سے ظاہر ہوتا ہے، عددی کانس ٹینس عددی اقدار ہیں، جو کانس ٹینس کے طور پر استعمال ہوتی ہیں۔ عددی کانس ٹینس کو مزید درجہ بندی کیا گیا ہے:

1. عددی کانس ٹینس (یا) فکسڈ پوائنٹ کانس ٹینس ۔
(Integer Constants (or) Fixed point constants)
2. اصلی کانس ٹینس (یا) فلوٹنگ پوائنٹ کانس ٹینس ۔
(Real Constants (or) Floating point constants)

(1) عددی کانس ٹینس (یا) فکسڈ پوائنٹ کانس ٹینس

عدد مکمل اعداد ہیں بغیر کسی کسر کے۔ ایک عدد کانس ٹینس میں کم از کم ایک ہندسہ ہونا چاہیے بغیر اعشاریہ کے۔ یہ دستخط شدہ یا غیر دستخط شدہ ہوسکتا ہے۔ دستخط شدہ عدد کو منفی سمجھا جاتا ہے، اس کے حصے کے طور پر کوما اور خالی جگہوں کی اجازت نہیں ہے۔ C++ میں، عددی کانس ٹینس کی تین قسمیں ہیں: (i) اعشاریہ (Decimal) (ii) آکٹل (Octal) (iii) ہیکساڈیسیمل (Hexadecimal)

(i) اعشاریہ

ایک یا زیادہ ہندسوں کا کوئی بھی سلسلہ (0....9)

Invalid	Valid
7,500 (Comma is not allowed)	725
66 5 (Blank space is not allowed)	-27
9$ (Special Character not allowed)	4.56

اگر آپ **4.56** کو عددی اعشاریہ کانس ٹینس کے طور پر تفویض کرتے ہیں، تو مرتب کرنے والا صرف **4.56** کے عددی حصے کو قبول کرے گا یعنی **4**۔ یہ صرف **.56** نظر انداز کرے گا۔

نوٹس (Notes)

اگر اعشاریہ کانس ٹینٹ کو کسر کے ساتھ ظاہر کیا جاتا ہے، تو کمپائلر قدر کا صرف عددی حصہ لے گا اور اس کے جزوی حصے کو نظر انداز کر دے گا۔ اسے " اِمپلی سِٹ کنورشن (Implicit conversion)" کہا جاتا ہے۔ اس پر بعد میں بات کی جائے گی۔

(ii) آکٹل

ایک یا ایک سے زیادہ آکٹل اقدار (0.....7) کی کوئی بھی ترتیب جو 0 سے شروع ہوتی ہے اسے آکٹل کانس ٹینٹ (Octal Constant) سمجھا جاتا ہے۔

11th_Comp Science_UM_Ch 9.indd 117 18-04-2023 15:04:16

Invalid	Valid
05,600(Commas is not allowed)	012
04.56 (Decimal point is not allowed)**	-027
0158 (8 is not a permissible digit in octal system)	+0231

نوٹس

** جب آپ ایک جزوی نمبر استعمال کرتے ہیں جو 0 سے شروع ہوتا ہے، تو C++ عدد کو عددی شمار کرتا ہے نہ کہ آکٹل۔

(iii) ہیکساڈیسیمل (Hexadecimal)

ایک یا زیادہ ہیکساڈیسیمل اقدار (9.....0, A.....F) کا کوئی بھی سلسلہ جو 0x یا 0X سے شروع ہوتا ہے جسے ہیکساڈیسیمل کانس ٹینٹ (Hexadecimal Constant) سمجھا جاتا ہے۔

Invalid	Valid
0x1,A5 (Commas is not allowed)	0x123
0x.14E (Decimal point is not allowed like this)	0X568

لاحقہ L یا l اور U یا u کسی بھی مستقل قوت کے ساتھ جوڑا جاتا ہے اسے بالترتیب ایک طویل یا غیر دستخط شدہ مستقل کے طور پر ظاہر کیا جاتا ہے۔

(2) ریئل کانس ٹینٹس (یا) فلوٹنگ پوائنٹ کانس ٹینٹس

(Real Constants (or) Floating Point Constants)

ریئل یا فلوٹنگ پوائنٹ کانس ٹینٹ ایک عددی کانس ٹینٹ ہے، جس میں فریکشنل جزء (Fractional Component) کا ہوتاہے۔ ان کانس ٹینٹ کو جزوی شکل میں یا ایکسپونٹ (Exponent) شکل میں لکھا جا سکتا ہے۔

ایک حقیقی کانس ٹینٹ کی جزوی شکل ہندسوں کی ایک دستخط شدہ یا غیر دستخط شدہ ترتیب ہے جس میں ہندسوں کے درمیان اعشاریہ بھی شامل ہے۔ اس میں اعشاریہ سے پہلے اور بعد میں کم از کم ایک ہندسہ ہونا چاہیے۔ یہ + یا - نشان کے ساتھ سابقہ ہو سکتا ہے۔ بغیر کسی نشان کے حقیقی کانس ٹینٹ کو مثبت سمجھا جائے گا۔

حقیقی کانس ٹینٹ کی ایکسپونٹ (Exponent) شکل دو حصوں پر مشتمل ہے: (1) مین ٹیسا (Mantissa) اور (2)اکسپونینٹ (Exponent) مین ٹیسا یا تو ایک عدد یا حقیقی کانس ٹینٹ ہونا ضروری ہے۔ مین ٹیسا کے بعد ایک حرف E یا e اور اکسپونینٹ بھی ایک عدد ہونا چاہیے۔

مثال کے طور پر، 58000000.00 کو 0.58×10^8 یا 0.58E8 لکھا جا سکتا ہے۔

Exponent (After E)	Mantissa (Before E)
8	0.58

مثال :

5.864 E1	$10^1 \times 5.864$	➔	58.64
5864 E-2	$10^{-2} \times 5864$	➔	58.64
0.5864 E2	$10^2 \times 0.5864$	➔	58.64

بولین لٹرلس (Boolean literals)

بولین لٹرلس کا استعمال بولین قدروں میں سے کسی ایک (صحیح یا غلط) کی نمائندگی کے لیے کیا جاتا ہے۔ اندرونی طور پر صحیح کی قدر 1 ہے اور غلط کی قدر 0 ہے۔

کیریکٹر کانس ٹینٹ (Character constant)

C++ میں ایک کیریکٹر کانس ٹینٹ کوئی بھی صحیح ایک کیریکٹر ہے جو ایک اقتباس میں بند ہے۔

صحیح کیریکٹر کانس ٹینٹس : '$', '2', 'A'

غلط کیریکٹر کانس ٹینٹس : "A"

ایک واحد کیریکٹر کانس ٹینٹ کی قدر ایک مساوی ASCII قدر رکھتی ہے۔ مثال کے طور پر، 'A' کی قدر 65 ہے۔

اسکیپ سیکوئنس(یا) غیر گرافک کیریکٹرس

(Escape sequences (or) non-graphic characters)

C++ بعض غیر پرنٹ ایبل (Non-Printable)حروف کی اجازت دیتا ہے جس کی نمائندگی کیریکٹر کانس ٹینٹس کے طور پر کی جاتی ہے۔ غیر پرنٹ ایبل حروف کو غیر گرافک کیریکٹرس (non-graphic characters) بھی کہا جاتا ہے۔ غیر پرنٹ ایبل حروف وہ حروف ہیں جو C++ میں کسی پروگرام کے عمل کے دوران براہ راست کی بورڈ سے ٹائپ نہیں کیے جا سکتے ہیں، مثال کے طور پر: بیک اسپیس (back space)، ٹیبز (tabs) وغیرہ۔ ان غیر پرنٹ ایبل حروف کو اسکیپ سیکوئنس کا استعمال کرکے دکھایا جا سکتا ہے۔بچنے کی ترتیب کو بیک سلیش (Back slash) کے ذریعے دکھایا جاتا ہے جس کے بعد ایک یا دو حروف ہوتے ہیں۔

11th_Comp Science_UM_Ch 9.indd 118 18-04-2023 15:04:16

ٹیبل 9.2 اسکیپ سیکوئنس

Non-graphical character	Escape sequence
Audible or alert bell	\a
Backspace	\b
Form feed	\f
Newline or linefeed	\n
Carriage return	\r
Horizontal tab	\t
Vertical tab	\v
Backslash	\\
Single quote	\'
Double quote	\"
Question Mark	\?
Octal number	\On
Hexadecimal number	\xHn
Null	\0

اگرچہ اسکیپ سیکوئنس میں دو حروف ہوتے ہیں، لیکن انہیں ایک اقتباسات میں بند کیا جانا چاہیے کیونکہ، C++ اسکیپ سیکوئنس کو کریکٹر کانس ٹینس کے طور پر سمجھتا ہے اور ASCII نمائندگی میں ایک بائٹ مختص کرتا ہے۔

کیا آپ جانتے ہیں؟

ASCII (امریکن اسٹینڈرڈ کوڈ فار انفارمیشن انٹرچینج American standard code for information exchange) پہلی بار 1963 میں X3 کمیٹی کے ذریعہ تیار اور شائع کیا گیا تھا، جو امریکن اسٹینڈرڈس اسوسی ایشن (ASA – American standards Association) کا ایک حصہ ہے۔

خود کا محاسبہ کیجیے

1. لٹرلس سے کیا مراد ہے؟ C++ میں عددی لٹرلس کی کتنی اقسام پائی جاتی ہیں؟
2. درج ذیل میں کانس ٹینٹس کی کون سی قسم ہے؟
   (i) 26 (ii) 015 (iii) 0xF (iv) 014.9
3. C++ میں کریکٹر کانس ٹینٹ کیا ہے؟
4. C++ میں غیر گرافک کیریکٹرس کی نمائندگی کیسے کی جاتی ہے؟
5. درج ذیل ریئل کانس ٹینٹس کو ایکسپونٹ کی شکل میں لکھیں:
   (i) 32.179 (ii) 8.124 (iii) 0.00007
6. درج ذیل ریئل کانس ٹینٹس کو جزوی شکل میں لکھیں:
   (i) 0.23E4 (ii) 0.517E-3 (iii) 0.5E-5
7. سٹرنگ میں \0 (null) کریکٹر کی کیا اہمیت ہے؟

سٹرنگ لٹرلس (String literals)

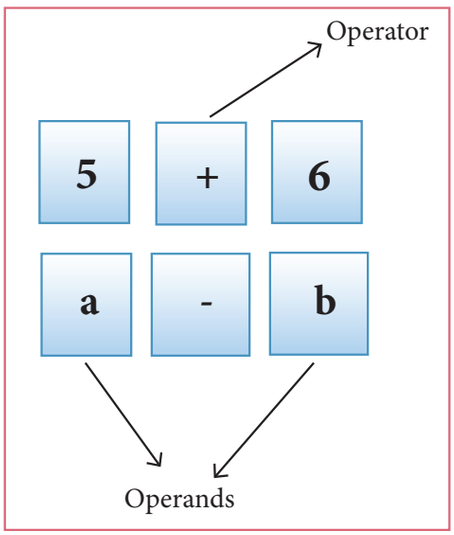
دوہرے اقتباسات کے اندر بند حروف کی ترتیب کو سٹرنگ لٹرلس (String literals) کہا جاتا ہے۔ پہلے سے طے شدہ طور پر، سٹرنگ لٹرلس خود بخود ایک خاص حرف (Null) '\0' کے ساتھ آخر میں شامل ہو جاتے ہیں۔ لہذا، سٹرنگ " welcome " کو دراصل میموری میں "welcome\0" کے طور پر دکھایا جائے گا اور اس سٹرنگ کا سائز 7 نہیں بلکہ 8 حروف یعنی آخری کریکٹر سمیت \0 ہے۔

صحیح سٹرنگ لٹرلس: "A", "welcome" "1234"

غلط سٹرنگ لٹرلس: 'welcome', '1234'

9.3.4 آپریٹرز (Operators)

وہ علامتیں جو کچھ ریاضیاتی یا منطقی کارروائیوں کے لیے استعمال ہوتی ہیں انہیں "آپریٹرز" کہا جاتا ہے۔ ڈیٹا آئٹمز یا اقدار جن پر آپریٹر عمل کرتے ہیں انہیں "آپرینڈز (Operands)" کہا جاتا ہے۔



C++ میں، آپریٹرز کو آپرینڈز کی تعداد کی بنیاد پر درجہ بندی کیا جاتا ہے۔

(i) یونری آپریٹرز (Unary Operators) - صرف ایک آپرینڈ کی ضرورت ہے۔

(ii) بائنری آپریٹرز (Binary Operators) - دو آپرینڈز کی ضرورت ہوتی ہے۔

(iii) ٹرنری آپریٹرز (Ternary Operators) - تین کی ضرورت ہے۔

11th_Comp Science_UM_Ch 9.indd 119 18-04-2023 15:04:16

C++ بائنری آپریٹرز کی درجہ بندی اس طرح کی گئی ہے:

(1) ارتھ میٹک آپریٹرز (Arithmetic Operators)
(2) ریلیشنل آپریٹرز (Relational Operators)
(3) لوجیکل آپریٹرز (Logical Operators)
(4) اسائنمنٹ آپریٹرز (Assignment Operators)
(5) کنڈیشنل آپریٹر (Conditional Operators)

(1) ارتھ میٹک آپریٹرز (Arithmetic Operators)

ارتھ میٹک آپریٹرز سادہ ریاضی کی کارروائیاں انجام دیتے ہیں جیسے جمع، گھٹاؤ، ضرب، تقسیم وغیرہ،

Example	Operation	Operator
10 + 5 = 15	جمع	+
10 – 5 = 5	گھٹاؤ	-
10 * 5 = 50	ضرب	*
10 / 5 = 2 (تقسیم کی مقدار)	تقسیم	/
10 % 3 = 1 (بقیہ ڈویژن)	ماڈیولس (بقیہ تقسیم کو تلاش کرنے کے لیے)	%

- مذکورہ ارتھ میٹک آپریٹرز بائنری آپریٹرز ہیں جن کے لیے کم از کم دو آپرینڈز (operands) کی ضرورت ہوتی ہے۔

انکریمنٹ اور ڈیکریمنٹ آپریٹرز (Increment and Decrement Operators)

++ (پلس، پلس) انکریمنٹ آپریٹر

-- (مائنس، مائنس) ڈیکریمنٹ آپریٹر

ایک انکریمنٹ یا ڈیکریمنٹ آپریٹر ایک ہی آپرینڈ پر عمل کرتا ہے اور ایک نئی قدر واپس کرتا ہے۔ اس طرح، یہ آپریٹرز یونری آپریٹرز ہیں۔ انکریمنٹ آپریٹر اپنے آپرینڈ میں 1 کا اضافہ کرتا ہے اور ڈیکریمنٹ آپریٹر اپنے آپرینڈ سے 1 کو گھٹاتا ہے۔ مثال کے طور پر،

- **x++ یا ++x وہی ہے جو x = x+1;**
  یہ x کی موجودہ قدر میں 1 کا اضافہ کرتا ہے۔
- **x-- یا --x وہی ہے جیسا کہ x = x–1;**
  یہ x کی موجودہ قدر سے 1 کو گھٹاتا ہے

++ یا -- آپریٹرز کو متغیر میں یا تو سابقہ (before) یا لاحقہ (postfix) کے طور پر رکھا جا سکتا ہے۔ سابقہ ورژن کے ساتھ، C++ اوپرینڈ استعمال کرنے سے پہلے انکریمنٹ/ڈیکریمنٹ کو انجام دیتا ہے۔

(2) ریلیشنل آپریٹرز (Relational Operators)

ریلیشنل آپریٹرز کو اس کے آپرینڈز کے درمیان تعلق کا تعین کرنے کے لیے استعمال کیا جاتا ہے۔ جب ریلیشنل آپریٹرز کو دو آپرینڈز پر لاگو کیا جاتا ہے، تو نتیجہ بالترتیب صحیح یا غلط کی نمائندگی کرنے کے لیے بولین ویلیو (boolean value) یعنی 1 یا 0 ہو گا۔ C++ چھ ریلیشنل آپریٹرز فراہم کرتا ہے۔ وہ ہیں،

Example	Operation	Operator
a > b	Greater than	>
a < b	Less than	<
a >= b	Greater than or equal to	>=
a <= b	Less than or equal to	<=
a == b	Equal to	==
a != b	Not equal	!=

- مندرجہ بالا مثالوں میں، آپرینڈ a کا موازنہ b کے ساتھ کیا گیا ہے اور تعلق کی بنیاد پر، نتیجہ یا تو 1 یا 0 ہو گا۔ یعنی، صحیح کے لیے 1، غلط کے لیے 0۔
- تمام چھ ریلیشنل آپریٹرز بائنری آپریٹرز ہیں۔

(3) لوجیکل آپریٹرز

ایک لوجیکل آپریٹر کا استعمال منطقی اور متعلقہ اظہارات کی جانچ کے لیے کیا جاتا ہے۔ لوجیکل آپریٹرز ان آپرینڈز پر عمل کرتے ہیں جنہیں خود منطقی اظہار (logical expressions) کہا جاتا ہے۔ C++ تین لوجیکل آپریٹرز فراہم کرتا ہے۔

ٹیبل 9.3 لوجیکل آپریٹرز

آپریٹر	آپریشن	تفصیل
&&	AND	لوجیکل AND دو مختلف متعلقہ اظہارات کو ایک میں جوڑتا ہے۔ یہ 1 (صحیح) لوٹاتا ہے، اگر دونوں اظہار درست ہیں، بصورت دیگر یہ 0 (غلط) لوٹاتا ہے۔

11th_Comp Science_UM_Ch 9.indd 120 18-04-2023 15:04:16

||	OR	لوجیکل OR دو مختلف متعلقہ اظہارات کو ایک میں جوڑتا ہے۔ یہ 1 (صحیح) لوٹاتا ہے، اگر اظہار میں سے کوئی ایک صحیح ہے۔ یہ 0 (غلط) لوٹاتا ہے، اگر دونوں اظہارار غلط ہیں۔
!	NOT	NOT کسی ایک اظہار/آپرینڈ پر کام کرتا۔ یہ صرف کی قدر کی نفی کرتا ہے یا اسے الٹ دیتا ہے۔ یعنی، اگر آپرینڈ/اظہار 1 (صحیح) ہے تو یہ آپریٹر 0 (غلط) اور اس کے برعکس لوٹاتا ہے۔

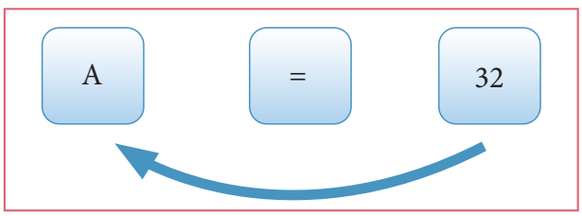
- AND, OR دونوں بائنری آپریٹر ہیں جہاں NOT ایک یونری آپریٹر (unary operator) ہے۔

مثال: a = 5, b = 6, c = 7;

Result	Expression
1 (True)	(a<b) && (b<c)
0 (False)	(a>b) && (b<c)
1 (True)	(a<b) || (b>c)
1 (True)	!(a>b)

(4) اسائنمنٹ آپریٹر:

اسائنمنٹ آپریٹر کا استعمال کسی متغیر کو قدر کی تفویض کرنے کے لیے کیا جاتا ہے جو اسائنمنٹ اسٹیٹمنٹ کے بائیں جانب ہوتا ہے۔ = (برابر) عام طور پر تمام کمپیوٹر پروگرامنگ زبانوں میں اسائنمنٹ آپریٹر کے طور پر استعمال ہوتا ہے۔ یہ آپریٹر آپریٹر کے دائیں جانب والی قدر کو بائیں جانب متغیر میں کاپی کرتا ہے۔ یہ ایک بائنری آپریٹر بھی ہے۔



C++ مختلف قسم کے اسائنمنٹ آپریٹرز استعمال کرتا ہے۔ انہیں شارٹ ہینڈ اسائنمنٹ آپریٹرز (Shorthand assignment operators) کہا جاتا ہے۔

Example	Name of Operator	Operator

a = 10;
c = a += 5;
(ie, a = a + 5)
c = 15
	Addition Assignment	+=

a = 10;
c = a -= 5;
(ie. a = a – 5)
c = 5
	Subtraction Assignment	-=

a = 10;
c = a *= 5;
(ie. a = a * 5)
c = 50
	Multiplication Assignment	*=

a = 10;
c = a /= 5;
(ie. a = a / 5)
c = 2
	Division Assignment	/=

a = 10;
c = a %= 5;
(ie. a = a % 5)
c = 0
	Modulus Assignment	%=

= اینڈ == آپریٹرز کے درمیان فرق پر بحث کریں۔

(5) کنڈیشنل آپریٹر:

C++ میں، صرف ایک کنڈیشنل آپریٹر ہے ?: جو مشروط آپریٹر ہے جسے ٹرنری آپریٹر (Ternary operator) بھی کہا جاتا ہے۔ اس آپریٹر کو if ... else کنٹرول اسٹیٹمنٹ کے متبادل کے طور پر استعمال کیا جاتا ہے۔ ہم اس آپریٹر کے بارے میں if ... else کے ساتھ بعد کے ابواب میں مزید جانیں گے۔

دوسرے آپریٹرز:

کوما آپریٹر (Comma operator)	کوما (,) C++ میں ایک آپریٹر ہے جو کئی اظہار کو ایک ساتھ جوڑنے کے لیے استعمال ہوتا ہے۔ کوما سے الگ کیے گئے اظہار کے گروپ کا بائیں سے دائیں تک جائزہ لیا جاتا ہے۔
سائز آف (Sizeof)	اسے کمپائل ٹائم آپریٹر (Compile time operator) کہا جاتا ہے۔ یہ بائٹس میں متغیر کا سائز لوٹاتا ہے۔
پوائنٹر (Pointer)	* متغیر کی طرف اشارہ & اڈریس آف

11th_Comp Science_UM_Ch 9.indd 121 18-04-2023 15:04:16

اجزاء کا انتخاب (component selection)	.	براہ راست اجزا کا انتخاب کنندہ یا
	<-	بالواسطہ اجزاء کا انتخاب کنندہ
کلاس ممبر آپریٹرس (Class member operators)	::	دائرہ کار تک رسائی / قرار داد
	.*	ڈیریفرنس (Deference)
	<-*	کلاس ممبر کے لیے ڈیریفرنس پوائنٹر

آپریٹرز کی ترجیح: (Precedence of Operators)

آپریٹرز کو ترجیح کی ترتیب میں عمل کیا جاتا ہے۔ آپرینڈز اور آپریٹرز کو تشخیص کے لیے ایک مخصوص منطقی انداز میں گروپ کیا گیا ہے۔ اس منطقی گروہ بندی کو انجمن (Association) کہا جاتا ہے

ترجیحی ترتیب: (The order of precedence)

( ) [ ]	قوسین کے اندر آپریٹرز کو پہلے انجام دیا جاتا ہے۔	ہائر
++, --	پوسٹ فکس (بعد) انکریمنٹ / ڈکریمنٹ	
++, --	پری فکس (پہلے) انکریمنٹ / ڈکریمنٹ	
*, /, %	ضرب، تقسیم، ماڈیولس	
+, -	جمع، گھٹاؤ	
<, <=, >, >=	اس سے کم، اس سے کم یا برابر، اس سے بڑا، اس سے بڑا یا برابر	
==, !=	کے برابر، برابر نہیں۔	
&&	لوجیکل AND	
||	لوجیکل OR	
?:	کنڈیشنل آپریٹر	
=	سادہ اسائنمنٹ	
+=, -=, *=, /=	شارٹ ہینڈ آپریٹرز	
,	کوما آپریٹر	لوور

9.3.5 اوقاف (Punctuators)

اوقاف وہ علامت ہیں، جو C++ پروگرام کی تعمیر کے دوران حد بندی کے طور پر استعمال ہوتے ہیں۔ انہیں "سیپریٹرس - Separators" بھی کہا جاتا ہے۔ درج ذیل اوقاف C++ میں استعمال ہوتے ہیں۔ ان میں سے زیادہ تر علامتیں C اور Java سے بہت ملتی جلتی ہیں۔

سیپریٹر	تفصیل	مثال

گھونگرالی قوسین { } (curly braces)	گھونگرالی قوسین کو کھولنا اور بند کرنا کوڈ کے بلاک کے آغاز اور اختتام کی نشاندہی کرتا ہے۔ کوڈ کا ایک بلاک جس میں ایک سے زیادہ قابل عمل بیانات ہوں۔ ان بیانات کو ایک ساتھ "compound statement" کہا جاتا ہے	int main ( )
		{
		int x=10, y=20, sum;
		sum = x + y;
		cout << sum;
		}

قوسین ( )	کھولنے اور بند ہونے والے قوسین کا اشارہ ہے۔ فنکشن کالز اور فنکشن پیرامیٹرز۔	clrscr( ); int main ( )
مربع بریکٹ [ ]	یہ واحد اور کثیر جہتی صفوں کی نشاندہی کرتا ہے۔	int num[5]; char name[50];
کوما،	یہ ایک اظہار میں ایک جداکار کے طور پر استعمال ہوتا ہے	int x=10, y=20, sum;

11th_Comp Science_UM_Ch 9.indd 122 18-04-2023 15:04:17

سیمی کولن ؛	C++ میں ہر قابل عمل بیان کو سیمی کولن کے ساتھ ختم ہونا چاہئے۔	int main ( )
		{
		int x=10, y=20, sum;
		sum = x + y;
		cout << sum;
		}

کولن:	یہ ایک بیان کو لیبل کرنے کے لئے استعمال کیا جاتا ہے۔	private:

تبصرے
//
/* */
	کوئی بھی بیان جو // سے شروع ہوتا ہے تبصرے سمجھا جاتا ہے۔ تبصرے صرف مرتب کرنے والوں کے ذریعہ نظر انداز کیے جاتے ہیں۔ یعنی، کمپائلر کسی بھی بیان پر عمل نہیں کرتا ہے جو // سے شروع ہوتا ہے
	// سنگل لائن تبصرہ
	/*.....*/ ملٹی لائن تبصرہ
		/* This is written by me to learn CPP */
		int main ( )
		{
		int x=10, y=20, sum;
		// to sum x and y
		sum = x + y;
		cout << sum;
		}

C++ میں، ایک یا دو آپریٹرز کو مختلف جگہوں پر مختلف معنی کے ساتھ استعمال کیا جا سکتا ہے۔
مثال کے طور پر: اسٹرسک (Asterisk) (*) ضرب کے ساتھ ساتھ متغیر کی طرف اشارہ کرنے کے لیے بھی استعمال کیا جاتا ہے۔

خود کا محاسبہ کیجیے

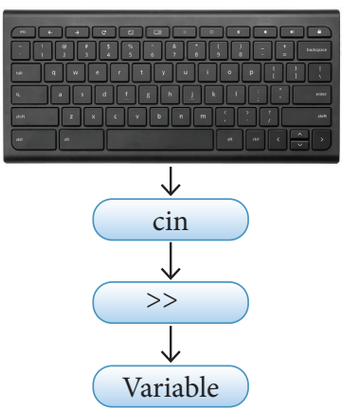
1. آپریٹرز کا استعمال کیا ہے؟
2. بائنری آپریٹرز کیا ہیں؟ ارتھ میٹک بائنری آپریٹرز کی مثالیں دیں؟
3. ماڈیولس آپریٹر % کیا کرتا ہے؟
4. 2 %8.5 کا نتیجہ کیا ہوگا؟
5. بتائیں کہ i = 8, j = 10, k = 8, مندرجہ ذیل اظہار کا نتیجہ کیا ہوگا؟
   i > = k (iii) i < j (ii) i < k (i)
   j ! = k (v) i = = j (iv)
6. درج ذیل تاثرات کے لیے تشخیص کا حکم کیا ہوگا؟
   a +10 < p - 3 + 2 q (ii) i + 3 >= j - 9 (i)
7. اگر نمبر 75 ہیں اور گریڈ 'A' ہے تو ٹیسٹ کرنے کے لیے ایک لوجیکل آپریٹر پر مشتمل ایک اظہار لکھیں۔

9.4 آئی / اُو آپریٹرس (I /O Operators)

9.4.1 ان پٹ آپریٹر (Input operator)

C++ آپریٹر >> فراہم کرتا ہے ان پٹ حاصل کرنے کے لیے۔ یہ کی بورڈ کے ذریعے قدر نکالتا ہے اور اسے اپنے دائیں جانب متغیر کو تفویض کرتا ہے۔ لہذا، اسے **"سٹریم ایکسٹرکشن (Stream extraction) یا "گیٹ فرام (get from)"** آپریٹر کہا جاتا ہے۔

یہ ایک بائنری آپریٹر ہے یعنی اسے دو آپرینڈز کی ضرورت ہوتی ہے۔ پہلا آپرینڈ پہلے سے طے شدہ شناخت کنندہ cin ہے (جسے C-In کہا جاتا ہے) جو کی بورڈ کو ان پٹ ڈیوائس کے طور پر شناخت کرتا ہے۔ دوسرا آپرینڈ متغیر ہونا چاہیے۔



تصویر 9.4 cin کے کام کرنے کا عمل

ایک وقت میں ایک سے زیادہ قدر حاصل کرنے یا نکالنے کے لیے، >> آپریٹر ہر متغیر کے لیے استعمال کیا جانا چاہیے۔ اس کو آپریٹر کا کیسکیڈنگ (Cascading of operator) کہا جاتا ہے۔

مثال:

cin >> num;	پہلے سے طے شدہ آبجیکٹ cin کی بورڈ پر ٹائپ کردہ قدر کو نکالتا ہے اور اسے متغیر نمبر میں اسٹور کرتا ہے۔

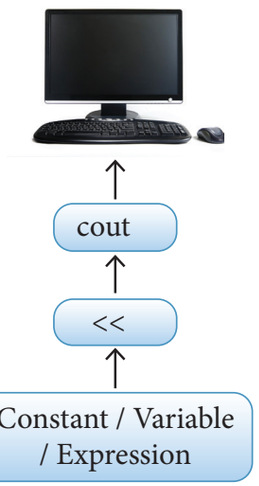
11th_Comp Science_UM_Ch 9.indd 123 18-04-2023 15:04:17

cin >>x >> y;	یہ دو قدروں کو نکالنے کے لیے استعمال ہوتا ہے۔ cin پہلی قدر پڑھتا ہے اور اسے فوری طور پر تفویض کرتا ہے۔ متغیر x تک؛ اگلا، یہ پڑھتا ہے،دوسری قدر جو اسپیس(space) کے بعد ٹائپ کی جاتی ہے اور اسے y کو تفویض کرتی ہے۔ اسپیس ہر ان پٹ کے لیے الگ کرنے والے کے طور پر استعمال ہوتی ہے۔

9.4.2 آؤٹ پٹ آپریٹر (Output Operator)

C++ آؤٹ پٹ آپریشن کرنے کے لیے >> آپریٹر فراہم کرتا ہے۔ آپریٹر >> کو "سٹریم انسرشن (Stream Insertion)" یا "put to" آپریٹر کہا جاتا ہے۔ اس کا استعمال اس کے دائیں جانب متغیرات کے تاروں یا قدروں کو اس کے بائیں جانب آبجیکٹ پر بھیجنے کے لیے کیا جاتا ہے۔ >> ایک بائنری آپریٹر ہے۔

پہلا آپرینڈ پہلے سے طے شدہ شناخت کنندہ cout ہے (جسے C-Out کہا جاتا ہے) جو مانیٹر کو معیاری آؤٹ پٹ آبجیکٹ کے طور پر شناخت کرتا ہے۔ دوسرا آپرینڈ ایک کانس ٹینٹ(constant)، ویریبل (variable)یا اکسپریشن (expression) ہوسکتا ہے۔



تصویر 9.5 cout کے کام کرنے کا عمل

ایک وقت میں ایک سے زیادہ قدر بھیجنے کے لیے، >> آپریٹر کو ہر کانس ٹینٹ (constant)، ویریبل (variable)یا اکسپریشن (expression) کے لیے استعمال کیا جانا چاہیے۔ اس کو **آپریٹر کا کیسکیڈنگ (cascading of operator)** کہا جاتا ہے۔

مثال:

cout << "Welcome";	پہلے سے طے شدہ آبجیکٹ cout دی گئی سٹرنگ " welcome " کو سکرین پر بھیجتا ہے۔
cout << "The sum = " << sum;	سب سے پہلے، cout سٹرنگ "The Sum = " اسکرین پر بھیجتا ہے اور پھر متغیر رقم کی قدر بھیجتا ہے۔ عام طور پر، cout ہر چیز کو دوہرے اقتباسات یا سنگل اقتباسات یعنی سٹرنگ یا کریکٹر کانس ٹینٹس کے اندر بھیجتا ہے سوائے غیر گرافک حروف کے۔
cout <<"\n The Area: " <<3.14*r*r;	پہلے cout، \n کے علاوہ ڈبل اقتباسات میں بیان کردہ ہر چیز کو اسکرین پر بھیجتا ہے اور پھر یہ ایکسپریشن 3.14*r*r کا جائزہ لیتا ہے اور نتیجہ اسکرین پر بھیجتا ہے۔ \n - ایک نئی لائن کو شامل کرنے کے لیے ایک غیر گرافک کریکٹر کانس ٹینٹ ہے۔
cout << a + b ;	Cout، a اور b کا مجموعہ آؤٹ پٹ کنسول (مانیٹر) کو بھیجتا ہے۔

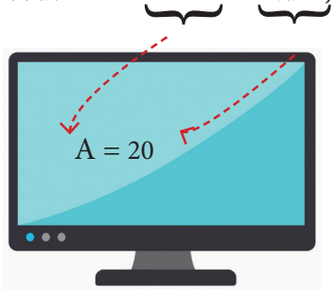
9.4.3 آئی / او آپریٹرس کی کیسکیڈنگ (Cascading of I/O operators)

ان پٹ اور آؤٹ پٹ آپریٹرز کا ایک سے زیادہ استعمال جیسا کہ >> اور << ایک ہی بیان میں I/O آپریٹرز کی کیسکیڈنگ کے نام سے جانا جاتا ہے۔

کیسکیڈنگ کاؤٹ (cascading cout) :

```
int Num=20;
cout << "A=" << Num;
```

تصویر 9.6 کا استعمال Cascading cout بیان کے کام کو سمجھنے کے لیے کیا جاتا ہے۔



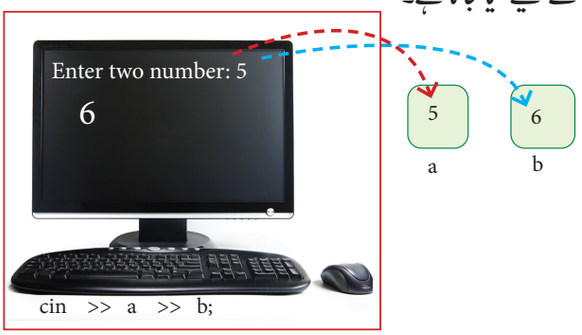
تصویر 9.6 کیسکیڈنگ cout

11th_Comp Science_UM_Ch 9.indd 124 18-04-2023 15:04:17

کیسکیڈنگ cin - مثال:

```
cout << "Enter two number: ";
cin >> a >> b;
```

تصویر 9.7 کا استعمال کیسکیڈنگ cin اسٹیٹمنٹ کے کام کو سمجھنے کے لیے کیا جاتا ہے۔



تصویر 9.7 کیسکیڈنگ cin

9.5 سیمپل پروگرام - C++ پروگرام کی پہلی جھلک
(Sample Program – A first look at C++ Program)

آئیے ہم اپنا پہلا C++ پروگرام شروع کرتے ہیں جو اسکرین پر " Welcome to Programming in ++C" سٹرنگ پرنٹ کرتا ہے۔

Sample Prog 9_5.cpp

```
// C++ program to print a string
#include <iostream>
using namespace std;
int main()
{
    cout << "Welcome to Programming in C++";
    return 0;
}
```

مندرجہ بالا پروگرام مندرجہ ذیل آؤٹ پٹ تیار کرتا ہے:

C++ میں پروگرامنگ میں خوش آمدی د

(Welcome to Programming in C++)

یہ بہت آسان C++ پروگرام ہے جس میں بنیادی عناصر شامل ہیں جو ہر C++ پروگرام میں ہوتے ہیں۔ آئیے ان عناصر پر ایک نظر ڈالتے ہیں:

1	**// C++ program to print a string**
یہ ایک تبصرہ بیان ہے۔ کوئی بھی بیان جو // سے شروع ہوتا ہے تبصرے (comments) سمجھا جاتا ہے۔ کمپائلر پروگرام کے حصے کے طور پر کوئی تبصرہ نہیں کرتا ہے اور یہ صرف نظر انداز کرتا ہے۔ اگر ہمیں تبصرے کی ایک سے زیادہ سطریں لکھنی ہوں تو ہم استعمال کر سکتے ہیں /* ..... */۔	
2	**# include <iostream>**
عام طور پر تمام C++ پروگرام # (ہیش/پاؤنڈ) سے شروع ہونے والے بیانات کے ساتھ شروع ہوتے ہیں۔ علامت # پری پروسیسر کے لیے ایک ہدایت ہے۔ اس کا مطلب ہے کہ ان بیانات پر تالیف کا عمل شروع ہونے سے پہلے عمل کیا جاتا ہے۔ **#include <iostream> بیان کمپائلر کے پری پروسیسر کو پروگرام میں ہیڈر فائل "iostream" کو شامل کرنے کے لیے کہتا ہے۔** ہیڈر فائل iostream کو ہر C++ پروگرام میں شامل ہونا چاہیے تاکہ ان پٹ/آؤٹ پٹ افعال کو لاگو کیا جا سکے۔ آسان الفاظ میں، iostream ہیڈر فائل میں اس کے ممبر آبجیکٹ cin اور cout کی تعریف ہوتی ہے۔ اگر آپ اپنے پروگرام میں iostream شامل کرنے میں ناکام رہتے ہیں، تو cin اور cout پر ایک غلط پیغام آئے گا۔ اور ہم کوئی ان پٹ حاصل کرنے یا کوئی آؤٹ پٹ بھیجنے کے قابل نہیں ہوں گے۔	
3	**using namespace std;**
لائن using namespace std; مرتب کرنے والے کو معیاری namespace استعمال کرنے کو کہتا ہے۔ Namespace کلاس، آبجیکٹ اور متغیرات کے لیے استعمال ہونے والے شناخت کنندگان کو جمع کرتی ہے۔ Namespaces بڑے منصوبوں میں نام کے تنازعات کو روکنے کا ایک طریقہ فراہم کرتی ہیں۔ یہ ایک نیا تصور ہے جسے C++ ANSI معیارات کمیٹی نے متعارف کرایا ہے۔	

11th_Comp Science_UM_Ch 9.indd 125 18-04-2023 15:04:17

4 int main ( )

C++ پروگرام افعال کا مجموعہ ہے۔ ہر C++ پروگرام کا ایک مین فنکشن (main function) ہونا ضروری ہے۔ () main فنکشن نقطہ آغاز ہے جہاں تمام C++ پروگرام اپنے عمل کو شروع کرتے ہیں۔ لہذا، قابل عمل بیانات () main فنکشن کے اندر ہونے چاہئیں۔

```
5  {
6      cout << "Welcome to Programming in C++";
7      return 0;
8  }
```

منحنی قوسین کے درمیان بیانات (لائن نمبر 5 سے 8) قابل عمل بیانات ہیں۔ یہ اصل میں کوڈ کے بلاک کے طور پر کہا جاتا ہے. لائن 6 میں، cout صرف "Welcome to Programming in C ++" سٹرنگ کو اسکرین پر بھیجتا ہے۔ جیسا کہ ہم پہلے ہی بحث کر چکے ہیں، ہر قابل عمل بیان کو سیمی کولن کے ساتھ ختم ہونا چاہیے۔ لائن 7 میں، واپسی ایک کلیدی لفظ ہے جو اس قدر کو واپس کرنے کے لیے استعمال ہوتا ہے جو آپ نے کسی فنکشن میں بیان کیا ہے۔ اس صورت میں، یہ 0 کو () main فنکشن میں واپس کر دے گا۔

9.6 C++ پروگرام کا عمل :
(Execution of C++ program)

C++ پروگرام بنانے اور اس پر عمل کرنے کے لیے، چار اہم مراحل پر عمل کرنا ضروری ہے۔

(1) سورس کوڈ بنانا

تخلیق کرنے میں C++ کمپائلر کے اصولوں کے مطابق درست C++ کوڈ کو ٹائپ کرنا اور اس میں ترمیم کرنا شامل ہے۔

(2) extension.cpp کے ساتھ سورس کوڈ کو محفوظ کرنا

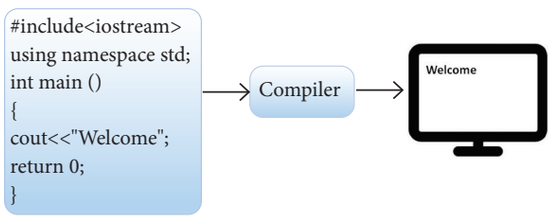
ٹائپ کرنے کے بعد سورس کوڈ کو extension.cpp کے ساتھ محفوظ کرنا چاہیے۔

(3) تالیف (compilation)

یہ ایک پروگرام کی تعمیر میں ایک اہم قدم ہے۔ تالیف میں، کمپائلر لائبریری فائلوں کو سورس کوڈ سے جوڑتا ہے اور کوڈ کی ہر لائن کی تصدیق کرتا ہے۔ اگر کوئی غلطی یا غلطی پائی جاتی ہے تو یہ غلط پیغام دکھائے گا۔ اگر کوئی خرابی نہیں ہے، تو یہ ماخذ کوڈ کو مشین پڑھنے کے قابل آبجیکٹ فائل میں extension.obj کے ساتھ ترجمہ کرتا ہے۔

(4) عمل درآمد (Execution)

یہ C++ پروگرام کا آخری مرحلہ ہے۔ اس مرحلے میں، آبجیکٹ فائل extension.exe کے ساتھ ایک قابل عمل فائل بن جاتی ہے۔ ایک بار جب پروگرام ایک قابل عمل فائل بن جاتا ہے ، جو پروگرام کا ایک آزاد وجود ہے۔ اس کا مطلب ہے، آپ کسی کمپائلر(compiler) یا IDE کی مدد کے بغیر اپنی ایپلی کیشن (Application)چلا سکتے ہیں۔



تصویر 9.8 عمل درآمد

9.7 C++ ترقیاتی ماحول
(C++ Development Environment)

C++ کے لیے بہت سے IDE پروگرام دستیاب ہیں۔ IDE، C++ پروگرام بنانا، مرتب کرنا اور اس پر عمل درآمد کرنا آسان بناتا ہے۔ زیادہ تر IDEs اوپن سورس ایپلی کیشنز ہیں (یعنی) وہ مفت دستیاب ہیں۔

9.7.1 IDE کے ساتھ واقف C++ کمپائلرس (مرتب کرنے والے)

ٹیبل 9.4 اوپن سورس کمپائلرس

کمپائلر (Compiler)	دستیابی (Availability)
Dev C++	اوپن سورس
Geany	اوپن سورس
Code::blocks	اوپن سورس
Code Lite	اوپن سورس

11th_Comp Science_UM_Ch 9.indd 126 18-04-2023 15:04:18

Net Beans	اوپن سورس
Digital Mars	اوپن سورس
Sky IDE	اوپن سورس
Eclipse	اوپن سورس

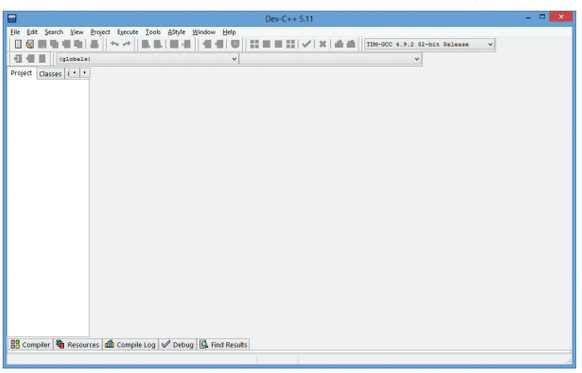
9.7.2 Dev C++ کے ساتھ کام کرنا

درجنوں IDEs میں سے، ہم C++ پروگرام بنانے کے لیے Dev C++ کمپائلر لیتے ہیں۔ اس کتاب کے پروگرامنگ کی تکنیک اور تصویری پروگرام "Dev ++C" کمپائلر پر مبنی ہیں۔

Dev C++ ایک اوپن سورس، کراس پلیٹ فارم (cross platform) ہے (لینکس کے لیے الفا ورژن دستیاب ہے)، مکمل خصوصیات والا انٹیگریٹڈ ڈیولپمنٹ انوائرمنٹ (IDE – Integrated Development Environment) ہے جسے C اور C++ میں پروگرامنگ کے لیے GNU جنرل پبلک لائسنس (General Public License) کے ساتھ تقسیم کیا گیا ہے۔ یہ ڈیلفی (Delphi) میں لکھا ہے۔

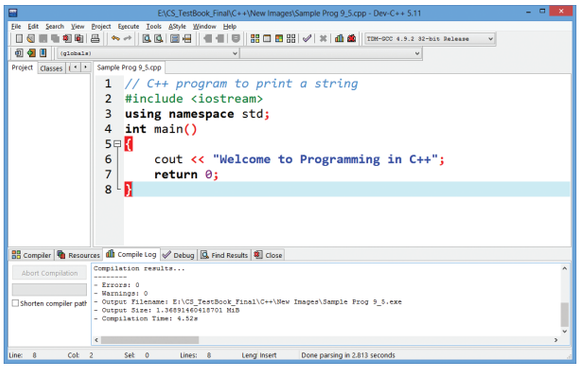
اسے **http://www.bloodshed.net/dev/devcpp.html** سے ڈاؤن لوڈ کیا جا سکتا ہے۔

1. انسٹالیشن (installation) کے بعد Dev C++ آئیکن ڈیسک ٹاپ پر دستیاب ہے۔ IDE کھولنے کے لیے ڈبل کلک کریں۔ Dev C++ IDE ظاہر ہوتا ہے جیسا کہ ذیل میں دیا گیا ہے۔



تصویر 9.9 Dev C ++ کا آغاز ونڈو

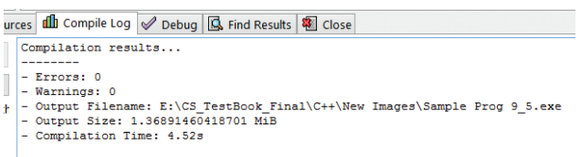
2. سورس فائل بنانے کے لیے، فائل ← (file) نیو ← (New) سورس فائل (source file) یا Ctrl + N دباکر منتخب کریں۔
3. ظاہر ہونے والی اسکرین میں، اپنا C++ پروگرام ٹائپ کریں، فائل ← (file) سیو (save) یا Ctrl + S کو دباکر فائل کو محفوظ کریں۔ یہ آپ کے سورس کوڈ فائل کے آخر میں .cpp کو بطور ڈیفالٹ شامل کرے گا۔ .cpp کو آپ کی فائل کے نام کے ساتھ ٹائپ کرنے کی ضرورت نہیں۔



تصویر 9.10 ایک پروگرام کے ساتھ Dev C++ IDE

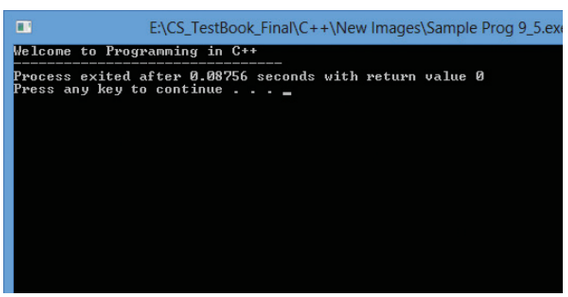
4. **محفوظ کرنے کرنے کے بعد اکزیکیوٹ ← (Execute) کمپائل (Compile) پر کلک کریں اور رن (Run) یا F11 کلید (key) دبائیں۔**

اگر آپ کے پروگرام میں کوئی خامی ہے تو، یہ کمپائل لاگ (compile log) کے تحت غلطیوں کو دکھاتا ہے۔ اگر آپ کا پروگرام بغیر کسی خامی کے ہے تو ڈسپلے اس طرح ظاہر ہو گا۔



تصویر 9.11 Dev C++ کمپائل لاگ

5. **کامیاب تالیف کے بعد، آؤٹ پٹ، آؤٹ پٹ کنسول (output console) میں ظاہر ہو گا، جیسا کہ درج ذیل میں ہے۔**



تصویر 9.12 Dev C++ آؤٹ پٹ ونڈو

11th_Comp Science_UM_Ch 9.indd 127 18-04-2023 15:04:18

9.8 غلطیوں کی اقسام
(Types of errors)

غلطیوں کی کچھ عام اقسام ذیل میں دی گئی ہیں:

غلطی کی قسم	تفصیل
سنٹیکس کی غلطی (Syntax Error)	• سنٹیکس ایک پروگرام کی تعمیر کے لیے گرائمر کے اصولوں کا ایک مجموعہ ہے۔ ہر پروگرامنگ زبان میں سورس کوڈ بنانے کے لیے منفرد اصول ہوتے ہیں۔ • سنٹیکس کی غلطیاں اس وقت ہوتی ہیں جب C++ کے گرامر کے اصولوں کی خلاف ورزی کی جاتی ہے۔ • مثال کے طور پر: اگر آپ مندرجہ ذیل ٹائپ کرتے ہیں تو C++ ایک غلطی ظاہر کرے گا۔ cout << "Welcome to Programming in C++" • C++ کے گرامر کے اصولوں کے مطابق، ہر قابل عمل بیان کو سیمی کولن کے ساتھ ختم ہونا چاہیے۔ لیکن، یہ بیان سیمی کولن کے ساتھ ختم نہیں ہوتا ہے۔
سیمنٹک غلطی (Semantic Error)	پروگرام گرائمری طور پر درست ہونے کے باوجود پروگرام کا متوقع نتیجہ نہیں نکلا ہے۔ یہ متغیر / آپریٹر / عمل درآمد کے آرڈر وغیرہ کے غلط استعمال سے ہو سکتا ہے۔ اس کا مطلب ہے، پروگرام گرامر کے لحاظ سے درست ہے، لیکن اس میں کچھ منطقی غلطی ہے۔ لہذا، سیمنٹک کی غلطی کو "منطق کی غلطی (Logic Error)" بھی کہا جاتا ہے
رن ٹائم غلطی (Run – time Error)	• ایک پروگرام کے عمل کے دوران ایک رن ٹائم غلطی واقع ہوتی ہے۔ یہ کچھ غیر قانونی آپریشن کی وجہ سے ہوتا ہے۔ • مثال کے طور پر، اگر کوئی پروگرام کسی ایسی فائل کو کھولنے کی کوشش کرتا ہے جو موجود نہیں ہے، تو اس کے نتیجے میں رن ٹائم غلطی ہوتی ہے۔

یاد رکھنے کے لیے نکات

- C++ کو Bjarne Stroustrup نے 1979 کے دوران AT&T Bell Labs میں تیار کیا تھا۔
- کریکٹر سیٹ ان حروف کا مجموعہ ہے جنہیں C++ پروگرام لکھنے کی اجازت ہے۔
- انفرادی عناصر کو اجتماعی طور پر لغوی اکائیاں یا لغوی عناصر یا ٹوکن کہا جاتا ہے۔
- کلیدی الفاظ وہ محفوظ الفاظ ہیں جو C++ مرتب کرنے والے کو مخصوص معنی دیتے ہیں۔
- شناخت کنندگان صارف کے بیان کردہ نام ہیں جو C++ پروگرام کے مختلف حصوں کو دیئے گئے ہیں۔ متغیرات، افعال، اررِیس(arrays)، کلاسز وغیرہ،
- لٹریلس (Literals) ڈیٹا آئٹمز ہیں جن کی قدریں کسی پروگرام کے عمل کے دوران تبدیل نہیں ہوتی ہیں۔ اس لیے لٹریلس کو Constants کہا جاتا ہے۔
- C++ میں مختلف قسم کے لٹریلس استعمال ہوتے ہیں (انٹیجر، فلوٹ، کریکٹر، سٹرنگ)
- وہ علامتیں جو کچھ ریاضیاتی، منطقی کارروائیوں کو کرنے کے لیے استعمال ہوتی ہیں انہیں "آپریٹرز (Operators)" کہا جاتا ہے۔
- اوقاف (punctuators) علامتیں ہیں، جو C++ پروگراموں کی تعمیر میں حد بندی کے طور پر استعمال ہوتے ہیں۔ انہیں "سیپریٹرس (Separators)" بھی کہا جاتا ہے۔
- ایکسٹریکشن آپریٹر (extraction operator) (>>) اور انسرشن آپریٹر (insertion operator) (<<) کو C++ میں ان پٹ حاصل کرنے اور آؤٹ پٹ بھیجنے کے لیے استعمال کیا جاتا ہے۔

11th_Comp Science_UM_Ch 9.indd 128 18-04-2023 15:04:18



عملی مشق

- درج ذیل C++ پروگراموں کو Dev C ++ IDE میں ٹائپ کریں اور عملی مشق کریں۔ اگر کمپائلر کوئی غلطی دکھاتا ہے تو اسے درست کرنے کی کوشش کریں اور اس وقت تک بار بار عمل کریں جب تک کہ آپ کو متوقع نتیجہ نہ ملے۔

1. C++ پروگرام تین مضامین کے کل نمبر تلاش کرنے کے لیے:

```
#include <iostream>
using namespace std;
int main()
{
        int m1, m2, m3, sum;
        cout << "\n Enter Mark 1: ";
        cin >> m1;
        cout << "\n Enter Mark 2: ";
        cin >> m2;
        cout << "\n Enter Mark 3: ";
        cin >> m3;
        sum = m1 + m2 + m3;
        cout << "\n The sum = " << sum;
}
```

- تمام دیے گئے نمبروں کی اوسط حاصل کرنے کے لیے مذکورہ کوڈ میں تبدیلیاں کریں۔

2. دائرے کا رقبہ معلوم کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int radius;
        float area;
        cout << "\n Enter Radius: ";
        cin >> radius;
        area = 3.14 * radius * radius;
        cout << "\n The area of circle = " << area;
}
```

3. مندرجہ ذیل پروگرام میں غلطیوں کی نشاندہی کریں۔

```
Using namespace std;
int main( )
{
cout << “Enter a value ”;
cin << num1 >> num2
num+num2=sum;
cout >> “\n The Sum= ” >> sum;
```


11th_Comp Science_UM_Ch 9.indd 129 18-04-2023 15:04:18

4. درج ذیل پروگرام میں غلطی کی قسم کی نشاندہی کریں:

```
#include <iostream>
using namespace std;
int main()
{
        int h=10; w=12;
        cout << "Area of rectangle " << h+w;
}
```





تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں:

1. کس نے C++ ایجاد کیا؟
   (a) چارلس بے بیج (Charles Babbage) (b) بیارنے سٹروسٹرپ (Bjarne Stroustrup)
   (c) بل گیٹس (Bill Gates) (d) سندر پی چئے (Sundar Pichai)

2. C++ کو دیا جانے والا اصل نام کیا تھا؟
   CPP (a) Advanced C (b) C with Classes (c) Class with C (d)

3. کس نے C++ کا اختراع کیا؟
   (a) رک مے سٹی (Rick Mascitti) (b) رک بیارنے (Rick Bjarne)
   (c) بل گیٹس (Bill Gates) (d) ڈینس رچی (Dennis Ritchie)

4. پروگرام میں سب سے چھوٹی انفرادی اکائی ہے:
   Program (a) Algorithm (b) Flowchart (c) Tokens (d)

5. درج ذیل میں سے کون سا آپریٹر C++ میں ایکسٹرکشن آپریٹر ہے؟
   >> (a) << (b) <> (c) ^^ (d)

6. درج ذیل میں سے کون سا بیان درست نہیں ہے؟
   (a) کلیدی الفاظ وہ محفوظ الفاظ ہیں جو C++ مرتب کرنے والے کو مخصوص معنی دیتے ہیں۔
   (b) محفوظ الفاظ یا مطلوبہ الفاظ کو شناخت کنندہ نام کے طور پر استعمال کیا جا سکتا ہے۔
   (c) ایک عدد مستقل میں کم از کم ایک ہندسہ ہونا چاہیے بغیر اعشاریہ کے۔
   (d) حقیقی مستقل کی ایکسپونٹ(exponent) شکل دو حصوں پر مشتمل ہوتی ہے۔

7. درج ذیل میں سے کون سا درست سٹرنگ لٹریل ہے؟
   'A' (a) 'Welcome' (b) 1232 (c) "1232" (d)

8. اعلیٰ سطحی زبان میں لکھے گئے پروگرام کو کہا جاتا ہے۔
   Object code (a) Source code (b) Executable code (c) All the above (d)

9. فرض کریں ; a=5, b=6 تو a&b کا نتیجہ کیا ہو گا؟
   4 (a) 5 (b) 1 (c) 0 (d)

10. درج ذیل میں سے کس کو کمپائل ٹائم آپریٹرز کہا جاتا ہے؟
    sizeof (a) pointer (b) virtual (c) this (d)

11th_Comp Science_UM_Ch 9.indd 130 18-04-2023 15:04:18

سیکشن - B

بہت مختصر جوابات دیں

1. ٹوکن سے کیا مراد ہے؟ C++ میں دستیاب ٹوکن کا نام بتائیں۔
2. کلیدی الفاظ کیا ہیں؟ کیا مطلوبہ الفاظ کو شناخت کنندہ کے طور پر استعمال کیا جا سکتا ہے؟
3. مندرجہ ذیل میں کانسٹینٹ کون سی قسم کے ہیں؟

   39 (i) 032 (ii) 0XCAFE (iii) 04.14 (iv)
4. درج ذیل ریئل کانسٹینٹ کو ایکسپونٹ فارم میں لکھیں:

   23.197 (i) 7.214 (ii) 0.00005 (iii) 0.319 (iv)
5. فرض کریں ; n=10 تو n ++ اور n -- کا نتیجہ کیا ہو گا؟
6. درج ذیل کو جوڑیں

B	A
(1) Tokens	(a) Modulus
(2) Remainder of a division	(b) Separators
(3) Punctuators	(c) Stream extraction
(4) get from	(d) Lexical Units

سیکشن - C

مختصر جوابات

1. مطلوبہ الفاظ اور شناخت کنندگان کے درمیان فرق بیان کریں؟
2. کیا C++ کیس حساس ہے؟ "حساس کیس(case sensitive)" کی اصطلاح سے کیا مراد ہے؟
3. "=" اور "==" میں فرق کریں۔
4. ہیڈر فائل کا استعمال کیا ہے؟
5. مین فنکشن خاص کیوں ہے؟

سیکشن - D

تفصیل سے بتائیں

1. C++ میں استعمال ہونے والے بائنری آپریٹرز کے بارے میں لکھیں۔
2. غلطیوں کی اقسام کیا ہیں؟

حوالہ جات:

(1) Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.

(2) The Complete Reference C++ (Forth Edition), Herbert Schildt.Mc.Graw Hills.

(3) Computer Science with C++ (A text book of CBSE XI and XII), Sumita Arora, Dhanpat Rai & Co.

11th_Comp Science_UM_Ch 9.indd 131 18-04-2023 15:04:18

ڈیٹا کی اقسام، متغیرات اور اظہارات
(Data Types, Variables And Expressions)

9.10 تعارف

ہر پروگرامنگ زبان میں دو بنیادی عناصر ہوتے ہیں، یعنی ڈیٹا کی اقسام اور متغیرات۔ وہ انتہائی ابتدائی پروگراموں کو بھی لکھنے کے لیے بہت ضروری عناصر ہیں۔ C++ ڈیٹا آئٹمز کو سنبھالنے کے لیے ڈیٹا کی اقسام کا ایک پیش وضاحتی سیٹ فراہم کرتا ہے۔ اس طرح کے ڈیٹا کی اقسام کو بنیادی (fundamental) یا بلٹ ان ڈیٹا کی اقسام (Built-in data types) کے نام سے جانا جاتا ہے۔ بلٹ ان ڈیٹا کی اقسام کے علاوہ، ایک پروگرامر اپنی ڈیٹا کی قسمیں بھی بنا سکتا ہے جسے یوزر ڈیفائنڈ ڈیٹا ٹائپ (User-defined data types) کہتے ہیں۔ اس باب میں، ہم بلٹ ان ڈیٹا کی اقسام کے بارے میں جاننے جا رہے ہیں۔

9.11 ڈیٹا کی اقسام کا تصور

آئیے درج ذیل مثال کو دیکھتے ہیں،

نام = رام

عمر = 15

اوسط_مارک = 85.6

مندرجہ بالا مثال میں، نام، عمر، اوسط_مارک وہ فیلڈز ہیں جو بالترتیب رام، 15 اور 85.6 جیسی اقدار رکھتے ہیں۔

ایک پروگرامنگ زبان میں، فیلڈز (fields) کو متغیر (Variables) کہا جاتا ہے اور اقدار (values) کو ڈیٹا (Data) کہا جاتا ہے۔

مندرجہ بالا مثال میں ہر ڈیٹا آئٹم مختلف نظر آتا ہے۔ یعنی "رام" حروف تہجی کی ایک ترتیب ہے اور دیگر دو ڈیٹا آئٹمز نمبر ہیں۔ پہلی قدر ایک مکمل نمبر ہے اور دوسری ایک جزوی نمبر ہے۔ حقیقی دنیا کے منظرناموں میں، ہم اپنی روزمرہ کی زندگی میں بہت سے مختلف قسم کے ڈیٹا کو سنبھالتے ہیں۔ ڈیٹا آئٹم کی نوعیت یا قسم مختلف ہوتی ہے، مثال کے طور پر فاصلہ (آپ کے گھر سے اسکول تک)، ٹکٹ کا کرایہ، قلم کی قیمت، نشانات، درجہ حرارت وغیرہ،

C++ پروگرامنگ میں، کسی بھی ڈیٹا کو سنبھالنے سے پہلے، اسے زبان کے مرتب کرنے والے کو واضح طور پر بتانا چاہیے،اس حوالے سے کہ ڈیٹا کی قسموں کے کچھ پہلے سے طے شدہ سیٹ کے ساتھ یہ کس قسم کا ڈیٹا ہے ۔

9.12 C++ ڈیٹا کی اقسام

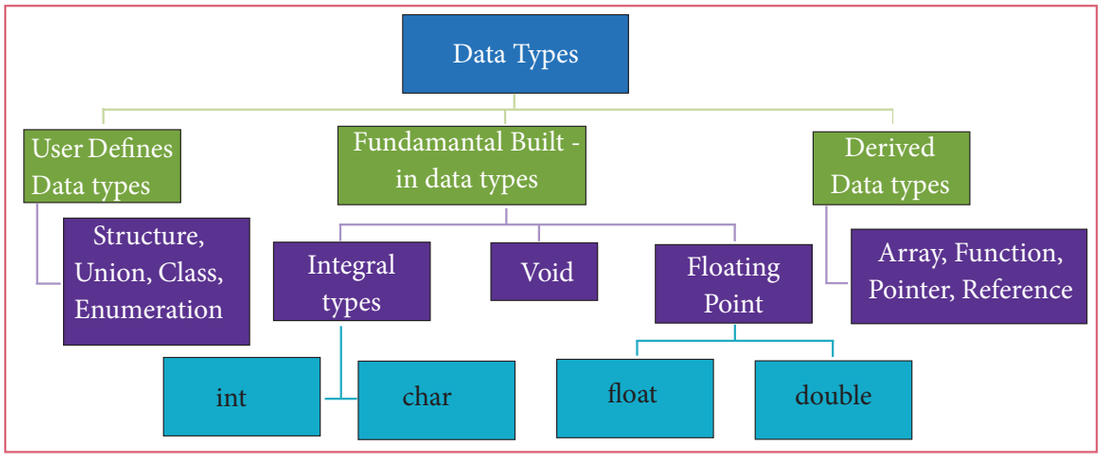
C++ میں، ڈیٹا کی اقسام کو تین اہم زمروں میں درجہ بندی کیا گیا ہے۔

(1) بنیادی ڈیٹا کی اقسام (fundamental data types)

(2) صارف کی وضاحت کردہ ڈیٹا کی اقسام (User-defined data types) اور

(3) اخذ کردہ ڈیٹا کی اقسام (Derived data types)۔

11th_Comp Science_UM_Ch 9.indd 132 18-04-2023 15:04:18



تصویر 9.13 C++ میں ڈیٹا کی اقسام

اس باب میں، ہم صرف بنیادی ڈیٹا کی اقسام کے بارے میں جاننے جا رہے ہیں۔

ڈیٹا کی اقسام کے کام کو سمجھنے کے لیے، ہمیں متغیرات کے بارے میں جاننے کی ضرورت ہے۔ متغیرات مخصوص ڈیٹا کی اقسام کی قدروں کو رکھنے کے لیے نامزد کردہ میموری کے مقامات ہیں۔ C++ میں، متغیرات کو اصل میں استعمال کرنے سے پہلے ان کے ڈیٹا کی اقسام کے ساتھ واضح طور پر اعلان کیا جانا چاہیے۔

متغیر کا اعلان کرنے کا سنٹیکس (Syntax):

<data type> <variable name>;

مثال:

int num1;

ایک سے زیادہ متغیرات کا اعلان کرنے کے لیے جو کہ ایک ہی ڈیٹا کی قسم کے ہیں ایک ہی سٹیٹمنٹ کا استعمال کرتے ہوئے، اسے کوما کا استعمال کرتے ہوئے متغیرات کو الگ کر کے اعلان کیا جا سکتا ہے۔

مثال:

int num1, num2, sum;

مثال کے طور پر، اپنے کمپیوٹر سائنس کے مارکس کو ذخیرہ کرنے کے لیے پہلے آپ کو ایک متغیر کا اعلان کرنا چاہیے تاکہ آپ کے نمبرز کو مناسب ڈیٹا ٹائپ کے ساتھ رکھا جائے۔ ڈیٹا کی مناسب قسم کا انتخاب کرنے کے لیے مزید علم اور تجربے کی ضرورت ہوتی ہے۔ عام طور پر، نشانات کو پورے نمبر کے طور پر دکھایا جاتا ہے۔ اس طرح، کمپیوٹر سائنس کے نشانات کو ذخیرہ کرنے کے لیے متغیر عددی ڈیٹا کی قسم کا ہونا چاہیے۔

مثال:

int comp_science_marks;

اب **comp_science_marks** نام کا ایک متغیر آپ کے نشانات کو ذخیرہ کرنے کے لیے تیار ہے۔

ہم بعد میں اس باب میں متغیرات کے بارے میں مزید جانیں گے۔

9.12.1 بنیادی ڈیٹا کی اقسام کا تعارف:

بنیادی (اٹومک-atomic) ڈیٹا کی قسمیں پہلے سے طے شدہ ڈیٹا کی قسمیں ہیں جو C++ کے ساتھ دستیاب ہیں۔ C++ میں ڈیٹا کی پانچ بنیادی اقسام ہیں: **char, in, float, double** اور **void**۔ دراصل، یہ ڈیٹا کی اقسام کی وضاحت کے لیے مطلوبہ الفاظ ہیں۔

(1) int ڈیٹا کی قسم: (int data type)

انٹیجر ڈیٹا کی قسم صرف انٹیجر نمبروں کو قبول کرتی ہے اور واپس کرتی ہے۔ اگر کسی متغیر کو int قرار دیا جاتا ہے، تو C++ کمپائلر اس میں صرف عددی اقدار کو ذخیرہ کرنے کی اجازت دیتا ہے۔ اگر آپ کسی int قسم کے متغیر میں فریکشنل ویلیو کو اسٹور کرنے کی کوشش کرتے ہیں تو یہ صرف انٹیجر والے حصے کو قبول کرے گا اور فرکشنل حصے کو نظر انداز کر دیا جائے گا۔

11th_Comp Science_UM_Ch 9.indd 133 18-04-2023 15:04:18

مثال کے طور پر
int num=12;

num1 متغیر کو عددی اقسام کے طور پر قرار دیا گیا ہے۔لہذا، یہ انٹیجر ویلیو کو محفوظ کر سکتا ہے۔

(2) چار ڈیٹا کی قسم: (Char data type)

کریکٹر ڈیٹا کی قسم تمام درست ASCII حروف کو قبول اور واپس کرتی ہے۔ کریکٹر ڈیٹا ٹائپ کو اکثر انٹیجر ٹائپ کہا جاتا ہے، کیونکہ تمام حروف کو ان کے متعلقہ ASCII کوڈز کے ذریعے میموری میں دکھایا جاتا ہے۔ اگر ایک متغیر کو چار کے طور پر قرار دیا جاتا ہے، تو C++ کسی حرف یا عددی قدر کو ذخیرہ کرنے کی اجازت دیتا ہے۔

مثال 1

```
char ch='A';
cout<<ch ;
```

مندرجہ بالا کوڈ میں، ch کو ایک کریکٹر رکھنے کے لیے چار قسم (Char type)کا متغیر قرار دیا گیا ہے۔ یہ کریکٹر A دکھاتا ہے۔

مثال 2:-

```
char ch='A';
ch=ch+1;
cout<<ch;
```

مندرجہ بالا بیانات میں ، ch کی قدر میں 1 کا اضافہ ہوا ہے اور نئی قدر کو واپس اسی متغیر ch میں محفوظ کیا گیا ہے۔ (یاد رکھیں کہ، ریاضی کی کارروائیاں صرف اعداد پر کی جاتی ہیں حروف تہجی کے ساتھ نہیں) لہذا یہ B دکھاتا ہے۔

ایک اور پروگرام یہ بتاتا ہے کہ کس طرح int اور char ڈیٹا کی اقسام ایک ساتھ کام کر رہی ہیں۔

مثال 9.3: ASCII قدر حاصل کرنے اور متعلقہ کریکٹر کو ظاہر کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
        int n;
        char ch;
        cout << "\n Enter an ASCII code (0 to 255): ";
        cin >> n;
        ch = n;
        cout << "\n Equivalent Character: " << ch;
}
```

The output

```
Enter an ASCII code (0 to 255): 100
Equivalent Character: d
```

مندرجہ بالا پروگرام میں، متغیر n کو int قسم اور ایک اور **متغیر ch** کو char type کے طور پر قرار دیا گیا ہے۔ عملدرآمد کے دوران، پروگرام صارف کو ASCII قدر درج کرنے کا اشارہ کرتا ہے۔ اگر صارف ASCII قدر بطور عدد داخل کرتا ہے، تو اسے **متغیر n** میں محفوظ کیا جائے گا۔ بیان میں ;ch = n، n کی قدر ch میں تفویض کی گئی ہے۔ یاد رکھیں کہ، **ch** ایک char type کا متغیر ہے۔

11th_Comp Science_UM_Ch 9.indd 134 18-04-2023 15:04:19

مثال کے طور پر، اگر کوئی صارف 100 بطور ان پٹ داخل کرتا ہے۔ ابتدائی طور پر، 100 متغیر n میں محفوظ ہوتا ہے۔ اگلے بیان میں، n یعنی 100 کی قدر ch کو تفویض کی گئی ہے۔ چونکہ، ch ایک char type ہے؛ یہ متعلقہ ASCII کریکٹر کو آؤٹ پٹ کے طور پر دکھاتا ہے۔ (100 کے لیے مساوی ASCII کریکٹر d ہے)۔

(3) فلوٹ ڈیٹا کی قسم: (float data type)

اگر ایک متغیر کو فلوٹ کے طور پر قرار دیا جاتا ہے، تو تمام اقدار کو فلوٹنگ پوائنٹ ویلیو کے طور پر محفوظ کیا جائے گا۔

فلوٹ ڈیٹا کی اقسام کو استعمال کرنے کے دو فائدے ہیں۔

(1) وہ عدد کے درمیان اقدار کی نمائندگی کر سکتے ہیں۔

(2) وہ قدروں کی ایک بہت بڑی حد کی نمائندگی کر سکتے ہیں۔

اسی وقت، فلوٹنگ پوائنٹ آپریشنز انٹیجر قسم کے مقابلے میں زیادہ وقت لیتا ہے، یعنی فلوٹنگ پوائنٹ آپریشن انٹیجر آپریشنز سے سست ہوتے ہیں۔ یہ فلوٹنگ پوائنٹ آپریشن کا ایک نقصان ہے۔

مثال کے طور پر

```
float num=13.4;
```

مندرجہ بالا مثال میں، num متغیر کو float type قرار دیا گیا ہے۔

(4) ڈبل ڈیٹا کی قسم: (Double data type)

یہ دوہری درستگی کے فلوٹنگ پوائنٹ نمبرز کے لیے ہے۔ (درستگی سے مراد اعشاریہ کے بعد اہم اعداد)۔ ڈبل فلوٹنگ پوائنٹ نمبرز کو سنبھالنے کے لیے بھی استعمال کیا جاتا ہے۔ لیکن، یہ قسم فلوٹ قسم سے دگنی جگہ پر قبضہ کرتی ہے۔ اس کا مطلب ہے، فلوٹ کی قسم کے مقابلے میں زیادہ حصوں کو ڈبل میں ایڈجسٹ کیا جا سکتا ہے۔ ڈبل قسم فلوٹ سے بڑا اور سست ہے۔ ڈبل فلوٹ ڈیٹا ٹائپ کی طرح استعمال ہوتا ہے۔

(5) وائڈ ڈیٹا کی قسم: (void data type)

وائڈ کے لغوی معنی 'خالی جگہ (empty space)' کے ہیں۔ یہاں، C++ میں، void ڈیٹا کی قسم اقدار کے خالی سیٹ کی وضاحت کرتی ہے۔ یہ ان افعال کے لیے واپسی کی قسم کے طور پر استعمال ہوتا ہے جو کوئی قدر واپس نہیں کرتے ہیں۔



خود کا محاسبہ کریں

1. بنیادی ڈیٹا کی اقسام سے آپ کا کیا مطلب ہے؟
2. ڈیٹا کی قسم char کو حروف کی نمائندگی کرنے کے لیے استعمال کیا جاتا ہے۔ پھر اسے اکثر عددی قسم کیوں کہا جاتا ہے؟
3. انٹیجرز پر فلوٹنگ پوائنٹ نمبرز کا کیا فائدہ ہے؟
4. ڈیٹا ٹائپ ڈبل ایک اور فلوٹنگ پوائنٹ ٹائپ ہے۔ اسے ایک الگ ڈیٹا ٹائپ کیوں سمجھا جاتا ہے؟
5. وائڈ (void) ڈیٹا کی قسم کا استعمال کیا ہے؟

9.12.2 بنیادی ڈیٹا کی اقسام کی میموری کی نمائندگی:

متغیر کو کسی خاص ڈیٹا کی قسم کے طور پر قرار دینے کی سب سے اہم وجہ میموری میں مناسب جگہ مختص کرنا ہے۔ ذخیرہ شدہ پروگرام کے تصور کے مطابق، ہر ڈیٹا کو پروسیس ہونے سے پہلے مین میموری میں رکھا جانا چاہیے۔ لہٰذا، C++ کمپائلر کمپائلر کے معیارات کے مطابق سنبھالے گئے ہر ڈیٹا کے لیے مخصوص میموری کی جگہ مختص کرتا ہے۔

11th_Comp Science_UM_Ch 9.indd 135 18-04-2023 15:04:19

مندرجہ ذیل ٹیبل 9.5 ظاہر کرتا ہے کہ ہر بنیادی ڈیٹا کی قسم کے لیے کتنی میموری کی جگہ مختص کی گئی ہے۔یاد رکھیں کہ ہر ڈیٹا کمپیوٹر میموری میں بائنری ہندسوں (binary digits) کی تصویر میں محفوظ ہوتا ہے (دیکھیں یونٹ I باب 2)۔

ٹیبل 9.5 بنیادی ڈیٹا کی اقسام کے لیے میموری مختص کرنا

Data type	Space in memory		Range of value
	in terms of bytes	in terms of bits	
char	1 byte	8 bits	-128 to 127
int	2 bytes	16 bits	-32,768 to 32,767
float	4 bytes	32 bits	3.4×10^{-38} to 3.4×10^{38} -1
double	8 bytes	64 bits	1.7×10^{-308} to 1.7×10^{308}-1

9.12.3 ڈیٹا کی قسم میں ترمیم کرنے والے: (Data type modifiers)

موڈیفائرس (modifiers) کا استعمال وائڈ قسم کے علاوہ کسی بنیادی ڈیٹا کی قسم کی ذخیرہ کرنے کی صلاحیت کو تبدیل کرنے کے لیے کیا جاتا ہے۔ عام طور پر، ہر بنیادی ڈیٹا کی قسم میں ڈیٹا آئٹمز کو میموری میں ذخیرہ کرنے کے لیے قدروں کی ایک مقررہ حد ہوتی ہے۔ مثال کے طور پر، int ڈیٹا کی قسم صرف دو بائٹس ڈیٹا کو محفوظ کر سکتی ہے۔ حقیقت میں، کچھ عددی اعداد و شمار کی لمبائی زیادہ ہو سکتی ہے اور میموری میں زیادہ جگہ کی ضرورت ہو سکتی ہے۔ اس صورت حال میں، ہمیں بڑی عددی اقدار کو ایڈجسٹ کرنے کے لیے میموری کی جگہ کو تبدیل کرنا چاہیے۔ **موڈیفائرس (modifiers) کا استعمال کسی بھی بنیادی ڈیٹا کی قسم کی میموری ایلوکیشن میں ترمیم (توسیع یا کم) کرنے کے لیے کیا جا سکتا ہے۔ انہیں کوالیفائر (Qualifiers) بھی کہا جاتا ہے۔**

C++ میں چار ترمیم کار استعمال ہوتے ہیں۔ وہ ہیں:

(1) دستخط شدہ (signed) (2) غیر دستخط شدہ (unsigned) (3) طویل (long) (4) مختصر (short)

ان چاروں ترمیم کاروں کو کسی بھی بنیادی ڈیٹا کی قسم کے ساتھ استعمال کیا جا سکتا ہے۔ مندرجہ ذیل ٹیبل 9.6 میں ترمیم کرنے والوں کے ساتھ اور اس کے بغیر ہر ڈیٹا کی قسم کے لیے میموری مختص کی گئی ہے۔

عددی قسم

ٹیبل 9.6 ڈیٹا کی اقسام کے لیے میموری کی تقسیم

Data type		Space in memory		Range of value
		in terms of bytes	in terms of bits	
short	short is a short name for **short int**	2 bytes	16 bits	-32,768 to 32,767
unsigned short	an integer number without minus sign.	2 bytes	16 bits	0 to 65535
signed short	An integer number with minus sign	2 bytes	16 bits	-32,768 to 32,767
Both short and signed short are similar				
int	An integer may or may not be signed	2 bytes	16 bits	-32,768 to 32,767

11th_Comp Science_UM_Ch 9.indd 136 18-04-2023 15:04:19

unsigned int	An integer without any sign (minus symbol)	2 bytes	16 bits	0 to 65535
signed int	An integer with sign	2 bytes	16 bits	-32,768 to 32,767
Both short and int are similar				
long	long is short name for long int	4 bytes	32 bits	-2147483648 to 2147483647
unsigned long	A double spaced integer without any sign	4 bytes	32 bits	0 to 4,294,967,295
signed long	A double spaced integer with sign	4 bytes	32 bits	-2147483648 to 2147483647

مندرجہ بالا ٹیبل واضح طور پر ظاہر کرتا ہے کہ ایک عددی قسم صرف 2 بائٹس ڈیٹا کو قبول کرتی ہے جب کہ ایک long int ڈیٹا کو قبول کرتا ہے جو اس سائز سے دوگنا ہوتا ہے یعنی 4 بائٹس ڈیٹا۔ لہذا، ہم ایک long int میں زیادہ ہندسوں کو ذخیرہ کر سکتے ہیں. (long ایک ترمیم کنندہ ہے اور int ایک بنیادی ڈیٹا کی قسم ہے)۔

چار قسم (Char type)

ٹیبل 9.7 چار ڈیٹا کی اقسام (Char data type) کے لیے میموری مختص کرنا

Data type		Space in memory		Range of value
		in terms of bytes	in terms of bits	
char	Signed ASCII character	1 byte	8 bits	-128 to 127
unsigned char	ASCII character without sign	1 byte	8 bits	0 to 255
signed char	ASCII character with sign	1 byte	8 bits	-128 to 127

فلوٹنگ پوائنٹ کی قسم

ٹیبل 9.8 فلوٹنگ پوائنٹ ڈیٹا کی اقسام (floating point data type) کے لیے میموری مختص

Data type		Space in memory		Range of value
		in terms of bytes	in terms of bits	
float	signed fractional value	4 bytes	32 bits	3.4×10^{-38} to $3.4\times10^{38}-1$
double	signed more precision fractional value	8 bytes	64 bits	1.7×10^{-308} to $1.7\times10^{308}-1$
long double	signed more precision fractional value	10 bytes	80 bits	3.4×10^{-4932} to $1.1\times10^{4932}-1$

یادداشت کی تخصیص استعمال کیے جانے والے کمپائلر کی قسم کی بنیاد پر مختلف ہوتی ہے۔یہاں، دی گئی قدریں **Turbo C++** کمپائلر کے مطابق ہیں۔ Dev C++، int اور long ڈبل قسموں کو کچھ اور جگہ فراہم کرتا ہے۔ مندرجہ ذیل ٹیبل 9.9، Turbo C++ اور Dev C++ میموری کی مختص کے درمیان فرق کو ظاہر کرتی ہیں۔

11th_Comp Science_UM_Ch 9.indd 137 18-04-2023 15:04:19

ٹیبل 9.9 Turbo C++ اور Dev C++ کی میموری ایلوکیشن

Data type	Memory size in bytes	
	Turbo C++	Dev C++
short	2	2
unsigned short	2	2
signed short	2	2
int	2	**4**
unsigned int	2	**4**
signed int	2	**4**
long	4	4
unsigned long	4	4
signed long	4	4
char	1	1
unsigned char	1	1
signed char	1	1
float	4	4
double	8	8
long double	10	**12**

چونکہ، Dev C++ int اور long کو 4 بائٹس فراہم کرتا ہے، ان میں سے کسی ایک کو بھی Dev C++ میں پروگرام لکھتے وقت بڑی عددی اقدار کو سنبھالنے کے لیے استعمال کیا جا سکتا ہے۔

نوٹ: sizeof() ایک آپریٹر ہے جو ڈیٹا کی قسم کا سائز دیتا ہے۔

C++ میں نمبر لاحقے (Number Suffixes in C++)

انٹیجر اور فلوٹنگ پوائنٹ نمبرز کے لیے مختلف لاحقے ہیں۔ لاحقہ کو ایک مختلف قسم کے طور پر ایک ہی قدر تفویض کرنے کے لیے استعمال کیا جا سکتا ہے۔ مثال کے طور پر، اگر آپ 45 کو unsigned int، long اور int، unsigned long int میں ذخیرہ کرنا چاہتے ہیں، تو آپ 45 کے ساتھ لاحقہ حرف L یا U (کسی بھی صورت میں) استعمال کر سکتے ہیں یعنی **45L** یا **45U**۔ اس قسم کا اعلامیہ مرتب کرنے والے کو ہدایت دیتا ہے کہ دی گئی قدروں کو long and unsigned ذخیرہ کرے۔ 'F' کو فلوٹنگ پوائنٹ ویلیوس (floating point values) کے لیے استعمال کیا جا سکتا ہے، مثال : 3.14F

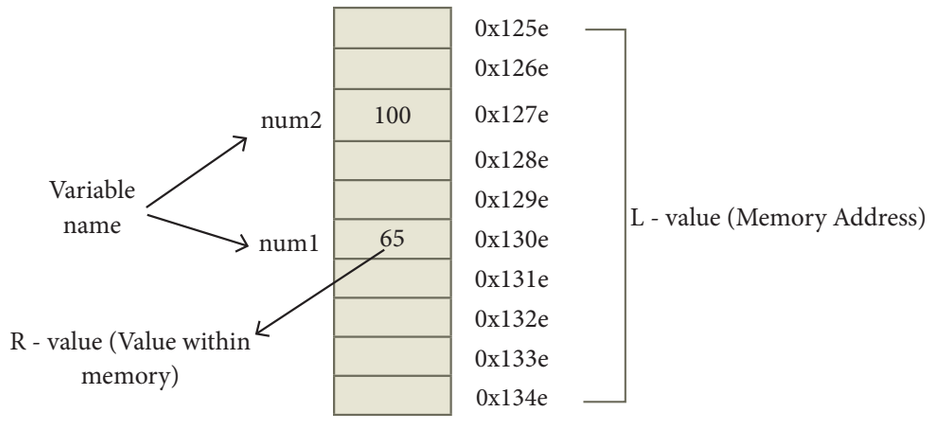
9.13 متغیرات (Variables)

متغیرات صارف کے متعین کردہ نام ہیں جو میموری کے مخصوص مقامات پر تفویض کیے گئے ہیں جن میں قدریں محفوظ ہیں۔متغیرات بھی شناخت کنندہ (identifiers) ہیں؛ اور اس لیے، متغیر کا نام دیتے وقت شناخت کنندگان کے نام کے اصولوں پر عمل کیا جانا چاہیے۔ یہ علامتی متغیرات (symbolic variables) کہلاتے ہیں کیونکہ یہ مقامات نامزد ہیں۔

علامتی متغیر کے ساتھ دو قدریں وابستہ ہیں۔ وہ R-value اور L-value ہیں۔

- R-value ایک میموری کی جگہ پر ذخیرہ شدہ ڈیٹا ہے۔
- L-value وہ میموری ایڈریس ہے جس میں R-value کو محفوظ کیا جاتا ہے۔

11th_Comp Science_UM_Ch 9.indd 138 18-04-2023 15:04:19



تصویر 9.14 متغیر کی میموری ایلوکیشن

یاد رکھیں، میموری ایڈریس ہیکساڈیسیمل اقدار (Hexadecimal decimal values) کی تصویر میں ہیں۔

9.13.1 متغیرات کا ڈکلیریشن (Declaration of Variables) سن

ہر متغیر کو پروگرام میں استعمال کرنے سے پہلے ان کا اظہار کیا جانا چاہیے۔ڈیکلریشن ایک ایسا عمل ہے جس میں کمپائلر کو اس قسم کے مطابق میموری مختص کرنے کی ہدایت کی جاتی ہے جو متغیر نام کے ساتھ بیان کی گئی ہے۔ مثال کے طور پر، اگر آپ کسی متغیر کو int قسم کے طور پر قرار دیتے ہیں، تو Dev C ++ میں، مرتب کرنے والا میموری کے 4 بائٹس مختص کرتا ہے۔ اس طرح، ہر متغیر کو ذخیرہ کرنے کی قدر کی قسم کے ساتھ اظہار کیا جانا چاہئے۔

ایک سے زیادہ متغیر کا ڈکلیریشن:

انفرادی متغیرات کو الگ کرنے والے کوما کا استعمال کرتے ہوئے ایک ہی قسم کے ایک سے زیادہ متغیر کو ایک ہی بیان کے طور پر قرار دیا جا سکتا ہے۔

سنٹیکس

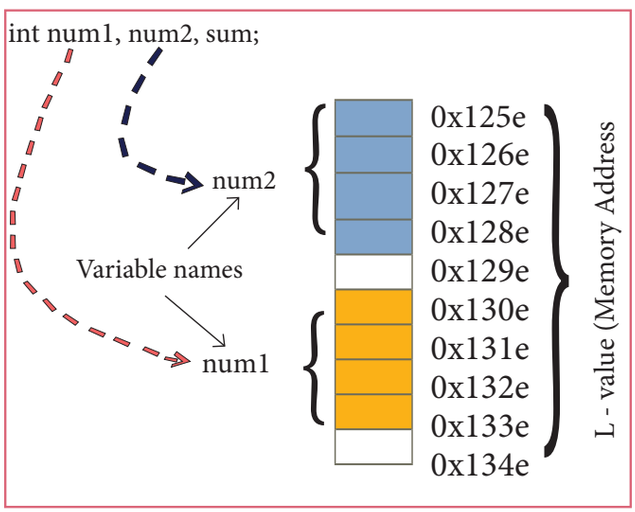
<data type> <var1>, <var2>, <var3> ..... <var_n>;

مثال:

int num1, num2, sum;

مندرجہ بالا بیان میں، تین متغیرات ہیں جنہیں int قسم کے طور پر قرار دیا گیا ہے۔ جس کا مطلب ہے، num2 ، num1 اور sum میں، آپ صرف عددی اقدار کو محفوظ کر سکتے ہیں۔

مندرجہ بالا اعلان کے لیے، C++ کمپائلر ہر متغیر کے لیے 4 بائٹس میموری (یعنی 4 میموری بکس) مختص کرتا ہے۔



تصویر 9.15 int قسم کے متغیرات کی میموری ایلوکیشن

11th_Comp Science_UM_Ch 9.indd 139 18-04-2023 15:04:19

اگر آپ کسی ابتدائی قدر کے بغیر کسی متغیر کا اعلان کرتے ہیں، تو اس متغیر کے لیے مختص میموری کی جگہ کچھ نامعلوم قدر کے ساتھ قابض ہو جائے گی۔ ان نامعلوم اقدار کو "جنک (Junk)" یا "کوڑا کرکٹ (Garbage)" اقدار کہا جاتا ہے۔

```
#include <iostream>
using namespace std;
int main()
{
        int num1, num2, sum;
        cout << num1 << endl;
        cout << num2 << endl;
        cout << num1 + num2;
}
```

مندرجہ بالا پروگرام میں، کچھ نامعلوم قدریں میموری میں موجود ہوں گی جو **num1** اور **num2** متغیرات اور سٹیٹمنٹ **cout << num1 + num2** کے لیے مختص کی گئی ہیں۔ ان نامعلوم فضول قدروں کا مجموعہ ظاہر کرے گا۔

9.13.2 متغیرات کا آغاز

کسی متغیر کو اس کے اعلان کے دوران ابتدائی قدر تفویض کرنا "انیسیالائی زیشن (Initialization)" کہلاتا ہے۔

مثالیں:

```
int num = 100;
float pi = 3.14;
double price = 231.45;
```

یہاں، اعلان کے دوران متغیرات num، pi، اور قیمت کا آغاز کیا گیا ہے۔ان ابتدائی اقدار کو بعد میں پروگرام کے عمل کے دوران تبدیل کیا جا سکتا ہے۔

مثال 9.6 سلنڈر (CSA = 2 pi r * h) کے خمیدہ سطح کا رقبہ تلاش کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
   float pi = 3.14, radius, height, CSA;
   cout << "\n Curved Surface Area of a cylinder";
   cout << "\n Enter Radius (in cm): ";
   cin >> radius;
   cout << "\n Enter Height (in cm): ";
   cin >> height;
   CSA = (2*pi*radius)*height;
   system("cls");
   cout << "\n Radius: " << radius <<"cm";
   cout << "\n Height: " << height << "cm";
   cout << "\n Curved Surface Area of a Cylinder is " << CSA <<" sq. cm.";
}
```

Output:

```
Curved Surface Area of a cylinder
Enter Radius (in cm): 7
Enter Height (in cm): 20
Radius: 7cm
Height: 20cm
Curved Surface Area of a Cylinder is 879.2 sq. cm.
```

ایک ہی قسم کے متغیرات کو ایک ہی بیان میں شروع کیا جا سکتا ہے۔

11th_Comp Science_UM_Ch 9.indd 140 18-04-2023 15:04:19

مثال:

int x1 = -1, x2 = 1, x3, n;

9.13.3 متحرک ابتداء:

ایک متغیر ایک پروگرام کے عمل کے دوران شروع کیا جا سکتا ہے. اسے "متحرک ابتداء(Dynamic Initialization)" کہا جاتا ہے۔

مثال کے طور پر،

int num1, num2, sum;
sum = num1 + num2;

مندرجہ بالا دو بیانات کو مندرجہ ذیل کے طور پر یکجا کیا جا سکتا ہے:

int sum = num1+num2;

یہ عمل کے دوران num1 اور num2 کی معلوم قدروں کا استعمال کرتے ہوئے رقم کو شروع کرتا ہے۔

مثال 9.7 متحرک ابتداء کو واضح کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int num1, num2;
        cout << "\n Enter number 1: ";
        cin >> num1;
        cout << "\n Enter number 2: ";
        cin >> num2;
        int sum = num1 + num2; // Dynamic initialization
        cout << "\n Average: " << sum /2;
}
```

Output:
Enter number 1: 78
Enter number 2: 65
Average: 71

مندرجہ بالا پروگرام میں، num1 اور num2 کی قدریں حاصل کرنے کے بعد، sum کا اعلان کیا جاتا ہے۔اور ان دو تغیرات کے اضافے کے ساتھ شروع کیا گیا۔ اس کے بعد، اسے 2 سے تقسیم کیا جاتا ہے۔

مثال 9.8: نیم دائرے کا رقبہ اور دائرہ معلوم کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int radius;
        float pi = 3.14;
        cout << "\n Enter Radius (in cm): ";
        cin >> radius;
        float perimeter = (pi+2)*radius;   // dynamic initialization
        float area = (pi*radius*radius)/2; // dynamic initialization
        cout << "\n Perimeter of the semicircle is " << perimeter << " cm";
        cout << "\n Area of the semicircle is " << area << " sq.cm";
}
```

Output:
Enter Radius (in cm): 14
Perimeter of the semicircle is 71.96 cm
Area of the semicircle is 307.72 sq.cm

11th_Comp Science_UM_Ch 9.indd 141 18-04-2023 15:04:19

9.13.4 رسائی موڈیفائر کانسٹ (The Access modifier const)

کانسٹ (Const) وہ کلیدی لفظ ہے جو کانسٹینٹ (Constant) کا اعلان کرنے کے لیے استعمال ہوتا ہے۔ آپ نے پہلے ہی پچھلے باب میں کانسٹینٹ کے بارے میں سیکھا ہے۔ کانسٹ(Const) کلیدی لفظ متغیر کی رسائی کو تبدیل / محدود کرتا ہے۔ لہذا، اسے رسائی موڈیفائر (Access modifier) کے نام سے جانا جاتا ہے۔

مثال کے طور پر،

int num = 100;

مذکورہ بیان ابتدائی قدر 100 کے ساتھ متغیر نمبر کا اعلان کرتا ہے۔تاہم، عمل درآمد کے دوران نمبر کی قدر کو تبدیل کیا جا سکتا ہے۔ اگر آپ مندرجہ بالا تعریف میں ترمیم کرتے ہیں تو; **const int num = 100** متغیر نمبر ایک کانسٹینٹ بن جاتا ہے اور اس کی قدر پورے پروگرام میں 100 رہے گی، اور اسے عمل درآمد کے دوران کبھی تبدیل نہیں کیا جا سکتا ہے۔

```
#include <iostream>
using namespace std;
int main()
{
        const int num=100;
        cout << "\n Value of num is = " << num;
        num = num + 1; // Trying to increment the constant
        cout << "\n Value of num after increment " << num;
}
```

مندرجہ بالا کوڈ میں، ٹربو کمپائلر **(turbo compiler)** میں **"Cannot modify the const object"** اور C Dev ++ میں "اسائنمنٹ آف ریڈ اونلی میموری نمبر **(Assignment of read only memory num)**" کے طور پر ایک ایرر میسج (error message) ظاہر ہو گا۔

9.13.5 حوالہ جات (References)

ایک حوالہ پہلے سے متعین متغیر کے لیے ایک عرف فراہم کرتا ہے۔حوالہ کا اعلان بنیادی قسم اور ایک& (ایمپرسینڈampersand -) علامت پر مشتمل ہوتا ہے۔ حوالہ متغیر کا نام پہلے اعلان کردہ متغیر کی قدر تفویض کیا جاتا ہے۔

سنٹیکس:

<type> <& reference_variable> = <original_variable>;

مثال 9.9: حوالہ متغیر کا اعلان کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int num;
        int &temp = num; //declaration of a reference variable temp
        num = 100;
cout << "\n The value of num = " << num;
cout << "\n The value of temp = " << temp;
}
```

The output of the above program will be

The value of num = 100

The value of temp = 100

11th_Comp Science_UM_Ch 9.indd 142 18-04-2023 15:04:19



خود کا محاسبہ کیجیے

1. موڈی فائر کیا ہیں؟ موڈی فائر کا استعمال کیا ہے؟
2. درج ذیل C++ بیان میں کیا غلط ہے:
   long float x;
3. متغیر کیا ہے؟ ایک متغیر کو علامتی متغیر کیوں کہا جاتا ہے؟
4. متغیر کی متحرک ابتدا سے آپ کا کیا مطلب ہے؟ ایک مثال دیں۔
5. درج ذیل بیان میں کیا غلط ہے؟
   const int x;

9.14 فارمیٹنگ آؤٹ پٹ
(Formatting Output)

آسانی سے پڑھنے اور سمجھنے کے لیے آؤٹ پٹ اسکرینوں کی ترتی میں فارمیٹنگ آؤٹ پٹ بہت اہم ہے۔ مینی پولیٹرس (manipulators) کا استعمال کسی بھی C++ پروگرام کے آؤٹ پٹ کو فارمیٹ کرنے کے لیے کیا جاتا ہے۔مینی پولیٹرس (manipulators) ایسے فنکشنز ہیں جنہیں خاص طور پر انسرشن (<<) اور ایکسٹرکشن (>>) آپریٹرز کے ساتھ استعمال کرنے کے لیے ڈیزائن کیا گیا ہے۔

C++ فارمیٹنگ کے لیے کئی ان پٹ اور آؤٹ پٹ مینی پو لیٹرس پیش کرتا ہے۔ عام طور پر استعمال ہونے والے مینی پولیٹرس یہ ہیں: **endl, setw, setfill, setprecision and setf.**، ان مینی پولیٹرس کو استعمال کرنے کے لیے، آپ کو مناسب ہیڈر فائل شامل کرنی چاہیے۔ endl مینی پولیٹر ایک ائی او اسٹریم (iostream)ہیڈر فائل (header file) کا رکن ہے۔ setw, setfill, setprecision and setf مینی پولیٹرس آئی او مینپ(iomanip) ہیڈر فائل کے رکن ہیں۔

endl (اینڈ دی لائن) (End the Line)

endl کو C++ میں لائن فیڈر(line feeder) کے طور پر استعمال کیا جاتا ہے۔ اسے '\n' کے متبادل کے طور پر استعمال کیا جا سکتا ہے۔ دوسرے الفاظ میں، endl ایک نئ لائن داخل کرتا ہے اور پھر کرسر کو اگلی لائن کے آغاز کی طرف اشارہ کرتا ہے۔ endl اور '\n' میں فرق ہے، حالانکہ وہ ایک جیسے کام انجام دے رہے ہیں۔

- endl - ایک نئ لائن داخل کرتا ہے اور بفر (buffer) کو فلش (flush) کرتا ہے (فلش کا مطلب ہے – صاف کرنا)
- '\n' - صرف ایک نئ لائن داخل کرتا ہے۔

مثال:

```
cout << "\n The value of num = " << num;
cout << "The value of num = " << num <<endl;
```

یہ دونوں بیانات ایک ہی آؤٹ پٹ کو ظاہر کرتے ہیں۔

setw ( )

setw مینی پولیٹر آؤٹ پٹ کے لیے تفویض کردہ فیلڈ کی چوڑائی کا تعین کرتا ہے۔ فیلڈ کی چوڑائی آؤٹ پٹ میں لکھے جانے والے حروف کی کم از کم تعداد کا تعین کرتی ہے۔

سنٹیکس (syntax):

setw(number of characters)

11th_Comp Science_UM_Ch 9.indd 143 18-04-2023 15:04:19

مثال 9.10: خالص تنخواہ کا حساب لگانے کا پروگرام

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
        float basic, da, hra, gpf, tax, gross, np;
        char name[30];
        cout << "\n Enter Basic Pay: ";
        cin >> basic;
        cout << "\n Enter D.A : ";
        cin >> da;
        cout << "\n Enter H.R.A: ";
        cin >> hra;
        gross = basic+da+hra; // sum of basic, da and hra
        gpf = (basic+da) * 0.10; // 10% 0f basic and da
        tax = gross * 0.10; //10% of gross pay
        np = gross - (gpf+tax); //netpay = earnings - deductions
        cout << setw(25) << "Basic Pay : " << setw(10)<< basic<< endl;
        cout << setw(25) << "Dearness Allowance : "<< setw(10)<<da<< endl;
        cout<<setw(25)<<"House Rent Allowance : "<<setw(10)<< hra<<endl;
        cout << setw(25) << "Gross Pay : " << setw(10) << gross << endl;
        cout << setw(25) << "G.P.F : " << setw(10) << gpf << endl;
        cout << setw(25) << "Income Tax : " << setw(10)<< tax << endl;
        cout << setw(25) << "Net Pay : " << setw(10) << np << endl;
}
```

The output will be,

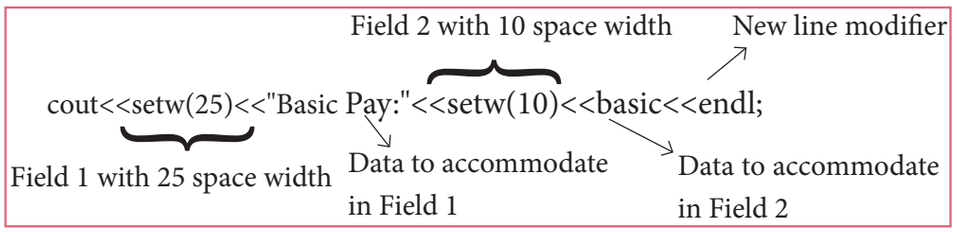
```
Enter Basic Pay: 12000
Enter D.A : 1250
Enter H.R.A : 1450
                    Basic Pay  :      12000
          Dearness Allowance   :       1250
       House Rent Allowance    :       1450
                    Gross Pay  :      14700
                        G.P.F  :       1325
                   Income Tax  :       1470
                      Net Pay  :      11905
```

(HOT: Try to make multiple output statements as a single cout statement)

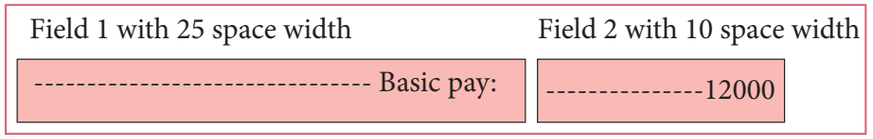
مندرجہ بالا پروگرام میں، ہر آؤٹ پٹ سٹیٹمنٹ میں دو setw() manipulators ہوتے ہیں۔ پہلا 25 (setw(25 اسپیس (space) کے ساتھ فائل بناتا ہے اور دوسرا 10 (setw(10 اسپیس کے ساتھ ایک اور فیلڈ بناتا ہے۔ جب آپ ان فیلڈز میں کسی قدر کی نمائندگی کرتے

11th_Comp Science_UM_Ch 9.indd 144 18-04-2023 15:04:19

ہیں، تو یہ فیلڈ کے اندر دائیں سے بائیں قدر دکھائے گا۔



فیلڈ 1 اور فیلڈ 2 میں، سٹرنگ "بنیادی تنخواہ:" اور بنیادی تنخواہ کی قدر کو دکھایا گیا ہے جیسا کہ ذیل میں تصویر 9.16 میں دیا گیا ہے۔



تصویر 9.16 setw( ) فنکشن

setprecision ( )

اس کا استعمال ہندسوں کی مخصوص تعداد میں فریکشن کے ساتھ نمبروں کو ظاہر کرنے کے لیے کیا جاتا ہے۔

سنٹیکس:

setprecision (number of digits);

مثال:

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{    float hra = 1200.123;
     cout << setprecision (5) << hra; }
```

مندرجہ بالا کوڈ میں، دی گئی ویلیو 1200.123 کو 5 ہندسوں میں دکھایا جائے گا جس میں فریکشن بھی شامل ہیں۔ لہذا، آؤٹ پٹ **1200.1** ہو جائے گا

setprecision( ) اقدار کو بائیں سے دائیں پرنٹ کرتا ہے۔ مندرجہ بالا کوڈ کے لیے، پہلے، یہ 4 ہندسے لے گا اور پھر فریکشنل (fractional) حصے سے ایک ہندسہ پرنٹ کرے گا۔

سیٹ پریسی شن(setprecision) کا استعمال اعشاریہ کی تعداد کو ظاہر کرنے کے لیے بھی کیا جا سکتا ہے۔ اس کام کو کرنے کے لیے، آپ کو setf() مینی پولیٹر (manipulator) کے اندر ایک ios فلیگ(پرچم) سیٹ کرنا ہوگا۔اسے دو شکلوں میں استعمال کیا جا سکتا ہے:

(i) فکسڈ (fixed) اور (ii) سائنٹیفک (scientific)۔

دونوں شکلیں اس وقت استعمال ہوتی ہیں جب فکسڈ یا سائنٹیفک مطلوبہ الفاظ کو سیٹ پریسی شن مینی پولیٹر(setprecision manipulator) سے پہلے مناسب طریقے سے استعمال کیا جاتا ہے۔

مثال:

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{    cout.setf(ios::fixed);
     cout << setprecision(2)<<0.1; }
```


11th_Comp Science_UM_Ch 9.indd 145 18-04-2023 15:04:20

مندرجہ بالا پروگرام میں، ios فلیگ کو فکسڈ ٹائپ پر سیٹ کیا گیا ہے۔ یہ فلوٹنگ پوائنٹ(floating point) نمبر کو فکسڈ نوٹیشن (fixed notation)میں پرنٹ کرتا ہے۔ تو، آؤٹ پٹ 0.10 ہو گا۔

```
cout.setf(ios::scientific);
cout << setprecision(2) << 0.1;
```

مندرجہ بالا بیانات میں، ios فلیگ سائنٹیفک (scientific) قسم پر سیٹ کیا گیا ہے؛ یہ فلوٹنگ پوائنٹ نمبر کو سائنٹیفک نوٹیشن (scientific notation) میں پرنٹ کرے گا۔ تو، آؤٹ پٹ 1.00e-001 ہو گا۔

9.15 اکسپریشن (Expression)

ایکسپریشن : آپریٹرز (operators)، کانسٹینٹ (constants) اور متغیرات (variables) کا مجموعہ ہے جو C++ کے اصولوں کے مطابق ترتیب دیا گیا ہے۔ اس میں فنکشن کالز (functions calls) بھی شامل ہو سکتی ہیں جو قدر واپس کرتی ہیں۔ (فنکشنز آئندہ ابواب میں سیکھے جائیں گے)۔

ایک ایکسپریشن ایک یا زیادہ آپرینڈز پر مشتمل ہو سکتا ہے، اور قدر پیدا کرنے کے لیے صفر یا زیادہ آپریٹرز پر مشتمل ہو سکتا ہے۔ C++ میں، اکسپریشن کی سات اقسام ہیں، اور وہ یہ ہیں:

(i) کانسٹینٹ اکسپریشن (Constant Expression)
(ii) انٹیجر اکسپریشن (Integer Expression)
(iii) فلوٹنگ اکسپریشن (Floating Expression)
(iv) ریلیشنل اکسپریشن (Relational Expression)
(v) لوجیکل اکسپریشن (Logical Expression)
(vi) بٹوائس اکسپریشن (Bitwise Expression)
(vii) پوائنٹر اکسپریشن (Pointer Expression)

نمبر شمار	اکسپریشن	تفصیل	مثال
1	کانسٹینٹ اکسپریشن (Constant Expression)	کانسٹینٹ اکسپریشن صرف کانسٹینٹ اقدار پر مشتمل ہوتا ہے۔	int num=100;
2	انٹیجر اکسپریشن (Integer Expression)	عددی نتائج پیدا کرنے کے لیے سمپل ارتھ میٹک آپریٹرز کے ساتھ انٹیجر اور کیریکٹر کی قدروں اور/یا متغیرات کا مجموعہ۔	sum=num1+num2; avg=sum/5;
3	فلوٹ اکسپریشن (Float Expression)	فلوٹنگ پوائنٹ کے نتائج پیدا کرنے کے لیے سمپل ارتھ میٹک آپریٹرز کے ساتھ فلوٹنگ پوائنٹ ویلیوز اور/یا متغیرات کا مجموعہ	Area=3.14*r*r;
4	ریلیشنل اکسپریشن (Relational Expression)	نتائج کے طور پر bool (صحیح کا مطلب 1 یا غلط کا مطلب 0) اقدار پیدا کرنے کے لیے متعلقہ آپریٹرز کے ساتھ اقدار اور/یا متغیرات کا مجموعہ۔	x>y; a+b==c+d;
5	لوجیکل اکسپریشن (Logical Expression)	لوجیکل آپریٹرز کے ساتھ اقدار اور/یا متغیرات کا مجموعہ جس سے نتائج کے طور پر bool قدریں پیدا ہوتی ہیں۔	(a>b)&& (c==10);
6	بٹوائس اکسپریشن (Bitwise Expression)	بٹ وائس آپریٹرز کے ساتھ اقدار اور/یا متغیرات کا مجموعہ	x>>3; a<<2;
7	پوائنٹر اکسپریشن (Pointer Expression)	پوائنٹر ایک متغیر ہے جو میموری ایڈریس رکھتا ہے۔ پوائنٹر متغیر کا اعلان (*) علامت کا استعمال کرتے ہوئے کیا جاتا ہے۔	int *ptr;

ٹیبل 9.10: اکسپریشن کے اقسام

11th_Comp Science_UM_Ch 9.indd 146 18-04-2023 15:04:20

9.16 ٹائپ کنورژن
(Type Conversion)

ایک بنیادی ڈیٹا کی قسم کو دوسرے میں تبدیل کرنے کے عمل کو "ٹائپ کنورژن – Type Conversion " کہا جاتا ہے۔ C++ دو قسم کے کنورژن فراہم کرتا ہے۔

(1) امپلی سٹ ٹائپ کنورشن (Implicit type conversion)

(2) اکسپلی سٹ ٹائپ کنورشن (Explicit type conversion)

(1) امپلی سٹ ٹائپ کنورشن

امپلی سٹ ٹائپ کنورشن ایک تبدیلی ہے جو مرتب کرنے والے کے ذریعہ خود بخود انجام دی جاتی ہے۔لہذا، امپلی سٹ کنورشن کو "**آٹو میٹک کنورژن – Automatic conversion**" بھی کہا جاتا ہے۔

اس قسم کی تبدیلی عام طور پر لاگو ہوتی ہے جب بھی مختلف ڈیٹا کی اقسام کو ایک اکسپریشن میں ملایا جاتا ہے۔ اگر آپرینڈز کی قسم مختلف ہوتی ہے، تو کمپائلر ان میں سے ایک کو دوسرے سے مماثل کرنے کے لیے تبدیل کرتا ہے، اس اصول کا استعمال کرتے ہوئے کہ "چھوٹی" قسم کو "وسیع" قسم میں تبدیل کر دیا جاتا ہے، جسے "**ٹائپ پروموشن – Type Promotion**" کہا جاتا ہے۔

مثال کے طور پر:

```
#include <iostream>
using namespace std;
int main()
{
        int a=6;
        float b=3.14;
        cout << a+b;
}
```

مندرجہ بالا پروگرام میں، اوپرینڈ a ایک int قسم ہے اور b ایک فلوٹ قسم ہے۔دوران پروگرام پر عمل درآمد کرتے ہوئے، int کو فلوٹ میں تبدیل کر دیا جاتا ہے، کیونکہ ایک فلوٹ int سے زیادہ چوڑا ہوتا ہے۔ لہذا، مندرجہ بالا پروگرام کا آؤٹ پٹ ہوگا: 9.14

درج ذیل ٹیبل 9.11 آپ کو کنورژن کا نمونہ دکھاتا ہے۔

RHO / LHO	char	short	int	long	float	double	long double
char	int	int	int	long	float	double	long double
short	int	int	int	long	float	double	long double
int	int	int	int	long	float	double	long double
long	long	long	long	long	float	double	long double
float	float	float	float	float	float	double	long double
double	double	double	double	double	double	double	long double
long double	long double	long double	long double	long double	long double	long double	long double

(RHO – دائیں ہاتھ کا آپرینڈ؛ LHO – بائیں ہاتھ کا آپرینڈ)

ٹیبل 9.11: مخلوط آپرینڈز کی امپلی سٹ کنوژن

11th_Comp Science_UM_Ch 9.indd 147 18-04-2023 15:04:20

(2) اکسپلی سِٹ ٹائپ کنورژن (Explicit type conversion)

++C پروگرامر کے ذریعہ متغیرات یا اظہارات کو ایک ڈیٹا کی قسم سے دوسری مخصوص ڈیٹا کی قسم میں تبدیل کرنے کی اجازت دیتا ہے۔ اسے **"ٹائپ کاسٹنگ – type casting"** کہا جاتا ہے۔

سنٹیکس:

```
(type-name) expression;
```

جہاں ٹائپ کا نام ایک درست ++C ڈیٹا کی قسم ہے جس میں تبدیلی کی جانی ہے۔

مثال:

```
#include <iostream>
using namespace std;
int main( )
{
    float varf=78.685;
    cout << (int) varf;
}
```

مندرجہ بالا پروگرام میں، **متغیر varf** کو ابتدائی قدر 78.685 کے ساتھ **فلوٹ (float)** کے طور پر قرار دیا گیا ہے۔ varf کی قدر واضح طور پر cout اسٹیٹمنٹ میں int قسم میں تبدیل ہوجاتی ہے۔ اس طرح، حتمی پیداوار 78 ہو جائے گا۔

اکسپلی سٹ کنورژن کے دوران، اگر آپ زیادہ رینج والی کسی قسم کو قدر تفویض کرتے ہیں، تو اس سے کوئی مسئلہ نہیں ہوتا۔ لیکن، ایک چھوٹی قسم کو بڑی قسم کی قدر تفویض کرنے کے نتیجے میں پریسی شن ویلیوس (precision Value) کھو سکتی یا ضائع ہو سکتی ہیں۔

نمبر شمار	اکسپلی سٹ کنورژن	مسئلہ
1	ڈبل ٹو فلوٹ (double to float)	**پریسی شن کا نقصان۔ اگر اصل قدر ہدف کی قسم کے لیے حد سے باہر ہے، تو نتیجہ غیر واضح ہو جاتا ہے۔**
2	فلوٹ ٹو انٹ (float to int)	**جزوی حصے کا نقصان۔ اگر اصل قدر ہدف کی قسم کے لیے حد سے باہر ہو، تو نتیجہ غیر متعین ہو جاتا ہے۔**
3	لونگ ٹو شارٹ (long to short)	**ڈیٹا کا نقصان**

ٹیبل 9.12 - اکسپلی سٹ کنورژن کے مسائل

مثال:

```
#include <iostream>
using namespace std;
int main()
{
        double varf=178.25255685;
        cout << (float) varf << endl;
        cout << (int) varf << endl;
}
```

Output:

178.253
178

11th_Comp Science_UM_Ch 9.indd 148 18-04-2023 15:04:20

خود کا محاسبہ کریں

1. ٹائپ کنورژن سے کیا مراد ہے؟
2. امپلی سٹ ٹائپ کنورشن ,اکسپلی سٹ ٹائپ کنورشن سے کس طرح مختلف ہے؟
3. endl اور \n میں کیا فرق ہے؟
4. حوالہ جات کا استعمال کیا ہے؟
5. setprecision( ) کا استعمال کیا ہے؟



عملی مشق

1. دو متغیر کی قدروں کا تبادلہ کرنے کے لیے C++ پروگرام لکھیں۔
   a. تیسرے متغیر کا استعمال کرتے ہوئے
   b. تیسرا متغیر استعمال کیے بغیر
2. درج ذیل کام کرنے کے لیے C++ پروگرام لکھیں:
   a. کواڈرینٹ (quadrant) کا دائرہ (perimeter) اور رقبہ (area) معلوم کرنے کے لیے۔
   b. مثلث (triangle) کا رقبہ معلوم کرنے کے لیے۔
   c. درجہ حرارت کو سیلشیس (Celsius) سے فارن ہائیٹ (Fahrenheit) میں تبدیل کرنا۔
3. 10ویں اسٹینڈرڈ پبلک امتحان سے حاصل کردہ نمبروں کی کل اور فیصد معلوم کرنے کے لیے C++ لکھیں۔ کل اور فیصد کے ساتھ ایک ایک کرکے تمام نشانات دکھائیں۔ فارمیٹنگ کے افعال کا اطلاق کریں۔

یاد رکھنے کے لیے نکات

- ہر پروگرامنگ زبان میں دو بنیادی عناصر ہوتے ہیں، یعنی ڈیٹا کی اقسام اور متغیر۔
- C++ میں، ڈیٹا کی اقسام کو تین اہم زمروں میں درجہ بندی کیا گیا ہے۔ (1) بلٹ ان ڈیٹا کی اقسام (built -in data types) (2) صارف کی وضاحت کردہ ڈیٹا کی اقسام (User-define data types) (3) اخذ کردہ ڈیٹا کی اقسام (Derived data types)
- متغیرات مخصوص ڈیٹا کی قسم کی قدروں کو رکھنے کے لیے نامزد کردہ جگہ ہیں۔
- C++ میں ڈیٹا کی پانچ بنیادی اقسام ہیں: int، char، double، float، اور void۔
- C++ کمپائلر کمپائلر کے معیارات کے مطابق سنبھالے گئے ہر ڈیٹا کے لیے مخصوص میموری کی جگہ مختص کرتا ہے۔
- متغیرات صارف کے متعین کردہ نام ہیں جو میموری کے مقام پر تفویض کیے گئے ہیں جس میں قدریں محفوظ ہیں۔
- ڈیکلریشن (Declaration) ایک ایسا عمل ہے جو کمپائلر کو متغیر نام کے ساتھ مخصوص قسم کے مطابق میموری مختص کرنے کی ہدایت کرتا ہے۔
- مینی پولیٹرس (Manipulators) کا استعمال کسی بھی C++ پروگرام کے آؤٹ پٹ کو فارمیٹ کرنے کے لیے کیا جاتا ہے۔ مینی پولیٹر ایسے فنکشنز ہیں جنہیں خاص طور پر انسرشن (<<) اور ایکسٹرکشن (>>) آپریٹرز کے ساتھ استعمال کرنے کے لیے ڈیزائن کیا گیا ہے۔
- ایک اکسپریشن (expression) آپریٹرز، کانسٹینٹ اور متغیرات کا مجموعہ ہے جو C++ کے اصولوں کے مطابق ترتیب دیا گیا ہے۔
- ایک بنیادی ڈیٹا کی قسم کو دوسری میں تبدیل کرنے کے عمل کو "ٹائپ کنورژن - Type conversion " کہا جاتا ہے۔ C++ دو قسم کے تبادلے فراہم کرتا ہے۔ (1) امپلی سٹ ٹائپ کنورژن اور (2) اکسپلی سٹ ٹائپ کنورژن۔

11th_Comp Science_UM_Ch 9.indd 149 18-04-2023 15:04:20

تشخیص

U2QAC

سیکشن - A

صحیح جواب کا انتخاب کریں

1. C++ میں ڈیٹا کے اقسام کی کتنی قسمیں دستیاب ہیں؟

5 (a) 4 (b) 3 (c) 2 (d)

2. درج ذیل میں سے کونسی ڈیٹا کی قسم بنیادی قسم نہیں ہے؟

signed (a) int (b) float (c) char (d)

3. درج ذیل اسٹیٹمنٹ کا نتیجہ کیا ہو گا؟

```
char ch= 'B';
cout << (int) ch;
```

B (a) b (b) 65 (c) 66 (d)

4. فلوٹنگ پوائنٹ ویلیو کو ظاہر کرنے کے لیے کس حرف کو بطور لاحقہ استعمال کیا جاتا ہے؟

F (a) C (b) L (c) D (d)

5. اگر آپ Dev C ++ کا استعمال کر رہے ہیں تو درج ذیل متغیر اعلان(variable declaration) کے لیے میموری کے کتنے بائٹس مختص کیے گئے ہیں؟ short int x ;

2 (a) 4 (b) 6 (c) 8 (d)

6. درج ذیل ٹکڑوں کا آؤٹ پٹ کیا ہے؟

```
char ch = 'A';
ch = ch + 1;
```

B (a) A1 (b) F (c) 1A (d)

7. مندرجہ ذیل میں سے کون سا ڈیٹا ٹائپ موڈیفائر نہیں ہے؟

signed (a) long (c) int (b) short (d)

8. مندرجہ ذیل میں سے کون سا آپریٹر ڈیٹا کی قسم کا سائز واپس کرتا ہے؟

sizeof( ) (a) int ( ) (b) long ( ) (c) double ( ) (d)

9. متغیر کے حوالہ تک رسائی کے لیے کون سا آپریٹر استعمال ہوتا ہے؟

$ (a) # (b) & (c) ! (d)

10. یہ endl کمانڈ کے متبادل کے طور پر استعمال کیا جا سکتا ہے:

\t (a) \b (b) \0 (c) \n (d)

سیکشن - B

بہت ہی مختصر جوابات دیں۔

1. مثال کے ساتھ const کلیدی لفظ پر ایک مختصر نوٹ لکھیں۔
2. setw () فارمیٹ مینی پولیٹر کا استعمال کیا ہے؟
3. char کو اکثر انٹیجر ڈیٹا ٹائپ کیوں سمجھا جاتا ہے؟
4. حوالہ متغیر (reference variable) کیا ہے؟ اس کا استعمال کیا ہے؟
5. درج ذیل C++ بیان پر غور کریں۔ کیا وہ برابر ہیں؟

char ch = 67 ; char ch = 'C' ;

11th_Comp Science_UM_Ch 9.indd 150 18-04-2023 15:04:20

6. 56L اور 56 میں کیا فرق ہے؟

7. اس بات کا تعین کریں کہ درج ذیل میں سے کون سے درست کانسٹینٹ ہیں؟ اور ان کی قسم کی وضاحت کریں۔

0.5 (i) 'Name' (ii) '\t' (iii) 27,822 (iv)

8. فرض کریں کہ x اور y دو ڈبل قسم کے متغیر ہیں جنہیں آپ انٹیجر کے طور پر شامل کرنا چاہتے ہیں اور انٹیجر متغیر کو تفویض کرنا چاہتے ہیں۔ اوپر کرنے کے لیے ایک C++ اسٹیٹمنٹ بنائیں۔

9. اگر num=6 شروع میں درج کریں تو کیا نتیجہ نکلے گا۔

cout << num; (a)

cout << (num==5); (b)

10. درج ذیل دو اسٹیٹ منٹس میں سے کون سا درست ہے؟ کیوں؟ ان کا نتیجہ بھی لکھیں۔

int a; a = 3,014; (i) int a; a=(3,014); (ii)

سیکشن C-

مختصر جوابات دیں۔

1. C++ میں ارتھ میٹک آپریٹرس کیا ہیں؟ یونری اور بائنری ارتھ میٹک آپریٹرس میں فرق کریں۔ ان میں سے ہر ایک کے لیے مثال دیں۔

2. ریلیشنل آپریٹرز اور لوجیکل آپریٹرس ایک دوسرے سے کیسے متعلق ہیں؟

3. درج ذیل C++ اکسپریشنس کی تشخیص کریں جہاں x,y, z انٹیجرس ہیں اور m,n فلوٹنگ پوائنٹ نمبرس ہیں۔ x,y, اور m کی قدریں درج ذیل ہیں۔ x=5, y=4 اور ; m=25

n = x + y / x; (i)

z = m * x + y; (ii)

z *= x * m + x; (iii)

حوالہ جات:

(1) Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.

(2) The Complete Reference C++ (Forth Edition), Herbert Schildt. Mc.Graw Hills.

(3) Computer Science with C++ (A text book of CBSE XI and XII), Sumita Arora, Dhanpat Rai & Co.

11th_Comp Science_UM_Ch 9.indd 151 18-04-2023 15:04:20

فلو آف کنٹرول
(Flow of Control)

سیکھنے کے مقاصد

اس باب کو سیکھنے کے بعد طلباء کے قابل ہو جائیں گے۔

- مختلف قسم کے اسٹیٹ منٹس کو سمجھنے
- C++ میں کنٹرول اسٹیٹ منٹس کے مختلف فلو کی تشکیل کرنے

10.1 تعارف

پچھلے ابواب میں آپ نے C++ پروگرامنگ کے بنیادی تصورات سیکھے جیسے متغیرات (variables)، کانس ٹینس (constants)، آپریٹرس (operators)، ڈیٹا کی قسمیں وغیرہ۔ عام طور پر ایک پروگرام شروع سے آخر تک اپنے اسٹیٹ مینٹس کو ترتیب وار انجام دیتا ہے۔ تاہم، اس طرح کی سخت ترتیب وار ترتیب محدود اور کم مفید ہے۔ بہت سارے حالات ہیں جہاں کسی خاص شرط کی بنیاد پر کوڈ بلاک کا فیصلہ کرنا مفید ہے۔ ایسے حالات میں، کنٹرول فلو کوڈ کے ایک حصے سے کوڈ کے دوسرے حصے میں چھلانگ لگاتا ہے۔ پروگرام اسٹیٹ مینٹس جو اس طرح کے چھلانگوں کا سبب بنتے ہیں انہیں "کنٹرول فلو – Control flow" کہا جاتا ہے۔ یہ باب کنٹرول اسٹرکچر کی بنیادی باتوں سے متعلق ہے جیسے کہ "selection (انتخاب)"، "Iteration (اعادہ)" اور "Jump" اسٹیٹمنٹ ۔

10.2 اسٹیٹ منٹس
(statements)

کمپیوٹر پروگرام کسی مخصوص کام کو انجام دینے کے لیے اسٹیٹ مینٹس یا ہدایات کا مجموعہ ہے۔ ان اسٹیٹ مینٹس کا مقصد مخصوص کارروائی کرنا ہے۔ کارروائی متغیر اعلانات، اکسپریشن کی تشخیص، اسائنمنٹ آپریشنز، فیصلہ سازی، لوپنگ اور اسی طرح کی ہو سکتی ہے۔

C++ میں دو قسم اسٹیٹ منٹس کے استعمال ہوتے ہیں۔

(i) نل اسٹیٹمنٹ (Null statement)

(ii) کمپاؤنڈ اسٹیٹمنٹ (Compound statement)

10.2.1 نل اسٹیٹمنٹ

" نل یا خالی اسٹیٹمنٹ (null or empty statement)

" ایک اسٹیٹمنٹ ہے جس میں صرف ایک سیمی کولن ہوتا ہے۔ یہ ذیل کی شکل لیتا ہے:

```
; // it is a null statement
```

نل اسٹیٹمنٹ عام طور پر آئی ٹریشن اسٹیٹ مینٹس میں پلیس ہولڈرز (place holders) کے طور پر یا ایسے اسٹیٹ مینٹس کے طور پر استعمال ہوتے ہیں جن پر مرکب اسٹیٹ مینٹس یا فنکشنز کے آخر میں لیبل لگانا ہے۔

10.2.2 کمپاؤنڈ (بلاک) اسٹیٹمنٹ

C++ اسٹیٹمنٹ کے ایک گروپ کی اجازت دیتا ہے جو منحنی قوسین {} (braces) سے منسلک ہوتا ہے ۔ اسٹیٹ مینٹس کے اس گروپ کو کمپاؤنڈ اسٹیٹمنٹ یا بلاک کہا جاتا ہے۔

کمپاؤنڈ اسٹیٹمنٹ کی عمومی شکل یہ ہے:

```
{
        statement1;
        statement2;
        statement3;
}
```

For example

```
{
        int x, y;
        x = 10;
        y = x + 10;
}
```


11th_Comp Science_UM_Ch 10.indd 152 18-04-2023 15:18:32

کمپاؤنڈ اسٹیٹمنٹ یا بلاک کو ایک اکائی کے طور پر سمجھا جاتا ہے اور یہ پروگرام میں کہیں بھی ظاہر ہوسکتا ہے۔

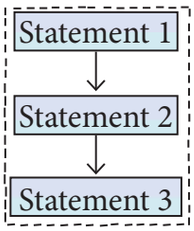
10.3 کنٹرول اسٹیٹ منٹس

کنٹرول اسٹیٹ مینٹس وہ اسٹیٹ مینٹس ہیں جو ہدایات کے جاری کرنے کی ترتیب کو تبدیل کرتے ہیں۔

ایک پروگرام میں، اسٹیٹ مینٹس کو ترتیب وار، منتخب یا اعادہ طور پر عمل میں لایا جا سکتا ہے۔ ہر پروگرامنگ زبانیں ترتیب، انتخاب (برانچنگ - branching) اور اعادہ کی حمایت کے لیے اسٹیٹ مینٹس فراہم کرتی ہیں۔

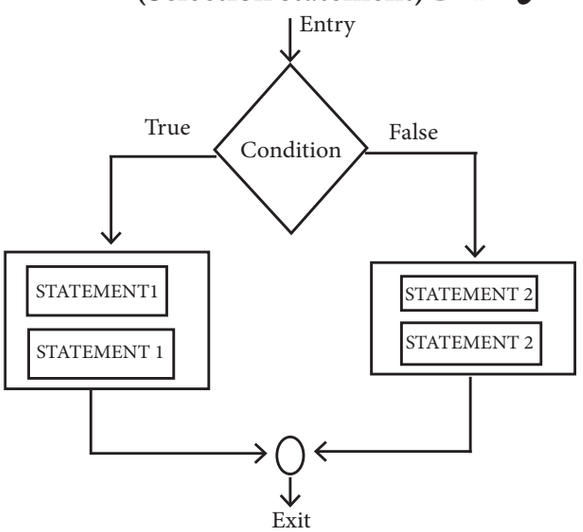
اگر اسٹیٹ مینٹس کو ترتیب وار عمل میں لایا جاتا ہے، تو فلو کو ترتیب فلو (sequential flow) کہا جاتا ہے۔ کچھ حالات میں، اگر اسٹیٹ مینٹس عملدرآمد کے فلو کو تبدیل کرتے ہیں جیسے برانچنگ (branching)، اعادہ (iteration)، جمپنگ (jumping) اور فنکشن کالز (function calls)، اس فلو کو کنٹرول فلو (control flow) کہا جاتا ہے۔

ترتیب وار اسٹیٹمنٹ (Sequence statement)



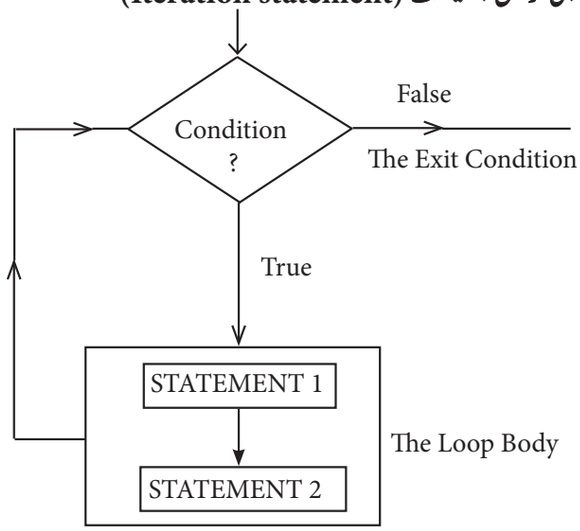
ترتیب وار اسٹیٹمنٹ وہ اسٹیٹ مینٹس ہیں، جو اوپر سے نیچے تک صرف ایک بار ایک کے بعد ایک عمل میں آتے ہیں۔ یہ اسٹیٹ مینٹس عملدرآمد کے فلو کو تبدیل نہیں کرتے ہیں۔ ان اسٹیٹ مینٹس کو ترتیب وار فلو اسٹیٹ مینٹس (sequential flow statement) کہا جاتا ہے۔ وہ ہمیشہ سیمی کولن (؛) کے ساتھ ختم ہوتے ہیں۔

سلیکشن اسٹیٹمنٹ (Selection statement)



سلیکشن سٹیٹمنٹ کا مطلب ہے کہ عمل میں لایا گیا اسٹیٹمنٹ کسی شرط پر منحصر ہے۔ اگر کوئی شرط درست ہے تو، ایک حقیقی بلاک (اسٹیٹ مینٹس کا ایک سیٹ) کو عمل میں لایا جائے گا، ورنہ ایک غلط بلاک کو عمل میں لایا جائے گا۔ اس اسٹیٹمنٹ کو ڈیسیژن اسٹیٹمنٹ (Decision statement) یا سلیکشن اسٹیٹمنٹ (selection statement) بھی کہا جاتا ہے کیونکہ یہ فیصلہ کرنے میں مدد کرتا ہے کہ اسٹیٹ مینٹس کے کون سے سیٹ پر عمل درآمد کیا جانا ہے۔

آئی ٹریشن اسٹیٹمنٹ (Iteration statement)



آئی ٹریشن اسٹیٹمنٹ وہ اسٹیٹمنٹ کا ایک مجموعہ ہے جو شرائط کی بنیاد پر دوہرایا جاتا ہے۔ اگر کسی شرط کی درست کی تشخیص ہوتی ہے تو، اسٹیٹ مینٹس کا سیٹ (ٹرو بلاک - True block) بار بار عمل میں آتا ہے۔ جیسے ہی شرط باطل ہو جاتی ہے، دوہراؤ رک جاتی ہے۔ اسے لوپنگ اسٹیٹمنٹ (looping statement) یا آئی ٹریشن اسٹیٹمنٹ (iteration statement) بھی کہا جاتا ہے۔

اسٹیٹ مینٹس کا مجموعہ جو بار بار انجام دیا جاتا ہے اسے لوپ کی باڈی (Body of the loop) کہا جاتا ہے۔ وہ حالت جس پر عمل درآمد یا لوپ سے باہر نکلنا ایگزٹ کنڈیشن (exit condition) یا ٹیسٹ کنڈیشن (test-condition) کہلاتا ہے۔

عام طور پر، تمام پروگرامنگ زبانیں اس قسم کے اسٹیٹ مینٹس کو پروگرام لکھنے کے لیے اس مسئلے کی بنیاد پر سپورٹ کرتی ہیں۔ C++ اس قسم کے اسٹیٹ مینٹس کی بھی حمایت کرتا ہے۔ ان اسٹیٹ مینٹس پر آنے والے حصوں میں بحث کی جائے گی۔

11th_Comp Science_UM_Ch 10.indd 153 18-04-2023 15:18:32

نوٹ

C++ میں، کسی بھی غیر صفر کو سچ سمجھا جاتا ہے بشمول منفی نمبر اور صفر کو غلط سمجھا جاتا ہے۔

سلیکشن اسٹیٹ مینٹس اور آئی ٹریشن اسٹیٹ مینٹس کنڈیشنل اکسپریشن (conditional expression) پر منحصر ہوتے ہیں۔ کنڈیشنل اکسپریشن صحیح یا غلط کا اندازہ کرتا ہے۔

10.4 سلیکشن اسٹیٹ منٹس

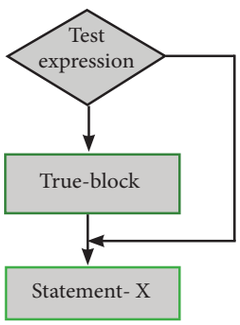
ایک پروگرام میں ایک فیصلہ ،پروگرام کے مختلف حصے میں ایک بار چھلانگ کا سبب بنتا ہے۔ C++ میں فیصلے کئی طریقوں سے کیے جاتے ہیں، سب سے اہم بات یہ ہے کہ if ... else ... statements جو دو متبادلوں میں سے انتخاب کرتا ہے۔ ایک اور ڈیسیژن اسٹیٹمنٹ سوئچ (decision statement switch)، ایک واحد متغیر کی قدر پر منحصر ہے، کوڈ کے متعدد متبادل حصوں کے لیے برانچس بناتا ہے۔

10.4.1 if اسٹیٹمنٹ

if سٹیٹمنٹ کسی شرط کا اندازہ لگاتا ہے، اگر شرط صحیح ہے تو پھر ایک صحیح بلاک (ایک اسیٹمنٹ یا اسٹیٹ مینٹس کا مجموعہ) عمل میں لایا جاتا ہے، بصورت دیگر صحیح بلاک کو چھوڑ دیا جاتا ہے۔ if اسٹیٹمنٹ کا عمومی سنٹیکس یہ ہے:

```
if (expression)
    true-block;
statement-x;
```

مندرجہ بالا سنٹیکس میں، if ایک کلیدی لفظ ہے جس میں اکسپریشن یا شرط ہونا چاہیے جو قوسین کے اندر بند ہو۔ اگر ایکسپریشن صحیح ہے تو (nonzero) تو پھر ٹرو بلاک (true block) کو عمل میں لایا جاتا ہے اور اس کے بعد سٹیٹمنٹ-x کو بھی عمل میں لایا جاتا ہے، ورنہ کنٹرول، اسٹیٹمنٹ-x کو جاتا ہے۔ ٹرو بلاک ایک اسٹیٹمنٹ، ایک کمپاؤنڈ اسٹیٹمنٹ (compound statement) یا ایمٹی اسٹیٹمنٹ (empty statement) پر مشتمل ہو سکتا ہے۔ if اسٹیٹمنٹ کا کنٹرول فلو اور متعلقہ فلو چارٹ ذیل میں دکھایا گیا ہے۔



مثال 10.1 C++ پروگرام یہ جانچنے کے لیے کہ آیا کوئی شخص if اسٹیٹمنٹ کا استعمال کرکے ووٹ دینے کا اہل ہے۔

```
#include <iostream>
using namespace std;
int main()
{
        int age;
        cout<< "\n Enter your age: ";
        cin>> age;
        if(age>=18)
                cout<< "\n You are eligible for voting ....";
        cout<< "This statement is always executed.";
        return 0;
}
```

The pair of braces is not required because if condition followed by only one statement

Output

```
Enter your age: 23
You are eligible for voting….
This statement is always executed.
```

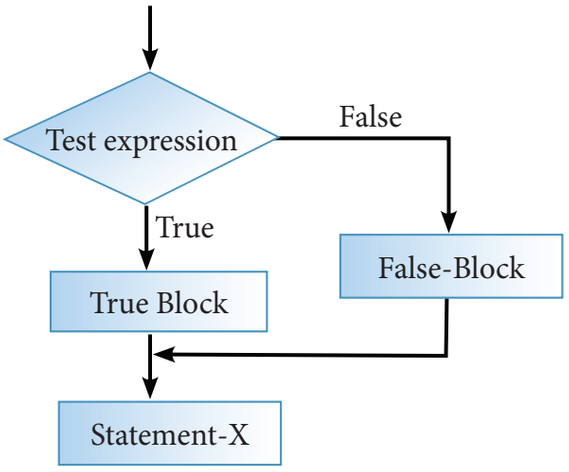

11th_Comp Science_UM_Ch 10.indd 154 18-04-2023 15:18:32

10.4.2 if-else اسٹیٹمنٹ

اوپر دی گئی if کی مثالوں میں، آپ نے دیکھا ہے کہ، اسٹیٹ مینٹس کا ایک بلاک صرف اس صورت میں عمل میں لایا جاتا ہے جب شرط کے درست ہونے کا اندازہ ہوتا ہے۔ کیا ہوگا اگر کوئی اور طریقہ کار اختیار کیا جائے اگر شرط غلط ہونے کا اندازہ لگاتی ہے۔ اس if کی ایک اور شکل ہے جو کسی اور شق فراہم کرکے اس قسم کی یا شرط کی اجازت دیتا ہے۔ if-else اسٹیٹمنٹ کا سنٹیکس ذیل میں دیا گیا ہے:

```
if ( expression)
{
        True-block;
}
else
{
        False-block;
}
Statement-x
```

if-else اسٹیٹمنٹ میں، سب سے پہلے اکسپریشن یا حالت کا اندازہ یا تو غلط یا صحیح پر کیا جاتا ہے۔ اگر نتیجہ درست ہے، تو صحیح بلاک کے اندر اسٹیٹ مینٹس کو انجام دیا جاتا ہے اور غلط بلاک کو چھوڑ دیا جاتا ہے۔ اگر نتیجہ غلط ہے، تو غلط بلاک کے اندر اسٹیٹمنٹ پر عمل درآمد کیا جاتا ہے، یعنی صحیح بلاک کو چھوڑ دیا جاتا ہے۔



مثال 10.2 if-else سٹیٹمنٹ کا استعمال کرتے ہوئے C++ پروگرام یہ معلوم کرنے کے لیے ہے کہ آیا دیا گیا نمبر جفت نمبر ہے یا طاق نمبر۔

```
#include <iostream>
using namespace std;
int main()
{
        int num, rem;
        cout<< "\n Enter a number: ";
        cin>>num;
        rem = num % 2;
        if (rem==0)
                cout<< "\n The given number" <<num<< " is Even";
        else
                cout<< "\n The given number "<<num<< " is Odd";
        return 0;
}
```

Output

```
Enter number: 10
The given number 10 is Even
```

مذکورہ پروگرام میں، دیئے گئے نمبر کا بقیہ حصہ ریم (rem) میں محفوظ کیا جاتا ہے۔ اگر ریم کی قدر صفر (zero) ہے، تو دیے گئے نمبر کو جفت نمبر کے طور پر نکالا جاتا ہے ورنہ، اس کا تخمینہ طاق نمبر پر لگایا جاتا ہے۔

11th_Comp Science_UM_Ch 10.indd 155 18-04-2023 15:18:32

10.4.3 نیسٹڈ if (Nested if)

ایک if اسٹیٹمنٹ جس میں ایک اور if اسٹیٹمنٹ ہو اسے نیسٹڈ if – nested if کہا جاتا ہے۔ نیسٹڈ کی درج ذیل تین شکلوں میں سے ایک ہو سکتی ہے۔

1. اگر نیسٹیڈ if part کے اندر ہو (If nested inside if part)
2. اگر نیسٹیڈ else part کے اندر ہو (If nested inside else part)
3. اگر نیسٹیڈ if part اور else part کے اندر ہو
(If nested inside both if part and else part)

The syntax of the nested if:

If nested inside if part

```
if (expression-1)
{
    if (expression-2)
    {
     True_Part_Statements;
    }
    else
    {
     False_Part_Statements;
    }
}
else
     body of else part;
```

If nested inside else part

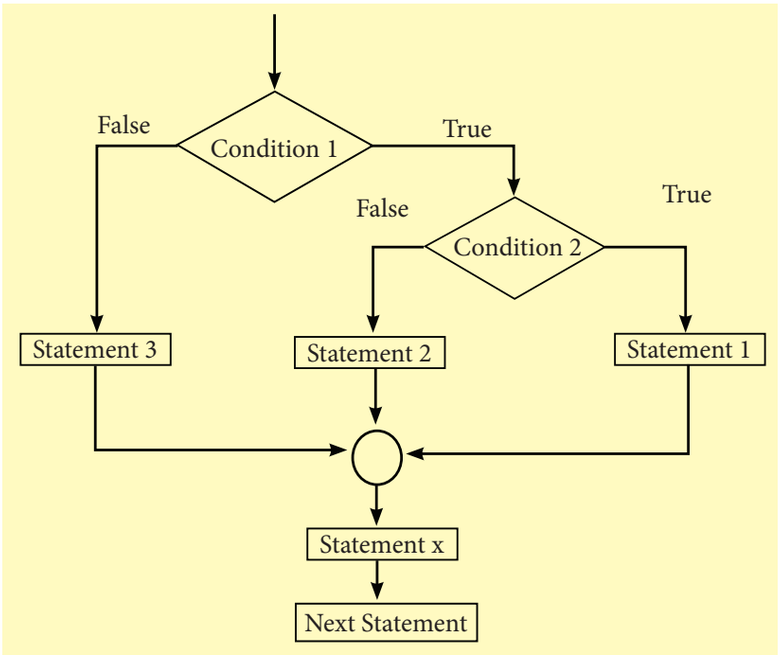
```
if (expression-1)
{
     body of true part;
}
else
{
     if (expression-2)
     {
      True_Part_Statements;
     }
     else
     {
      False_Part_Statements;
     }
}
```

If nested inside both if part and else part

```
if (expression)
{
      if (expression)
      {
       True_Part_Statements;
      }
      else
      {
       False_Part_Statements;
      }
}
else
{
      if (expression)
      {
       True_Part_Statements;
      }
      else
      {
       False_Part_Statements;
      }
}
```

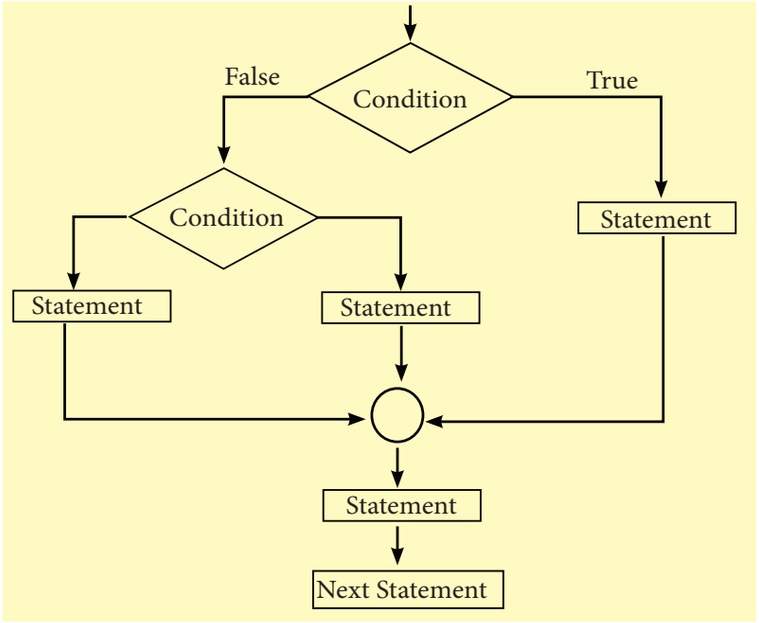
نیسٹڈ کے پہلے سنٹیکس میں اگر ایکسپریشن- 1 کا اوپر ذکر کیا گیا ہے اور ایکسپریشن کا نتیجہ غلط ہے تو کنٹرول اسٹیٹمنٹ-m کو جاتا ہے۔ ورنہ، ایکسپریشن -2 کا جائزہ لیا جاتا ہے، اگر شرط درست ہے، تو Nested-True-block کو عمل میں لایا جاتا ہے، اگلی سٹیٹمنٹ-n کو بھی عمل میں لایا جاتا ہے۔ ورنہ Statement-n، Nested-False-Block اور Statement-m کو عمل میں لایا جاتا ہے۔

مذکورہ بالا کے کام کا طریقہ کار if ... else کا خاکہ ذیل میں فلو چارٹ (flow chart) کے طور پر دیا گیا ہے:

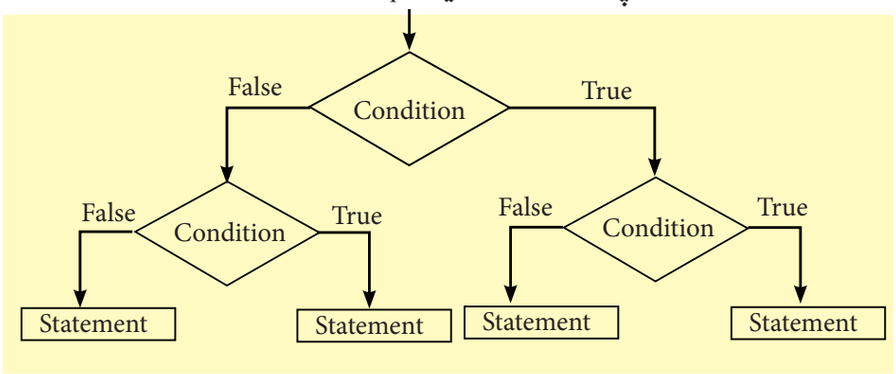


11th_Comp Science_UM_Ch 10.indd 156 18-04-2023 15:18:32

فلو چارٹ 10.1 اگر نیسٹیڈ if part کے اندر ہو



فلو چارٹ 10.2 اگر نیسٹیڈ else part کے اندر ہو



فلو چارٹ 10.3 اگر نیسٹیڈ if part اور else part کے اندر ہو

مثال 10.3 – nested if اسٹیٹمنٹ کا استعمال کرتے ہوئے گریڈ کے مطابق کمیشن کا حساب لگانے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int sales, commission;
        char grade;
        cout << "\n Enter Sales amount: ";
        cin >> sales;
        cout << "\n Enter Grade: ";
        cin >> grade;
        if (sales > 5000)
        {
                commission = sales * 0.10;
                cout << "\n Commission: " << commission;
        }
```


11th_Comp Science_UM_Ch 10.indd 157 18-04-2023 15:18:32

```
        else
        {
                commission = sales * 0.05;
                cout << "\n Commission: " << commission;
        }
        cout << "\n Good Job ..... ";
        return 0;
}
```

Output:

```
Enter Sales amount: 6000
Enter Grade: A
Commission: 600
Good Job .....
```

10.4.4 اف- ایلس – اف لیڈر (if – else – if ladder)

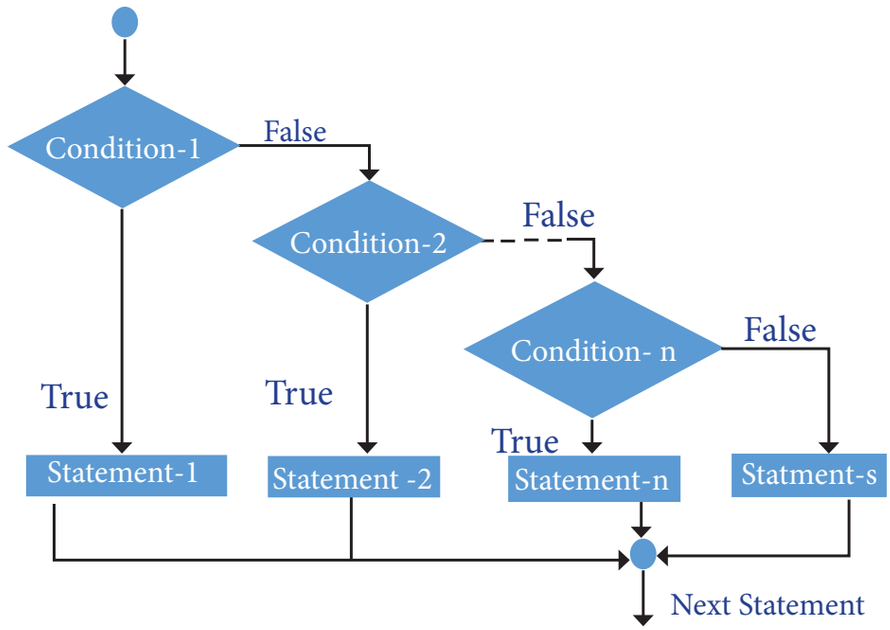
if-else لیڈر ایک کثیر راستہ فیصلہ سازی کا اسٹیٹمنٹ ہے۔ اس قسم کے اسٹیٹمنٹ میں 'if' کے بعد ایک یا ایک سے زیادہ if-else ہوتا ہے اور آخر میں else پر ختم ہوتا ہے۔

if-else لیڈر کا سنٹیکس:

```
if (expression 1)
{
  Statement-1
}
else
        if( expression 2)
        {
                Statement-2
        }
        else
                if ( expression 3)
                {
                        Statement-3
                }
                else
                {
                        Statement-4
                }
```

جب متعلقہ اکسپریشن درست ہو جاتا ہے، تو بلاک کے ساتھ منسلک اسٹیٹمنٹ پر عمل درآمد کیا جاتا ہے، اور باقی لیڈر(ladder) کو نظرانداز کر دیا جاتا ہے۔ اگر ان میں سے کوئی بھی شرط درست نہیں ہے، تو آخریels e اسٹیٹمنٹ پر عمل کیا جائے گا۔

11th_Comp Science_UM_Ch 10.indd 158 18-04-2023 15:18:32



فلو چارٹ 10.4 if-else لیڈر کا فلو چارٹ

مثال 10.4 if-else لیڈر کا استعمال کرتے ہوئے اپنا گریڈ تلاش کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int marks;
cout<<" Enter the Marks :";
cin>>marks;
if( marks >= 60 )
        cout<< "Your grade is 1st class !!" <<endl;
                else if( marks >= 50 && marks < 60)
                        cout<< "your grade is 2nd class !!" <<endl;
                                else if( marks >= 40 && marks < 50)
                                        cout<< "your grade is 3rd class !!" <<endl;
else
        cout<< "You are fail !!" <<endl;
return 0;
}
```

Output

```
Enter the Marks :60
Your grade is 1st class !!
```

جب نمبر 60 سے زیادہ یا اس کے برابر ہوں تو پیغام "آپ کا گریڈ 1st کلاس ہے!!" ظاہر ہوتا ہے اور باقی لیڈر کو نظرانداز کیا جاتا ہے۔ جب نمبر 50 اور 59 کے درمیان ہوں تو پیغام "آپ کا گریڈ 2nd کلاس ہے!!" ظاہر ہوتا ہے، اور دوسری لیڈر کو نظرانداز کیا جاتا ہے۔ جب نمبر 40 سے 49 کے درمیان ہوتا ہے تو پیغام آتا ہے "آپ کا گریڈ 3rd کلاس ہے!!" ظاہر ہوتا ہے، بصورت دیگر، پیغام "آپ ناکام ہیں!!" ظاہر کیا جاتا ہے

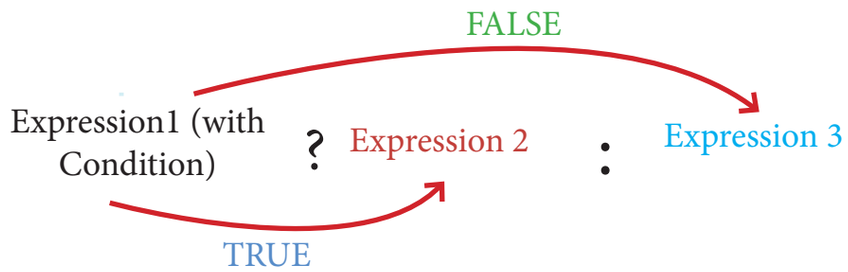
11th_Comp Science_UM_Ch 10.indd 159 18-04-2023 15:18:32

10.4.5 دی؟: اف ایلس کا متبادل ہے (The ?: Alternative to if – else)

کنڈیشنل آپریٹر (یا ٹرنری آپریٹر) 'if else' اسٹیٹمنٹ کا متبادل ہے۔ کنڈیشنل آپریٹر جو دو علامتوں (?:) پر مشتمل ہوتا ہے۔ یہ تین آرگیومنٹس لیتا ہے۔کنڈیشنل آپریٹر کا کنٹرول فلو ذیل میں دکھایا گیا ہے:

کنڈیشنل آپریٹر کا سنٹیکس ہے:

```
expression 1? expression 2 : expression 3
```



مندرجہ بالا سنٹیکس میں،اکسپریشن 1 ایک شرط ہے جس کا اندازہ کیا جاتا ہے، اگر شرط درست ہے (نان صفر – Non zero)، تو کنٹرول ایکسپریشن 2 میں منتقل ہو جاتا ہے، ورنہ، کنٹرول ایکسپریشن 3 میں چلا جاتا ہے۔

مثال 10.5 – کنڈیشنل آپریٹر کا استعمال کرتے ہوئے دو نمبروں میں سے سب سے زیادہ تلاش کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int a, b, largest;
        cout << "\n Enter any two numbers: ";
        cin >> a >> b;
         largest = (a>b)? a : b;
        cout << "\n Largest number : " << largest;
        return 0;
}
```

Output:

```
 Enter any two numbers: 12 98
 Largest number : 98
```

10.4.6 سوئچ اسٹیٹمنٹ (Switch statement)

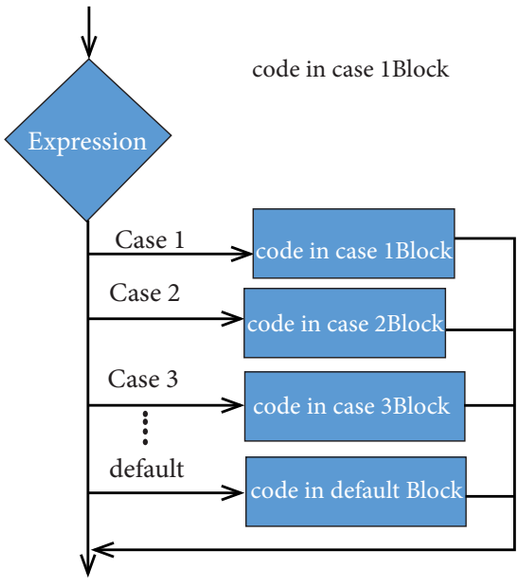
سوئچ اسٹیٹمنٹ ایک ملٹی وے برانچ اسٹیٹمنٹ (muti-way branch statement) ہے۔ یہ ایکسپریشن کی قدر کی بنیاد پر کوڈ کے مختلف حصوں پر عمل درآمد بھیجنے کا ایک آسان طریقہ فراہم کرتا ہے۔ سوئچ اسٹیٹمنٹ متعدد if-else ترتیب کی جگہ لے لیتا ہے۔

11th_Comp Science_UM_Ch 10.indd 160 18-04-2023 15:18:32

سوئچ اسٹیٹمنٹ کا سنٹیکس ہے؛

```
switch(expression)
{
        case constant 1:
                statement(s);
                break;
        case constant 2:
                statement(s);
                break;
        .
        .
        .
        .
        default:
                statement(s);
}
```

مندرجہ بالا سنٹیکس میں، اکسپریشن کا اندازہ لگایا جاتا ہے اور اگر اس کی قدر کسی ایک کیس سٹیٹمنٹ میں متعین مستقل قدر سے مماثل ہوتی ہے، تو اسٹیٹ مینٹس کے متعلقہ سیٹ کو عمل میں لایا جاتا ہے۔ ورنہ، ڈیفالٹ آپشن کے تحت اسٹیٹ مینٹس پر عمل درآمد کیا جاتا ہے۔ سوئچ اسٹیٹمنٹ اور فلو چارٹ کا ورک فلو (workflow) نیچے دکھایا گیا ہے۔



فلو چارٹ 10.5: سوئچ اور فلو چارٹ کا ورک فلو

اصول:

1. سوئچ میں فراہم کردہ اکسپریشن کے نتیجے میں ایک مستقل قدر ہونی چاہیے ورنہ یہ درست نہیں ہوگا۔
2. نقلی کیس ویلیوز(duplicate case values) کی اجازت نہیں ہے۔
3. پہلے سے طے شدہ اسٹیٹمنٹ اختیاری ہے۔
4. بریک اسٹیٹمنٹ (Break statement)اسٹیٹمنٹ کی ترتیب کو ختم کرنے کے لیے سوئچ کے اندر استعمال کیا جاتا ہے۔ جب بریک اسٹیٹمنٹ تک پہنچ جاتا ہے، تو سوئچ ختم ہوجاتا ہے، اور کنٹرول فلو سوئچ اسٹیٹمنٹ کے بعد اگلی لائن پر چھلانگ لگا دیتا ہے

11th_Comp Science_UM_Ch 10.indd 161 18-04-2023 15:18:32

5. بریک اسٹیٹمنٹ اختیاری ہے۔ اگر چھوڑ دیا گیا تو، اگلے کیس میں عمل جاری رہے گا۔ بریک تک پہنچنے تک کنٹرول فلو بعد کے آنے والے کیسس (cases) میں گر جائے گا۔

6. سوئچ اسٹیٹ منٹس کے نیسٹنگ (nesting) کی بھی اجازت ہے۔

مثال 10.6 – سوئچ اسٹیٹمنٹ کو ظاہر کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int num;
        cout << "\n Enter week day number: ";
        cin >> num;
        switch (num)
        {
                case 1 : cout << "\n Sunday"; break;
                case 2 : cout << "\n Monday"; break;
                case 3 : cout << "\n Tuesday"; break;
                case 4 : cout << "\n Wednessday"; break;
                case 5 : cout << "\n Thursday"; break;
                case 6 : cout << "\n Friday"; break;
                case 7 : cout << "\n Saturday"; break;
                default: cout << "\n Wrong input....";
        }
}
```

Output:

```
Enter week day number: 6
Friday
```

10.4.7 سوئچ بمقابلہ if-else (Switch vs if-else)

"if-else" اور "switch" دونوں انتخابی اسٹیٹ مینٹس ہیں۔ انتخابی اسٹیٹ مینٹس، پروگرام کے فلوکو اسٹیٹ مینٹس کے مخصوص بلاک میں منتقل کرتے ہیں اس کی بنیاد پر کہ آیا شرط " صحیح" ہے یا "غلط"۔ تاہم، ان کی کارروائیوں میں کچھ اختلافات ہیں جو ذیل میں دیے گئے ہیں:

If-else اور سوئچ کے درمیان کلیدی فرق

شمار نمبر	if-else	سوئچ (Switch)
1	if اسٹیٹمنٹ کے اندر اکسپریشن یہ فیصلہ کرتا ہے کہ آیا if بلاک کو عمل میں لانا ہے یا انڈر بلاک کو۔	سوئچ اسٹیٹمنٹ کے اندر اکسپریشن یہ فیصلہ کرتا ہے کہ کون سا کیس عمل میں لانا ہے۔
2	ایک if – else اسٹیٹمنٹ متعدد انتخاب کے لیے متعدد اسٹیٹ مینٹس کا استعمال کرتا ہے۔	سوئچ اسٹیٹمنٹ ایک سے زیادہ انتخاب کے لیے واحد اکسپریشن کا استعمال کرتا ہے۔
3	if – else اسٹیٹمنٹ مساوات کے ساتھ ساتھ منطقی اکسپریشن کے لیے بھی چیک کرتا ہے۔	صرف برابری کے لیے چیک کو تبدیل کریں۔
4	if اسٹیٹمنٹ انٹیجر، کریکٹر، پوائنٹر یا فلوٹنگ پوائنٹ ٹائپ یا بولین ٹائپ کا اندازہ کرتا ہے۔	سوئچ اسٹیٹمنٹ صرف کریکٹر یا انٹیجر ڈیٹا کی قسم کا اندازہ کرتا ہے۔
5	اگر شرط غلط ہے تو باقی بلاک اسٹیٹ مینٹس پر عمل درآمد کیا جائے گا۔	اگر شرط غلط ہے تو پہلے سے طے شدہ اسٹیٹ مینٹس پر عمل درآمد کیا جاتا ہے.

11th_Comp Science_UM_Ch 10.indd 162 18-04-2023 15:18:32

if اسٹیٹمنٹ سوئچ اسٹیٹمنٹ سے زیادہ لچکدار ہے۔

10.5 آئی ٹریشن اسٹیٹ مینٹس (Iteration statements)

ایک آئی ٹریشن (یا لوپنگ) ایک یا زیادہ اسٹیٹ مینٹس کا ایک سلسلہ ہے جو بار بار اس وقت تک عمل میں لایا جاتا ہے جب تک کہ کوئی شرط پوری نہ ہوجائے۔ ان اسٹیٹ مینٹس کو کنٹرول فلو اسٹیٹ مینٹس (control flow statements) بھی کہا جاتا ہے۔ یہ کوڈ کی لمبائی کو کم کرنے، وقت کو کم کرنے، پروگرام کو عمل میں لانے کے لیے استعمال کیا جاتا ہے اور میموری کی جگہ کم لیتی ہے۔ C++ تین قسم کے آئی ٹریشن اسٹیٹ مینٹس کی حمایت کرتا ہے۔

- فار اسٹیٹمنٹ (for statement)
- وائل اسٹیٹمنٹ (while statement)
- ڈو وائل اسٹیٹمنٹ (do-while statement)

تمام لوپنگ سٹیٹ منٹس ایک سیٹ سٹیٹ منٹس کو دہراتے ہیں جب تک کہ ایک مخصوص شرط صحیح رہتی ہے۔ مخصوص حالت کو لوپ کنٹرول کہا جاتا ہے۔ تینوں لوپ اسٹیٹ منٹس کے لیے، ایک صحیح حالت کوئی بھی غیر صفر قدر ہے اور صفر کی قدر غلط حالت کو ظاہر کرتی ہے۔

10.5.1 لوپ کے حصے (Parts of the Loop)

ہر لوپ میں چار عناصر ہوتے ہیں جو مختلف مقاصد کے لیے استعمال ہوتے ہیں۔ یہ عناصر ہیں۔

- انی شیلائزیشن اکسپریشن (Initialisation expression)
- ٹیسٹ اکسپریشن (Test expression)
- اپ ڈیٹ اکسپریشن (update expression)
- لوپ کی باڈی (The body of the loop)

انی شیلائزیشن اکسپریشن: کنٹرول لوپ میں داخل ہونے سے پہلے کنٹرول متغیر کو شروع کرنا ضروری ہے۔ کنٹرول متغیر کا آغاز انی شیلائزیشن اکسپریشن کے تحت ہوتا ہے۔ انی شیلائزیشن اکسپریشن صرف ایک بار لوپ کے آغاز میں انجام دیا جاتا ہے۔

ٹیسٹ اکسپریشن: ٹیسٹ ایکسپریشن ایک ایسا اکسپریشن یا حالت ہے جس کی قدر یہ فیصلہ کرتی ہے کہ لوپ باڈی کو عمل میں لایا جائے گا یا نہیں۔ اگر اکسپریشن درست (یعنی، 1) پر تشخیص کرتا ہے، تو لوپ کی باڈی پر عمل درآمد ہو جاتا ہے، ورنہ لوپ ختم ہو جاتا ہے۔ انٹری کنٹرولڈ لوپ (entry-controlled loop) میں، لوپ میں داخل ہونے سے پہلے ٹیسٹ-اکسپریشن کی جانچ کی جاتی ہے جبکہ ایگزٹ-کنٹرولڈ لوپ (exit-controlled loop) میں، ٹیسٹ-اکسپریشن کی جانچ لوپ سے باہر نکلنے سے پہلے کی جاتی ہے۔

اپ ڈیٹ اکسپریشن : یہ لوپ متغیر کی قدر کو تبدیل کرنے کے لیے استعمال ہوتا ہے۔ اس اسٹیٹمنٹ کو لوپ کے آخر میں عمل میں لایا جاتا ہے جب لوپ کی باڈی کو عمل میں لایا جاتا ہے۔

لوپ کی باڈی : ایک اسٹیٹمنٹ یا اسٹیٹ مینٹس کا مجموعہ لوپ کی ایک باڈی بناتا ہے جسے بار بار انجام دیا جاتا ہے۔ اندراج پر قابو پانے والے لوپ میں، پہلے ٹیسٹ کے اکسپریشن کی جانچ کی جاتی ہے اور اگر یہ غیر صفر ہے، تو لوپ کی باڈی کو عمل میں لایا جاتا ہے بصورت دیگر لوپ کو ختم کر دیا جاتا ہے۔ ایگزٹ کنٹرولڈ لوپ میں، لوپ کے باڈی کو پہلے عمل میں لایا جاتا ہے پھر ٹیسٹ ایکسپریشن کا جائزہ لیا جاتا ہے۔ اگر ٹیسٹ-اکسپریشن درست ہے تو لوپ کی باڈی کو دہرایا جاتا ہے ورنہ لوپ ختم ہو جاتا ہے۔

10.5.2 فار لوپ (for loop)

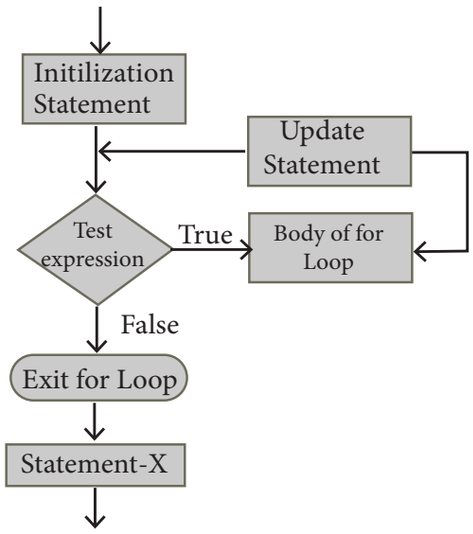
فار لوپ انٹری کنٹرولڈ لوپ (entry-controlled loop) ہے اور یہ سب سے آسان لوپنگ اسٹیٹمنٹ ہے جو کوڈ کو بار بار عمل میں لانے کی اجازت دیتا ہے۔ اس میں تین مختلف اسٹیٹ مینٹس (انی شیلائزیشن، کنڈیشن یا ٹیسٹ-اکسپریشن اور اپ ڈیٹ اکسپریشنس کو سیمی کولنز (semicolons) سے الگ کیا گیا ہے۔

عمومی سنٹیکس ہے: (The general syntax is:)

```
for (initialization(s); test-expression; update expression(s))
{
        Statement 1;
        Statement 2;
        ..............
}
Statement-x;
```


11th_Comp Science_UM_Ch 10.indd 163 18-04-2023 15:18:32

انی شیلائزیشن حصے کا استعمال متغیرات کو شروع کرنے یا متغیر کا اعلان کرنے کے لیے کیا جاتا ہے جو صرف ایک بار عمل میں آتا ہے، پھر کنٹرول ٹیسٹ-اکسپریشن پر گزر جاتا ہے۔ ٹیسٹ-اکسپریشن کی تشخیص کے بعد، اگر نتیجہ غلط ہے، تو کنٹرول اسٹیٹمنٹ-x میں منتقل ہو جاتا ہے۔ اگر نتیجہ درست ہے تو، لوپ کی باڈی کو عمل میں لایا جاتا ہے، اس کے بعد کنٹرول کو اپ ڈیٹ ایکسپریشن میں منتقل کیا جاتا ہے۔ اپ ڈیٹ ایکسپریشن حصے کی تشخیص کے بعد، کنٹرول ٹیسٹ ایکسپریشن حصے میں منتقل کر دیا جاتا ہے۔ اگلا مرحلہ 3 سے 5 دہرایا جاتا ہے۔ لوپ اور فلو چارٹ کے لیے ورک فلو ذیل میں دکھایا گیا ہے۔



فلو چارٹ 10.6: لوپ اور فلو چارٹ کا ورک فلو

مثال 10.7 لوپ کا استعمال کرتے ہوئے 0 سے 9 تک نمبروں کو ظاہر کرنے کے لیے C++ پروگرام

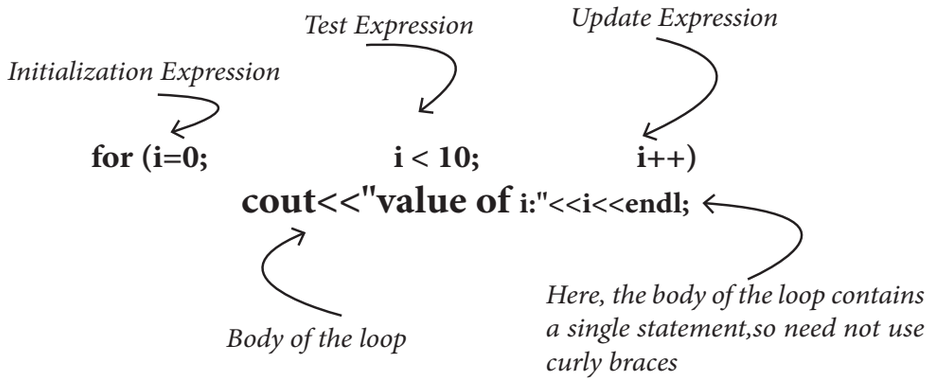
```
#include <iostream>
using namespace std;
int main ()
{
int i;
for(i = 0; i< 10; i ++ )
        cout<< "value of i : " <<i<<endl;
return 0;
}
```

Output

```
value of i : 0
value of i : 1
value of i : 2
value of i : 3
value of i : 4
value of i : 5
value of i : 6
value of i : 7
value of i : 8
value of i : 9
```


11th_Comp Science_UM_Ch 10.indd 164 18-04-2023 15:18:33

درج ذیل لائنیں لوپ کے لیے اوپر دیے گئے کام کو اسٹیٹمنٹ کرتی ہیں:



مندرجہ بالا پروگرام میں، پہلے متغیر i کو شروع کیا جاتا ہے، اس کے بعد i کا موازنہ 10 سے کیا جاتا ہے، اگر i دس سے کم ہو تو i کی قدر میں اضافہ ہوتا ہے۔ اس طرح 0 سے 9 تک کے اعداد ظاہر ہوتے ہیں۔ ایک بار i جب 10 ہو جاتا ہے، تو یہ 10 > نہیں رہتا۔ لہذا، کنٹرول فار لوپ سے باہر آجاتا ہے۔

مثال 10.8 فار لوپ کا استعمال کرتے ہوئے 1 سے 10 تک نمبروں کے مجموعہ کا C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int i,sum=0;
for(i=1; i<=10;i++)
 {
        sum=sum+i;
 }
cout<<"The sum of 1 to 10 is "<<sum;
return 0;
}
```

Output

The sum of 1 to 10 is 55

فار لوپ کے تغیرات (variations of for loop)

for، C++ میں سب سے اہم لوپنگ اسٹیٹمنٹ میں سے ایک ہے کیونکہ یہ متعدد تغیرات کی اجازت دیتا ہے۔ یہ تغیرات for loop کے لچک اور لاگو ہونے میں اضافہ کرتے ہیں۔ ان تغیرات پر ذیل میں اظہارِ خیال کیا جائے گا:

متعدد انی شیلائیزیشن اور متعدد اپ ڈیٹ اکسپریشنس

فار لوپ کے ابتدائی اور اپ ڈیٹ ایکسپریشنس میں متعدد اسٹیٹ مینٹس استعمال کیے جا سکتے ہیں۔ یہ متعدد انی شیلائیزیشن اور متعدد اپ ڈیٹ اکسپریشنس کوماس (commas) کے ذریعہ الگ کیے گئے ہیں۔ مثال کے طور پر،

```
#include<iostream>
using namespace std;
int main()
{
        int i, j;
            for(i=0, j=10 ;  i<j  ;  i++,j--)
            {
            cout<<"\nThe value of i is"<<i<<" The value of j is "<<j;
```

Multiple initialization expressions (separated by commas)

Multiple update expressions (separated by commas)

11th_Comp Science_UM_Ch 10.indd 165 18-04-2023 15:18:33

```
}
return 0;
}
```
Output
```
The value of i is 0  The value of j is 10
The value of i is 1  The value of j is 9
The value of i is 2  The value of j is 8
The value of i is 3  The value of j is 7
The value of i is 4  The value of j is 6
```

مندرجہ بالا مثال میں، ابتدائی حصے میں دو متغیر i اور j ہیں اور اپ ڈیٹ ایکسپریشن i++ اور j++ پر مشتمل ہے۔ یہ دونوں متغیرات کوما کے ذریعے الگ کیے گئے ہیں جو کہ ترتیب وار ترتیب میں عمل میں آتے ہیں، یعنی ابتدا کے دوران پہلے i=0 اس کے بعد j=10۔ اسی طرح، اپ ڈیٹ ایکسپریشن میں، سب سے پہلے ++ i کی تشخیص کی جاتی ہے اس کے بعد ++ j کی تشخیص کی جاتی ہے۔

پوسٹ فکس پر پریفکس آپریٹر کو ترجیح

عام طور پر، اپ ڈیٹ ایکسپریشن میں انکریمنٹ/ڈیکریمنٹ (increment / decrement) آپریٹر ہوتا ہے۔ (++ یا --)۔

اس حصے میں، جب اکیلے استعمال کیا جائے تو ہمیشہ پوسٹ فکس پر پریفکس انکریمنٹ/ڈیکریمنٹ آپریٹر کو ترجیح دیں۔ اس کے پیچھے وجہ یہ ہے کہ جب اکیلے استعمال کیا جاتا ہے تو، پریفکس آپریٹرز کو پوسٹ فکس سے زیادہ تیزی سے انجام دیا جاتا ہے۔

آپشنل اکسپریشن (Optional Expressions)

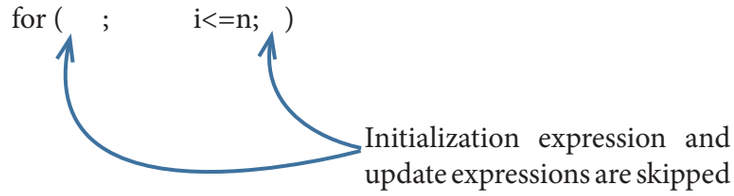
عام طور پر، لوپ کے لیے تین حصے ہوتے ہیں، انی شیلائزیشن اکسپریشن، ٹیسٹ اکسپریشن، اپ ڈیٹ اکسپریشن ۔ یہ تینوں اکسپریشنس ایک فار لوپ (for loop) میں اختیاری (optional) ہیں۔

مثال 10.9 1 سے n تک نمبروں کا مجموعہ کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int i, sum=0, n;
cout<<"\n Enter The value of n";
cin>>n;
i =1;
for ( ; i<=n; )
 {
        sum += i;
        ++i;
 }
cout<<"\n The sum of 1 to " <<n<<"is "<<sum;
return 0;
}
```
Output
```
Enter the value of n 5
The sum of 1 to 5 is 15
```


11th_Comp Science_UM_Ch 10.indd 166 18-04-2023 15:18:33

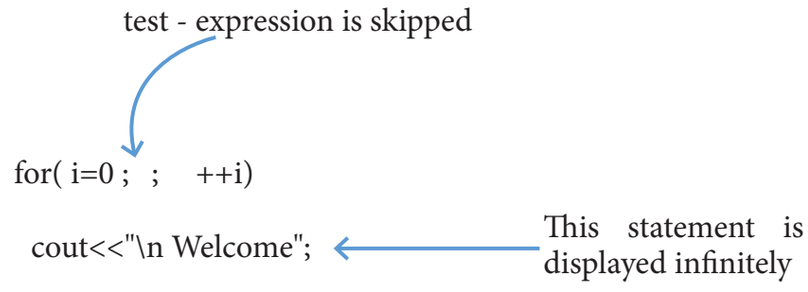
مندرجہ بالا کوڈ میں، اپ ڈیٹ اکسپریشن نہیں دیا گیا ہے، لیکن اپ ڈیٹ اکسپریشن سے پہلے ایک سیمی کولن ضروری ہے۔



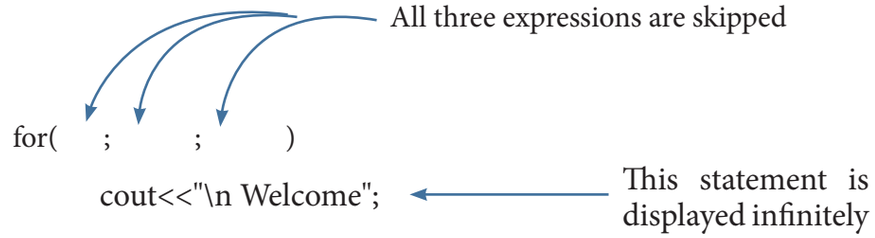
مندرجہ بالا کوڈ میں،فار لوپ (for loop) میں نہ تو ابتدا اور نہ ہی اپ ڈیٹ کا اکسپریشن دیا گیا ہے۔ اگر دونوں یا کوئی ایک اکسپریشن موجود نہ ہو تو کنٹرول مشروط حصے (conditional part) میں منتقل ہوجاتا ہے۔

لامحدود لوپ (infinite loop)

ایک لامحدود لوپ(infinite loop) بن جائے گا اگر ایک فار لوپ میں ٹیسٹ-اکسپریشن موجود نہ ہو۔ مثال کے طور پر

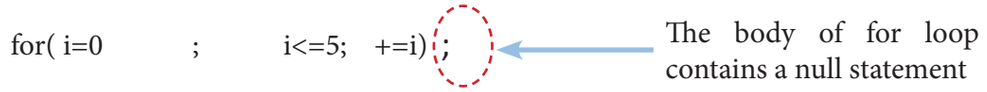


اسی طرح، فار لوپ درج ذیل کو بھی ایک لامحدود لوپ بناتا ہے۔



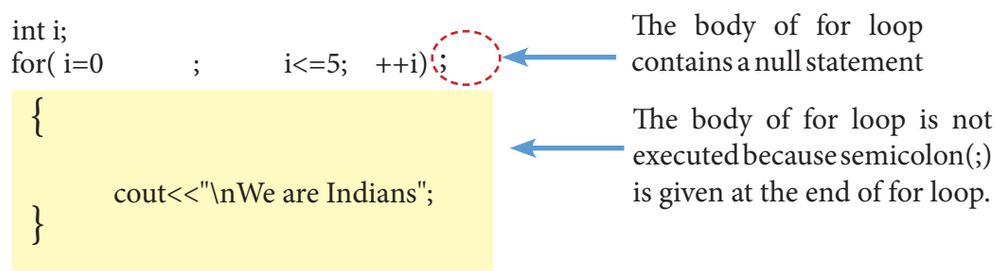
خالی لوپ (Empty loop)

خالی لوپ ایک لوپ کا مطلب ہے جس کی باڈی میں کوئی اسٹیٹمنٹ نہیں ہے اسے خالی لوپ کہتے ہیں۔ مندرجہ ذیل فار لوپ ایک خالی لوپ ہے۔



مندرجہ بالا کوڈ میں، for loop میں ایک نَل اسٹیٹمنٹ ہے، یہ ایک خالی لوپ ہے۔

اسی طرح، فار لوپ مندرجہ ذیل میں بھی ایک خالی لوپ ہے۔



11th_Comp Science_UM_Ch 10.indd 167 18-04-2023 15:18:33

مندرجہ بالا کوڈ میں، فار لوپ کی باڈی کو عمل میں نہیں لایا جاتا جو منحنی قوسین میں موجود ہے ، کیونکہ فار لوپ کے بعد ایک سیمی کولن دیا جاتا ہے۔

فار لوپ میں متغیر کا ڈکلریشن (Declaration of variable in a for loop)

C++ میں، متغیرات کا ڈکلریشن فار لوپ کے اندر بھی کیا جا سکتا ہے۔ مثال کے طور پر،

```
int main ()
{
   int sum = 0;
   for(int i=0;  i<=5;  ++i )
   {
         sum = sum + i;
   }
   cout<<"\nThe variable i cannot be accessed here";
   cout<<"\n The variable sum can be accessed here";
}
```

Variable (i)is declared within the for loop.

The variable i can be accessed only within the body of loop.

main() کے بلاک کے اندر اعلان کردہ ایک متغیر کو main() کے اندر کہیں بھی رسائی حاصل کی جا سکتی ہے یعنی main() میں متغیر کا دائرہ۔

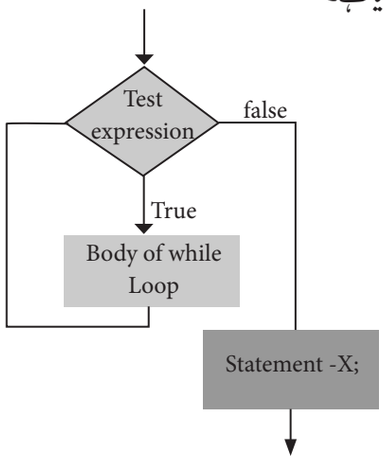
10.5.3 وائل لوپ (While loop)

ایک وائل لوپ ایک کنٹرول فلو اسٹیٹمنٹ ہے جو لوپ اسٹیٹ منٹس کو اس وقت تک عمل میں لانے کی اجازت دیتا ہے جب تک کہ حالت درست ہو۔ وائل لوپ ایک انٹری کنٹرولڈ لوپ (entry-controlled loop) ہے کیونکہ لوپ میں داخل ہونے سے پہلے ٹیسٹ ایکسپریشن (test – expression) کا جائزہ لیا جاتا ہے۔

وائل لوپ کا سنٹیکس یہ ہے۔

```
while ( Test expression )
{
        Body of the loop;
}
Statement-x;
```

وائل لوپ کا کنٹرول فلو اور فلو چارٹ نیچے دکھایا گیا ہے۔



فلو چارٹ 10.7: وائل لوپ کنٹرول فلو اور فلو چارٹ

11th_Comp Science_UM_Ch 10.indd 168 18-04-2023 15:18:33

وائل لوپ میں، ٹیسٹ ایکسپریشن کا جائزہ لیا جاتا ہے اور اگر ٹیسٹ ایکسپریشن کا نتیجہ درست ہے، تو لوپ کی باڈی کو عمل میں لایا جاتا ہے اور دوبارہ کنٹرول کووائل لوپ میں منتقل کر دیا جاتا ہے۔ جب ٹیسٹ ایکسپریشن کا نتیجہ غلط ہوتا ہے تو کنٹرول اسٹیٹمنٹ-x میں منتقل ہو جاتا ہے۔

مثال 10.10 وائل لوپ کا استعمال کرتے ہوئے 1 سے 10 تک نمبروں کے مجموعہ کا C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int i=1,sum=0;
while(i<=10)
{
        sum=sum+i;
        i++;
}
cout<<"The sum of 1 to 10 is "<<sum;
return 0;
}
```

Output

```
The sum of 1 to 10 is 55
```

مندرجہ بالا پروگرام میں، انٹیجر متغیر i کو 1 سے اور متغیر مجموعہ کو 0 سے شروع کیا گیا ہے۔وائل لوپ کنڈیشن کو چیک کرتا ہے، i<10، اگر کنڈیشن درست ہے تو i کی قدر، جو کہ رقم میں شامل کی جاتی ہے اور i کو 1 سے بڑھایا جاتا ہے۔ دوبارہ، شرط i<10 کو چیک کیا جاتا ہے۔ چونکہ 2<10، 2 کو رقم کی پہلے والی قدر میں شامل کیا جاتا ہے۔ یہ اس وقت تک جاری رہتا ہے جب تک i، 11 نہیں ہو جاتا۔ اس وقت، 11<10 کا اندازہ غلط ہوتا ہے اور وائل لوپ ختم ہو جاتا ہے۔ لوپ ختم ہونے کے بعد، رقم کی قدر ظاہر ہوتی ہے۔

مثال 10.11 لوپ کا استعمال کرتے ہوئے 5 نمبروں کا مجموعہ اور اوسط تلاش کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int i=1,num,avg,sum=0;
while(i<=5)
{
        cout<<"Enter the number : ";
        cin>>num;
        sum=sum+num;
        i++;
}
avg=sum/5;
cout<<"The sum is "<<sum<<endl;
cout<<"The average is "<<avg;
return 0;
}
```

Output

```
Enter the number : 1
Enter the number : 2
Enter the number : 3
Enter the number : 4
Enter the number : 5
The sum is 15
The average is 3
```


11th_Comp Science_UM_Ch 10.indd 169 18-04-2023 15:18:33

مندرجہ بالا پروگرام میں، عددی متغیرات num اور avg کا اعلان کیا جاتا ہے اور متغیر i کو 1 اور sum سے شروع کیا جاتا ہے۔ وائل لوپ کنڈیشن کو چیک کرتا ہے، چونکہ i <=> 5 شرط درست ہے، اس لیے صارف سے ایک نمبر پڑھا جاتا ہے اور اسے مجموعہ (sum) میں شامل کیا جاتا ہے اور i کو1 سے بڑھایا جاتا ہے۔ اب، شرط i <=> 5 ہے دوبارہ چیک کیا جاتا ہے۔ 5, > = 2 کے بعد سے، دوسرا نمبر صارف سے حاصل کیا جاتا ہے اور اسے مجموعہ میں شامل کیا جاتا ہے۔ یہ اس وقت تک جاری رہتا ہے، جب تک کہ i، 6 نہ ہو جائے، اس وقت وائل لوپ ختم ہو جاتا ہے۔ لوپ ختم ہونے کے بعد، اوسط(avg) کی گنتی کی جاتی ہے اور رمجموعہ اور اوسط دونوں دکھائے جاتے ہیں۔

وائل لوپ تغیر (while loop variation)

ایک وائل لوپ میں کئی تغیرات شامل ہو سکتے ہیں۔ یہ ایک خالی لوپ یا لامحدود لوپ ہوسکتا ہے۔ خالی وائل لوپ میں لوپ کے باڈی کے اندر کوئی اسٹیٹمنٹ نہیں ہوتا ہے سوائے نل سٹیٹمنٹ (null) کے، یعنی صرف ایک سیمی کولن۔

مثال کے طور پر

```
int main()
{

    int i=0;
    while(++i < 10000 ):

    return 0;

}
```

This is an empty loop because the while loop does not contain any statement

مندرجہ بالا کوڈ میں، لوپ ایک ٹائم ڈیلے لوپ (Time delay loop) ہے۔ پروگرام کو کچھ وقت کے لیے موقوف کرنے کے لیے ٹائم ڈیلے لوپ مفید ہے۔

وائل لوپ کی باڈی کے اندر کوئی اپ ڈیٹ اسٹیٹمنٹ نہیں دیا جاتا ہے تو ایک وائل لوپ لامحدود لوپ ہوسکتا ہے۔ مثال کے طور پر،

```
int main()
{

    int i = 0;
    while(i < =10)
      cout <<"The value of i is "<<i;
      i++;

    return 0;

}
```

This statement will be displayed infinitely because no update statement inside the body of the loop

This is not a part of the while loop statement because of missing curly braces

11th_Comp Science_UM_Ch 10.indd 170 18-04-2023 15:18:33

10.5.4 ڈو-وائل لوپ (do-while loop)

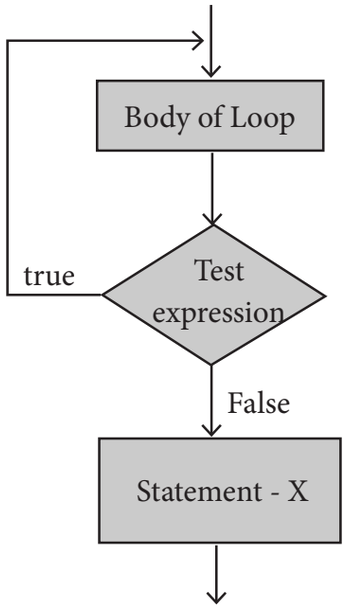
ڈو – وائل لوپ ایک ایگزٹ کنٹرولڈ لوپ (exit – controlled loop) ہے۔ ڈو-وائل لوپ میں، لوپ کی باڈی کو انجام دینے کے بعد لوپ کے نچلے حصے میں حالت کا جائزہ لیا جاتا ہے۔ اس کا مطلب ہے کہ لوپ کی باڈی کو کم از کم ایک بار عمل میں لایا جاتا ہے، یہاں تک کہ جب حالت پہلی اعادہ کے دوران غلط کا اندازہ کرتی ہے۔

ڈو وائل لوپ کا سنٹیکس یہ ہے:

```
do
{
        Body of the loop;

} while(condition);
```

ڈو-وائل لوپ کے لیے فلو کنٹرول اور فلو چارٹ نیچے دکھایا گیا ہے۔



فلو چارٹ 10.8: ڈو-وائل لوپ کنٹرول فلو اور فلو چارٹ

مثال 10.12 ڈو وائل لوپ کا استعمال کرتے ہوئے 10 سے 1 تک نمبر دکھانے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int n = 10;
do
{
        cout<<n<<", ";
        n--;
}while (n>0) ;
}
```

Output

10, 9, 8, 7, 6, 5, 4, 3, 2, 1

11th_Comp Science_UM_Ch 10.indd 171 18-04-2023 15:18:33

مندرجہ بالا پروگرام میں، انٹیجر متغیر n کو 10 سے شروع کیا جاتا ہے۔ اس کے بعد n کی قدر 10 کے طور پر ظاہر ہوتی ہے اور n کو 1 سے کم کیا جاتا ہے۔ اب، حالت کا جائزہ لیا جاتا ہے 0<9، سے دوبارہ 9 ظاہر ہوتا ہے اور n کو 8 تک کم کیا جاتا ہے۔ یہ اس وقت تک جاری رہتا ہے، جب تک n 0 کے کے برابر نہ ہو جائے، اس وقت، حالت 0 < n کا اندازہ غلط ہو جائے گا اور ڈو وائل لوپ ختم ہو جائے گا۔

10.5.5 لوپس کا نیسٹنگ (Nesting of loops)

ایک لوپ جس میں دوسرا لوپ ہوتا ہے اسے نیسٹڈ لوپ (nested loop) کہا جاتا ہے۔

سنٹیکس ذیل میں دیا گیا ہے:

```
for (initialization(s); test-expression; update expression(s))
{
        for (initialization(s); test-expression; update expression(s)
        {
        statement(s);
         }
statement(s);
}
```

```
while(condition)
{
        while(condition)
        {
        statement(s);
        }
statement(s);
}
```

```
do
{
statement(s);
        do
        {
        statement(s);
        }while(condition);
} while( condition );
```

مثال 10.13 نیسٹڈ فار لوپ کا استعمال کرتے ہوئے میٹرکس ضرب کی ٹیبل کو ظاہر کرنے کے لیے C++ پروگرام

```
#include<iostream>
using namespace std;
int main(void)
{
    cout<< "A multiplication table:" <<endl <<"  1\t2\t3\t4\t5\t6\t7\t8\t9" <<endl<< "" <<endl;
    for(int c = 1; c < 10; c++)
    {
    cout<< c << "| ";
    for(int i = 1; i< 10; i++)
    {
    cout<<i * c << '\t';
    }
    cout<<endl;
    }
return 0;
}
```


11th_Comp Science_UM_Ch 10.indd 172 18-04-2023 15:18:33

Output

A multiplication table:

1	2	3	4	5	6	7	8	9
1| 1	2	3	4	5	6	7	8	9
2| 2	4	6	8	10	12	14	16	18
3| 3	6	9	12	15	18	21	24	27
4| 4	8	12	16	20	24	28	32	36
5| 5	10	15	20	25	30	35	40	45
6| 6	12	18	24	30	36	42	48	54
7| 7	14	21	28	35	42	49	56	63
8| 8	16	24	32	40	48	56	64	72
9| 9	18	27	36	45	54	63	72	81

10.6 جمپ اسٹیٹ منٹس
(Jump statements)

جمپ اسٹیٹمنٹ پروگرام کے عام فلو میں خلل ڈالنے کے لیے استعمال ہوتے ہیں۔ جمپ اسٹیٹ منٹس کی اقسام یہ ہیں۔

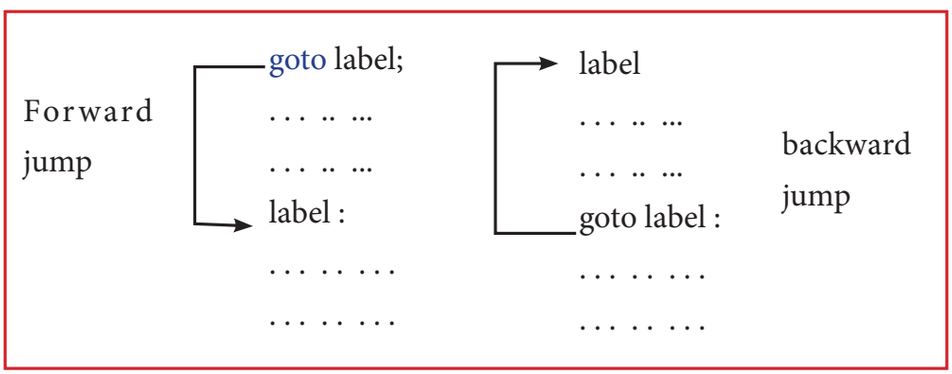
- گو ٹو اسٹیٹمنٹ (goto statement)
- بریک اسٹیٹمنٹ (break statement)
- کنٹینیو اسٹیٹمنٹ (continue statement)

10.6.1 گو ٹو اسٹیٹمنٹ

گوٹو اسٹیٹمنٹ ایک کنٹرول اسٹیٹمنٹ ہے جو کسی پروگرام میں بغیر کسی شرط کے کنٹرول کو ایک جگہ سے دوسری جگہ منتقل کرنے کے لیے استعمال ہوتا ہے

گو ٹو اسٹیٹمنٹ کا سنٹیکس ہے؛

Syntax1	Syntax2
goto label; ---------- ---------- ---------- **label:**	**label:** ---------- ---------- ---------- **goto label;**



11th_Comp Science_UM_Ch 10.indd 173 18-04-2023 15:18:33

اوپر دیے گئے سنٹیکس میں، لیبل ایک شناخت کنندہ ہے۔ جب گو ٹو لیبل (goto label)؛ کا سامنا ہوتا ہے تو پروگرام کا کنٹرول لیبل پر جمپ کرتا ہے: اور اس کے نیچے موجود کوڈ پر عمل درآمد کرتا ہے۔

مثال 10.14 گوٹو اسٹیٹمنٹ کا استعمال کرتے ہوئے پہلے پانچ طاق نمبروں کو ظاہر کرنے کے لیے C++ پروگرام

```
# include <iostream>
using namespace std;
int main()
{
int n=1;
jump:
{
 if(n<10)
 { // Control of the program move to jump:
        cout<<n<<'\t';
        n+=2;
        goto jump;
 }
 else
        return 0;
} }
```

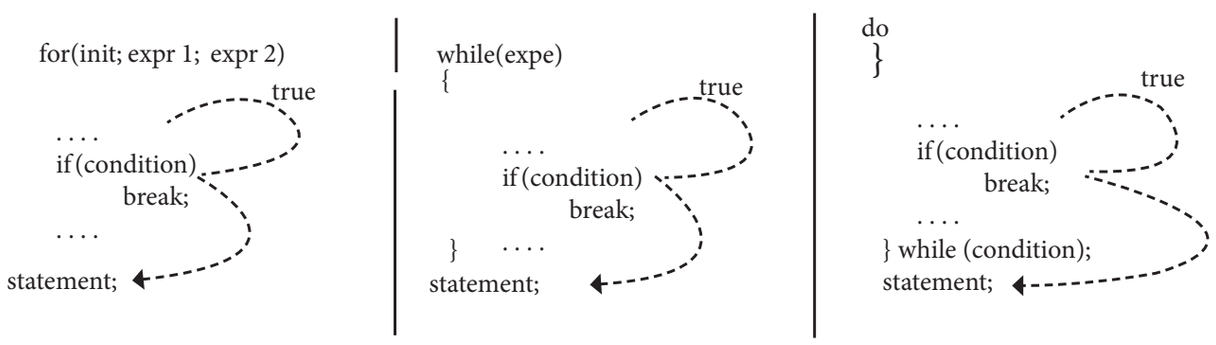
Output

```
1       3       5       7       9
```

مندرجہ بالا پروگرام میں پہلے پانچ طاق نمبر دکھائے جاتے ہیں۔ اگر n، 10 سے کم ہے گوٹو کنٹرول کو جمپ اسٹیٹمنٹ میں منتقل کرتا ہے۔ اگر n 10 سے زیادہ ہے تو کنٹرول لوپ سے باہر آجاتا ہے۔

10.6.2 بریک اسٹیٹمنٹ (break statement)

بریک اسٹیٹمنٹ ایک جمپ اسٹیٹمنٹ ہے جو لوپ کے عمل کو ختم کرتا ہے اور کنٹرول کو منتقل کیا جاتا ہے تاکہ لوپ کی باڈی کے بعد عام عمل کو دوبارہ شروع کیا جاسکے۔مندرجہ ذیل تصویر لوپنگ اسٹیٹ منٹس کے ساتھ بریک سٹیٹمنٹ کا کام دکھاتا ہے۔



بریک سٹیٹمنٹ for, while اور do-while لوپ میں

11th_Comp Science_UM_Ch 10.indd 174 18-04-2023 15:18:33

مثال 10.15 بریک اسٹیٹمنٹ کا استعمال کرتے ہوئے N نمبروں کی گنتی کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main ()
{
int count = 1;
do
{
cout<< "Count : " << count <<endl;
if( count >= 3)
 {
        break;
 }
count ++;
}while( count < 20 );
return 0;
}
```

Output

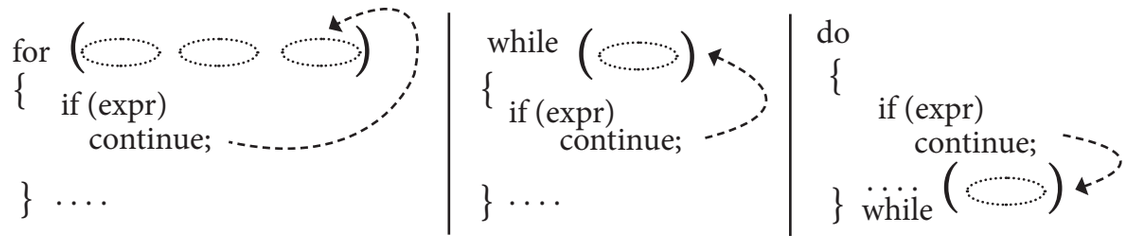
```
Count : 1
Count : 2
Count : 3
```

مندرجہ بالا مثال میں، وائل لوپ 20 بار اعادہ کرے گا، لیکن جیسے ہی گنتی 3 تک پہنچ جاتی ہے، بریک اسٹیٹمنٹ کی وجہ سے لوپ ختم ہو جاتا ہے۔

10.6.3 کنٹینیو اسٹیٹمنٹ (continue statement)

کنٹینیو اسٹیٹمنٹ کافی حد تک بریک اسٹیٹمنٹ سے ملتا جلتا کام کرتا ہے۔ لوپ (بریک اسٹیٹمنٹ) کو ختم کرنے کے بجائے، کنٹینیو اسٹیٹمنٹ لوپ کو اگلے اعادہ تک جاری رکھنے یا اس پر عمل کرنے پر مجبور کرتا ہے۔ جب کنٹینیو سٹیٹمنٹ لوپ میں عمل در آمد ہوتا ہے تو کنٹینو سٹیٹمنٹ کے بعد لوپ کے اندر موجود کوڈ کو چھوڑ دیا جائے گا اور لوپ کا اگلا اعادہ شروع ہو جائے گا۔

مندرجہ ذیل شکل کنٹینو اسٹیٹمنٹ کے ورکنگ فلو کو واضح کرتی ہے۔



کنٹینیو اسٹیٹمنٹ کا ورک فلو

مندرجہ بالا مثال میں، لوپ 10 بار اعادہ کرے گا لیکن، اگر i، 6 تک پہنچ جاتا ہے، تو کنٹینیو اسٹیٹمنٹ کی وجہ سے کنٹرول فار لوپ میں منتقل ہو جاتا ہے۔

11th_Comp Science_UM_Ch 10.indd 175 18-04-2023 15:18:33

مثال 10.16 کنٹینیو اسٹیٹمنٹ کا استعمال کرتے ہوئے 6 کے علاوہ 1 سے 10 تک نمبر ظاہر کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
 int main()
{
for (int i = 1; i<= 10; i++) {
if (i == 6)
continue;
else
cout<<i<< " ";
}
return 0;
}
```

Output

1 2 3 4 5 7 8 9 10

بریک (Break) اور کنٹینیو (Continue) کے درمیان فرق

بریک (Break)	کنٹینیو (Continue)
بریک کا استعمال لوپ کے عمل کو ختم کرنے کے لیے کیا جاتا ہے۔	لوپ کے عمل کو ختم کرنے کے لیے کنٹینیو کا استعمال نہیں کیا جاتا ہے
یہ اعادہ کو توڑ دیتا ہے۔	یہ اعادہ کو چھوڑ دیتا ہے۔
جب یہ اسٹیٹمنٹ عمل میں آتا ہے، توکنٹرول لوپ سے باہر آجائے گا اور لوپ کے فوراً بعد اسٹیٹمنٹ پر عمل درآمد کرتا ہے۔	جب یہ اسٹیٹمنٹ عمل میں آتا ہے، تو یہ لوپ سے باہر نہیں آئے گا بلکہ لوپ کے اگلے اعادہ پر بڑھتا/جمپ کرتا ہے۔
بریک کو لوپس کے ساتھ ساتھ سوئچ کیس کے ساتھ استعمال کیا جاتا ہے۔	کنٹینیو صرف لوپس میں استعمال ہوتا ہے، یہ سوئچ کیس میں استعمال نہیں ہوتا ہے۔



عملی مشق

درج ذیل مسائل کو حل کرنے کے لیے C++ پروگرام لکھیں:

1. کسی کیریکٹر کو داخل کرنے اور پرنٹ کرنے کا پروگرام کہ آیا دیا ہوا حرف حرف تہجی، ہندسہ یا کوئی اور حرف ہے۔
2. پرنٹ کرنے کا پروگرام کہ آیا دیا ہوا حرف بڑا ہے یا چھوٹا حرف یا ہندسہ یا کوئی اور حرف۔ اس کے لیے ASCII کوڈ استعمال کریں۔ ASCII کوڈز ذیل میں دیئے گئے ہیں:

حروف	ASCII رینج
'0' - '9'	48 - 57
'A' - 'Z'	65 - 90
'a' - 'z'	97 - 122

دیگر حروف 0 – 255 کو چھوڑ کر مذکورہ بالا کوڈز۔

3. ایک عدد کے فیکٹریل کا حساب لگانے کا پروگرام۔
4. فبونیکی سیریز (fabonacci series) کو پرنٹ کرنے کا پروگرام یعنی 0 1 1 2 3 5 8

11th_Comp Science_UM_Ch 10.indd 176 18-04-2023 15:18:33

5. نیسٹڈ لوپس کا استعمال کرتے ہوئے درج ذیل ڈیزائن تیار کرنے کے پروگرامس

(a)	(b)
A	5 4 3 2 1
A B	5 4 3 2
A B C	5 4 3
A B C D	5 4
A B C D E	5
A B C D E F	

یاد رکھنے کے نکات

- کمپیوٹر پروگرام کسی مخصوص کام کو انجام دینے کے لیے اسٹیٹ مینٹس یا ہدایات کا ایک مجموعہ ہے۔
- C ++ میں دو قسم کے اسٹیٹ مینٹس استعمال ہوتے ہیں، یعنی نل (Null) اور کمپاؤنڈ اسٹیمنٹ (Compound statement)۔
- کنٹرول اسٹیٹ مینٹس وہ اسٹیٹ مینٹس ہیں جو ہدایات کے فلو کی ترتیب کو تبدیل کرتے ہیں۔
- C++ میں تین قسم کے کنٹرول اسٹیمنٹ استعمال ہوتے ہیں۔ (1) سکوینس اسٹیمنٹ (2) سلیکشن اسٹیمنٹ (3) ائی ٹریشن اسٹیمنٹ
- اف اور سوئچ، سلیکشن اسٹیٹ مینٹس ہیں۔
- کنڈیشنل آپریٹر 'اف – ایلس اسٹیمنٹ' کا متبادل ہے۔
- سوئچ اسٹیمنٹ ایک ملٹی وے برانچنگ اسٹیمنٹ ہے۔
- ائی ٹریشن اسٹیمنٹ (لوپنگ) اسٹیٹ مینٹس کے ایک سیٹ کو بار بار انجام دینے کے لئے استعمال ہوتا ہے جب تک کہ کوئی شرط پوری نہ ہوجائے۔
- تین قسم کے تائید ی ائی ٹریشن اسٹیٹ مینٹس ہیں ۔ (1) فار (for) (2) وائل (while) (3) ڈو- وائل (do-while)۔
- C++ میں تین جمپ اسٹیمنٹ استعمال ہوتے ہیں۔ (1) گو ٹو اسٹیمنٹ (goto statement) (2) بریک اسٹیمنٹ (break statement) (3) کنٹینیو اسٹیمنٹ (continue statement)

تشخیص

84AUJ

سیکشن – A

صحیح جواب کا انتخاب کریں

1. نل سٹیمنٹ کا متبادل نام کیا ہے؟

No statement (A)

Empty statement (B)

Void statement (C)

Zero statement (D)

2. C++ میں، اسٹیٹ مینٹس کا گروپ اس کے اندر منسلک ہونا چاہیے:

{ } (A) [ ] (B)

( ) (C) < > (D)

3. اسٹیٹ مینٹس کا مجموعہ جو بار بار اعادہ میں عمل میں لایا جاتا ہے اسے کہا جاتا ہے:

condition (A) loop (B)

statement (C) body of loop (D)

4. ملٹی وے برانچ اسٹیمنٹ:

if (A) if … else (B)

switch (C) for (D)

5. آئی ٹریشن اسٹیٹ مینٹس کی کتنی اقسام ہیں؟

2 (A) 3 (B)

5 (D) 4 (C)

6. (int i=0; i<10; i++) کے لیے، درج ذیل لوپ کتنی بار عمل میں آئے گا؟

0 (A) 10 (B)

9 (C) 11 (D)

11th_Comp Science_UM_Ch 10.indd 177 18-04-2023 15:18:34

7. مندرجہ ذیل میں سے کون سا ایگزٹ کنٹرول لوپ ہے؟

for (A) while (B)
do…while (C) if…else (D)

8. جمپ اسٹیٹ منٹس کے کلیدی الفاظ سے طاق کی شناخت کریں:

break (A) switch (B)
goto (C) continue (D)

9. مندرجہ ذیل میں سے کس کو انٹری کنٹرول لوپ کہا جاتا ہے؟

do-while (A) switch (B)
while (C) if-else (D)

10. ایک لوپ جس کی باڈی کے اندر ایک اور لوپ ہوتا ہے:

Nested loop (A) Inner loop (B)
Inline loop (C) Nesting of loop (D)

سیکشن – B

بہت ہی مختصر جوابات دیں۔

1. نل اسٹیٹمنٹ اور کمپاؤنڈ اسٹیٹمنٹ کیا ہے؟
2. سلیکشن اسٹیٹمنٹ کیا ہے؟ اس کی اقسام لکھیں؟
3. درج ذیل کوڈ سیگمنٹ کو درست کریں:

```
if (x=1)
  p= 100;
  else
p = 10;
```

4. درج ذیل کوڈ کا آؤٹ پٹ کیا ہے؟

```
int year;
cin >> year;
if (year % 100 == 0)
    if ( year % 400 == 0)
            cout << "Leap";
else
      cout << "Not Leap year";
```

If the input given is (i) 2000 (ii) 2003 (iii) 2010?

5. درج ذیل کوڈ کا آؤٹ پٹ کیا ہے؟

```
for (int i=2; i<=10 ; i+=2)
    cout << i;
```

6. ایک فار لوپ لکھیں جو نمبر 21 سے 30 تک دکھاتا ہے۔
7. ایک وائل لوپ لکھیں جو نمبر 20.....8, 6, 4, 2 دکھاتا ہے۔
8. ایک if اور a کا موازنہ کریں؟ : آپریٹر

سیکشن – C

مختصر جوابات دیں

1. درج ذیل if-else کو ایک کنڈیشنل اسٹیٹمنٹ میں تبدیل کریں:

```
if (x >= 10)
        a = m + 5;
else
        a = m;
```

2. درج ذیل کوڈ کو دوبارہ لکھیں تاکہ یہ فعال ہو:

```
v = 5;
do;
{
  total += v;
  cout << total;
while v <= 10
```

3. دیے گئے نمبر کے ضرب کی ٹیبل کو پرنٹ کرنے کے لیے ایک C++ پروگرام لکھیں۔
4. سوئچ اسٹیٹمنٹ کا سنٹیکس اور مقصد لکھیں۔
5. درج ذیل سیریز کو پرنٹ کرنے کے لیے ایک مختصر پروگرام لکھیں:

40 .....10 7 4 1 (a)

سیکشن – D

تفصیل سے وضاحت کریں۔

1. مناسب مثال کے ساتھ کنٹرول اسٹیٹمنٹ کی وضاحت کریں۔

11th_Comp Science_UM_Ch 10.indd 178 18-04-2023 15:18:34

2. انٹری کنٹرول لوپ کیا ہے؟ مناسب مثال کے ساتھ انٹری کنٹرولڈ لوپ میں سے کسی ایک کی وضاحت کریں۔

3. دو نمبروں کے LCM اور GCD کو تلاش کرنے کے لیے ایک پروگرام لکھیں۔

4. درج ذیل سیریز کا مجموعہ تلاش کرنے کے لیے پروگرام لکھیں:

(a) $x - \frac{x^2}{2!} + \frac{x^3}{3!} - \frac{x^4}{4!} + \frac{x^5}{5!} - \frac{x^6}{6!}$

(b) $x + \frac{x^2}{2} + \frac{x^3}{3} + ... + \frac{x^n}{n}$

5. سیریز کا مجموعہ معلوم کرنے کے لیے ایک پروگرام لکھیں۔

$$S = 1 + x + x^2 + ... + x^n$$

حوالہ جات:

(1) Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.

(2) The Complete Reference C++ (Forth Edition), Herbert Schildt. Mc.Graw Hills.

(3) Computer Science with C++ (A text book of CBSE XI and XII), Sumita Arora, Dhanpat Rai & Co.

11th_Comp Science_UM_Ch 10.indd 179 18-04-2023 15:18:34

فنکشنس (Functions)

سیکھنے کے مقاصد



اس باب کو سیکھنے کے بعد طلباء کے قابل ہوجائیں گے۔

- فنکشن کی تعریف اور فنکشنس کے استعمال کو سمجھنے
- فنکشنس کے اقسام - پہلے سے وضاحت کردہ (pre-defined) اور صارف کی وضاحت کردہ (user-defined) فنکشنس کو سمجھنے
- مسائل کو حل کرنے کے لیے ریاضی کے فنکشنس کو عمل میں لانے
- سٹرنگ (string) اور کریکٹر ڈیٹا (character data) کی ہیرا پھیری کے لیے سٹرنگ اور کریکٹر فنکشنس کا استعمال کرنے
- فنکشنس بنا کر ماڈیولر پروگرامنگ (modular programming) لاگو کرنے
- آرگیومنٹس (arguments) کے کردار اور آرگیومنٹس کے مختلف طریقوں کا موازنہ کرنے
- ایک پروگرام میں متغیرات اور فنکشنس کے دائرہ کار کو پہچاننے

11.1 تعارف

ایک بڑے پروگرام کو چھوٹے ذیلی پروگراموں (بلاکس - blocks) میں تقسیم کیا جا سکتا ہے جسے فنکشنس (functions) کہا جاتا ہے جہاں ہر ذیلی پروگرام کچھ مخصوص فعالیت انجام دے سکتا ہے۔ فنکشنس پروگرام کے سائز اور پیچیدگی کو کم کرتے ہیں، اسے سمجھنا، جانچنا اور غلطیوں کی جانچ کرنا آسان بنا دیتے ہیں۔ جو فنکشنس بطور ڈیفالٹ دستیاب ہوتے ہیں وہ "بلٹ ان - Built-in" فنکشنس کے نام سے جانے جاتے ہیں اور صارف اپنے فنکشنس بنا سکتا ہے جسے "یوزر ڈیفائنڈ - User-defined" فنکشنس کہا جاتا ہے۔

- بلٹ ان فنکشنس- فنکشنس جو C ++ لینگویج کی معیاری لائبریری میں دستیاب ہیں۔
- صارف کی طرف سے وضاحت کردہ فنکشنس- صارفین کے ذریعہ تخلیق کردہ فنکشنس۔

11.2 فنکشنس کی ضرورت (Need for Functions)

پروگرام کے سائز اور پیچیدگی کو کم کرنے کے لیے ہم فنکشنس کا استعمال کرتے ہیں۔ پروگرامرز اپنے فنکشنس لکھ کر یا معیاری لائبریری سے کال کر کے ذیلی پروگراموں کا استعمال کر سکتے ہیں۔

1. تقسیم اور فتح (Divide and Conquer)

- پیچیدہ پروگراموں کو قابل انتظام ذیلی پروگراموں میں تقسیم کیا جا سکتا ہے جنہیں فنکشنس کہتے ہیں۔
- ایک پروگرامر انفرادی فنکشنس کو تیار کرنے، ڈی بگنگ (debugging) کرنے اور جانچنے پر توجہ دے سکتا ہے۔
- بہت سے پروگرامر بیک وقت مختلف فنکشنس پر کام کر سکتے ہیں۔

2. دوبارہ قابل استعمال (Reusability)

- کوڈ کی چند سطریں مختلف سیاق و سباق میں بار بار استعمال کی جا سکتی ہیں۔ ایک ہی کوڈ کی نقل کو فنکشنس کا استعمال کرکے ختم کیا جا سکتا ہے جو دیکھ بھال کو بہتر بناتا ہے اور پروگرام کا سائز کم کرتا ہے۔
- کچھ فنکشنس کو مختلف ان پٹ کے ساتھ متعدد بار کال کیا جا سکتا ہے۔

11th_Comp Science_UM_Ch 11.indd 180 18-04-2023 15:30:05

11.3 فنکشنس کی اقسام

فنکشنس کو دو اقسام میں تقسیم کیا جا سکتا ہے

1. پہلے سے وضاحت کردہ (Pre-defined) یا بلٹ ان (Built-in) یا لائبریری کے فنکشنس (Library functions)
2. صارف کی طرف سے وضاحت کردہ فنکشنس۔

C++ فنکشنس کا ایک بھرپور مجموعہ فراہم کرتا ہے جو مختلف کاموں کے لیے استعمال ہونے کے لیے تیار ہے۔ ان میں سے ہر ایک کے ذریعہ انجام دینے والے کام پہلے ہی لکھے ہوئے ہیں، ڈی بگڈ (debugged) ہیں اور مرتب کیے گئے ہیں، صرف ان کی تعریفیں ہیڈر فائلز (header files) نامی فائلوں میں گروپ اور اسٹور کی جاتی ہیں۔ اس طرح کے استعمال کے لیے تیار ذیلی پروگراموں کو پہلے سے وضاحت کردہ (Pre-defined) یا بلٹ ان (Built-in) فنکشنس کہا جاتا ہے۔

C++ صارف کی ضرورت کے مطابق مخصوص کام کے لیے نئے فنکشنس بنانے کی سہولت بھی فراہم کرتا ہے۔ ٹاسک کے نام اور مطلوبہ ڈیٹا (آرگیومنٹس) کا فیصلہ صارف کرتا ہے اور اس لیے وہ صارف کی وضاحت کردہ فنکشنس (User -defined functions) کے نام سے جانے جاتے ہیں۔

11.4 C++ ہیڈر فائلز اور بلٹ ان فنکشنس

ہیڈر فائلیں لائبریری کے فنکشنس کے لیے فنکشنس پروٹو ٹائپ (prototype) اوروضاحتیں فراہم کرتی ہیں۔ لائبریری کے فنکشنس کے ساتھ استعمال ہونے والے ڈیٹا کی اقسام اور کانس ٹینٹس (constants) بھی ان میں بیان کیے گئے ہیں۔ ایک ہیڈر فائل کی شناخت ان کی فائل ایکسٹینشن .h. سے کی جا سکتی ہے۔ ایک ہیڈر فائل میں متعدد بلٹ ان (built-in) فنکشنس شامل ہو سکتے ہیں۔

مثال کے طور پر : stdio.h ایک ہیڈر فائل ہے جو پہلے سے وضاحت کردہ "معیاری ان پٹ / آؤٹ پٹ- standard input/ output" فنکشنس پر مشتمل ہے۔

11.4.1 معیاری ان پٹ / آؤٹ پٹ (stdio.h) (Standard input / output)

یہ ہیڈر فائل معیاری I/O پہلے سے وضاحت کردہ فنکشنس getchar(), putchar(), gets(), puts () اور وغیرہ کی وضاحت کرتی ہے۔

11.4.1.1 getchar() اور putchar() فنکشنس

پہلے سے وضاحت کردہ فنکشنس getchar() کی بورڈ سے ایک کریکٹر حاصل کرنے کے لیے استعمال ہوتا ہے اور putchar() فنکشنس اسے ظاہر کرنے کے لیے استعمال کیا جاتا ہے۔

پروگرام 11.1 کسی کردار کو قبول کرنے اور اسے ڈسپلے کرنے کے لیے C++ کوڈ

```
#include<iostream>
#include<stdio.h>
using namespace std;
int main()
{
        cout<<"\n Type a Character : ";
        char ch = getchar();
        cout << "\n The entered Character is: ";
        putchar(ch);
        return 0;
}
```

Output:

```
Type a Character : T
The entered Character is: T
```

11.4.1.2 gets() اور puts() فنکشنس

فنکشنس gets() معیاری ان پٹ سے ایک سٹرنگ پڑھتا ہے اور اسے متغیر کے ذریعہ اشارہ کردہ سٹرنگ میں اسٹور کرتا ہے۔ فنکشنس puts() ایک نئی لائن میں gets() فنکشنس کے ذریعے پڑھی جانے والی اسٹرنگ کو پرنٹ کرتا ہے۔

11th_Comp Science_UM_Ch 11.indd 181 18-04-2023 15:30:05

پروگرام 11.2 سٹرنگ کو قبول کرنے اور ڈسپلے کرنے کے لیے C ++ کوڈ

```
#include<iostream>
#include<stdio.h>
using namespace std;
int main()
{
        char str[50];
        cout<<"Enter a string : ";
        gets(str);
        cout<<"You entered: "
        puts(str);
        return(0);
}
```

Output :

```
Enter a string : Computer Science
You entered: Computer Science
```

11.4.2 کیریکٹر فنکشنس (ctype.h) (Character functions)

یہ ہیڈر فائل کیریکٹر پر مختلف کارروائیوں کی وضاحت کرتی ہے۔ C++ میں دستیاب مختلف کریکٹر فنکشنس درج ذیل ہیں۔ ان فنکشنس کو پروگرام میں استعمال کرنے کے لیے ہیڈر فائل ctype.h کو شامل کرنا ہے۔

11.4.2.1 اسالنم () Isalnum( )

یہ فنکشنس یہ چیک کرنے کے لیے استعمال کیا جاتا ہے کہ آیا کوئی حرف نمبری (alphanumeric) ہے یا نہیں۔ یہ فنکشنس غیر صفر کی قدر واپس کرتا ہے اگر c ہندسہ یا حرف ہے، ورنہ یہ 0 لوٹاتا ہے۔

جنرل فارم:

```
int isalnum (char c);
```

مثال:

```
int r = isalnum('5');
cout << isalnum('A') <<'\t'<<r;
```

لیکن ذیل میں دیئے گئے بیانات متغیر n کو 0 تفویض کرتے ہیں، کیونکہ دیا ہوا حرف نہ تو حروف تہجی ہے اور نہ ہی کوئی ہندسہ۔

```
char c = '$';
int n = isalnum(c);
cout<<c;
```

Output:

```
0
```


11th_Comp Science_UM_Ch 11.indd 182 18-04-2023 15:30:05

پروگرام 11.3

```
#include<iostream>
#include<stdio.h>
#include<ctype.h>
using namespace std;
int main()
{
        char ch;
        int r;
        cout<<"\n Type a Character :";
        ch = getchar();
        r = isalnum(ch);
        cout<<"\nThe Return Value of isalnum(ch) is :"<<r;
}
```

Output-1:

```
Type a Character :A
The Return Value of isalnum(ch) is :1
```

Output-2:

```
Type a Character :?
The Return Value of isalnum(ch) is :0
```

11.4.2.2 اسالفا() isalpha( )

اسالفا() فنکشن یہ چیک کرنے کے لیے استعمال کیا جاتا ہے کہ آیا دیا ہوا حرف حرف تہجی ہے یا نہیں

جنرل فارم:

isalpha(char c);

یہ فنکشن 1 واپس کرے گا اگر دیا ہوا حرف حرف تہجی ہے اور 0 ورنہ 0۔ درج ذیل بیان متغیر n کو 0 تفویض کرتا ہے، کیونکہ دیا ہوا حرف حرف تہجی نہیں ہے۔

int n = isalpha('3');

لیکن، ذیل میں دیا گیا بیان 1 دکھاتا ہے، کیونکہ دیا ہوا حرف ایک حروف تہجی ہے۔

cout << isalpha('a');

11th_Comp Science_UM_Ch 11.indd 183 18-04-2023 15:30:05

پروگرام 11.4

```
#include<iostream>
#include<stdio.h>
#include<ctype.h>
using namespace std;
int main()
{
        char ch;
        cout << "\n Enter a charater: ";
        ch = getchar();
        cout<<"\n The Return Value of isalpha(ch) is :" << isalpha(ch) ;
}
```

Output-1:

```
        Enter a charater: A
        The Return Value of isalpha(ch) is :1
```

Output-2:

```
        Enter a charater: 7
        The Return Value of isalpha(ch) is :0
```

11.4.2.3 اس ڈیجٹ() isdigit()

یہ فنکشن یہ چیک کرنے کے لیے استعمال کیا جاتا ہے کہ آیا دیا گیا کریکٹر ہندسہ ہے یا نہیں۔ یہ فنکشن 1 واپس کرے گا اگر دیا ہوا حرف ایک ہندسہ ہے، ورنہ 0۔

جنرل فارم:

isdigit(char c);

پروگرام 11.5

```
using namespace std;
#include<iostream>
#include<ctype.h>
int main()
{
        char ch;
        cout << "\n Enter a Character: ";
        cin >> ch;
        cout<<"\n The Return Value of isdigit(ch) is :" << isdigit(ch) ;
}
```


11th_Comp Science_UM_Ch 11.indd 184 18-04-2023 15:30:05

Output-1

Enter a Character: 3
The Return Value of isdigit(ch) is :1

Output-2

Enter a Character: A
The Return Value of isdigit(ch) is :0

*** ریٹرن 0؛ (تازہ ترین مرتب کرنے والوں میں لازمی نہیں)**

11.4.2.4 اس لوور islower()

یہ فنکشن یہ جانچنے کے لیے استعمال کیا جاتا ہے کہ آیا کوئی حرف چھوٹے (چھوٹے حرف) میں ہے یا نہیں۔یہ فنکشن ایک غیر صفر کی قدر واپس کرے گا، اگر دیا ہوا حرف چھوٹا حرف تہجی ہے، ورنہ 0۔

جنرل فارم:

```
islower(char c)
```

مندرجہ ذیل بیانات پر عمل کرنے کے بعد، متغیر n کی قدر 1 ہو جائے گی کیونکہ دیا ہوا کریکٹر لوور کیس (lower case) میں ہے۔

```
char ch = 'n';
int n = islower(ch);
```

لیکن ذیل میں دیا گیا بیان متغیر n کو 0 تفویض کرے گا، کیونکہ دیا ہوا حرف ایک اپر کیس (uppercase) حرف ہے۔

```
int n = islower('P');
```

11.4.2.5 اس اپر isupper()

یہ فنکشن دیے گئے کیریکٹر کو اپر کیس کی جانچ کرنے کے لیے استعمال کیا جاتا ہے۔ یہ فنکشن 1 لوٹائے گا اگر صحیح ہے ورنہ 0۔

جنرل فارم:

```
isupper(char c)
```

درج ذیل مثالوں کے لیے قدر 1 کو n اور 0 کو m کے لیے تفویض کیا جائے گا۔

```
int n=isupper('A');
int m=isupper('a');
```

11.4.2.6 ٹو اپر toupper( )

یہ فنکشن دیئے گئے کریکٹر کو اپر کیس (uppercase) میں تبدیل کرنے کے لیے استعمال کیا جاتا ہے۔ یہ فنکشن دیئے گئے کریکٹر کے مساوی اپر کیس لوٹائے گا۔ اگر دیا گیا کریکٹر خود اپر کیس میں ہے، تو آؤٹ پٹ ایک جیسا ہو گا۔

جنرل فارم:

```
char toupper(char c);
```

درج ذیل بیان متغیر c کو کریکٹر کانسٹینٹ 'K' تفویض کرے گا۔

```
char c = toupper('k');
cout<<c;
```

11.4.2.7 ٹو لوور tolower()

یہ فنکشن دیئے گئے کریکٹر کو لوور کیس میں تبدیل کرنے کے لیے استعمال کیا جاتا ہے۔ یہ فنکشن دیے گئے کریکٹر کے لوور کیس کے برابر لوٹائے گا۔ اگر دیا گیا کریکٹر خود لوور کیس میں ہے تو آؤٹ پٹ ایک جیسا ہو گا۔

جنرل فارم:

```
char tolower(char c)
```

درج ذیل بیان متغیر c کو کریکٹر کانسٹینٹ 'k' تفویض کرے گا۔

```
char c = tolower('K');
cout <<c;
```

11.4.3 سٹرنگ ہیرا پھیری (String manipulation) (string.h)

لائبریری string.h (جسے cstring بھی کہا جاتا ہے) میں حروف کی ترتیب میں ذخیرہ شدہ اسٹرنگس سے نمٹنے کے لیے کئی عام کام ہوتے ہیں۔ string.h ہیڈر فائل کو کسی بھی سٹرنگ فنکشن کو استعمال کرنے سے پہلے شامل کرنا ہے۔

11th_Comp Science_UM_Ch 11.indd 185 18-04-2023 15:30:05

11.4.3.1 اسٹرپکپی () strcpy()

جنرل فارم:

strcpy(Target String, Source String);

اسٹرپکپی() فنکشن دو آرگیو منٹس (arguments) لیتا ہے: ہدف (target) اور ذریعہ (source)۔ یہ ماخذ کے ذریعہ اشارہ کردہ کریکٹر سٹرنگ کو ہدف کے ذریعہ اشارہ کردہ میموری کے مقام کی طرف نقل کرتا ہے۔ سٹرنگ کے ساتھ منسلک کالعدم کریکٹر (\0) کو بھی نقل کیا جاتا ہے۔

پروگرام 11.6

```
#include <string.h>
#include <iostream>
using namespace std;
int main()
{
        char source[] = "Computer Science";
        char target[20]="target";
        cout<<"\n String in Source Before Copied :"<<source;
        cout<<"\n String in Target Before Copied :"<<target;
        strcpy(target,source);
        cout<<"\n String in Target After strcpy function Executed :"<<target;
        return 0;
}
```

Output:

```
String in Source Before Copied :Computer Science
String in Target Before Copied :target
String in Target After strcpy function Executed :Computer Science
```

11.4.3.2 اسٹرلین () strlen()

اسٹرلین () اپنی دلیل کے طور پر ایک کالعدم سٹرنگ لیتا ہے اور اس کی لمبائی واپس کرتا ہے۔ لمبائی میں نل (\0) حرف شامل نہیں ہے۔

جنرل فارم:

strlen(string);

پروگرام 11.7

```
#include <string.h>
#include <iostream>
using namespace std;
int main()
{
        char source[ ] = "Computer Science";
        cout<<"\n Given String is "<<source<<" its Length is "<<strlen(source);
        return 0;
}
```

Output:

```
Given String is Computer Science its Length is 16
```


11th_Comp Science_UM_Ch 11.indd 186 18-04-2023 15:30:06

11.4.3.3 اسٹرمپ() strcmp()

اسٹرمپ() فنکشن دو آرگیومنٹس لیتا ہے:اسٹرنگ 1 اور اسٹرنگ 2۔ یہ اسٹرنگ 1 اور اسٹرنگ 2 کے لُغت نِگاری کے انداز سے موازنہ کرتا ہے۔

جنرل فارم:

strcpy(String1, String2);

اسٹرمپ() فنکشن یہ لوٹاتا ہے:

- مثبت قدر اگر سٹرنگ 1 میں پہلا مختلف کریکٹر سٹرنگ2 میں متعلقہ کریکٹر سے بڑا ہے۔ (ASCII اقدار کا موازنہ کیا جاتا ہے)
- منفی قدر اگر سٹرنگ 1 میں پہلا مختلف کریکٹر سٹرنگ 2 میں متعلقہ کریکٹر سے کم ہو۔
- 0 اگر اسٹرنگ 1 اور اسٹرنگ 2 برابر ہو۔

پروگرام 11.8

```
#include <string.h>
#include <iostream>
using namespace std;
int main()
{
        char string1[] = "Computer";
        char string2[] = "Science";
        int result;
        result = strcmp(string1,string2);
        if(result==0)
        {
        cout<<"String1 : "<<string1<<" and String2 : "<<string2 <<"Are Equal";
        }
        if (result<0)
        {
        cout<<"String1 :"<<string1<<" and String2 : "<<string2 <<" Are Not Equal";
        }
}
```

Output

```
        String1 : Computer and String2 : Science Are Not Equal
```

11.4.3.4 اسٹرکیٹ() strcat()

اسٹرکیٹ() فنکشن دو آرگیومنٹس لیتا ہے: ہدف اور ذریعہ۔ یہ فنکشن ذریعہ کی طرف سے اشارہ کردہ کریکٹر سٹرنگ کی کاپی کو ہدف کی طرف سے اشارہ کردہ سٹرنگ کے آخر میں شامل کرتا ہے۔

11th_Comp Science_UM_Ch 11.indd 187 18-04-2023 15:30:06

جنرل فارم:

strcat(Target, source);

پروگرام 11.9

```
#include <string.h>
#include <iostream>
using namespace std;
int main()
{
        char target[50] = "Learning C++ is fun";
        char source[50] = " , easy and Very useful";
        strcat(target, source);
        cout << target ;
        return 0;
}
```

Output

```
        Learning C++ is fun , easy and Very useful
```

11.4.3.5 اسٹروپر() strupr()

اسٹروپر() فنکشن دی گئی سٹرنگ کو اپر کیس (uppercase) حروف میں تبدیل کرنے کے لیے استعمال کیا جاتا ہے۔

جنرل فارم:

strcat(string);

پروگرام 11.10

```
using namespace std;
#include<iostream>
#include<ctype.h>
#include<string.h>
int main()
{
        char str1[50];
        cout<<"\nType any string in Lower case :";
        gets(str1);
        cout<<"\n Converted the Source string “<<str1<<into Upper Case is "<<strupr(str1);
        return 0;
}
```

Output:

```
        Type any string in Lower case : computer science
Converted the Source string computer science into Upper Case is COMPUTER SCIENCE
```

11.4.3.6 اسٹرل ور() strlwr()

اسٹرل ور() فنکشن دی گئی اسٹرنگ کولوور کیس میں تبدیل کرنے کے لیے استعمال ہوتا ہے۔

جنرل فارم:

strlwr(string);

11th_Comp Science_UM_Ch 11.indd 188 18-04-2023 15:30:06

پروگرام 11.11

```
using namespace std;
#include<iostream>
#include<ctype.h>
#include<string.h>
int main()
{
        char str1[50];
        cout<<"\nType any string in Upper case :";
        gets(str1);
        cout<<"\n Converted the Source string “<<str1<<into Lower Case is "<<strlwr(str1);
}
```

Output:

```
        Type any string in Upper case : COMPUTER SCIENCE
Converted the Source string COMPUTER SCIENCE into lower Case is computer science
```

11.4.4 ریاضی فنکشنس (Mathematical functions) (math.h)

زیادہ تر ریاضی فنکشنس کی وضاحت math.h ہیڈر فائل میں کی گئی ہے جس میں بنیادی ریاضی کے فنکشنس شامل ہیں۔

11.4.4.1 cos() فنکشن

cos() فنکشن ریڈیئنز(radians) میں ایک ہی آرگیو منٹ لیتا ہے۔ cos() فنکشن [-1,1] کی رینج میں قدر لوٹاتا ہے۔ لوٹائی گئی قدر یا تو ڈبل (double)، فلوٹ (float)، یا لانگ ڈبل (long double) میں ہے۔

پروگرام 11.12

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
        double x = 0.5, result;
        result = cos(x);
        cout << "COS("<<x<<")= "<<result;
}
```

Output:

```
        COS(0.5)= 0.877583
```

11.4.4.2 sqrt() فنکشن

sqrt() فنکشن دی گئی قدر کا اسکوائر روٹ (square root) لوٹا تا ہے۔ sqrt() فنکشن ایک واحد غیر منفی آرگیو منٹ لیتا ہے۔ اگر منفی قدر کو بطور آرگیومنٹ sqrt() فنکشن میں منتقل کیا جاتا ہے تو ڈومین (domain) کی خرابی ہوتی ہے۔

11th_Comp Science_UM_Ch 11.indd 189 18-04-2023 15:30:06

پروگرام 11.13

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
        double x = 625, result;
        result = sqrt(x);
        cout << "sqrt("<<x<<") = "<<result;
        return 0;
}
```

Output:

```
        sqrt(625) = 25
```

11.4.4.3 sin() فنکشن

sin() فنکشن ریڈینز (radians) میں ایک ہی آرگیو منٹ لیتا ہے۔ sin() فنکشن [-1,1] کی حد میں قدر واپس کرتا ہے۔ لوٹائی گئی قدر یا تو ڈبل، فلوٹ، یا لانگ ڈبل میں ہے۔

11.4.4.4 pow() فنکشن

pow() فنکشن ایکسپونٹ (exponent) کی پاور پر اٹھائے گئے بیس (base) کو لوٹاتا ہے۔ اگر pow() کو دی گئی کوئی آرگیو منٹ لانگ ڈبل ہے، تو ریٹرن کی قسم کو بڑھا کر لانگ ڈبل (long double) کر دیا جاتا ہے۔ اگر نہیں، تو ریٹرن کی قسم ڈبل ہے۔ pow() فنکشن دو آرگیو منٹس لیتا ہے:

- **بنیاد (base)** - بنیادی قدر
- **ایکسپونٹ (exponent)** - بنیادی ایکسپونٹ

پروگرام 11.14

```
#include <iostream>
#include <math.h>
using namespace std;
int main ()
{
        double base, exponent, result;
        base = 5;
        exponent = 4;
        result = pow(base, exponent);
        cout << "pow("<<base << "^" << exponent << ") = " << result;
        double x = 25;;
        result = sin(x);
        cout << "\nsin("<<x<<")= "<<result;
        return 0;
}
```

Output:

```
        pow(5^4) = 625
        sin(25)= -0.132352
```


11th_Comp Science_UM_Ch 11.indd 190 18-04-2023 15:30:06

11.5 صارف کی وضاحت کردہ فنکشنس (User – defined Functions)

11.5.1 تعارف

ہم کسی مخصوص کام کو انجام دینے کے لیے نئے فنکشنس کی بھی وضاحت کر سکتے ہیں۔ یہ صارف کی وضاحت کردہ فنکشنس (User – defined Functions) کے طور پر کہا جاتا ہے۔ صارف کی طرف سے مقرر کردہ فنکشنس صارف کی طرف سے بنائے جاتے ہیں۔ ایک فنکشن اختیاری طور پر ان پٹ پیرامیٹرز کی وضاحت کر سکتا ہے جو کال کرنے والوں کو فنکشن میں آرگیومنٹس دینے کے قابل بناتے ہیں۔ایک فنکشن اختیاری طور پر ایک قدر کو آؤٹ پٹ کے طور پر بھی واپس کر سکتا ہے۔فنکشنس ایک دوبارہ قابل استعمال بلاک میں عام کاموں کو سمیٹنے کے لیے مفید ہیں، مثالی اعتبار سے ایسے نام کے ساتھ جو واضح طور پر بیان کرتا ہے کہ فنکشن کیا کرتا ہے۔

11.5.2 فنکشن کی وضاحت (Function Definition)

C++ میں، کسی فنکشن کو پروگرام میں کہیں بھی استعمال کرنے سے پہلے اس کی وضاحت ہونی چاہیے۔ فنکشن کی وضاحت کا عمومی سنٹیکس ہے:

```
Return_Data_Type  Function_name(parameter list)
{
        Body of the function
}
```

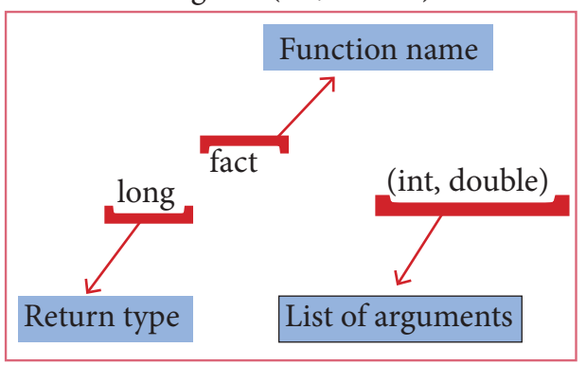
نوٹ:

1. Return_Data_Type کی کوئی بھی درست ڈیٹا C++ کی قسم ہے۔
2. Function_name صارف کی طرف سے بیان کردہ شناخت کنندہ ہے۔
3. پیرامیٹر (parameter) کی فہرست، جو اختیاری ہے، پیرامیٹرز کی ایک فہرست ہے، یعنی ڈیٹا کی قسموں سے پہلے متغیرات کی فہرست اور کوما (comma) سے الگ کی گئی ہے۔
4. فنکشن کا باڈی C++ اسٹیٹ منٹس (statements) پر مشتمل ہوتا ہے جو اس فنکشن کے مطلوبہ کام کو انجام دینے کے لیے درکار ہوتے ہیں۔

11.5.3 فنکشن پروٹو ٹائپ (Function Prototype)

C++ پروگرام کسی بھی تعداد میں فنکشنس پر مشتمل ہو سکتا ہے۔ لیکن، شروع کرنے کے لیے اس میں ہمیشہ صرف ایک مین() فنکشن پروگرام پر عملدرآمد کے لیے ہونا چاہیے۔ہم اپنی مرضی کے مطابق فنکشنس کی وضاحتیں کسی بھی ترتیب سے لکھ سکتے ہیں۔ ہم پہلے مین () فنکشن اور اس کے بعد دیگر تمام فنکشنس کی وضاحت کرسکتے ہیں یا مین () سے پہلے تمام ضروری فنکشنس کی وضاحت کرسکتے ہیں۔ متغیر اعلان کی طرح، پروگرام میں استعمال ہونے سے پہلے کسی فنکشن کا اعلان کرنا ضروری ہے۔ اعلامیہ کا بیان مین() فنکشن سے باہر دیا جا سکتا ہے۔

long fact (int, double)



تصویر 11.1

مندرجہ بالا پروٹو ٹائپ مرتب کرنے والے کو درج ذیل معلومات فراہم کرتا ہے۔

- فنکشن کی ریٹرن کی قدر لمبی قسم کی ہے۔
- حقیقت (fact) فنکشن کا نام ہے۔
- فنکشن کو دو آرگیو منٹس کے ساتھ کہا جاتا ہے: پہلی آرگیو منٹ int ڈیٹا کی قسم ہے۔ دوسری آرگیومنٹ ڈبل ڈیٹا (double data) کی قسم ہے۔

```
int display(int, int) // function prototype//
```

مندرجہ بالا فنکشن پروٹو ٹائپ ریٹرن ڈیٹا (return data) کی قسم، فنکشن کا نام اور رسمی پیرامیٹرز (formal parameters) یا آرگومینٹس (arguments) کی فہرست کے بارے میں تفصیلات فراہم کرتا ہے۔

11.5.4 کالعدم کمانڈ کا استعمال (Use of void command)

کالعدم قسم کے دو اہم مقاصد ہیں:

- اس بات کی نشاندہی کرنے کے لیے کہ فنکشن کوئی قدر واپس نہیں کرتا ہے۔
- ایک عام پوائنٹر (pointer) کا اعلان کرنا۔

11th_Comp Science_UM_Ch 11.indd 191 18-04-2023 15:30:06

11.5.5 فنکشن تک رسائی حاصل کرنا
(Accesing a function)

صارف کی وضاحت کردہ فنکشن کو واضح طور پر اس کا نام استعمال کرتے ہوئے اور مطلوبہ آرگیو منٹس کو پاس کرنے کے لیے بلایا جانا چاہیے۔ کمپائلر سے مراد فنکشن پروٹو ٹائپ ہے تاکہ یہ چیک کیا جا سکے کہ فنکشن کو صحیح طریقے سے بلایا گیا ہے یا نہیں۔ اگر آرگیومنٹ کی قسم پروٹو ٹائپ میں بیان کردہ ڈیٹا کی قسم سے بالکل مماثل نہیں ہے، تو کمپائلر ٹائپ کنورژن (type conversion) انجام دے گا، اگر ممکن ہو تو۔ اگر ٹائپ کنورژن ناممکن ہے تو کمپائلر ایک غلط پیغام تیار کرتا ہے۔

نوٹس

کالعدم ڈیٹا ٹائپ کمپائلر کی طرف اشارہ کرتا ہے کہ فنکشن قدر واپس نہیں کرتا ہے، یا بڑے سیاق و سباق میں کالعدم اشارہ کرتا ہے کہ اس میں کچھ نہیں ہے۔

مثال کے طور پر:

void fun(void)

مندرجہ بالا فنکشن پروٹو ٹائپ کمپائلر(compiler) کو بتاتا ہے کہ فنکشن () fun نہ تو کالنگ پروگرام (calling program)سے قدر وصول کرتا ہے اور نہ ہی کالنگ پروگرام کو کوئی قدر واپس کرتا ہے۔

مثال :

1	ڈسپلے()	بغیر کسی آرگیو منٹ کے اور واپسی کی قدر کے بغیر فنکشن کو کال کرنا
2	ڈسپلے (x, y)	آرگیومنٹ کے ساتھ اور واپسی کی قدر کے بغیر فنکشن کو کال کرنا
3	x = ڈسپلے ()	بغیر کسی آرگیو منٹ کے اور واپسی کی قدر کے ساتھ فنکشن کو کال کرنا
4	x = ڈسپلے (x, y)	آرگیومنٹ اور واپسی کی قدر کی ساتھ فنکشن کو کال کرنا

11.5.5.1 رسمی پیرامیٹرز اور اصل پیرامیٹرز یا آرگیو منٹس
(Formal parameters and actual parameters or Arguments)

آرگیومنٹس یا پیرامیٹرز کالنگ فنکشن (calling function) سے کالڈ فنکشن (called function) میں اقدار کو منتقل کرنے کا ذریعہ ہیں۔ پیرامیٹرز کے طور پر فنکشنس کی وضاحت میں استعمال ہونے والے متغیرات کو رسمی پیرامیٹرز (formal parameters) کے طور پر جانا جاتا ہے۔ فنکشن کال (function call) میں استعمال ہونے والے کانس ٹینٹس، متغیرات یا اکسپریشنس کو اصل پیرامیٹرز کے نام سے جانا جاتا ہے۔

```
#include <iostream>
Using namespace std;
int sum (int x, int y)
{
        return (x + y);
}
int main ()
{
        int a,b ;
        cout<<"\n Enter Number 1:";
        cin>>a;
        cout<<"\n Enter Number 2:";
        cin >>b;
        cout<<"\n The sum = "<<sum (a, b);
}
```

Function

Formal Parameter

Main Program

Actual Parameter

تصویر 11.2 رسمی اور حقیقی پیرامیٹرز

11th_Comp Science_UM_Ch 11.indd 192 18-04-2023 15:30:06

11.5.5.2 پہلے سے وضاحت کردہ آرگیو منٹس (Default Arguments)

C ++ میں، کوئی ایک فنکشن پروٹو ٹائپ کے رسمی پیرامیٹرز کو ڈیفالٹ ویلیوز (default values) تفویض کر سکتا ہے۔ پہلے سے وضاحت کردہ آرگیومنٹس فنکشن کو کال کرتے وقت کچھ آرگیومنٹس کو چھوڑنے کی اجازت دیتا ہے۔

کسی فنکشن کو کال کرتے وقت،

- کسی بھی گمشدہ آرگیو منٹس کے لیے، کمپائلر کالڈ فنکشن کے لیے پہلے سے وضاحت کردہ آرگیومنٹس میں اقدار کا استعمال کرتا ہے۔
- پہلے سے طے شدہ قدر متغیر شروعات کی شکل میں دی گئی ہے۔

مثال:

void defaultvalue(int n1=10, n2=100);

- پہلے سے وضاحت کردہ آرگیومنٹس جزوی یا بغیر آرگیومنٹ کے فنکشن کال اسٹیٹمنٹ (call statement) کی سہولت فراہم کرتے ہیں۔

مثال:

defaultvalue(x,y);
defaultvalue(200,150);
defaultvalue(150);
defaultvalue(x,150);

- ڈیفالٹ اقدار کو فنکشن پروٹو ٹائپ میں دائیں سے بائیں شامل کیا جا سکتا ہے، یعنی ہمارے پاس دلیل کی فہرست کے درمیان کسی دلیل کے لیے ڈیفالٹ قدر نہیں ہو سکتی۔

مثال:

void defaultvalue(int n1=10, n2);//
invalid prototype

void defaultvalue(int n1, n2 = 10);//
valid prototype

11.5.5.3 کانسٹنٹ آرگیومنٹس (Constant Arguments)

کانسٹنٹ متغیر کا اعلان const کی ورڈ (keyword) کا استعمال کرتے ہوئے کیا جا سکتا ہے۔ const کی ورڈ متغیر قدر کو مستحکم بناتا ہے۔ اعلان کرتے وقت کانسٹنٹ متغیر کو شروع کیا جانا چاہئے۔ کانسٹ موڈیفائر (const modifier) ایک متغیر کو ابتدائی قدر تفویض کرنے کے قابل بناتا ہے جسے بعد میں فنکشن کی باڈی (body of the function) میں تبدیل نہیں کیا جا سکتا۔

سنٹیکس:

<returntype><functionname> (const <datatype variable=value>)

مثال:

- int minimum(const int a=10);
- float area(const float pi=3.14, int r=5);

پروگرام 11.16

```
#include <iostream>
using namespace std;
double area(const double r,const double pi=3.14)
{
        return(pi*r*r);
}
int main ()
{
        double rad,res;
        cout<<"\nEnter Radius :";
        cin>>rad;
        res=area(rad);
        cout << "\nThe Area of Circle ="<<res;
        return 0;
}
```

Output:

```
Enter Radius :5
The Area of Circle =78.5
```


11th_Comp Science_UM_Ch 11.indd 193 18-04-2023 15:30:06

اگر متغیر قدر "r" کو r = 25 کے طور پر تبدیل کیا جاتا ہے؛ فنکشن "علاقہ – area " کی باڈی کے اندر پھر کمپائلر "صرف پڑھنے کے پیرامیٹر 'r' کی تفویض" کے طور پر ایک غلطی ظاہر کرے گا۔

```
double area(const double r,const double pi=3.14)
{
    r=25;
    return(pi*r*r);
}
```

11.6 فنکشنس کو کال کرنے کے طریقے
(Methods of calling functions)

C++ میں، آرگیومنٹس کو دو طریقوں سے فنکشن میں منتقل کیا جا سکتا ہے۔ آرگیومنٹس کو پاس کرنے کے طریقہ کار کی بنیاد پر، فنکشن کالنگ کے طریقوں کو کال بذریعہ ویلیو (call by Value) طریقہ اور کال بذریعہ ریفرنس (call by Reference) یا ایڈریس طریقہ (Address method) کے طور پر درجہ بندی کیا جا سکتا ہے۔

11.6.1 کال بذریعہ ویلیو طریقہ

یہ طریقہ ایک حقیقی پیرامیٹر (actual parameter) کی قدر کو فنکشن کے رسمی پیرامیٹر (formal parameter) میں کاپی کرتا ہے۔ اس صورت میں، فنکشن کے اندر رسمی پیرامیٹر میں کی گئی تبدیلیوں کا اصل پیرامیٹر پر کوئی اثر نہیں پڑے گا۔

پروگرام 11.17

```
#include<iostream>
using namespace std;
void display(int x)
{
        x=x*x;
        cout<<"\n\nThe Value inside display function (x*x):"<<x;
}
int main()
{
         int a;
        cout<<"\nExample : Function call by value:";
         cout<<"\n\nEnter the Value for A :";
         cin>>a;
        display(a);
        cout<<"\n\nThe Value inside main function "<<a;
        return(0);
}
```

Output :

```
Example : Function call by value
Enter the Value for A : 5
The Value inside display function (a * a) : 25
The Value inside main function 5
```


11th_Comp Science_UM_Ch 11.indd 194 18-04-2023 15:30:06

11.6.2 کال بذریعہ حوالہ یا اڈریس طریقہ
(Call by reference or address Method)

یہ طریقہ اصل آرگیومنٹ کے اڈریس کو رسمی پیرامیٹر میں کاپی کرتا ہے۔ چونکہ آرگیو منٹ کا اڈریس گزر چکا ہے، رسمی پیرامیٹر میں کی گئی کوئی بھی تبدیلی اصل پیرامیٹر میں ظاہر ہوگی۔

پروگرام 11.18

```
#include<iostream>
using namespace std;
void display(int &x) //passing address of a//
{
        x=x*x;
        cout<<"\n\nThe Value inside display function ( x*x) :"<<x ;
 }
int main()
{
int n1;
cout<<"\nEnter the Value for N1 :";
cin>>n1;
cout<<"\nThe Value of N1 is inside main function Before passing : "<< n1;
display(n1);
cout<<"\nThe Value of N1 is inside main function After passing (n1 x n1)  : "<< n1; return(0);
}
```

Output :

```
Enter the Value for N1 :45
The Value of N1 is inside main function Before passing : 45
The Value inside display function (n1 x n1) :2025
The Value of N1 is inside main function After passing (n1 x n1)  : 2025
```

نوٹ کریں کہ ڈسپلے () فنکشن میں تبدیلی صرف فنکشن ہیڈر (function header) میں ہے۔ پیرامیٹر x کے اعلان میں & علامت کا مطلب ہے کہ آرگیو منٹ ایک حوالہ متغیر (reference variable) ہے اور اس وجہ سے فنکشن کو حوالہ پاس کرکے کال کیا جائے گا۔ لہٰذا جب آرگیومنٹ n1 ڈسپلے() فنکشن میں منتقل کی جاتی ہے، متغیر x کو n1 کا پتہ ملتا ہے تاکہ مقام کا اشتراک کیا جائے۔ دوسرے الفاظ میں، متغیر x اور n1 ایک ہی میموری کی جگہ کا حوالہ دیتے ہیں۔ ہم مین () فنکشن میں نام n1 اور ڈسپلے() فنکشن میں نام x کو اسی اسٹوریج لوکیشن(storage location) کا حوالہ دینے کے لیے استعمال کرتے ہیں۔ لہٰذا، جب ہم x کی قدر کو تبدیل کرتے ہیں، تو ہم اصل میں n1 کی قدر کو تبدیل کر رہے ہیں۔

11.6.3 ان لائن فنکشن (Inline function)

عام طور پر کسی فنکشن میں کال اسٹیٹمنٹ (call statement) ایک کمپائلر کو فنکشن پر جانے کے لیے بناتا ہے (فنکشنس کی وضاحت STACKS میں محفوظ ہوتی ہے) اور کال اسٹیٹمنٹ کے بعد ہدایات پر واپس بھی جاتی ہے۔ اس سے پروگرام پر عمل درآمد کی رفتار کم ہو جاتی ہے۔ چھوٹے فنکشنس کی وضاحت کے لیے STACKS جیسے اوور ہیڈز (overheads) کو کم کرنے کے لیے ان لائن (Inline) فنکشنس کا استعمال کیا جا سکتا ہے۔

11th_Comp Science_UM_Ch 11.indd 195 18-04-2023 15:30:06

ایک ان لائن فنکشن سورس فائل میں عام فنکشن کی طرح لگتا ہے لیکن فنکشنس کا کوڈ براہ راست کالنگ پروگرام میں داخل کرتا ہے۔ فنکشن کو ان لائن کرنے کے لیے، فنکشن ہیڈر میں کلیدی لفظ ان لائن ڈالنا پڑتا ہے۔

سنٹیکس:

inline returntype functionname(datatype parameter 1, … datatype parameter n)

ان لائن فنکشنس کے فوائد:

- ان لائن فنکشنس تیزی سے کام کرتے ہیں لیکن میموری کی زیادہ جگہ درکار ہوتی ہے۔
- STACKS استعمال کرنے کی پیچیدگی کو کم کرتا ہے۔

پروگرام 11.19

```
#include <iostream>
using namespace std;
inline int add (int a , int b)
{
        int c=a+b;
        return(c);
}
int main ()
{
        int x,y,z;
        cout<<"\nEnter the First Number :";
        cin>>x;
        cout<<"\nEnter the second Number     :";
        cin>>y;
        z=add(x,y);
        cout << "\n sum of "<<x<<"+"<<y<<"="<<z;
        return 0;
}
```

Output:

```
Enter the First Number :10
Enter the second Number     :20
sum of 10+20=30
```

اگرچہ مندرجہ بالا پروگرام تالیف کے دوران نارمل فنکشن وضاحت فارمیٹ (definition format) میں لکھا گیا ہے فنکشن کوڈ a+b براہ راست کالنگ اسٹیٹمنٹ میں داخل کیا جائے گا یعنی **z=add(x,y);** یہ کالنگ اسٹیٹمنٹ کو **z = a+b;** کے طور پر تبدیل کرتا ہے۔

11.7 صارف کی وضاحت کردہ فنکشن ڈیکلریشنز کی مختلف شکلیں
(Different forms of User – defined function declarations)

11.7.1 ریٹرن کی قیمت کے بغیر اور پیرامیٹر کے بغیر ایک فنکشن

مندرجہ ذیل پروگرام ایک فنکشن کے لیے ایک مثال ہے جس میں کوئی ریٹرن اور کوئی آرگیومنٹ نہیں گزری ہے۔

11th_Comp Science_UM_Ch 11.indd 196 18-04-2023 15:30:06

فنکشن کا نام ڈسپلے() ہے، اس کے ریٹرن ڈیٹا کی قسم کالعدم (void) ہے اور اس میں کوئی آرگیومنٹ نہیں ہے۔

پروگرام 11.20

```
#include<iostream>
using namespace std;
void display()
{       cout<<"First C++ Program with Function"; }
int main()
{       display(); // Function calling statement//
        return(0);
}
```

Output :

```
        First C++ Program with Function
```

11.7.2 ریٹرن کی قدر کے ساتھ اور پیرامیٹر کے بغیر ایک فنکشن

(A Function with return value without parameter)

فنکشن کا نام ڈسپلے () ہے، اس کی ریٹرن کی قسم int ہے اور اس میں کوئی آرگیومنٹ نہیں ہے۔ ریٹرن اسٹیٹمنٹ کالنگ فنکشن میں ایک قدر واپس کرتا ہے اور پروگرام کنٹرول کو واپس کالنگ اسٹیٹمنٹ میں منتقل کرتا ہے۔

پروگرام 11.21

```
#include<iostream>
using namespace std;
int display()
{
        int a=10, b=5, s;
        s=a+b;
        return s;
}
int main()
{       int m=display();
        cout<<"\nThe Sum="<<m;
        return(0);
}
```

Output :

```
        The Sum=15
```


11th_Comp Science_UM_Ch 11.indd 197 18-04-2023 15:30:06

11.7.3 ریٹرن کی قدر کے بغیر اور پیرامیٹر کے ساتھ ایک فنکشن
(A function without return value and with parameter)

فنکشن کا نام ڈسپلے() ہے، اس کی ریٹرن کی قسم کالعدم ہے اور اس میں دو قدریں حاصل کرنے کے لیے دو پیرامیٹرز یا آرگیومینٹس x اور y ہیں۔ ریٹرن اسٹیٹمنٹ کنٹرول کو واپس کالنگ اسٹیٹمنٹ پر لوٹاتا ہے۔

پروگرام 11.22

```
#include<iostream>
using namespace std;
void display(int x, int y)
{
        int s=x+y;
        cout<<"The Sum of Passed Values: "<<s;
}
int main()
{
        int a=50,b=45;
          display(a,b);
        return(0);
}
```

Output :

```
The Sum of Passed Values: 95
```

11.7.4 ریٹرن کی قدر اور پیرامیٹر کے ساتھ ایک فنکشن
(A function with return value and with parameter)

فنکشن کا نام ڈسپلے () ہے، اس کی ریٹرن کی قسم int ہے اور اس میں دو قدریں حاصل کرنے کے لیے دو پیرامیٹرز یا آرگیومینٹس x اور y ہیں۔ ریٹرن اسٹیٹمنٹ کنٹرول کو واپس کالنگ اسٹیٹمنٹ پر لوٹاتا ہے۔

پروگرام 11.23

```
#include<iostream>
using namespace std;
int display(int x, int y)
{
        int s=x+y;
        return s;
}
int main()
{
        int a=45,b=20;
        int s=display(a,b);
        cout<<"\nExample:Function with Return Value and  with Arguments";
        cout<<"\nThe Sum of Passed Values: "<<s;
        return(0);
}
```


11th_Comp Science_UM_Ch 11.indd 198 18-04-2023 15:30:06

Output :

Example: Function with Return Value and with Arguments

The Sum of Passed Values: 65

11.8 فنکشن سے ریٹرن (Returning from function)

فنکشن سے ریٹرن، ریٹرن اسٹیٹمنٹ (return statement) کا استعمال کرکے کی جاتی ہے۔

ریٹرن اسٹیٹمنٹ عمل کو روکتا ہے اور کالنگ فنکشن پر واپس آجاتا ہے۔جب ایک ریٹرن اسٹیٹمنٹ عمل میں لایا جاتا ہے، اس وقت فنکشن کو فوری طور پر ختم کر دیا جاتا ہے۔

11.8.1 ریٹرن اسٹیٹمنٹ (The return statement)

ریٹرن اسٹیٹمنٹ کسی فنکشن سے ریٹرن کے لیے استعمال ہوتا ہے۔ اسے جمپ اسٹیٹمنٹ (jump statement) کے طور پر درجہ بندی کیا گیا ہے کیونکہ یہ فنکشن کے عمل کو ختم کرتا ہے اور کنٹرول کو کالڈ اسٹیٹمنٹ (called statement) میں منتقل کرتا ہے۔ ریٹرن میں اس کے ساتھ کوئی قدر وابستہ ہو سکتی ہے یا نہیں۔ اگر ریٹرن کی کوئی قدر اس سے وابستہ ہے، تو وہ قدر کالنگ اسٹیٹمنٹ (calling statement) کے لیے ریٹرن کی قدر بن جاتی ہے۔یہاں تک کہ وائڈ فنکشن (void function) کے لیے بھی پیرامیٹر کے بغیر ریٹرن اسٹیٹمنٹ فنکشن کو ختم کرنے کے لیے استعمال کیا جا سکتا ہے۔

سنٹیکس:

return expression/variable;

مثال :

```
return(a+b);   return(a);
return; // to terminate the function
```

11.8.2 ریٹرننگ اقدار: (Returning values)

وہ فنکشنس جو کوئی قدر نہیں لوٹاتا انہیں کالعدم (void) قرار دیا جاتا ہے۔ کسی فنکشن کی ڈیٹا ٹائپ کو int سمجھا جاتا ہے، اگر کسی ڈیٹا کی قسم کا واضح طور پر ذکر نہیں کیا جاتا ہے۔ مثال کے طور پر،

مثال کے طور پر :

```
int add (int, int);
add (int, int);
```

دونوں پروٹوٹائپس (prototypes) میں، ریٹرن کی قدر int ہے، کیونکہ C ++ میں کسی فنکشن کی ریٹرن کی قدربطور ڈیفالٹ int ہوتی ہے جب کوئی ریٹرن قدر واضح طور پر نہیں دی جاتی ہے۔ درج ذیل مثالوں کو دیکھیں:

نمبر شمار	فنکشنس پروٹو ٹائپ (Function Prototype)	ریٹرن کی قسم (return type)
1	int sum(int, float)	int
2	float area(float, float)	float
3	char result()	char
4	double fact(int n)	double

11th_Comp Science_UM_Ch 11.indd 199 18-04-2023 15:30:07

غیر عددی اقدار کو لوٹانا (Returning Non-integer values)

سٹرنگ کو کالنگ سٹیٹمنٹ میں بھی واپس کیا جا سکتا ہے۔

پروگرام 11.24

```
#include<iostream>
#include<string.h>
using namespace std;
char *display()
{       return (“chennai”);   }
int main()
{
        char s[50];
        strcpy(s,display());
        cout<<”\nExample:Function with Non Integer Return”<<s;
        return(0);}
```

Output :

Example: Function with Non Integer Return Chennai

11.9 ری کرسیو فنکشن
(Recursive Function)

ایک فنکشن جو خود کو کال کرتا ہے اسے ری کرسیو فنکشن کہا جاتا ہے۔ اور اس تکنیک کو ری کرسیو (اعادہ) کے نام سے جانا جاتا ہے۔

مثال 1: ری کرسیو کا استعمال کرتے ہوئے نمبر کا فیکٹوریل (factorial)

پروگرام 11.25

```
#include <iostream>
using namespace std;
int factorial(int); // Function prototype //
int main()
{
        int no;
        cout<<"\nEnter a number to find its factorial: ";
        cin >> no;
        cout << "\nFactorial of Number " << no <<" = " << factorial(no);
        return 0;
}
int factorial(int m)
{
        if (m > 1)
        {
        return m*factorial(m-1);
        }
        else
        {
        return 1;
}
}
```

Output :

Enter a number to find its factorial: 5
Factorial of Number 5 = 120

11th_Comp Science_UM_Ch 11.indd 200 18-04-2023 15:30:07

نوٹ: فنکشن پروٹو ٹائپ لازمی ہے کیونکہ فنکشن فیکٹوریل () ((factorial) مین () ((main) فنکشن کے بعد دیا جاتا ہے۔

11.10 متغیرات کے اسکوپ (حد) اصول
(Scope rules of Variables)

متغیر کی رسائی سے اسکوپ مراد ہے۔ C++ میں اسکوپس کی چار اقسام ہیں۔ وہ یہ ہیں: لوکل اسکوپ (Local scope)، فنکشن اسکوپ (Function scope)، فائل اسکوپ (file scope) اور کلاس اسکوپ (Class scope)۔

11.10.1 تعارف

اسکوپ، متغیر کا ایک خطہ یا زندگی ہے اور بڑے پیمانے پر بات کی جائے تو تین جگہیں ہیں، جہاں متغیرات کا اعلان کیا جا سکتا ہے،

- ایک بلاک کے اندر جسے لوکل متغیرات (local variables) کہا جاتا ہے۔
- کسی فنکشن کے اندر فنکشن متغیرات (function variables) کہلاتا ہے۔
- تمام فنکشنس کے باہر جنہیں عالمی متغیرات (global variables) کہا جاتا ہے۔
- کلاس کے اندر کلاس متغیر (class variable) یا ڈیٹا ممبر (data members) کہلاتا ہے۔

11.10.2 لوکل اسکوپ (local scope) :

- ایک لوکل متغیر کو بلاک کے اندر بیان کیا جاتا ہے۔ کوڈ کا ایک بلاک منحنی قوسین { } کے ساتھ شروع اور ختم ہوتا ہے۔
- لوکل متغیر کا اسکوپ وہ بلاک ہے جس میں اس کی وضاحت کی گئی ہے۔
- ایک لوکل متغیر تک اس کے اعلان (declaration) کے بلاک کے باہر سے رسائی حاصل نہیں کی جا سکتی ہے۔
- ایک لوکل متغیر اس کے بلاک میں داخل ہونے پر بنایا جاتا ہے اور باہر نکلنے پر ختم ہو جاتا ہے۔

11.10.3 فنکشن اسکوپ: (Function scope)

- کسی فنکشن کے اندر اعلان کردہ متغیرات کا اسکوپ فنکشن بلاک اور اس میں موجود تمام ذیلی بلاکس تک پھیلا ہوا ہے۔
- فنکشن اسکوپ متغیر کا لائف ٹائم (life time)، فنکشن بلاک کا لائف ٹائم ہے۔ رسمی پیرامیٹرز کا اسکوپ فنکشن اسکوپ ہے۔

11.10.4 فائل اسکوپ: (file scope)

- تمام بلاکس اور فنکشنس کے اوپر اعلان کردہ ایک متغیر (بشمول ()main) فائل کا اسکوپ رکھتا ہے۔ فائل اسکوپ متغیر کا لائف ٹائم پروگرام کا لائف ٹائم ہوتا ہے۔
- فائل اسکوپ متغیر کو عالمی متغیر بھی کہا جاتا ہے۔

پروگرام 11.26

```
//Demo to test all Scopes//
#include<iostream>
using namespace std;
int file_var=20;          //Declared within File - file scope variable
void add(int x)
{
        int m;            //Declaration of  variable m in add () - Function scope variable
        m=x+30+file_var;
        cout<<"\n The Sum = "<<m;
}
```


11th_Comp Science_UM_Ch 11.indd 201 18-04-2023 15:30:07

```
int main ( )
{
int a =5,b=10;
if(a>b)
{
        int t;              // local to this if block - Local variable
        t=a+20;
 }
cout<<t;
add(a);
cout<<m;
cout<<"\nThe File Variable = "<<file_var;
return(0);
}
```

Error
In function 'int main()':
[Error] 't' was not declared in main()

On compilation the Program 11.28, the compiler prompts an error message: The variable **t** is not accessible. Because the life time of a local variable is the life time of a block in its state of execution.

[Error] 'm' was not declared in this scope

The variable **m i**s not accessible. Because the life time of the function scope variable is the life time of a block in its state of execution.

11.10.5 کلاس اسکوپ: (class scope)

- ایک کلاس صارف کی وضاحت کردہ ڈیٹا کی قسم کو بنانے اور لاگو کرنے کا ایک نیا طریقہ ہے۔ کلاسز (classes) مختلف اقسام کے ڈیٹا کو ایک ساتھ پیک کرنے کا طریقہ فراہم کرتی ہیں۔
- ڈیٹا ممبران (Data members) ڈیٹا متغیرات ہیں جو کلاس کی نقوش یا خاصیات کی نمائندگی کرتے ہیں۔

کلاس کے طالب علم میں mark1، mark2 اور کل ڈیٹا متغیر ہوتے ہیں۔ اس کا اسکوپ صرف کلاس کے طالب علم کے اندر ہے۔	class student
	{
	private :
	int mark1, mark2, total;
	};

نوٹ: کلاس اسکوپ پر بعد میں باب "کلاسز اور آبجیکٹ-classes and object" میں بحث کی جائے گی۔

11th_Comp Science_UM_Ch 11.indd 202 18-04-2023 15:30:07

11.10.6 اسکوپ ریزولوشن آپریٹر (Scope resolution operator)

اسکوپ آپریٹر متغیر کے پوشیدہ اسکوپ کو ظاہر کرتا ہے۔ اسکوپ ریزولوشن آپریٹر (::) کو درج ذیل مقاصد کے لیے استعمال کیا جاتا ہے۔

- عالمی متغیر تک رسائی کے لیے جب ایک ہی نام کا مقامی متغیر ہو۔ اسکوپ ریزولوشن آپریٹر کا استعمال کرتے ہوئے ایک مثال۔

پروگرام 11.27

```
// Program to show that we can access a global variable
// using scope resolution operator :: when there is a local
// variable with same name //
#include<iostream>
using namespace std;
int x=45;  // Global Variable x
int main()
{
 int x = 10; // Local Variable x
 cout << "\nValue of global x is " << ::x;
 cout << "\nValue of local x is " << x;
 return 0;
}
```

Output:

```
Value of global x is 45
Value of local x is 10
```

یاد رکھنے کے نکات

- ایک بڑے پروگرام کو عام طور پر چھوٹے سائز کے بلاکس میں تقسیم کیا جا سکتا ہے جسے فنکشنس کہتے ہیں۔
- فنکشنس کو پہلے سے وضاحت کردہ (pre-defined) یا بلٹ ان (built-in) یا لائبریری فنکشنس (library functions) اور یوزر ڈیفائنڈ (User-defined) فنکشنس میں درجہ بندی کیا جا سکتا ہے۔
- یوزر ڈیفائنڈ فنکشنس صارف کے ذریعے بنائے جاتے ہیں۔
- وائڈ فنکشن کمپائلر کو بتاتا ہے کہ فنکشن کچھ نہیں لوٹاتا ہے۔
- ریٹرن اسٹیٹمنٹ (return statement) کالنگ فنکشن (calling function) کی قدر واپس کرتا ہے اور پروگرام کنٹرول کو کالنگ فنکشن میں واپس منتقل کرتا ہے۔
- C ++ میں کسی فنکشن کی ڈیفالٹ ریٹرن قسم int ہوتی ہے۔
- ایک فنکشن جو خود کو کال کرتا ہے اسے ری کرسیو فنکشن (recursive function) کے نام سے جانا جاتا ہے۔
- متغیر کی رسائی سے اسکوپ مراد ہے۔
- اسکوپس کی چار اقسام ہیں۔ وہ ہیں: لوکل اسکوپ، فنکشن اسکوپ، فائل اسکوپ اور کلاس اسکوپ۔
- اسکوپ آپریٹر (::) متغیر کے پوشیدہ اسکوپ کو ظاہر کرتا ہے۔

11th_Comp Science_UM_Ch 11.indd 203 18-04-2023 15:30:07



عملی مشق

درج ذیل مسائل کو حل کرنے کے لیے C ++ پروگرام لکھیں:

1. وہ پروگرام جو دو اسٹرنگ کو پڑھتا ہے اور پہلی سٹرنگ کو دوسری سے جوڑتا ہے۔ مثال کے طور پر، اگر پہلی سٹرنگ بطور تمل (Tamil) اور دوسری سٹرنگ بطور ناڈو (nadu) درج کی گئی ہے، تو پروگرام کو تمل ناڈو (Tamilnadu) پرنٹ کرنا چاہیے۔ سٹرنگ لائبریری ہیڈر استعمال کریں۔
2. پروگرام جو سٹرنگ کو پڑھتا ہے اور اسے بڑے حروف میں تبدیل کرتا ہے۔ مطلوبہ ہیڈر فائلیں شامل کریں۔
3. پروگرام جو یہ جانچتا ہے کہ آیا دیا گیا حرف حرف تہجی ہے یا نہیں۔ اگر یہ حروف تہجی ہے، چاہے یہ چھوٹے حروف ہوں یا بڑے حرف؟ مطلوبہ ہیڈر فائلیں شامل کریں۔
4. دو آرگیومنٹس / پیرامیٹرز – ڈبل x اور int n کے ساتھ ، C++ میں فنکشن سم سیریز ( ) (function sumseries()) کی وضاحت لکھیں - فنکشنس کو ڈبل قسم کی قدر واپس کرنی چاہئے اور اسے درج ذیل سیریز کا مجموعہ انجام دینا چاہئے:

   x-x2 /3! + x3 / 5! - x4 / 7! + x5 / 9! -... upto n terms.

5. پروگرام جو ایک فنکشن کیلک ( ) کو طلب کرتا ہے جس میں دو عدد اور ایک ارتھ میٹک (arithmetic) آپریٹر شامل ہوتا ہے اور متعلقہ نتیجہ پرنٹ کرتا ہے۔





تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. درج ذیل میں سے کون سی ہیڈر فائل معیاری I/O پہلے سے وضاحت کردہ فنکشنس کی وضاحت کرتی ہے؟
   stdio.h (A) math.h (B) string.h (C) ctype.h (D)
2. یہ چیک کرنے کے لیے کون سا فنکشن استعمال ہوتا ہے کہ آیا کوئی حرف حرفی نمبر ہے یا نہیں۔
   isalpha() (A) isdigit() (B) isalnum() (C) islower() (D)
3. کون سا فنکشنس پروگرام پر عمل درآمد شروع کرتا ہے؟
   isalpha() (A) isdigit() (B) main() (C) islower() (D)
4. مندرجہ ذیل میں سے کون سا فنکشن ریٹرن قدر کے ساتھ اور بغیر کسی آرگیو منٹ کے ہے؟
   x=display(int, int) (A) x=display() (B) y=display(float) (C) display(int) (D)
5. add(int, int); کے فنکشن پروٹو ٹائپ کی ریٹرن ڈیٹا قسم کون سی ہے؟
   int (A) float (B) char (C) double (D)
6. درج ذیل میں سے کون سا اسکوپ آپریٹر ہے؟
   > (A) & (B) % (C) :: (D)

11th_Comp Science_UM_Ch 11.indd 204 18-04-2023 15:30:07

سیکشن - B

بہت مختصر جوابات دیں

1. فنکشنس کی وضاحت کریں۔
2. ()اسٹرلین ()(strlen) فنکشنس کے بارے میں لکھیں۔
3. کالعدم (void) ڈیٹا کی قسم کی کیا اہمیت ہے؟
4. پیرامیٹر کیا ہے اور اس کی اقسام کی فہرست بنائیں؟
5. لوکل اسکوپ پر ایک نوٹ لکھیں۔

سیکشن - C

مختصر جوابات دیں

1. بلٹ ان فنکشنس کیا ہے؟
2. ()isupper اور ()toupper فنکشنس میں کیا فرق ہے؟
3. ()strcmp فنکشنس کے بارے میں لکھیں۔
4. C ++ میں ()pow فنکشن پر مختصر نوٹ لکھیں۔
5. پروٹو ٹائپ کمپائلر کو کیا معلومات فراہم کرتا ہے؟
6. پہلے سے وضاحت کردہ آرگیومنٹس کیا ہیں؟ مثال دیں۔

سیکشن - D

تفصیل سے بتائیں۔

1. مناسب مثال کے ساتھ قدر کے طریقہ سے کال (call by value method) کی وضاحت کریں۔
2. ری کرشن (recursion) کیا ہے؟ری کرشن کا استعمال کرتے ہوئے دیئے گئے نمبر کے فیکٹوریل کو تلاش کرنے کے لئے ایک پروگرام لکھیں۔
3. فنکشنس کی ریٹرن کی مختلف شکلیں کیا ہیں؟ مثال کے ساتھ وضاحت کریں۔
4. مثال کے ساتھ متغیر کے اسکوپ کی وضاحت کریں۔
5. کسی بھی عدد کو قبول کرنے کے لیے ایک پروگرام لکھیں اور اسے ریورس (reverse) کریں۔

11th_Comp Science_UM_Ch 11.indd 205 18-04-2023 15:30:07

باب 12

اریز اور اسٹرکچرس
(Arrays and Structures)

سیکھنے کے مقاصد



اس باب کو سیکھنے کے بعد طلباء

- اریز کا استعمال کرتے ہوئے اسٹرکچرڈ ڈیٹا (structured data) کی قسم جاننے
- اریز کی اقسام کو جاننے
- مختلف قسم کی اریز کو جوڑنے کے لیے پروگرام لکھنے

کے قابل ہوجائیں گے۔

12.1 تعارف

متغیرات کو ڈیٹا ذخیرہ کرنے کے لیے استعمال کیا جاتا ہے۔ یہ متغیرات C++ میں بنیادی تعمیراتی بلاکس میں سے ایک ہیں۔ایک واحد متغیر ایک واحد قدر کو ذخیرہ کرنے کے لیے استعمال کیا جاتا ہے جسے میموری میں کہیں بھی استعمال کیا جا سکتا ہے۔ کچھ حالات میں، ہمیں ایک ہی قسم کی متعدد اقدار کو ذخیرہ کرنے کی ضرورت ہے۔ اس صورت میں، اسے ایک ہی ڈیٹا کی قسم کے متعدد متغیرات کی ضرورت ہے۔ تمام اقدار میموری میں کہیں بھی بغیر کسی ترتیب کے محفوظ کی جاتی ہیں۔

مثال کے طور پر، 100 طلباء کے رول نمبرز کو ذخیرہ کرنے کے لیے، اسے 100 متغیرات کی ضرورت ہے جن کا نام رول1،رول 2، رول 3،..... رول 100۔ 100 متغیرات کا اعلان کرنا اور تمام رول نمبرز کو ذخیرہ کرنا بہت مشکل ہو جاتا ہے۔ C++میں، اریز کا تصور ایک متغیر میں متعدد اقدار کو ذخیرہ کرنے میں مدد کرتا ہے۔ لفظی طور پر، ارے (array) کا معنی ہے "ایک سے زیادہ"۔ دوسرے الفاظ میں، ارےایک ہی قسم کی ایک سے زیادہ قدروں کو ذخیرہ کرنے کا ایک آسان طریقہ ہے جس کا حوالہ ایک عام نام سے ہے۔ C++میں ارے اخذ کردہ ڈیٹا کی قسم بھی ہے۔

"ایک ارے (array) ایک ہی قسم کے متغیرات کا مجموعہ ہے جس کا حوالہ عام نام سے دیا جاتا ہے"۔ایک ارے میں، قدروں کو ایک ہی قسم کے عناصر کی ایک مقررہ تعداد میں ترتیب وار میموری میں محفوظ کیا جاتا ہے۔لہذا، ایک عددی ارے میں عدد کی ترتیب ہوتی ہے۔ ایک کریکٹرارے حروف کی ایک ترتیب رکھتی ہے، وغیرہ۔ ارے کے سائز کو اس کے طول و عرض (dimension) کے طور پر کہا جاتا ہے۔

12.2 اَریز کی اقسام:
(Types of Arrays)

C++ میں استعمال ہونے والی ارےوں کی مختلف اقسام ہیں۔ وہ ہیں:

- ایک جہتی اَریز (One -dimensional Arrays)
- دو جہتی اریز (Two -dimensional Arrays)
- کثیر جہتی اریز (Multi -dimensional Arrays)

12.2.1 ایک جہتی اَرے
(One -dimensional Array)

یہ ایک ارے کی سب سے آسان شکل ہے۔ ایک جہتی ارے ان اقدار کی نمائندگی کرتی ہے جو ایک رو (rows) میں یا ایک کالم (column) میں محفوظ ہوتی ہیں۔

اعلامیہ

سنٹیکس:

```
<data type><array_name> [<array_size>];
```

data type اَریز کی بنیادی قسم کااعلان کرتا ہے، جو کہ ارے میں موجود ہر عنصر کی قسم ہے۔

array_name اس نام کی وضاحت کرتا ہے جس کے ساتھ ارے کا حوالہ دیا جائے گا۔

array_size وضاحت کرتا ہے کہ ارے میں کتنے عناصر ہوں گے۔ سائز مربع بریکٹ کے ساتھ بیان کیا جانا چاہئے [ ]۔

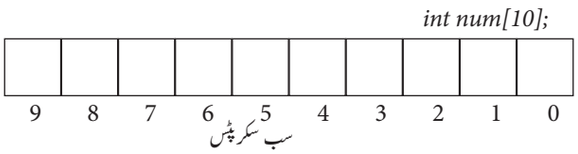
مثال:

```
int num[10];
```


11th_Comp Science_UM_Ch 12.indd 206 19-04-2023 09:45:24

مندرجہ بالا اعلامیہ میں، "num" نامی ایک ارے کااعلان 10 عناصر (10 مختلف اقدار کو ذخیرہ کرنے کے لیے میموری کی جگہ) کے ساتھ عددی قسم کے طور پر کیا گیا ہے۔

مندرجہ بالاڈیکلریشن (declaration) کے لیے، مرتب کرنے والا 10 میموری لوکیشنز (باکسس) مختص کرتا ہے جس کا حوالہ ایک عام نام "num" سے دیا گیا ہے جیسا کہ ذیل میں دیا گیا ہے۔



ہر عنصر (میموری باکس) کا ایک منفرد انڈیکس نمبر ہوتا ہے جو 0 سے شروع ہوتا ہے جسے "سب اسکرپٹ – subscript " کہا جاتا ہے۔سب سکرپٹ ہمیشہ 0 سے شروع ہوتا ہے اور یہ ایک غیر دستخط شدہ عددی قدر ہونی چاہیے۔ ایک ارے کے ہر عنصر کو اس کے نام سے اسکوائر بریکٹ کے اندر سب اسکرپٹ انڈیکس (index) کے ساتھ حوالہ دیا جاتا ہے۔ مثال کے طور پر، num [3] سے مراد اَریز میں 4th عنصر ہے۔

مختلف ڈیٹا کی اقسام کے ساتھ کچھ مزید ارے کے اعلانات:

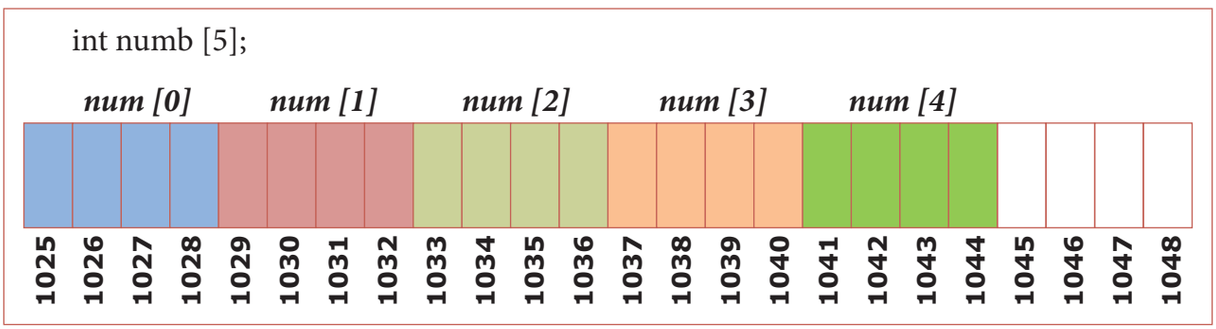
```
char emp_name[25];          // character array named emp_name with size 25

float salary[20];           // floating-point array named salary with size 20

int a[5], b[10], c[15]; // multiple arrays are declared of type int
```

ایک جہتی ارے کی میموری کی نمائندگی

ایک ارے کو رکھنے کے لیے درکار اسٹوریج کی مقدار براہ راست قسم اور سائز سے متعلق ہے۔ مندرجہ ذیل اعداد و شمار پانچ عناصر کے ساتھ ایک ارے کی میموری مختص کو ظاہر کرتا ہے۔



مندرجہ بالا اعداد و شمار واضح طور پر ظاہر کرتا ہے کہ، ارے نمبر 5 عناصر کے ساتھ ایک عددی ارے ہے۔ Dev-C++ مرتب کرنے والے کے مطابق، ہر int قسم کے متغیر کے لیے 4 بائٹس مختص کیے گئے ہیں۔یہاں، ارے میں مکمل طور پر 5 عناصر ہیں، جہاں ہر عنصر کے لیے، 4 بائٹس مختص کیے جائیں گے۔ مجموعی طور پر، اس ارے کے لیے 20 بائٹس مختص کیے جائیں گے۔

Datatype	Turbo C++	Dev C++
char	1	1
int	2	4
float	4	4
long	4	4
double	8	8
long double	10	10

ایک ارے کے لیے مختص میموری کی جگہ کا حساب درج ذیل فارمولے سے لگایا جا سکتا ہے۔

11th_Comp Science_UM_Ch 12.indd 207 19-04-2023 09:45:25

ارے کی قسم کے لیے مختص کردہ بائٹس کی تعداد × ابتدائی عناصر کی تعداد

آغاز

ایک ارے کو اس کے ڈیکلریشن کے وقت شروع کیا جا سکتا ہے۔ جب تک ایک ارے شروع نہیں کی جاتی ہے، تمام ارے کے عناصر ردی کی قیمتوں پر مشتمل ہوتا ہے۔

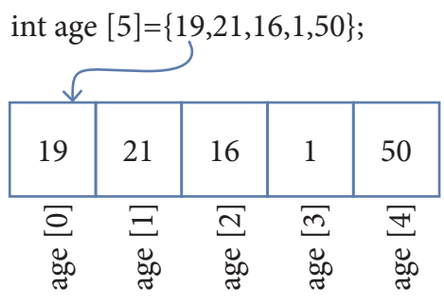
سنٹیکس:

<datatype> <array_name> [size] = {value 1, value 2, ……, value n};

مثال

int age[5]={19,21,16,1,50};

مندرجہ بالا مثال میں، ارے کا نام 'عمر' ہے جس کا سائز 5 ہے۔ اس صورت میں، پہلا عنصر 19، عمر[0] میں ذخیرہ کیا جاتا ہے، دوسرا عنصر21، عمر[1] میں ذخیرہ ہوتا ہے اور اسی طرح جیسا کہ تصویر 12.1 میں دکھایا گیا ہے۔



تصویر 12.1

ایک ارے میں اقدار کااعلان اور آغاز کرتے وقت، قدروں کو منحنی قوسین (curly braces) کے اندر دیا جانا چاہئے یعنی۔ {...}

ایک ارے کا سائز اختیاری ہو سکتا ہے جب اعلان کے دوران ارے شروع کی جاتی ہے۔

مثال:

int age[]={ 19,21,16,1,50};

اوپر کی ابتدا میں، ابتدائی کے ساتھ اعلان (declaration) میں ارے کا سائز براہ راست متعین نہیں کیا گیا ہے۔ لہذا، سائز کا تعین کمپائلر (compiler) کے ذریعہ کیا جاتا ہے جو اقدار کی کل تعداد پر منحصر ہوتا ہے۔ اس صورت میں، ارے کا سائز پانچ ہے۔ ارے شروع کرنے کی مزید مثالیں:

float x[5] = {5.6, 5.7, 5.8, 5.9, 6.1};

char vowel[6] = {'a', 'e', 'i', 'o', 'u', '\0'};

رن ٹائم (run time) کے دوران ایک ارے میں اقدار کو قبول کرنا:

رن ٹائم کے دوران ارے کی خلیوں (cells) میں قدریں داخل کرنے کے لیے متعدد تفویض بیانات درکار ہیں۔ لوپ کے لیے مثالی طور پر ارے کے عناصر کے ذریعے اعادہ کرنے کے لیے موزوں ہے۔

```
// عمل درآمد کے دوران اقدار درج کرنا
#include <iostream>
using namespace std;
int main()
{
        int num[5];
        for(int i=0; i<5; i++)
        {
                cout<< "\n Enter value " << i+1 << "= ";
                cin>>num[i];
        }
}
```

مندرجہ بالا پروگرام میں، لوپ کے اندر اسٹیٹمنٹ کو 5 بار انجام دینے کے لیے ایک فار لوپ (for loop) بنایا گیا ہے۔ لوپ کے ہر اعادہ کے دوران، کاؤٹ (cout) اسٹیٹمنٹ آپ کو " ویلیو درج کریں ....." کا اشارہ کرتا ہے۔ اور cin قدر حاصل کرتا ہے اور اسے num [i]; میں محفوظ کرتا ہے۔

مندرجہ ذیل ٹیبل مندرجہ بالا کوڈ بلاک (code block) کے عمل کو ظاہر کرتا ہے۔

11th_Comp Science_UM_Ch 12.indd 208 19-04-2023 09:45:25

Iteration	i <5	cout << "\n Enter value " << i+1 << "= ";	cin>>num [i];	Received value stored in memory		i++ (i=i+1)
1	5 > 0 (T)	Enter value 1 =	num[0] = 5	num[0]	5	1
2	5 > 1 (T)	Enter value 2 =	num[1] = 10	num[1]	10	2
3	5 > 2 (T)	Enter value 3 =	num[2] = 15	num[2]	15	3
4	5 > 3 (T)	Enter value 4 =	num[3] = 20	num[3]	20	4
5	5 > 4 (T)	Enter value 4 =	num[25 = [4	num[4]	25	5
6	5 > 5 (F)	Exit from Loop				

نوٹ

فار لوپ میں، انڈیکس i **(index i)** کا اعلان ابتدائی قدر 0 (زیرو) کے ساتھ کیا جاتا ہے۔ چونکہ زیادہ تر معاملات میں، لوپ انڈیکس کی ابتدائی قدر کو ارے سب اسکرپٹ (subscript) کی نمائندگی کے طور پر استعمال کیا جائے گا۔

ارے کے عناصر کی رسائی (Accessing array elements)

ارے عناصر کو پروگرام میں کہیں بھی استعمال کیا جا سکتا ہے جیسا کہ ہم عام متغیر کی صورت میں کرتے ہیں۔ ایک ارے کے عناصر کی ارے نام کے ساتھ رسائی حاصل کی جاتی ہے جس کے بعد مربع بریکٹ میں سب اسکرپٹ انڈیکس (subscript index) ہوتا ہے۔

مثال:

```
cout<<num[3];
```

مندرجہ بالا اسٹیٹمنٹ میں، num[3] سے مراد ارے کا چوتھا عنصر ہے اور cout اسٹیٹمنٹ num [3] کی قدر کو ظاہر کرتا ہے۔

نوٹ

بریکٹ (bracket) میں سب اسکرپٹ ایک متغیر، ایک کانس ٹینٹ یا ایک ایکسپریشن ہوسکتا ہے جو ایک عدد کا اندازہ کرتا ہے۔

// ارے عناصر کی رسائی

```
#include <iostream>
using namespace std;
int main()
{
        int num[5] = {10, 20, 30, 40, 50};
        int t=2;
        cout<<num[2] <<endl; // S1
        cout<<num[3+1] <<endl; // S2
        cout<<num[t=t+1]; // S3
}
```

output:

```
30
50
40
```


11th_Comp Science_UM_Ch 12.indd 209 19-04-2023 09:45:25

مذکورہ پروگرام میں، اسٹیٹمنٹ **S1** تیسرے عنصر (سب اسکرپٹ انڈیکس 2) کی قدر دکھاتا ہے۔ **S2**، پانچویں عنصر کی قدر ظاہر کرے گا (یعنی سب اسکرپٹ ویلیو 1+3=4 ہے)۔ اسی طرح اسٹیٹمنٹ **S3** چوتھے عنصر کی قدر ظاہر کرے گا۔

10 اقدار کو داخل کرنے اور طاق اور جفت نمبروں کی تعداد کو شمار کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int num[10], even=0, odd=0;
        for (int i=0; i<10; i++)
        {
                cout<< "\n Enter Number " << i+1 <<"= ";
                cin>>num[i];
                if (num[i] % 2 == 0)
                 ++even;
                else
                 ++odd;
        }
        cout << "\n There are "<< even <<" Even Numbers";
        cout << "\n There are "<< odd <<" Odd Numbers";
}
```

Output:

```
Enter Number 1= 78
Enter Number 2= 51
Enter Number 3= 32
Enter Number 4= 66
Enter Number 5= 41
Enter Number 6= 68
Enter Number 7= 27
Enter Number 8= 65
Enter Number 9= 28
Enter Number 10= 94
There are 6 Even Numbers
There are 4 Odd Numbers
```

(ہاٹس: (HOTS) if کے بجائے کنڈیشنل آپریٹر کا استعمال کرتے ہوئے مذکورہ پروگرام کو دوبارہ لکھیں)

ایک جہتی ارے میں تلاش کرنا:

تلاش کرنا، نمبروں کے دیئے گئے سیٹ میں موجود کسی خاص قدر کو دریافت کرنے کا عمل ہے۔ لکیری تلاش (linear search) یا ترتیب وار تلاش (sequential search) فہرست کے ہر عنصر کا اس قدر کے ساتھ موازنہ کرتی ہے جسے اس وقت تک تلاش کرنا ہوتا ہے جب تک کہ ارے میں موجود تمام عناصر کو منتقل اور موازنہ نہ کیا جائے۔

11th_Comp Science_UM_Ch 12.indd 210 19-04-2023 09:45:25

لکیری تلاش کے لیے پروگرام

```
#include <iostream>
using namespace std;
int main()
{
        int num[10], val, id=-1;
        for (int i=0; i<10; i++)
        {
        cout<< "\n Enter value " << i+1 <<"= ";
        cin>>num[i];
        }
        cout<< "\n Enter a value to be searched: ";
        cin>>val;
    for (int i=0; i<10; i++)
    {
          if (arr[i] == val)
          {    id= i;
               break;
          }
    }
        if(id==-1)
        cout<< "\n Given value is not found in the array..";
        else
        cout<< "\n The value is found at the position" << id+1;
        return 0;
}
```

مندرجہ بالا پروگرام ایک ارے کو پڑھتا ہے اور قدروں کو تلاش کرنے کا اشارہ کرتا ہے۔ یہ فہرست کے ہر عنصر کا اس قدر سے موازنہ کرتا ہے جس کی تلاش اس وقت تک کی جاتی ہے جب تک کہ ارے میں موجود تمام عناصر کو منتقل اور موازنہ نہ کیا جائے۔

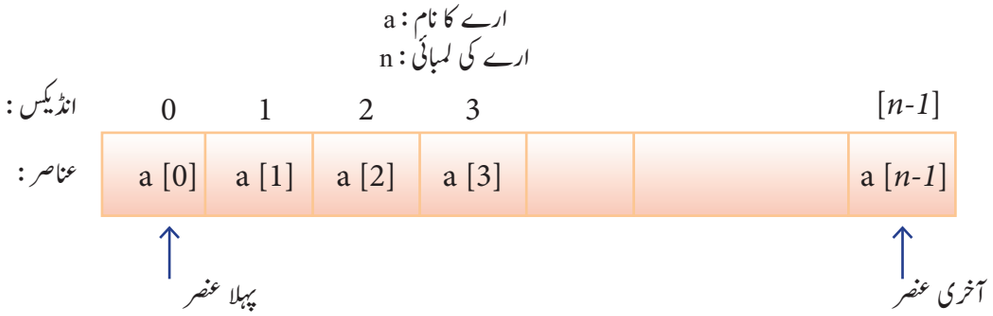
سٹرنگس (Strings)

سٹرنگ کو حروف کی ترتیب کے طور پر وضاحت کیا جاتا ہے جہاں ہر کریکٹر ایک حرف، نمبر یا علامت ہو سکتا ہے۔ ہر عنصر میموری کی ایک بائٹ پر قبضہ کرتا ہے۔ ہر اسٹرنگ کو ایک null (‘\0’, ASCII code 0) کریکٹر کے ذریعے ختم کیا جاتا ہے جسے سٹرنگ کے آخر میں شامل کرنا ضروری ہے۔ C ++ میں، سٹرنگ کی نمائندگی کرنے کے لیے کوئی بنیادی ڈیٹا ٹائپ نہیں ہے۔ اس کے بجائے، یہ سٹرنگ کو ایک جہتی کیریکٹر ارے کے طور پر لاگو کرتا ہے۔ کریکٹر ارے (character array) کا اعلان کرتے وقت، اسے آخر میں ایک null کریکٹر بھی رکھنا پڑتا ہے، اور اس طرح، کریکٹر ارے کا سائز سٹرنگ کی لمبائی سے ایک کریکٹر لمبا ہونا چاہیے۔

کریکٹر ارے (سٹرنگ) کی تخلیق (Character Array (string) creation)

کسی بھی قسم کی ارے (array) بنانے کے لیے، ارے کا سائز (لمبائی) پہلے سے معلوم ہونا ضروری ہے، تاکہ میموری کے مقامات کو ارے کے سائز کے مطابق مختص کیا جا سکے۔ ایک بار ایک ارے بن جانے کے بعد، اس کی لمبائی طے ہو جاتی ہے اور رن ٹائم (run time) کے دوران اسے تبدیل نہیں کیا جا سکتا۔ یہ تصویر 12.2 میں دکھایا گیا ہے۔

11th_Comp Science_UM_Ch 12.indd 211 19-04-2023 09:45:25



تصویر 12.2

سنٹیکس

ارے کا ڈیکلریشن یہ ہے:

char array_name[size];

مندرجہ بالا ڈیکلریشن میں، ارے کا سائز ایک غیر دستخط شدہ عددی قدر ہونا چاہیے۔

مثال کے طور پر،

char country[6];

یہاں، ارے حروف کی ترتیب کو ذخیرہ کرنے کے لیے 6 بائٹس میموری محفوظ رکھتی ہے۔ سٹرنگ کی لمبائی 5 حروف سے زیادہ نہیں ہو سکتی اور آخر میں ایک جگہ نل کریکٹر (null character) کے لیے مخصوص ہے۔

// کیریکٹر ارے کو ظاہر کرنے کے لیے پروگرام۔

```
#include <iostream>
using namespace std;
int main()
 {
        char country[6];
        cout<< "Enter the name of the country: ";
        cin>>country;
        cout<<" The name of the country is "<<country;
}
```

OUTPUT

```
Enter country the name: INDIA
The country name is INDIA
```

آغاز (Initialization)

کیریکٹر ارے کو اس کے ڈیکلریشن کے وقت شروع کیا جا سکتا ہے۔ سنٹیکس ذیل میں دکھایا گیا ہے:

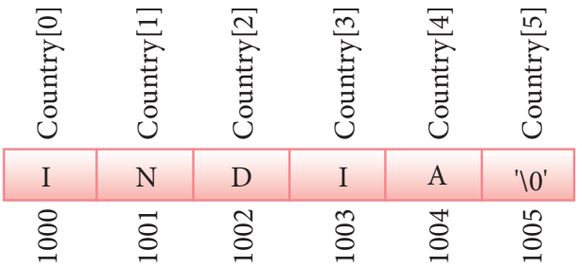
char array_name[size]={ list of characters separated by comma or a string } ;

مثال کے طور پر،

char country[6]="INDIA";

مندرجہ بالا مثال میں، "INDIA" کے متن میں 5 حروف ہیں جو ملک کے لیے ابتدائی قدر کے طور پر تفویض کیے گئے ہیں۔ متن دوہرے اقتباسات میں بند ہے۔ میموری کی نمائندگی کو تصویر 13.3 میں دکھایا گیا ہے۔

11th_Comp Science_UM_Ch 12.indd 212 19-04-2023 09:45:25



تصویر 12.3

مندرجہ بالا میموری کی نمائندگی میں، ہر کیریکٹر میموری میں ایک بائٹ پر قبضہ کرتا ہے۔ سٹرنگ کے آخر میں، کمپائلر کے ذریعہ ایک نل کیریکٹر (null character) خود بخود شامل ہو جاتا ہے۔

C++ کیریکٹر ارے کو شروع کرنے کے دوسرے طریقے بھی فراہم کرتا ہے:

char country[6]={‘I’, ‘N’, ‘D’, ‘I’, ‘A’, ‘\0’};

char country[]=“INDIA”;

char country[]={‘I’, ‘N’, ‘D’, ‘I’, ‘A’, ‘\0’};

اگر ارے کے سائز کا واضح طور پر تذکرہ نہیں کیا گیا ہے، تو کمپائلر فہرست میں موجود عناصر کی تعداد کی بنیاد پر خود بخود ارے کے سائز کا حساب لگاتا ہے اور اس کے مطابق جگہ مختص کرتا ہے۔

سٹرنگ کی ابتدا میں، اگر تمام حروف شروع نہیں کیے گئے ہیں، تو باقی حروف NULL سے بھر جائیں گے۔

مثال:

```
char str[5]={'5','+','A'};
str[0]; ---> 5
str[1]; ---> +
str[2]; ---> A
str[3]; ---> NULL
str[4]; ---> NULL
```

نوٹ

ابتداء کے دوران، عناصر کی ارے کو اس کے سائز سے زیادہ شروع نہیں کیا جا سکتا۔

مثال کے طور پر

```
char str[2]={'5','+','A','B'}; // Invalid
```

مندرجہ بالا مثال میں، کمپائلر "ارے کے چارس (array of chars) کے لیے ابتداء –اسٹرنگ بہت لمبی ہے" غلط پیغام دکھاتا ہے۔

پیلینڈروم ہے یا نہیں ، چیک کرنے کے لیے پروگرام لکھیں

```
#include<iostream>
#include<string.h>
using namespace std;
int main( )
{
        int i, j, len, flag =1;
        char a [20];
        cout<<"Enter a string:";
        cin>>a;
        for(len=strlen(a);a[len]!='\0';++len)
        for(i=0,j=len-1;i<len/2;++i,--j)
        {
        if(a[j]!=a[i])
                        flag=0;
                        break;
        }
```


11th_Comp Science_UM_Ch 12.indd 213 19-04-2023 09:45:25

```
        if(flag==1)
                cout<<"\n The String is palindrome";
        else
                cout<<"\n The String is not palindrome";
        return 0;
}
```

Output:

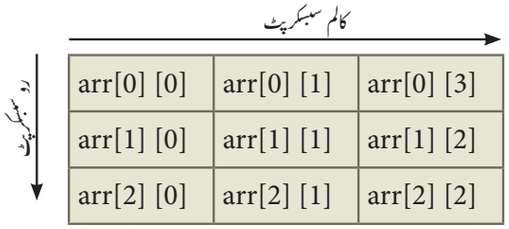
```
        Enter a string : madam
        The String is palindrome
```

12.3 دو - جہتی ارے
(Two – dimensional array)

دو جہتی (2D) اریز ملتے جلتے عناصر کا مجموعہ ہیں جہاں عناصر کو رووں اور کالموں کی مخصوص تعداد میں ذخیرہ کیا جاتا ہے۔ ایک مثال m x n میٹرکس (matrix) جہاں m رووں کی تعداد کو ظاہر کرتا ہے اور n کالموں کی تعداد کو ظاہر کرتا ہے جیسا کہ تصویر 12.4 میں دکھایا گیا ہے۔

int arr[3][3];

2D ارے تصوراتی میموری کی نمائندگی



ارے arr کو میٹرکس کی تصوراتی شکل میں 3 رووں اور 3 کالموں کے ساتھ دیکھا جا سکتا ہے۔ یہاں نوٹ کرنے کی بات یہ ہے کہ سب اسکرپٹ 0 [0] [0] arr سے شروع ہوتا ہے جو پہلے عنصر کی نمائندگی کرتا ہے۔

تصویر 12.4

12.3.1 2-D ارے کا ڈکلریشن (Declaration of 2-D array)

2-D ارے کا ڈکلریشن یہ ہے

data-type array_name[row-size][col-size];

مندرجہ بالاڈیکلریشن میں، ڈیٹا کی قسم سے مراد کسی بھی درست C++ ڈیٹا کی قسم ہے، ارے _ نام سے مراد 2-D ارے کا نام ہے، رو کے سائز سے مراد رووں کی تعداد ہے اور2-D ارے میں کول سائز (col-size) سے مراد کالموں کی تعداد ہے۔

مثال کے طور پر

int A[3][4];

مندرجہ بالا مثال میں، A ایک 2-D ارے ہے، 3 رووں کی تعداد کو ظاہر کرتا ہے اور 4 کالموں کی تعداد کو ظاہر کرتا ہے۔ یہ ارے زیادہ سے زیادہ 12 عناصر کو رکھ سکتی ہے۔

نوٹ

ارے سائز ایک غیر دستخط شدہ عددی قدر ہونا چاہیے جو 0 سے زیادہ ہے۔ ارے میں، کالم کا سائز لازمی ہے لیکن رو کا سائز اختیاری ہے۔

11th_Comp Science_UM_Ch 12.indd 214 19-04-2023 09:45:25

2-D ارے کی دیگر مثالیں یہ ہیں:

```
int A[3][3];
float x[2][3];
char name[5][20];
```

12.3.2 دو-جہتی ارے کا آغاز

2-D ارے کے ڈیکلریشن کے وقت ارے کو ایک سے زیادہ طریقوں سے شروع کیا جا سکتا ہے۔

مثال کے طور پر

```
int matrix[4][3]={
{10,20,30},// Initializes row 0
{40,50,60},// Initializes row 1
{70,80,90},// Initializes row 2
{100,110,120}// Initializes row 3
};
int matrix[4][3]={10,20,30,40,50,60,70,80,90,100,110,120};
```

ارے کی رو کا سائز اختیاری ہے لیکن کالم کا سائز لازمی ہے

مثال کے طور پر

```
int matrix[][3]={
{10,20,30},// row 0
{40,50,60},// row 1
{70,80,90},// row 2
{100,110,120}// row 3
};
```

12.3.3 دو-جہتی ارے کی رسائی

دو-جہتی ارے اس میں کسی خاص عنصر تک رسائی کے لیے دو اشاریہ کی قدروں (index values) کا استعمال کرتی ہے، جہاں پہلا اشاریہ رو کی قدر کی وضاحت کرتا ہے اور دوسرا اشاریہ کالم کی قدر کی وضاحت کرتا ہے۔

```
matrix[0][0]=10;// Assign 10 to the first element of the first row
matrix[0][1]=20;// Assign 20 to the second element of the first row
matrix[1][2]=60;// Assign 60 to the third element of the second row
matrix[3][0]=100;// Assign 100 to the first element of the fourth row
```


11th_Comp Science_UM_Ch 12.indd 215 19-04-2023 09:45:25

دو میٹرکس کے اضافے کے لیے ایک پروگرام لکھیں

```
#include<iostream>
#include<conio>
using namespace std;
int main()
{
        int row, col, m1[10][10], m2[10][10], sum[10][10];
        cout<<"Enter the number of rows : ";
        cin>>row;
        cout<<"Enter the number of columns : ";
        cin>>col;
        cout<< "Enter the elements of first matrix: "<<endl;
        for (int i = 0;i<row;i++ )
        for (int j = 0;j <col;j++ )
        cin>>m1[i][j];
        cout<< "Enter the elements of second matrix: "<<endl;
        for (int i = 0;i<row;i++ )
        for (int j = 0;j<col;j++ )
        cin>>m2[i][j];
        cout<<"Output: "<<endl;
        for (int i = 0;i<row;i++ )
        for (int j = 0;j<col;j++ )
        {
        sum[i][j]=m1[i][j]+m2[i][j];
        cout<<sum[i][j]<<" ";
        }
        cout<<endl<<endl;
        }
getch();
return 0;
}
```

```
Enter the number of rows :   2
Enter the number of column : 2
Enter the elements of first matrix:
1
1
1
1
Enter the elements of second matrix:
1
1
1
1
Output:
2   2
2   2
```


11th_Comp Science_UM_Ch 12.indd 216 19-04-2023 09:45:25

12.3.4 2-D ارے کی میموری نمائندگی

عام طور پر، دو- جہتی ارے کو میٹرکس (matrix) کے طور پر دیکھا جا سکتا ہے۔ 2-D ارے کا تصوراتی منظر ذیل میں دکھایا گیا ہے:

int A[4][3];

A[0][0]	A[0][1]	A[0][2]
A[1][0]	A[1][1]	A[1][2]
A[2][0]	A[2][1]	A[2][2]
A[3][0]	A[3][1]	A[3][2]

مندرجہ بالا مثال میں، 2-D ارے نام A میں 4 رویں اور 3 کالم ہیں۔

ایک- جہتی کی طرح، 2-D ارے عناصر مسلسل میموری میں محفوظ ہوتے ہیں۔

2-D ارے میموری نمائندگی کی دو قسمیں ہیں۔ وہ یہ ہیں:

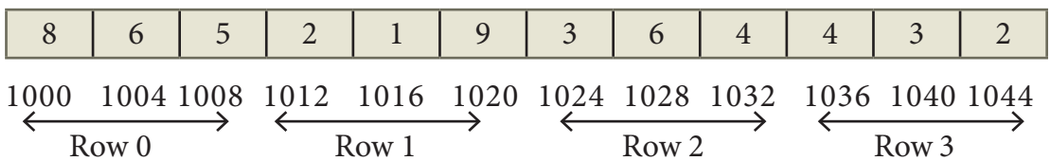
- رو- میجر آرڈر(Row-major order)
- کالم- میجر آرڈر (Column-major order)

مثال کے طور پر

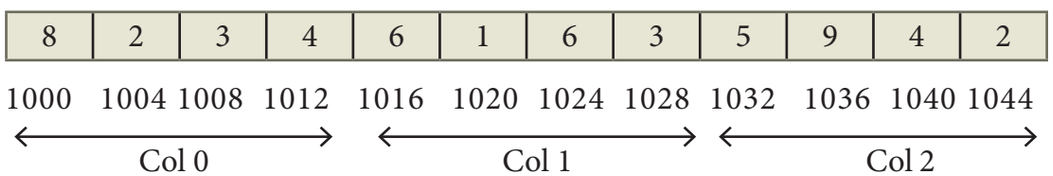
int A[4][3]={ { 8,6,5}, { 2,1,9}, {3,6,4}, {4,3,2} }

رو میجر آرڈر

رو میجر آرڈر، تمام عناصر کو رو در رو مسلسل میموری (continuous memory) جگہوں پر ذخیرہ کیا جاتا ہے، یعنی تمام عناصر پہلی رو میں، پھر دوسری رو میں وغیرہ۔ رو میجر آرڈر کی میموری نمائندگی، جیسا کہ ذیل میں دکھایا گیا ہے۔



کالم میجر آرڈر



12.4 اسٹرنگس کی ارے
(Array of strings)

اسٹرنگس کی ایک ارے دو- جہتی کیریکٹر کی ارے ہے۔ پہلے انڈیکس (روز-rows) کا سائز سٹرنگس کی تعداد کو ظاہر کرتا ہے اور دوسرے انڈیکس (کالم) کا سائز ہر اسٹرنگ کی زیادہ سے زیادہ لمبائی کو ظاہر کرتا ہے۔ عام طور پر، ہر اسٹرنگ کے آخر میں نل کیریکٹر کو ایڈجسٹ کرنے کے لیے سٹرنگس کی ارے کا اعلان اس طرح کیا جاتا ہے۔ مثال کے طور پر، 2-D ارے میں ڈکلریشن ہے:

char Name[6][10];

مندرجہ بالا ڈکلریشن میں، 2-D ارے میں دو اشاریے ہیں جو رو کے سائز اور کالم کے سائز کا حوالہ دیتے ہیں، یعنی 6 سے مراد روز (rows) کی تعداد اور 10 سے مراد کالموں کی تعداد ہے

11th_Comp Science_UM_Ch 12.indd 217 19-04-2023 09:45:25

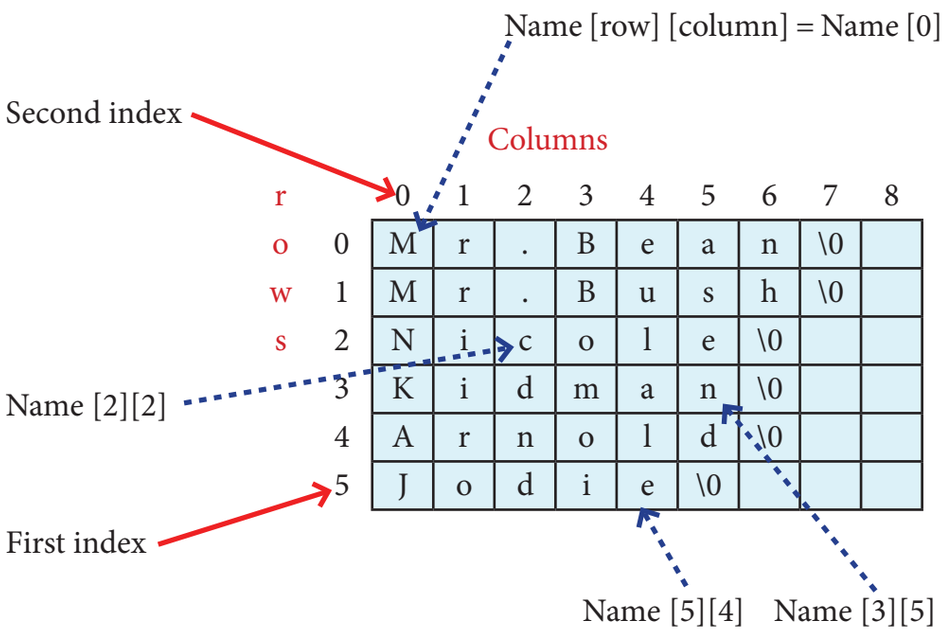
12.4.1 آغاز (Initialization)

مثال کے طور پر

```
char Name[6][10] = {"Mr. Bean", "Mr.Bush", "Nicole", "Kidman", "Arnold", "Jodie"};
```

مندرجہ بالا مثال میں، 2-D ارے کو 6 سٹرنگس کے ساتھ شروع کیا گیا ہے، جہاں ہر اسٹرنگ زیادہ سے زیادہ 9 کیریکٹرس لمبی ہے، کیونکہ آخری کریکٹر نل (null) ہے۔

2-D ارے کا میموری ترتیب ذیل میں دکھایا گیا ہے اور تمام اسٹرنگس (strings) مسلسل جگہوں پر محفوظ ہیں۔



2-D کریکٹر ارے کا استعمال کرتے ہوئے اسٹرنگس کی ارے کو ظاہر کرنے کے لیے C++ پروگرام

```
#include<iostream>
using namespace std;
int main( )
{
        // initialize 2d array
        char colour [4][10]={"Blue","Red","Orange",
                        "yellow"};

        // printing strings stored in 2d array
        for (int i=0;  i <4; i++)
        cout  <<  colour [i] << "\n";
}
```

Output:

```
Blue
Red
Orange
Yellow
```


11th_Comp Science_UM_Ch 12.indd 218 19-04-2023 09:45:25

کیس اسٹڈی:

(1) 10 طلباء کے نمبروں کو قبول کرنے کے لیے ایک پروگرام لکھیں اور اوسط، زیادہ سے زیادہ اور کم سے کم نمبر تلاش کریں۔

اسٹرکچرس (Structures)

12.5 اسٹرکچرس کا تعارف

اسٹرکچر صارف کی طرف سے واضح کردہ ہے جس میں ڈیٹا کی مختلف اقسام کے ساتھ ڈیٹا آئٹمز کا مجموعہ ہوتا ہے۔ یہ مخلوط ڈیٹا کی اقسام کے متغیرات کو ایک اکائی میں اکٹھا کرنے کی اجازت دیتا ہے۔

12.5.1 اسٹرکچرس کا مقصد

کسی بھی صورت حال میں جب یکساں ڈیٹا-قسم کی اشیاء کی نمائندگی کرنے کے لیے ایک سے زیادہ متغیرات کی ضرورت ہوتی ہے،تو ارے کو استعمال کیا جا سکتا ہے۔ اگر عناصر مختلف ڈیٹا کی اقسام کے ہیں، تو ارے سپورٹ نہیں کر سکتی۔ اگر ایک سے زیادہ متغیر استعمال کیے جاتے ہیں، تو وہ میموری میں محفوظ کیے جا سکتے ہیں لیکن ملحقہ مقامات (adjacent locations) پر نہیں۔ یہ تلاش کے دوران وقت کی کھپت کو بڑھاتا ہے۔یہ اسٹرکچر مختلف ڈیٹا کی اقسام کو ایک ہی منطقی عنصر کے حصے کے طور پر ایک دوسرے سے ملحق ایک میموری حصے میں ذخیرہ کرنے کی سہولت فراہم کرتا ہے۔

12.5.2 اسٹرکچرس کا ڈیکلریشن اور وضاحت

اسٹرکچر کا اعلان کلیدی لفظ 'struct' کا استعمال کرتے ہوئے کیا جاتا ہے۔ اسٹرکچر بنانے کا سنٹیکس ذیل میں دیا گیا ہے۔

```
struct structure_name {
        type member_name1;
        type member_name2;
} reference_name;
```

اسٹرکچر کی تعریف کے ساتھ ڈیکلریشن کردہ اشیاء کو عالمی آبجیکٹ کہا جاتا ہے۔

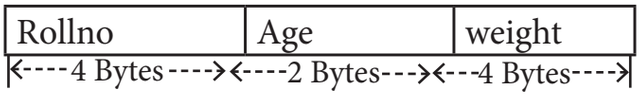
ایک اختیاری فیلڈ حوالہ_ نام کا استعمال اسٹرکچر کی قسم کی اشیاء کو براہ راست اعلان کرنے کے لیے کیا جا سکتا ہے۔

مثال:

```
struct Student
{
        long rollno;
        int age;
        float weight;
} ;
```

اسٹرکٹ (struct) کے اوپر بیان میں، تین متغیرات رول نمبر ، عمر، اور وزن استعمال کیے گئے ہیں۔اسٹرکچرس کے اندر موجود یہ متغیرات (ڈیٹا عنصر) کو اراکین (یا فیلڈس-fields) کہا جاتا ہے۔ طالب علم کے اسٹرکچرس کو استعمال کرنے کے لیے، اسٹوڈنٹ قسم کے ایک متغیر کاڈیکلریشن کیا جاتا ہے اور میموری کی تقسیم کو تصویر 12.5 میں دکھایا گیا ہے۔

11th_Comp Science_UM_Ch 12.indd 219 19-04-2023 09:45:25



تصویر 12.5 میموری ایلوکیشن

struct Student balu; // create a Student structure for Balu

یہ طالب علم کی قسم کے متغیر کی وضاحت کرتا ہے جس کا نام بالو ہے۔ عام متغیر کی طرح، اسٹرکچر متغیر خود اس متغیر کے لیے میموری مختص کرتا ہے۔ ایک ہی اسٹرکٹ کی قسم کے متعدد متغیرات کی وضاحت کرنا ممکن ہے:

struct Student frank; // create a structure for Student Frank.

مثال کے طور پر، اسٹرکچر آبجیکٹ بالو (Balu) اور فرینک (Frank) کو بھی اسٹرکچر ڈیٹا کی قسم کے طور پر اعلان کیا جا سکتا ہے:

```
struct Student
{
long rollno;
int age;
float weight;
} balu, frank;
```

12.5.3 اسٹرکچر کے عناصر کا حوالہ دینا

ایک بار جب طالب علم کی اسٹرکچر کی قسم کی دو اشیاء (بالو اور فرینک) کا اعلان کر دیا جاتا ہے، تو ان کے اراکین تک براہ راست رسائی حاصل کی جا سکتی ہے۔ اس کے لیے سنٹیکس آبجیکٹ کے نام اور ممبر کے نام کے درمیان ایک ڈاٹ (.)(dot) استعمال کر رہا ہے ۔ مثال کے طور پر، اسٹوڈنٹ کے اسٹرکچرس کے عناصر تک اس طرح رسائی حاصل کی جا سکتی ہے:

```
balu.rollno
balu.age
balu.weight
frank.rollno
frank.age
frank.weight
```


11th_Comp Science_UM_Ch 12.indd 220 19-04-2023 09:45:26

(گمنام اسٹرکچر بمقابلہ نامزد اسٹرکچر)

بغیر نام/ٹیگ کے اسٹرکچر کو گمنام اسٹرکچر کہا جاتا ہے۔

```
struct
{
long rollno;
    int age;
    float weight;
} student;
```

طالب علم کو مندرجہ بالا اسٹرکچر کے حوالہ نام کے طور پر کہا جا سکتا ہے اور عناصر تک طالب علم کی طرح رسائی حاصل کی جا سکتی ہے۔ طالب علم .رول نمبر، طالب علم. عمر اور طالب علم. وزن۔

12.5.4 اسٹرکچر کے عناصر کو شروع کرنا

قدروں کو اسٹرکچر عناصر کو تفویض کیا جا سکتا ہے جیسا کہ متغیرات کو قدریں تفویض کرنا۔

مثال

```
balu.rollno= “702016”;
balu.age= 18;
balu.weight= 48.5;
```

نیز، قدروں کو براہ راست تفویض کیا جا سکتا ہے جیسا کہ اریز (arrays) کو قدریں تفویض کرنا۔

```
balu={702016, 18, 48.5};
```

12.5.5 اسٹرکچر کی تفویض

عناصر کی قدروں کو انفرادی طور پر تفویض کرنے کے بجائے اسٹرکچرس کو براہ راست تفویض کیا جا سکتا ہے۔

مثال

اگر مہیش (Mahesh) اور پراوین (Praveen) کی عمر ایک جیسی ہے اور ایک ہی قد اور وزن ہے تو مہیش کی اقدار کو پراوین میں نقل کیا جا سکتا ہے۔

```
struct Student
{
    int  age;
    float height, weight;
}mahesh;
```

Structure assignment is possible only if both structure variables/ objects are same type.

مہیش کی عمر 17 سال ہے اور قد اور وزن بالترتیب 164.5 اور 52.5 ہے۔ درج ذیل بیان تفویض کو انجام دے گا۔

```
mahesh = {17, 164.5, 52.5};
praveen =mahesh;
```

پراوین کو وہی عمر، قد اور وزن تفویض کرے گا۔

11th_Comp Science_UM_Ch 12.indd 221 19-04-2023 09:45:26

مثالیں:

درج ذیل C++ پروگرام کی بورڈ (keyboard) کے ذریعے طلبہ کی معلومات پڑھتا ہے اور وہی دکھاتا ہے

```
#include <iostream>
using namespace std;
struct Student
{
        int  age;
        float height, weight;
} mahesh;
void main( )
{
        cout<< " Enter the age:"<<endl;
        cin>>mahesh.age;
        cout<< "Enter the height:"<<endl;
        cin>>mahesh.height;
        cout<< "Enter the weight:"<<endl;
        cin>>mahesh.weight;
        cout<< "The values entered  for Age, height and weight are"<<endl;
        cout<<mahesh.age<<     "\t"<<mahesh.height<<     "\t"<<Mahesh.
weight;
}
```

Output:

```
Enter the age:
18
Enter the height:
160.5
Enter the weight:
46.5
The values entered  for Age, height and weight are
18      160.5           46.5
```

یاد رکھنے کے نکات

- اسٹرکچر صارف کی طرف سے وضاحت کردہ ہے جس میں ڈیٹا کی مختلف اقسام کے ساتھ ڈیٹا آئٹمز کا مجموعہ ہوتا ہے۔
- اسٹرکچر کا اعلان کلیدی لفظ 'struct' کا استعمال کرتے ہوئے کیا جاتا ہے۔
- اسٹرکچر کے عناصر کا حوالہ اس کے آبجیکٹ کے نام کے بعد ڈاٹ (.) آپریٹر اور پھر ممبر کا نام استعمال کرتے ہوئے دیا جاتا ہے۔
- بغیر نام/ٹیگ کے اسٹرکچرس کو گمنام اسٹرکچر کہا جاتا ہے۔
- اسٹرکچر کے عناصر کو یا تو علیحدہ تفویض بیانات کا استعمال کرتے ہوئے یا ڈیکلریشن کے وقت اس کی اقدار کو منحنی قوسین سے گھیر کر شروع کیا جا سکتا ہے۔
- ایک اسٹرکچر آبجیکٹ کو دوسرے اسٹرکچرس والے آبجیکٹ کو بھی تفویض کیا جاسکتا ہے صرف اسی صورت میں جب دونوں اشیاء ایک ہی اسٹرکچر کی قسم کی ہوں۔
- دوسرے اسٹرکچرس کے اندر اعلان کردہ اسٹرکچر کو نیسٹڈ اسٹرکچر (nested structure) کہا جاتا ہے۔
- ایک اسٹرکچر اس کے رکن عنصر کے طور پر ارے پر مشتمل ہو سکتا ہے۔
- اسٹرکچر متغیر کی ارے (array) بھی بنائی جا سکتی ہے۔

11th_Comp Science_UM_Ch 12.indd 222 19-04-2023 09:45:26

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں۔

1. مندرجہ ذیل میں سے کون سا ایک ہی قسم کے متغیرات کا مجموعہ ہے جس کا حوالہ عام نام سے دیا جاتا ہے؟

int (a) float (b) Array (c) class (d)

2. int age[]={6,90,20,18,2}; اس ارے میں کتنے عناصر ہیں؟

2 (a) 5 (b) 6 (c) 4 (d)

3. cin>>n[3]; ، یہ بیان کس عنصر کی قدر کو قبول کرتا ہے؟

2 (a) 3 (b) 4 (c) 5 (d)

4. پہلے سے طے شدہ طور پر، ایک اسٹرنگ کس کیریکٹر کے ساتھ ختم ہوتا ہے؟

\0 (a) \t (b) \n (c) \b (d)

5. اسٹرکچر کی وضاحت کس سے ختم ہوتی ہے؟

: (a) } (b) ; (c) :: (d)

6. جب اسٹرکچر کا اعلان کیا جائے گا تو کیا ہو گا؟

(a) یہ کوئی میموری مختص نہیں کرے گا (b) یہ میموری کو مختص کرے گا

(c) اس کا اعلان اور آغاز کیا جائے گا (d) اس کا صرف اعلان کیا جائے گا

7. ایک اسٹرکچر کا ڈیکلریشن ذیل میں دیا گیا ہے۔

```
struct Time
{
int hours;
int minutes;
int seconds;
}t;
```

مندرجہ بالا ڈیکلریشن کا استعمال کرتے ہوئے درج ذیل میں سے کون سا سیکنڈز کا حوالہ دیتا ہے۔

(a) ٹائم . سیکنڈس (b) ٹائم:: سیکنڈس (c) سیکنڈس (d) ٹی. سیکنڈس

8. مندرجہ ذیل میں سے کون سا اسٹرکچر مناسب طریقے سے وضاحت کیا گیا ہے؟

struct {int num;} (a) struct sum {int num;} (b)

struct sum int sum; (c) struct sum {int num;}; (d)

11th_Comp Science_UM_Ch 12.indd 223 19-04-2023 09:45:26

9. اسٹرکچر کا ڈیکلریشن ذیل میں دیا گیا ہے۔

```
struct employee
{
int empno;
char ename[10];
}e[5];
```

مندرجہ بالا ڈکلریشن کا استعمال کرتے ہوئے درج ذیل میں سے کون سا بیان درست ہے۔

(a) cout<<e[0].empno<<e[0].ename; (b) cout<<e[0].empno<<ename;

(c) cout<<e[0]->empno<<e[0]->ename; (d) cout<<e.empno<<e.ename;

10. اسٹرکچر کے رکن تک رسائی حاصل کرتے وقت، ڈاٹ آپریٹر کے بائیں جانب شناخت کنندہ کا نام کیا ہوتا ہے؟

(a) اسٹرکچر ویری ایبل (b) اسٹرکچر ٹیگ

(c) اسٹرکچر ممبر (d) اسٹرکچر فنکشن

سیکشن - B

بہت ہی مختصر جوابات دیں

1. ایک ارے میں ٹراورسل (Traversal) کیا ہے؟
2. اسٹرنگس کیا ہے؟
3. دو -جہتی ارے کا اعلان کرنے کے لیے سنٹیکس کیا ہے؟
4. اسٹرکچر کی وضاحت کریں ۔ اس کا استعمال کیا ہے؟
5. درج ذیل اسٹرکچرس کی وضاحت میں کیا خرابی ہے۔

```
struct employee{ in teno;char ename[20];char dept; }
Employee e1,e2;
```

سیکشن - C

مختصر جوابات دیں

1. ایک ارے کی وضاحت کریں؟ اقسام کیا ہیں؟
2. اسٹرنگس کی ارے (array of strings) کے متعلق ایک نوٹ لکھیں۔
3. درج ذیل کوڈ میں 'S' سے شروع ہونے والے تمام طلباء کے ناموں کا خلاصہ ہوتا ہے اور اسے ظاہر کرتے ہیں۔ مطلوبہ بیانات کے ساتھ خالی جگہ پر کریں۔

```
struct student {int exam no,lang,eng,phy,che,mat,csc,total;char name[15];};
int main()
{
student s[20];
for(int i=0;i<20;i++)
{          …….. //accept student details          }
for(int i=0;i<20;i++)
```


11th_Comp Science_UM_Ch 12.indd 224 19-04-2023 09:45:26

```
{
…….. //check for name starts with letter “S”
……. // display the detail of the checked name
}
return 0;
}
```

4. اسٹرکچر کے ارکان تک کیسے رسائی حاصل کی جائے؟ مثال دیں۔
5. گمنام اسٹرکچر (anonymous structure) کسے کہتے ہیں ؟ ایک مثال دیں۔

سیکشن – D

تفصیل سے بتائیں

1. دو میٹرکس کے درمیان فرق معلوم کرنے کے لیے C++ پروگرام لکھیں۔
2. درج ذیل اسٹرکچر کی وضاحت کا استعمال کرتے ہوئے دو فاصلے شامل کرنے کے لیے C++ پروگرام لکھیں۔

```
struct Distance{
int feet;
float inch;
}d1 , d2, sum;
```

3. درج ذیل C++ پروگرام کا آوٹ پٹ لکھیں

```
#include<iostream>
#include<stdio.h>
#include <string.h>
#include<conio.h>
using namespace std;
struct books {
char name[20], author[20];
} a[2];
int main()
{  cout<< "Details of Book No " << 1 << "\n";
cout<< "-----------------------\n";
cout<< "Book Name :"<<strcpy(a[0].name,"Programming  ")<<endl;
cout<< "Book Author :"<<strcpy(a[0].author,"Dromy")<<endl;
cout<< "\nDetails of Book No " << 2 << "\n";
cout<< "-----------------------\n";
cout<< "Book Name :"<<strcpy(a[1].name,"C++programming" )<<endl;
cout<< "Book Author :"<<strcpy(a[1].author,"BjarneStroustrup ")<<endl;
cout<<"\n\n";
cout<< "================================================\n";
```


11th_Comp Science_UM_Ch 12.indd 225 19-04-2023 09:45:26

```
cout<< " S.No\t| Book Name\t|author\n";
cout<< "=================================================";
for (int i = 0; i < 2; i++) {
cout<< "\n  " << i + 1 << "\t|" << a[i].name << "\t| " << a[i].author;
}
cout<< "\n=================================================";
return 0;
}
```

4. درج ذیل C++ پروگرام کا آوٹ پٹ لکھیں

```
#include <iostream>
#include <string.h>
using namespace std;
struct student
{
        introll_no;
        char name[10];
        long phone_number;
};
int main(){
student p1 = {1,"Brown",123443},p2;
p2.roll_no = 2;
strcpy(p2.name ,"Sam");
p2.phone_number = 1234567822;
cout<< "First Student" <<endl;
cout<< "roll no : " << p1.roll_no <<endl<< "name : " << p1.name <<endl;
cout<< "phone no : " << p1.phone_number <<endl;
cout<< "Second Student" <<endl;
cout<< "roll no : " << p2.roll_no <<endl<< "name : " << p2.name <<endl;
cout<< "phone no : " << p2.phone_number <<endl;
return 0;
}
```


11th_Comp Science_UM_Ch 12.indd 226 19-04-2023 09:45:26

5. درج ذیل پروگرام میں غلطی کو ڈیبگ (Debug) کریں۔

```
#include <istream.h>
structPersonRec
{
        charlastName[10];
        chaefirstName[10];
int age;
}
PersonRecPeopleArrayType[10];
void main()
{
        PersonRecord people;
for (i = 0; i < 10; i++)
{
        cout<<people.firstName<< ‘ ‘ <<people.lastName  <<people.age;
}
for (int i = 0; i < 10; i++)
{
        cout<< "Enter first name: ";          cin<<peop[i].firstName;
        cout<< "Enter last name: ";           cin>>peop[i].lastName;
        cout<< "Enter age: ";          cin>> people[i].age;}
}
```

حوالہ جات :

1. Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.
2. The Complete Reference C++ (Forth Edition), Herbert Schildt.Mc.Graw Hills.
3. Computer Science with C++ (A text book of CBSE XI and XII), SumitaArora, DhanpatRai& Co.
4. The C++ Programming Language, Bjarne Stroustrup
5. https://www.tutorialspoint.com
6. http://www.cs.princeton.edu
7. https://www.programiz.com

11th_Comp Science_UM_Ch 12.indd 227 19-04-2023 09:45:26

باب 13

آبجیکٹ اورین ٹیڈ پروگرامنگ تکنیکس کا تعارف

سیکھنے کے مقاصد

اس باب کو سیکھنے کے بعد طلباء کے قابل ہوجائیں گے۔

- **OOPS** کے تصور کو سمجھنے
- پروسیجرل (procedural)، ماڈیولر (modular) اور آبجیکٹ اورین ٹیڈ پروگرامنگ کے درمیان فرق جاننے
- آبجیکٹ اورین ٹیڈ پروگرامنگ کے فوائد اور نقصانات کو سمجھنے

13.1 تعارف

آبجیکٹ- اورین ٹیڈ پروگرامنگ (OOP) وہ اصطلاح ہے جو کلاسوں اور آبجیکٹس پر مبنی پروگرامنگ کے نقطہ نظر کو بیان کرنے کے لیے استعمال ہوتا ہے۔ آبجیکٹ- اورین ٹیڈ پر مبنی نمونہ ہمیں سافٹ ویئر کو آبجیکٹس کے مجموعے کے طور پر ترتیب دینے کی اجازت دیتا ہے جو ڈیٹا اور رویے دونوں پر مشتمل ہو۔ یہ روایتی فنکشنل پروگرامنگ پریکٹس کے برعکس ہے، جو ڈیٹا اور رویے کو ڈھیلے طریقے سے جوڑتا ہے۔

1980 کے بعد سے لفظ 'آبجیکٹ (Object)' پروگرامنگ زبانوں کے سلسلے میں ظاہر ہوا ہے، 1990 کے بعد سے تقریباً تمام زبانیں آبجیکٹ- اورین ٹیڈ پر مبنی خصوصیات کی حامل ہیں۔ یہ باب عمومی OOP تصورات کو متعارف کراتا ہے۔

13.2 پروگرامنگ پیرا ڈائمز (Programming Paradigms)

پیراڈائم (Paradigm) کا مطلب پروگرام کے اصول کو منظم کرنا ہے۔ یہ پروگرامنگ کا ایک نقطہ نظر ہے۔ کمپیوٹر کا استعمال کرتے ہوئے مسئلہ حل کرنے کے لیے مختلف طریقے دستیاب ہیں۔وہ پروسیجرل پروگرامنگ، ماڈیولر پروگرامنگ اور آبجیکٹ اورین ٹیڈ پروگرامنگ ہیں۔

13.2.1 پروسیجرل پروگرامنگ (Procedural Programming)

پروسیجرل کا مطلب ہے کہ کمپیوٹر کو کچھ کرنے کے لیے ہدایات کی ایک فہرست دی گئی تھی۔ پروسیجرل پروگرامنگ کا مقصد پروسیجرس (Procedures) پر زیادہ ہے۔ یہ چیزیں کرنے پر زور دیتا ہے۔

پروسیجرل پروگرامنگ کی اہم خصوصیات

- پروگرام سب روٹینس (subroutines) یا سب پروگرامس (Sub programs) کی شکل میں ترتیب دیئے جاتے ہیں۔
- تمام ڈیٹا آئٹمز عالمی ہیں۔
- چھوٹے سائز کے سافٹ ویئر ایپلی کیشن کے لیے موزوں ہے۔
- پروگرام کے کوڈ کو برقرار رکھنا اور اس میں اضافہ کرنا مشکل ہے کیونکہ ڈیٹا کی قسم میں کسی بھی قسم کی تبدیلی کو ان تمام سب روٹینس پر پھیلانے کی ضرورت ہے، جو ایک ہی ڈیٹا کی قسم کا استعمال کرتے ہیں۔ یہ وقت طلب ہے۔
- مثال: **FORTRAN** اور **COBOL**

11th_Comp Science_UM_Ch 13.indd 228 19-04-2023 09:49:19

13.2.2 ماڈیولر پروگرامنگ:-
(Modular programming)

ماڈیولر پروگرامنگ ہدایات کی ایک فہرست پر مشتمل ہے جو کمپیوٹر کو کچھ کرنے کی ہدایت کرتی ہے۔ لیکن یہ پیراڈیم (Paradigm) متعدد ماڈیولز پر مشتمل ہے، ہر ماڈیول میں متعلقہ اقسام کے فنکشنس کا ایک سیٹ ہے۔ڈیٹا ماڈیولز کے نیچے چھپا ہوا ہے۔ ڈیٹا کی ترتیب کو صرف ماڈیول میں ترمیم کرکے تبدیل کیا جاسکتا ہے۔

ماڈیولر پروگرامنگ کی اہم خصوصیات

- ڈیٹا کے بجائے الگورتھم (algorithm) پر زور
- پروگراموں کو انفرادی ماڈیولز میں تقسیم کیا گیا ہے۔
- ہر ماڈیولس ایک دوسرے سے آزاد ہیں اور ان کا اپنا مقامی ڈیٹا ہے۔
- ماڈیولس اپنے ڈیٹا کے ساتھ ساتھ اس کو بھیجے گئے ڈیٹا کے ساتھ بھی کام کر سکتے ہیں۔
- مثال: پاسکل **(Pascal)** اور سی **(C)**

13.2.3 آبجیکٹ اورینٹیڈ پروگرامنگ
(Object Oriented Programming)

آبجیکٹ اورین ٹیڈ پروگرامنگ پیراڈیم الگورتھم کے بجائے ڈیٹا پر زور دیتا ہے۔ یہ کلاسز اور آبجیکٹس (classes and objects) کا استعمال کرتے ہوئے پروگراموں کو لاگو کرتا ہے۔

کلاس: کلاس C++ میں ایک تعمیر ہے جس کا استعمال ڈیٹا اور اس سے وابستہ فنکشن کو انکیپسولیشن(کیپسول کی طرح بند کرنا) (encapsulation) تصور کا استعمال کرتے ہوئے ایک اکائی میں ایک ساتھ باندھنے کے لیے کیا جاتا ہے۔ کلاس ایک صارف کی وضاحت شدہ ڈیٹا کی قسم ہے۔ کلاس اسی طرح کی آبجیکٹس کے ایک گروپ کی نمائندگی کرتا ہے۔

اس کی وضاحت ایک ٹیمپلیٹ (template) یا بلیو پرنٹ (blueprint) کے طور پر بھی کی جاسکتی ہے جو ایک گروپ آبجیکٹس کی نمائندگی کرتی ہے جو مشترکہ خصوصیات اور تعلقات کا اشتراک کرتے ہیں۔

آبجیکٹس: ڈیٹا اور اس سے وابستہ فنکشن کو ایک اکائی میں ایک ساتھ پیش کرتا ہے۔ آبجیکٹس OOP کی بنیادی اکائی ہیں۔ بنیادی طور پر ایک اعتراض کلاس سے بنایا گیا ہے۔ وہ کلاس کی مثالیں ہیں جنہیں کلاس متغیر بھی کہا جاتا ہے۔

کچھ خصوصیات اور طرز عمل کے ساتھ قابل شناخت ہستی کو آبجیکٹ کہا جاتا ہے۔

آبجیکٹ اورین ٹیڈ پروگرامنگ کی اہم خصوصیات

- الگورتھم کے بجائے ڈیٹا پر زور دیتا ہے۔
- ڈیٹا ابسٹریکشن (abstraction) کو پروسیجرل کے ابسٹریکشن کے علاوہ متعارف کرایا گیا ہے۔
- ڈیٹا اور اس سے منسلک آپریشنز کو ایک اکائی میں گروپ کیا گیا ہے۔
- پروگرامز آپریٹ کیے جانے والے ڈیٹا کے ارد گرد بنائے گئے ہیں۔
- اسی طرح کے، ابھی تک الگ الگ ڈیٹا کی قسم کے درمیان تعلقات بنائے جا سکتے ہیں۔
- مثال: C++, Java, VB.Net, Pyton وغیرہ۔

13.3 OOP کے بنیادی تصورات

آبجیکٹ اورین ٹیڈ پروگرامنگ کو پروسیجرل اور ماڈیولر پروگرامنگ کی خرابیوں کو دور کرنے کے لیے تیار کیا گیا ہے۔ یہ بڑے پیمانے پر قبول کیا جاتا ہے کہ آبجیکٹ – اورین ٹیڈ پرگرامنگ پر مبنی سافٹ ویئر بنانے کا سب سے اہم اور طاقتور طریقہ ہے۔

آبجیکٹ اورین ٹیڈ پروگرامنگ کی پہونچ بنیادی طور پر حوصلہ افزائی کرتا ہے:

- **ماڈیولرائزیشن (modularization):** جہاں پروگرام کو ماڈیولز میں تحلیل کیا جا سکتا ہے۔
- **سافٹ ویئر کا دوبارہ استعمال:** جہاں موجودہ اور نئے ماڈیولز سے ایک پروگرام بنایا جا سکتا ہے۔

آبجیکٹ اورین ٹیڈ پروگرامنگ کی اہم خصوصیات

- ڈیٹا ابسٹریکشن (Data Abstraction)
- انکیپسولیشن (Encapsulation)

11th_Comp Science_UM_Ch 13.indd 229 19-04-2023 09:49:19

- ماڈیولیریٹی (Modularity)
- انہیریٹنس (Inheritance)
- پولیمورفزم (Polymorphism)

13.3.1 انکیپسولیشن (Encapsulation)

وہ پروسیجرل جس کے ذریعے ڈیٹا اور فنکشنس کو ایک اکائی میں باندھا جاتا ہے اسے انکیپسولیشن کہا جاتا ہے۔ یہ تجرید کو لاگو کرتا ہے۔ انکیپسولیشن ڈیٹا متغیرات اور فنکشنس کو کلاس میں ایک ساتھ باندھنے کے بارے میں ہے۔ اسے ڈیٹا بائنڈنگ (data binding) بھی کہا جا سکتا ہے۔

انکیپسولیشن کلاس کی سب سے نمایاں خصوصیت ہے۔ بیرونی دنیا تک ڈیٹا کی رسائی نہیں ہے، اور صرف وہی فنکشنس جو کلاس میں لپیٹے ہوئے ہیں اس تک رسائی حاصل کر سکتے ہیں۔ یہ فنکشنس آبجیکٹ کے ڈیٹا اور پروگرام کے درمیان انٹرفیس (interface) فراہم کرتے ہیں۔ پروگرام کے ذریعے براہ راست رسائی سے ڈیٹا کی اس انکیپسولیشن کو ڈیٹا ہائڈنگ (data hiding) یا انفارمیشن ہائڈنگ (information hiding) کہا جاتا ہے۔

13.3.2 ڈیٹا ابسٹریکشن (Data Abstraction)

ابسٹریکشن سے مراد پس منظر کی تفصیلات کو ظاہر کیے بغیر صرف ضروری خصوصیات کو ظاہر کرنا ہے۔ کلاسیس تجریدی صفات (abstract attributes) اور فنکشن کی فہرست کی وضاحت کرنے کے لیے ابسٹریکشن کے تصور کا استعمال کرتے ہیں جو ان صفات پر کام کرتے ہیں۔ وہ اس چیز کی تمام ضروری خصوصیات کو سمیٹتے ہیں جو تخلیق کی جاتی ہیں۔ صفات کو ڈیٹا ممبر (data members) کہا جاتا ہے کیونکہ وہ معلومات رکھتے ہیں۔ ان اعداد و شمار پر کام کرنے والے فنکشنس کو میتھڈز یا ممبر فنکشن (method or member function) کہا جاتا ہے۔

13.3.3 ماڈیولائریٹی (Modularity)

ماڈیولائریٹی ایک ایسے سسٹم کو ڈیزائن کر رہی ہے جسے فنکشنل یونٹس (نام کے ماڈیولز) کے ایک سیٹ میں تقسیم کیا گیا ہے جسے ایک بڑی ایپلی کیشن میں بنایا جا سکتا ہے۔

13.3.4 انہیریٹنس (Inheritance)

انہیریٹنس ایک موجودہ کلاس (بیس کلاس – base class) سے نئی کلاسز (ڈرائیوڈ کلاس – derived class) بنانے کی تکنیک ہے۔ انہیریٹنس کا سب سے اہم فائدہ **کوڈ کو دوبارہ استعمال کرنا ہے۔**

13.3.5 پولیمورفزم (Polymorphism)

پولیمورفزم ایک پیغام یا فنکشن کی ایک سے زیادہ شکلوں میں ظاہر ہونے کی صلاحیت ہے۔

13.4 OOP کے فوائد:

دوبارہ استعمال کے قابل (Re-usability):

"ایک بار لکھیں اور اسے متعدد بار استعمال کریں" آپ کلاس کا استعمال کرکے یہ حاصل کرسکتے ہیں۔

دوہرانے کا عمل (Redundancy) :

انہیریٹنس ڈیٹا کو دوہرانے کے عمل کے لیے اچھی خصوصیت ہے۔ اگر آپ کو ایک سے زیادہ کلاس میں ایک ہی فعالیت کی ضرورت ہو تو آپ ایک ہی فعالیت کے لیے ایک عام کلاس لکھ سکتے ہیں اور اس کلاس کو سب کلاس (subclass) میں حاصل کر سکتے ہیں۔

آسان دیکھ بھال (Easy Maintenance):

موجودہ کوڈ کو برقرار رکھنا اور اس میں ترمیم کرنا آسان ہے کیونکہ موجودہ کوڈ سے چھوٹے فرق کے ساتھ نئی آبجیکٹس بنائی جا سکتی ہیں۔

سیکورٹی (Security):

ڈیٹا کو چھپانے اورابسٹریکشن کا استعمال کرتے ہوئے صرف ضروری ڈیٹا فراہم کیا جائے گا اس طرح ڈیٹا کی حفاظت برقرار رہتی ہے۔

13.5 OOP کے نقصانات (Disadvantages of OOP)

سائز (size):

آبجیکٹ اورین ٹیڈ پروگرامس دوسرے پروگراموں کے مقابلے میں بہت بڑے ہوتے ہیں۔

کوشش (Effort):

آبجیکٹ اورین ٹیڈ پروگرامس بنانے کے لیے بہت زیادہ محنت درکار ہوتی ہے۔

رفتار (Speed):

آبجیکٹ اورین ٹیڈ پروگرامس اپنے سائز کی وجہ سے دوسرے پروگراموں سے سست ہوتے ہیں۔

11th_Comp Science_UM_Ch 13.indd 230 19-04-2023 09:49:19

یاد رکھنے کے نکات

- پیراڈائم (Paradigm) کا مطلب ہے پروگرام کے اصول کو منظم کرنا۔ یہ پروگرامنگ کا ایک طریقہ ہے۔
- پروسیجرل یا ماڈیولر پروگرامنگ کا مطلب ہے کہ ہدایات کی ایک فہرست دی گئی تھی اور ہر ہدایات کمپیوٹر کو کچھ کرنے کو کہتا ہیں۔
- پروسیجرل پروگرامنگ کا مقصد زیادہ تر پروسیجرس (procedures) پر مبنی ہیں۔ اس میں پروگرام سب روٹینز (subroutines) یا سب پروگرامس (subprograms) کی شکل میں ترتیب دیئے جاتے ہیں۔
- ماڈیولر پروگرامنگ ایک ماڈیول میں متعلقہ پروسیجرس کو یکجا کرتا ہے اور ڈیٹا کو ماڈیول کے نیچے چھپاتا ہے۔
- آبجیکٹ اورین ٹیڈ پروگرامنگ پیراڈائم الگورتھم کے بجائے ڈیٹا پر زور دیتا ہے۔ یہ کلاسس اور آبجیکٹس کا استعمال کرتے ہوئے پروگراموں کو لاگو کرتا ہے۔
- کلاس ایک صارف کی وضاحت شدہ ڈیٹا کی قسم ہے۔ کلاس اسی طرح کی آبجیکٹس کے ایک گروپ کی نمائندگی کرتا ہے۔
- آبجیکٹس (objects) OPP کی بنیادی اکائی ہیں۔ یہ ڈیٹا اور متعلقہ فنکشن کو ایک اکائی میں ایک ساتھ پیش کرتا ہے۔
- وہ طریقہ جس کے ذریعے ڈیٹا اور فنکشنس کو ایک اکائی میں جوڑا جاتا ہے اسے انکیپسولیشن (ENCAPSULATION) کہا جاتا ہے۔ یہ ابسٹریکشن کو لاگو کرتا ہے۔
- ابسٹریکشن (abstraction) سے مراد پس منظر کی تفصیلات کو ظاہر کیے بغیر صرف ضروری خصوصیات دکھانا ہے۔
- ماڈیولائریٹی (modularity) ایک ایسے نظام کو ڈیزائن کر رہی ہے جسے فنکشنل یونٹس کے ایک سیٹ میں تقسیم کیا گیا ہے جسے ایک بڑی ایپلی کیشن میں بنایا جا سکتا ہے۔
- پولیمورفزم (polymorphism) ایک پیغام یا فنکشن کی ایک سے زیادہ شکلوں میں ظاہر ہونے کی صلاحیت ہے۔
- انہیریٹنس (Inheritance) ایک موجودہ کلاس (بیس کلاس - base class) سے نئی کلاسز (ڈرائیوڈ کلاس - derived class) بنانے کی تکنیک ہے۔ انہیریٹنس کا سب سے اہم فائدہ کوڈ کو دوبارہ استعمال کرنا ہے۔انہیریٹنس فطری طور پر منتقلی ہے۔

OOKGE

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. یہ اصطلاح کلاسز اور آبجیکٹس پر مبنی پروگرامنگ کی رسائی کو بیان کرنے کے لیے استعمال ہوتی ہے۔

OOP (A) POP (B) ADT (C) SOP (D)

2. وہ پیراڈائم جس کا مقصد پروسیجرس پر زیادہ ہے۔

(A) آبجیکٹ اورین ٹیڈ پروگرامنگ (B) پروسیجرل پروگرامنگ
(C) ماڈیولر پروگرامنگ (D) اسٹرکچرل پروگرامنگ

3. مندرجہ ذیل میں سے کون سا صارف کی کی وضاحت کردہ ڈیٹا کی قسم ہے؟

(A) کلاس (B) فلوٹ int (C) (D) آبجیکٹ

4. کچھ خصوصیات اور رویے کے ساتھ قابل شناخت شئ ہے۔

(A) کلاس (B) آبجیکٹ (C) اسٹرکچر (D) ممبر

5. وہ پروطریقۂ کار جس کے ذریعے ڈیٹا اور فنکشنس کو ایک اکائی میں باندھا جاتا ہے اسے کہا جاتا ہے۔

(A) انہیریٹنس (B) انکیپسولیشن
(C) پولیمورفیزم (D) ابسٹریکشن

6. پروگرام کے ذریعہ براہ راست رسائی سے ڈیٹا کی موصلیت کو یہ کہا جاتا ہے۔

(A) ڈیٹا ہائڈنگ (B) انکیپسولیشن
(C) پولیمورفیزم (D) ابسٹریکشن

11th_Comp Science_UM_Ch 13.indd 231 19-04-2023 09:49:19

7. مندرجہ ذیل میں سے کون سا تصور اس چیز کی تمام ضروری خصوصیات کو سمیٹتا ہے جو تخلیق کی جاتی ہیں؟
(A) کلاس (B) انکیپسولیشن
(C) پولیمورفیزم (D) ابسٹریکشن

8. مندرجہ ذیل میں سے کس انہیریٹنس کا سب سے اہم فائدہ ہے؟
(A) ڈیٹا ہائڈنگ (B) کوڈ کو دوبارہ استعمال کرنا
(C) کوڈ میں ترمیم (D) رسائی

9. "ایک بار لکھیں اور اسے متعدد بار استعمال کریں" کے ذریعہ یہ حاصل کیا جاسکتا ہے۔
(A) ریڈنڈینسی (B) ری یونیبلٹی
(C) موڈی فیکشن (D) کمپوزیشن

10. درج ذیل میں سے کون سا ڈیٹا فطری طور پر منتقلی کی حمایت کرتا ہے؟
(A) انہیریٹنس (B) انکیپسولیشن
(C) پولیمورفیزم (D) ابسٹریکشن

سیکشن – B

بہت ہی مختصر جوابات دیں

1. ماڈیولر پروگرامنگ، پروسیجرل پروگرامنگ پیراڈائم سے کیسے مختلف ہے؟
2. کلاسز اور آبجیکٹس میں فرق کریں۔
3. پولیمورفزم کیا ہے؟
4. انکیپسولیشن اور ابسٹریکشن کا آپس میں کیسا تعلق ہے؟
5. OOP کے نقصانات لکھیں۔

سیکشن – C

مختصر جوابات دیں

1. پیراڈائم کیا ہے؟پیراڈائم کی مختلف اقسام کا ذکر کریں۔
2. پروسیجرل پروگرامنگ کی خصوصیات پر ایک نوٹ لکھیں۔
3. ماڈیولر پروگرامنگ کی کچھ خصوصیات کی فہرست بنائیں۔
4. ماڈیولرائزیشن (modularization) اور سافٹ ویئر (software) کے دوبارہ استعمال سے آپ کا کیا مطلب ہے؟
5. انفارمیشن ہائڈنگ (information hiding) کی وضاحت کریں۔

سیکشن – D

تفصیل سے بتائیں

1. آبجیکٹ اورین ٹیڈ پروگرامنگ اور پروسیجرل پروگرامنگ کے درمیان فرق لکھیں۔
2. OOP کے کیا فائدے ہیں؟
3 بنیادی تصورات پر ایک نوٹ لکھیں جو OOP کی حمایت کرتے ہیں؟

حوالہ جات:

(1) Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.
(2) The Complete Reference C++ (Forth Edition), Herbert Schildt.Mc.Graw Hills.
(3) Computer Science with C++ (A text book of CBSE XI and XII), SumitaArora, DhanpatRai& Co.
(4) A text book of CBSE XI and XII computer science by PreetiArora and Pinky Gupta.
(5) Computer Science with C++ Reeta shoo and Gagansahoo
(6) The C++ Programming Language,BjarneStroustrup
(7) C++ Primer (5th Edition) by S. B. Lippman, J. Lajoie

11th_Comp Science_UM_Ch 13.indd 232 19-04-2023 09:49:19

کلاسز اور آبجیکٹس
(Classes and objects)



سیکھنے کے مقاصد

اس باب کو سیکھنے کے بعد طلباء کے قابل ہو جائیں گے۔

- کلاسز، آبجیکٹ کنسٹرکٹرس (constructors) اور ڈسٹرکٹرس (destructors) کے مقصد کو سمجھنے
- کنسٹرکٹرس اور ڈسٹرکٹرس کے ساتھ کلاسز کا استعمال کرتے ہوئے C++ پروگرام بنانے
- کنسٹرکٹرس اور ڈسٹرکٹرس کے ساتھ کلاس پروگرامس پر عمل درآمد اور ڈیبگ (debug) کرنے

14.1 کلاس کا تعارف

C++ کی سب سے اہم خصوصیت "کلاس" ہے۔ اس کی اہمیت اس حقیقت سے اجاگر ہوتی ہے کہ بی جارنے اسٹراسٹرپ (Bjarne Stroustrup) نے ابتدا میں 'C کے ساتھ کلاسس' کا نام دیا تھا۔ C++ کلاسس پیش کرتا ہے، جو OOP زبانوں میں عام طور پر موجود چار خصوصیات فراہم کرتا ہے جو درج ذیل ہیں:
ابسٹریکشن (Abstraction)، انکیپسولیشن (Encapsulation)، انہیریٹنس (Inheritance) اور پولیمورفیزم (Polymorphism)۔

14.1.1 کلاس کی ضرورت

کلاس ڈیٹا اور اس سے وابستہ افعال کو ایک ساتھ باندھنے کا ایک طریقہ ہے۔ حقیقی دنیا کے اداروں کی نمائندگی کرنے کے لیے کلاسس کی ضرورت ہوتی ہے جن میں نہ صرف ڈیٹا ٹائپ کی خصوصیات ہوتی ہیں بلکہ اس سے وابستہ آپریشنس بھی ہوتے ہیں۔ یہ صارف کی وضاحت کردہ ڈیٹا کی قسم بنانے کے لیے استعمال ہوتا ہے۔

14.1.2 کلاس کا اعلان (Declaration of a class)

کلاس کی وضاحت C++ میں کلیدی لفظ کلاس کا استعمال کرتے ہوئے کی جاتی ہے جس کے بعد کلاس کا نام آتا ہے۔ کلاس کی باڈی کو منحنی قوسین کے اندر بیان کیا جاتا ہے اور اسے یا تو سیمی کولن یا آخر میں اعلانات کی فہرست کے ذریعے ختم کیا جاتا ہے۔

نوٹ

اسٹرکچر اور کلاس کے درمیان فرق صرف یہ ہے کہ اسٹرکچر کے ارکان پہلے سے طے شدہ طور پر عوامی ہوتے ہیں جب یہ کلاس میں نجی ہوتا ہے۔

کلاس کی وضاحت کی عمومی شکل

```
class class_name
{
private:
        variable declaration;
        function declaration;
protected:
        variable declaration;
        function declaration;
public:
        variable declaration;
        function declaration;
};
```

- کلاس باڈی میں اپنے ممبروں کا اعلان (ڈیٹا ممبر اور ممبر کے افعال) ہوتا ہے۔

11th_Comp Science_UM_Ch 14.indd 233 19-04-2023 10:06:38

- کلاس باڈی کے پاس رسائی کے تین تصریحات ہیں (مرئیت کے لیبل) یعنی نجی (private)، عوامی (public) اور محفوظ (Protected)۔

14.1.3 کلاس تک رسائی کی وضاحت کرنے والے

ڈیٹا ہائڈنگ آبجیکٹ اورین ٹیڈ پروگرامنگ کی ایک اہم خصوصیت ہے جو کسی پروگرام کے افعال کو براہ راست کلاس کی قسم کی اندرونی نمائندگی تک رسائی سے روکنے کی اجازت دیتا ہے۔ کلاس ممبران تک رسائی کی پابندی کلاس باڈی کے اندر پبلک، پرائیویٹ، اور محفوظ سیکشنس (sections) کے ذریعے بیان کی جاتی ہے۔ کلیدی الفاظ عوامی، نجی، اور محفوظ کو رسائی کی وضاحت کرنے والے (Access Specifiers) کہتے ہیں۔ اراکین کے لیے پہلے سے طے شدہ رسائی کی وضاحت کردہ نجی ہے۔

عوامی اراکین (The Public Members)

ایک عوامی اراکین کلاس کے باہر کہیں سے بھی قابل رسائی ہے لیکن پروگرام کے اندر۔ آپ کسی بھی ممبر فنکشن کو استعمال کیے بغیر بھی پبلک ڈیٹا ممبرز کی ویلیو سیٹ اور حاصل کر سکتے ہیں۔

نجی اراکین (The Private Members)

کلاس کے باہر سے نجی اراکین تک رسائی حاصل نہیں کی جا سکتی۔ صرف کلاس ممبر فنکشنس ہی پرائیویٹ ممبرس تک رسائی حاصل کر سکتے ہیں۔ بطور ڈیفالٹ کلاس کے تمام ممبرس پرائیویٹ ہوں گے۔

محفوظ اراکین (The Protected Members)

ایک محفوظ اراکین اور نجی اراکین سے بہت ملتا جلتا ہے، لیکن یہ ایک اضافی فائدہ فراہم کرتا ہے کہ ان تک بچوں کی کلاسوں میں رسائی حاصل کی جاسکتی ہے جنہیں ڈرائیوڈ کلاسس (derived classes) (انہیریڈیٹ کلاسس – inherited classes) کہا جاتا ہے۔

مثال

Keyword class intimates the compiler that it is a class definition

Class name or tag name acts as a user defined data type. Using this, object of the same class type will be created.

```
class student
{
private:
        char name [10];
        int rollno, mark1, mark2, total;
protected:
        void accept();
        void compute();
public:
        void display();
};
```

These are private access specifier members

That means these members cannot be accessed from outside

These are protected access specifier members

These members also cannot be accessed from outside

Members under this specifier can be accessed from outside

11th_Comp Science_UM_Ch 14.indd 234 19-04-2023 10:06:38

نوٹ

اگر کلاس کے تمام ممبران کو نجی طور پر وضاحت کیا جاتا ہے، تو کلاس کا آبجیکٹ (object) کلاس سے کسی بھی چیز تک رسائی حاصل نہیں کر سکتا۔

سرگرمی 1

درج ذیل کوڈ کے حصہ کے غلط ہونے کی وجہ بتائیں

(i)	(ii)

class count
{
int first;
int second;
public:
int first;
};
	class item
	{
	int prd;
	};
	item int prdno;

14.1.4 کلاس ممبران کی وضاحت

کلاس ممبران پر مشتمل ہے۔ ممبران کو ڈیٹا ممبرس اور ممبر فنکشنس کے طور پر درجہ بندی کیا جاتا ہے۔ ڈیٹا ممبران ڈیٹا متغیرات ہیں جو کلاس کی خصوصیات یا خصوصیات کی نمائندگی کرتے ہیں۔ ممبر فنکشنس وہ فنکشن ہیں جو کلاس میں مخصوص کام انجام دیتے ہیں۔ ممبران کے افعال کو میتھڈس (methods) کہا جاتا ہے، اور ڈیٹا کے ممبران کو ایٹری بیوٹس (attributes) بھی کہا جاتا ہے۔

مثال

```
Class result
{
Private:
        char name [10];
        int rollno,mark1, mark2, total;

Public:
        void accept();
        void display();
};
```

ڈیٹا ممبران (Data members)

ممبر فنکشنس (Member function)

نوٹ

کلاسز میں کچھ خاص ممبر فنکشن بھی ہوتے ہیں جنہیں کنسٹرکٹر (constructors) اور ڈسٹرکٹر (destructors) کہا جاتا ہے

14.1.5 کلاس کے طریقوں کی وضاحت

طریقوں (فنکشنس) کی وضاحت کیے بغیر، کلاس کی وضاحت نامکمل ہو جائے گی۔ کلاس کے ممبر کے افعال کو دو طریقوں سے بیان کیا جا سکتا ہے۔

(1) کلاس کی وضاحت کے اندر

(2) کلاس کی وضاحت سے باہر

(1) کلاس کی وضاحت کے اندر

جب کسی ممبر فنکشن کو کلاس کے اندر بیان کیا جاتا ہے، تو یہ اِن لائن فنکشنس (inline functions) کی طرح برتاؤ کرتا ہے۔ یہ اِن لائن ممبر فنکشنس کہلاتے ہیں۔

نوٹ

اگر کوئی فنکشن ان لائن ہے تو، کمپائلر کوڈ کی ایک کاپی ہر اس مقام پر رکھتا ہے جہاں کمپائل کے وقت فنکشن کو بلایا جاتا ہے۔

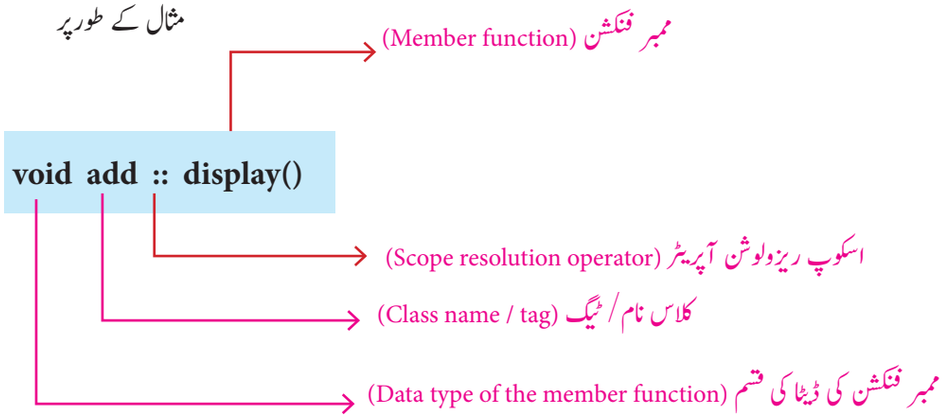
(2) کلاس کی وضاحت سے باہر

جب ممبر فنکشن کی وضاحت کلاس کے باہر بالکل عام فنکشن کی وضاحت کی طرح کی جاتی ہے (فنکشن کی وضاحت جس سے آپ واقف ہیں) تو اسے آؤٹ لائن ممبر فنکشن (outline member function) یا نان اِن لائن ممبر فنکشن (non-inline member function) کہا جاتا ہے۔ اس مقصد کے لیے اسکوپ ریزولوشن آپریٹر (::) (scope resolution operator) استعمال کیا جاتا ہے۔ آؤٹ لائن ممبر فنکشن کی وضاحت کے لیے سنٹیکس یہ ہے۔

11th_Comp Science_UM_Ch 14.indd 235 19-04-2023 10:06:38

```
return_type class_name :: function_name (parameter list)
{
        function definition
}
```

مثال کے طور پر



مثال 14.1 ان لائن اور آؤٹ لائن ممبر فنکشن

```
# include <iostream>
using namespace std;
class Box
{
        double width;          // no access specifier mentioned
public:
        double length;
        void printWidth( )              //inline member function definition
        {
                cout<<"\n The width of the box is..."<<width;
        }
        void  setWidth( double w );          //prototype of the function
};
void Box :: setWidth(double w)      // outline member function definition
{
        width=w;
}
int main( )
{
Box b;          // object for class Box
b.setWidth(10.0);      // Use member function to set the width.
b.printWidth( );               //Use member function to print the width.
return 0;
}
```

Absence of access specifier means that members are private by default..

Output:

The width of the box is... 10

11th_Comp Science_UM_Ch 14.indd 236 19-04-2023 10:06:38

نوٹ

لوپنگ کنسٹرکٹ (looping construct)، سوئچ (switch) یا گوٹو اسٹیٹمنٹ (goto statement) والے ممبر فنکشن کو ان لائن قرار دینا مناسب نہیں ہے۔

14.2 آبجیکٹس بنانا

کلاس کی تفصیلات صرف کلاس کی خصوصیات کی وضاحت کرتی ہے۔ کسی کلاس کو استعمال کرنے کے لیے، اس کلاس کی قسم کے متغیرات کا اعلان کرنا ہوگا۔ کلاس متغیرات کو آبجیکٹ کہا جاتا ہے۔ آبجیکٹس کو کلاس کی مثال (instance of class) بھی کہا جاتا ہے۔

مثال کے طور پر

```
student s;
```

مندرجہ بالا بیان میں s کلاس کے طالب علم کی ایک مثال ہے۔

آبجیکٹس کو دو طریقوں سے بنایا جا سکتا ہے،

(1) عالمی آبجیکٹ (Global object)

(2) مقامی آبجیکٹ (Local object)

(1) عالمی آبجیکٹ (Global object)

اگر کسی آبجیکٹ کو تمام فنکشن باڈیز سے باہر قرار دیا جاتا ہے یا کلاس ڈیکلریشن کے اختتامی قوسین کے فوراً بعد ان کے نام رکھ کر اسے گلوبل آبجیکٹ کہا جاتا ہے۔ ان آبجیکٹس کو پروگرام میں کسی بھی فنکشن کے ذریعے استعمال کیا جا سکتا ہے۔

(2) مقامی آبجیکٹ (Local object)

اگر کسی آبجیکٹ کو کسی فنکشن کے ساتھ اعلان کیا جائے تو اسے لوکل آبجیکٹ کہا جاتا ہے۔ فنکشن کے باہر سے اس کی رسائی حاصل نہیں کی جا سکتی۔

مثال 14.2 مقامی اور عالمی آبجیکٹ کا استعمال

```
# include <iostream>
# include <conio.h>
using namespace std;
class add
{
  int a,b;
  public:
    int sum;
  void getdata()
  {
    a=5;
    b=10;
    sum = a+b;
  }
} a1;                    //global object
add a2;                  //global object
int main()
{
add a3;          // Local object
a1.getdata();
a2.getdata();
a3.getdata();
cout<<a1.sum;
cout<<a2.sum;
cout<<a3.sum;
return 0;
}
```

Output:

151515

سرگرمی 2

درج ذیل کوڈ کے حصہ میں غلطی کی نشاندہی کریں

```
class A
{
        float x;
        void init()
        {
        A a1;
        X1.5=1;
        }};
void main()
{    A1.init(); }
```


11th_Comp Science_UM_Ch 14.indd 237 19-04-2023 10:06:38

14.3 آبجیکٹس کی میموری مختص کرنا
(Memory allocation of objects)

ممبر فنکشنس بنائے جاتے ہیں اور میموری اسپیس میں صرف اسی وقت رکھے جاتے ہیں جب انہیں کلاس کی تفصیلات کے ایک حصے کے طور پر بیان کیا جاتا ہے۔ چونکہ اس کلاس سے تعلق رکھنے والی تمام آبجیکٹس ایک ہی ممبر فنکشن کا استعمال کرتی ہیں، اس لیے جب آبجیکٹ بنائے جاتے ہیں تو ممبر فنکشنس کے لیے کوئی الگ جگہ مختص نہیں کی جاتی ہے۔ ممبر متغیرات کے لیے درکار میموری کی جگہ صرف ہر آبجیکٹ کے لیے الگ الگ مختص کی جاتی ہے کیونکہ ممبر متغیرات مختلف آبجیکٹ کے لیے مختلف ڈیٹا ویلیوز رکھتے ہیں۔

مثال 14.3 آبجیکٹس کی میموری مختص کرنا

```
# include <iostream>
using namespace std;

class product
{
        int code, quantity;
        float price;
        public:
        void assignData();
        void Print();
};
int main()
{
        product p1, p2;
        cout<<"\n Memory allocation for object p1 " <<sizeof(p1);
        cout<<"\n Memory allocation for object p2 " <<sizeof(p2);
        return 0;
}
```

Output:

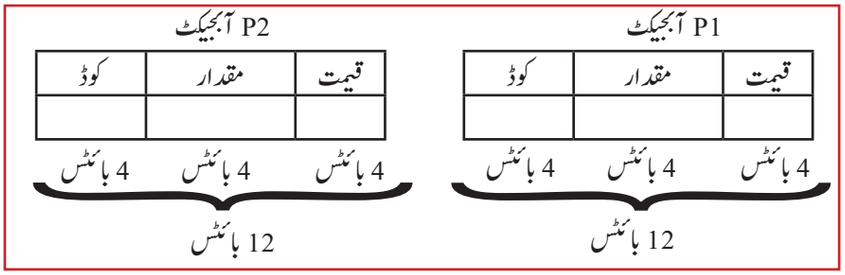
```
Memory allocation for object p1  12
Memory allocation for object p2  12
```

ممبر فنکشن ڈیٹا () اور پرنٹ () کو تفویض کرتا ہے جو کامن پول (common pool) سے تعلق رکھتے ہیں اس لحاظ سے دونوں آبجیکٹس p1 اور p2 کو مشترکہ پول کے کوڈ ایریا (code area) تک رسائی حاصل ہوگی۔

نوٹ

ممبران کو کلاس ٹائپ آبجیکٹ بنانے کے بعد ہی میموری کی جگہ مختص کی جائے گی۔

P1 اور P2 کے آبجیکٹ کے لیے میموری کی مثال دی گئی ہے:



11th_Comp Science_UM_Ch 14.indd 238 19-04-2023 10:06:39

سرگرمی 3

s1 اور s2 آبجیکٹ کا سائز کیا ہے؟

```
class sum
{
        int n1,n2;
        public:
        void add(){int n3=10;n1=n2=10;}
} s1,s2;
```

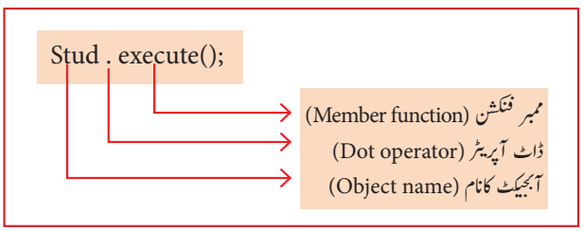
14.4 کلاس ممبران کا حوالہ

کلاس کے ممبران کو کلاس کے آبجیکٹ کے بعد ڈاٹ (ممبر شپ) آپریٹر اور ممبر کا نام استعمال کرکے حوالہ دیا جاتا ہے۔

ممبر فنکشن کو کال کرنے کا عمومی سنٹیکس یہ ہے:

Object_name . function_name(actual parameter);

مثال کے طور پر درج ذیل مثال پر غور کریں۔



مثال 14.4 آبجیکٹ کے مواصلات کی وضاحت کرنے کے لیے C++ پروگرام

```
#include<iostream>
using namespace std;
class compute
{
 int n1,n2;       //private by default
 public :
 int n;
 int add (int a, int b)   //inline member  function
 {
 int c=a+b;       //int c ; local variable for this function
 return c;
 }

}c1,c2;
int main()
{
 c1.n =c1.add(12,15); //member function is called
 c2.n =c2.add(8,4);
 cout<<"\n Sum of object-1  "<<c1.n;
 cout<<"\n Sum of object-2  "<<c2.n;
 cout<<"\n Sum of the two objects are  "<<c1.n+c2.n;
          return 0;
}
```

Output:

```
Sum of object-1  27
Sum of object-2  12
Sum of the two objects are  39
```

نوٹ

یہاں تک کہ کلاس کے لیے آبجیکٹ کی ایک ارے بھی بنائی جا سکتی ہے۔ اس کا اعلان اوروضاحت اسی طرح کی گئی ہے جس طرح کسی بھی دوسری قسم کی ارے کی وضاحت کی گئی ہے۔

مثال :

student s[10];

جہاں طالب علم کلاس کا نام ہے اور s[10] طالب علم کی کلاس کے لیے بنائے گئے 10 آبجیکٹ ہیں۔

14.5 کنسٹرکٹرس (constructors) کا تعارف

کلاس کی وضاحت صرف ایک نئی صارف کی وضاحت کردہ ڈیٹا کی قسم بناتی ہے۔ کلاس کی قسم کی مثالیں فوری کی جانی چاہئے (تخلیق اور ابتدا)۔ کنسٹرکٹر کا استعمال کرتے ہوئے آبجیکٹ کوفوری طور پر کیا جاتا ہے۔

11th_Comp Science_UM_Ch 14.indd 239 19-04-2023 10:06:39

14.5.1 کنسٹرکٹرس کی ضرورت

C++ میں ایک ارے (array) یا اسٹرکچر (structure) ان کے اعلان کے وقت شروع کیا جا سکتا ہے۔
مثال کے طور پر

```
struct sum
  {
        int n1,n2;
  };
class add
  {
        int num1,num2;
  };
int  main()
  {
        int arr[]={1,2,3};        //declaration and initialization of array
        sum s1={1,1};            //declaration and initialization of structure object
        add  a1={0,0};          // class object declaration and initialization throws
                                                          compilation error
   }
```

Member function of a class can access all the members irrespective of their associated access specifier.

کسی اسٹرکچر یا ارے سے ملتے جلتے اعلان کے وقت کلاس قسم کے آبجیکٹ کا آغاز ممکن نہیں ہے کیونکہ کلاس کے ممبران کے پاس ان کے متعلقہ رسائی کی وضاحت کرنے والے (نجی یا محفوظ یا عوامی) ہوتے ہیں۔ لہذا کلاسس میں خصوصی ممبر فنکشن شامل ہوتے ہیں جنہیں کنسٹرکٹر (constructor) کہا جاتا ہے۔کنسٹرکٹر فنکشن کلاس آبجیکٹ کو شروع کرتا ہے۔

14.6 اعلان اور وضاحت

جب کسی کلاس کی مثال دائرہ کار میں آتی ہے تو، کنسٹرکٹر نامی ایک خاص فنکشن عمل میں آتا ہے۔ کنسٹرکٹر فنکشن کا نام کلاس کے نام جیسا ہی ہے کنسٹرکٹرس کچھ بھی واپس نہیں کرتے ہیں۔ وہ کسی بھی قسم کے ڈیٹا سے وابستہ نہیں ہیں۔ اس کی وضاحت یا تو کلاس کی وضاحت کے اندر یا کلاس کی وضاحت سے باہر کی جا سکتی ہے۔

مثال 1:

مثال 14.5 ایک کنسٹرکٹر کی کلاس کی تفصیلات کے اندر وضاحت

```
#include<iostream>
using namespace std;
class Sample
{
        int i,j;
        public :
          int k;
          Sample()
          {
             i=j=k=0;   //constructor defined inside the class
          }
};
```


11th_Comp Science_UM_Ch 14.indd 240 19-04-2023 10:06:39

14.6.1 کنسٹرکٹر کے افعال

جیسا کہ اب ہم جانتے ہیں کہ کنسٹرکٹر کلاس کا ایک خاص ابتدائی ممبر فنکشن ہے جو کہ جب بھی کسی کلاس کی مثال کا اعلان یا تخلیق کیا جاتا ہے تو خود بخود بلایا جاتا ہے۔ کنسٹرکٹر کا بنیادی کام ہے۔

1) آبجیکٹ کے لیے میموری کی جگہ مختص کرنا اور

2) کلاس آبجیکٹ کے ڈیٹا ممبر کو شروع کرنا

کلاس آبجیکٹ کو شروع کرنے کا ایک متبادل طریقہ ہے، لیکن اس صورت میں ہمیں ممبر فنکشن کو واضح طور پر کال کرنا ہو گا۔

14.7 کنسٹرکٹرس کی اقسام

کنسٹرکٹرس کی مختلف اقسام ہیں۔

- **ڈیفالٹ کنسٹرکٹرس (Default constructor)**

ایک کنسٹرکٹر جو کوئی پیرامیٹر قبول نہیں کرتا اسے ڈیفالٹ کنسٹرکٹر کہا جاتا ہے۔ مثال کے طور پر کلاس ڈیٹا، ڈیٹا ::، ڈیٹا () ڈیفالٹ کنسٹرکٹر ہے۔ اس کنسٹرکٹر کا استعمال کرتے ہوئے آبجیکٹ اسی طرح بنائے جاتے ہیں جس طرح دوسرے ڈیٹا کی اقسام کے متغیرات بنائے جاتے ہیں۔ اگر کسی کلاس میں واضح کنسٹرکٹر (یوزر ڈیفائنڈ کنسٹرکٹر) نہیں ہوتا ہے تو کمپائلر خود بخود ڈیفالٹ کنسٹرکٹر تیار کرتا ہے۔

- **پیرامیٹرائزڈ کنسٹرکٹرس (Parameterized constructors)**

ایک کنسٹرکٹر جو دلائل لے سکتا ہے اسے پیرامیٹرائزڈ کنسٹرکٹر کہا جاتا ہے ۔اس قسم کا کنسٹرکٹر مختلف ابتدائی اقدار کے ساتھ اشیاء بنانے میں مدد کرتا ہے۔ یہ فنکشن میں پیرامیٹرس کو پہونچاتے ہوئے حاصل کیا جاتا ہے۔

مثال :

```
Data :: Data(int,int);
```

- **کاپی کنسٹرکٹرس (Copy constructors)**

ایک کنسٹرکٹر جس کی اپنی کلاس کے پہلے سے موجود آبجیکٹ کا حوالہ ہو اسے کاپی کنسٹرکٹر کہا جاتا ہے۔ یہ عام طور پر ڈیٹا (ڈیٹا&) کی شکل میں ہوتا ہے، جہاں ڈیٹا کلاس کا نام ہوتا ہے۔

ایک کاپی کنسٹرکٹر کو کئی طریقوں سے بلایا جا سکتا ہے:

1) جب کسی چیز کو رکن کے کسی بھی فنکشن میں پیرامیٹر کے طور پر پہونچادیاجاتاہے۔

مثال void Data::putdata(Data x);

2) جب ممبر فنکشن کسی چیز کو لوٹاتا ہے

مثال Data getdata() { }

3) جب کسی چیز کو اس کی اپنی کلاس کی مثال کے حوالے سے پہونچایا جاتاہے۔

مثال کے طور پر، Data d1, d2 (d1); // d2(d1) کاپی کنسٹرکٹر کو بلاتا ہے۔

11th_Comp Science_UM_Ch 14.indd 241 19-04-2023 10:06:39

مثال 14.6 کنسٹرکٹر کی اقسام

```
#include<iostream>
using namespace std;
class Data
 {
       int i, j;
     public:
       int k;
        Data()
        {
             cout<<"Non Parametrerized constructor";
              i=0;
               j=0;
         }
         Data(int a,int b)
         {
             cout<<"Parametrerized constructor";
              i=a;
               j=b;
         }
        Data(Data &a)
         {
             cout<<"Copy constructor";
              i=a.i;
          }

          void display()            //member function
           {
              cout<< i<<j;
           }
       };
int main()
{
          Data  d1,d2(10,20),d3(d2);
          d1.display();
          d2.display();
          d3.display();
           return 0;

}
```

14.8 کنسٹرکٹرس کی درخواست
(Invocation of constructors)

پیرامیٹرائزڈ کنسٹرکٹر (parameterized constructor) کا استعمال کرتے ہوئے آبجیکٹ بنانے کے دو طریقے ہیں۔

- مضمر کال (Implicit call)
- واضح کال (Explicit call)

11th_Comp Science_UM_Ch 14.indd 242 19-04-2023 10:06:39

14.8.1 مضمر کال (Implicit call)

اس طریقہ میں، جب بھی کوئی چیز بنائی جاتی ہے تو پیرامیٹرائزڈ کنسٹرکٹر کو خود بخود طلب کیا جاتا ہے۔ مثال کے طور پر سادہ s1(10,20);، اس میں آبجیکٹ بنانے کے لیے s1 پیرامیٹرائزڈ کنسٹرکٹر کو خود بخود طلب کیا جاتا ہے۔

14.8.2 واضح کال (Explicit call)

اس طریقہ میں، کنسٹرکٹر کا نام واضح طور پر پیرامیٹرائزڈ کنسٹرکٹر کو پکارنے کے لیے دیا جاتا ہے تاکہ آبجیکٹ کو بنایا اور شروع کیا جا سکے۔ مثال کے طور پر

```
simple s1=simple(10,20);        //explicit call
```

واضح کال کا طریقہ سب سے موزوں طریقہ ہے کیونکہ یہ ایک عارضی آبجیکٹ بناتا ہے، ڈیٹا کے ضائع ہونے کا امکان پیدا نہیں ہوتا ہے۔ ایک عارضی آبجیکٹ میموری میں اس وقت تک زندہ رہتا ہے جب تک اسے اظہار میں استعمال کیا جاتا ہے۔ اس کے بعد یہ ختم ہو جاتا ہے۔

14.9 آبجیکٹس کی متحرک ابتدا
(Dynamic initialization of objects)

جب رن ٹائم کے دوران ابتدائی قدریں فراہم کی جاتی ہیں تو اسے ڈائنامک انیشی لائزیشن (Dynamic initialization) کہا جاتا ہے۔

مثال 14.7 متحرک ابتداء کو واضح کرنے کے لیے

```
#include<iostream>
using namespace std;
class X
{   int n;
    float avg;
    public:
    X(int p,float q)
    {  n=p;
       avg=q;  }
    void disp()
    {    cout<<"\n Roll numbe:- " <<n;
         cout<<"\nAverage :- "<<avg;          } };
int main()
{
int a ; float b;
         cout<<"\nEnter the Roll Number";
         cin>>a;
         cout<<"\nEnter the Average";
         cin>>b;
         X x(a,b);        // dynamic initialization
         x.disp();
         return 0;
}
```

Output:

```
Enter the Roll Number 1201
Enter the Average 98.6
 Roll numbe:- 1201
Average :- 98.6
```


11th_Comp Science_UM_Ch 14.indd 243 19-04-2023 10:06:39

14.10 کنسٹرکٹرس کی خصوصیات
(Characteristics of constructors)

- کنسٹرکٹر کا نام کلاس کے نام جیسا ہی ہونا چاہیے
- کنسٹرکٹر کے لیے واپسی کی کوئی قسم نہیں بتائی جا سکتی ہے
- ایک کنسٹرکٹر پیرامیٹر کی فہرست رکھ سکتا ہے
- کنسٹرکٹر فنکشن کو اورلوڈیڈ (overloaded) کیا جا سکتا ہے
- انہیں حاصل نہیں کیا جاسکتا لیکن اخذ شدہ کلاس بیس کلاس کنسٹرکٹر کو کال کر سکتا ہے۔
- صارف کی وضاحت کردہ کنسٹرکٹر کی غیر موجودگی میں ،کمپائلر کنسٹرکٹر بناتا ہے۔
- کمپائلر کنسٹرکٹر پبلک ممبر فنکشن ہے۔
- جب آبجیکٹ بنتا ہے تو کنسٹرکٹر کو خود بخود عمل میں لایا جاتا ہے۔
- ایک کنسٹرکٹر کو واضح طور پر اس کی کلاس قسم کی نئی آبجیکٹ بنانے کے لیے استعمال کیا جا سکتا ہے۔

14.11 ڈسٹرکٹرس
(Destructors)

جب کوئی کلاس آبجیکٹ دائرہ کار سے باہر ہو جاتا ہے تو، ایک خاص فنکشن جسے ڈسٹرکٹر (destructor) کہتے ہیں، عمل میں لایا جاتا ہے۔ ڈسٹرکٹر کا کلاس ٹیگ نام جیسا ہی ہے ، لیکن اس کا سابقہ ~(tilde) کے ساتھ ہے۔ڈسٹرکٹر فنکشن بھی کچھ نہیں لوٹاتا ہے اور یہ کسی بھی قسم کے ڈیٹا سے وابستہ نہیں ہے۔

14.11.1 ڈسٹرکٹرس کی ضرورت

ڈسٹرکٹرس کا مقصد ان وسائل کو آزاد کرنا ہے جو آبجیکٹ نے اپنی زندگی کے دوران حاصل کیے ہوں گے۔ ڈسٹرکٹر فنکشن کسی آبجیکٹ کی میموری کو ہٹاتا ہے جو کسی چیز کو بنانے کے وقت کنسٹرکٹر نے مختص کیا تھا۔

14.11.2 اعلان اور وضاحت

ڈسٹرکٹر ایک خاص ممبر فنکشن ہے جسے اس وقت بلایا جاتا ہے جب کسی آبجیکٹ کی زندگی ختم ہو جاتی ہے اور کنسٹرکٹر کی طرف سے بنائی گئی چیز کو تباہ کر دیتی ہے۔ عام طور پر عوام کے تحت اعلان کیا جاتا ہے۔

مثال 14.8 کلاس میں ڈسٹرکٹر فنکشن کو واضح کرنا

```
#include<iostream>
using namespace std;
class simple
{
private:
int a, b;
public:
simple()
{
a= 0 ;
b= 0;
cout<< "\n Constructor of class-simple ";
}
void getdata()
{
cout<<"\n Enter values for a and b ";
cin>>a>>b;
}
void putdata()
{
cout<<"\nThe two integers are .. ";
cout<<a<<'\t'<< b<<endl;
cout<<"\n The sum = "<<a+b;
}
~simple()
{   cout<<"\n Destructor is executed ";}
};
int main()
{
simple s;
s.getdata();
s.putdata();
return 0;
}
```

Output:

```
Constructor of class-simple
Enter values for a and b  6 7
The two integers are .. 6      7
The sum = 13
Destructor is executed
```

14.12 ڈسٹرکٹرس کی خصوصیات
(Characteristics of Destructors)

- ڈسٹرکٹر کا وہی نام ہے جو اس کلاس کا ہے جو ٹلڈ کریکٹر '~' (tilde character) کے ذریعے پہلے لگایا گیا ہے
- ڈسٹرکٹر کے پاس دلائل نہیں ہو سکتے
- اس کی واپسی کی کوئی قسم نہیں ہے
- ڈسٹرکٹرس کو اورلوڈیڈ (overloaded) نہیں کیا جا سکتا
- وضاحت کردہ صارف ڈسٹرکٹر کی غیر موجودگی میں، یہ کمپائلر کے ذریعہ تیار کیا جاتا ہے۔
- ڈسٹرکٹر کو خود بخود اس وقت عمل میں لایا جاتا ہے جب کنٹرول آبجیکٹ کو تباہ کرنے کے لیے کلاس اسکوپ کے اختتام تک پہنچ جاتا ہے۔
- اسے حاصل نہیں کیا جاسکتا

11th_Comp Science_UM_Ch 14.indd 244 19-04-2023 10:06:39

یاد رکھنے کے نکات

- ایک کلاس ڈیٹا اور متعلقہ افعال کو ایک ساتھ باندھتی ہے۔
- C++ میں ایک کلاس صارف کو ڈیٹا کی قسم بناتی ہے جس کا استعمال کرتے ہوئے اس قسم کی آبجیکٹس بنائی جا سکتی ہیں۔
- کلاس ڈیٹا ممبرز کا اعلان کرتے وقت، ممبر کے فنکشنس،رسائی کی وضاحت کرنے والے اور کلاس ٹیگ کا نام دیا جاتا ہے۔
- کلاس کے ممبر کے افعال یا تو کلاس (اِن لائن) وضاحت کے اندر یا کلاس کی و ضاحت سے باہر بیان کیے جا سکتے ہیں۔
- کلاس کے عوامی اراکین کو اس کلاس قسم کے آبجیکٹ کا استعمال کرکے براہ راست کلاس سے باہر رسائی حاصل کی جاسکتی ہے۔
- ایک کلاس ،ڈیٹا اور فنکشنس کو ایک ساتھ منسلک کرکے OOP کی خصوصیات انکیپسولیشن کو سپورٹ کرتی ہے۔
- ایک کلاس نجی اور محفوظ اراکین کے ذریعے بیرونی دنیا سے معلومات چھپا کر ڈیٹا چھپانے کی حمایت کرتی ہے۔
- جب ایک ممبر فنکشن کو اسی کلاس کے دوسرے ممبر فنکشن کے ذریعہ بلایا جاتا ہے، تو اسے ممبر فنکشنس کا نیسٹنگ (nesting of member functions) کہا جاتا ہے۔
- اسکوپ ریزولوشن آپریٹر (::)، جب کلاس کے نام کے ساتھ استعمال ہوتا ہے تو یہ ظاہر کرتا ہے کہ ممبران اس کلاس سے تعلق رکھتے ہیں جیسا کہ

  class_name:: function_name اور صرف متغیر نام کے ساتھ استعمال کیا جاتا ہے جیسا کہ

  variable-name :: s
- عالمی متغیر کو ظاہر کرتا ہے۔
- جب کسی کلاس کی مثال دائرہ کار میں آتی ہے، تو کنسٹرکٹر (constructor) نامی ایک خاص فنکشن عمل میں آتا ہے۔
- کنسٹرکٹر فنکشن میموری کو مختص کرتا ہے اور کلاس آبجیکٹ کو شروع کرتا ہے۔
- جب کوئی کلاس آبجیکٹ دائرہ کار سے باہر ہو جاتا ہے، تو ایک خاص فنکشن جسے ڈسٹرکٹر (destructor) کہتے ہیں، عمل میں لایا جاتا ہے۔
- کنسٹرکٹر فنکشن کا نام اور ڈسٹرکٹر کا کلاس ٹیگ جیسا ہی نام ہے۔
- پیرامیٹرز کے بغیر کنسٹرکٹر کو ڈیفالٹ کنسٹرکٹر (default constructor) کہا جاتا ہے۔
- ڈیفالٹ آرگیومنٹ والا کنسٹرکٹر ڈیفالٹ کنسٹرکٹر کے برابر ہوتا ہے۔
- کنسٹرکٹر اور ڈسٹرکٹر دونوں کچھ بھی واپس نہیں کرتے۔ وہ کسی بھی قسم کے ڈیٹا سے وابستہ نہیں ہیں۔
- آبجیکٹ کو متحرک طور پر شروع کیا جا سکتا ہے۔

11th_Comp Science_UM_Ch 14.indd 245 19-04-2023 10:06:39

عملی مشق

1. درج ذیل تصریح کے ساتھ کلاس ملازم کی وضاحت کریں۔

کلاس ملازم کے نجی ممبران

empno- integer

ename – 20 characters

basic – float

netpay, hra, da, - float

کیلکولیٹ () - فلوٹ ریٹرن ٹائپ کے ساتھ basic+hra+da تلاش کرنے کے لیے ایک فنکشن

کلاس ملازم کے عوامی ممبر کے افعال

ہیو ڈیٹا ()(havedata)- کے لیے اقدار empno, ename, basic, hra, da and کو قبول کرنے کے لیے ایک فنکشن

کال کیلکولیٹ ()(call calculate) -کو نیٹ پے(netpay) کا حساب کرنے کے لیے استعمال کریں۔

ڈسپ ڈیٹا ()(dispdata) - اسکرین پر تمام ڈیٹا ممبران کو ظاہر کرنے کے لیے ایک فنکشن

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. کلاس کے اندر اعلان کردہ متغیرات کو اس نام سے جانا جاتا ہے۔

(A) ڈیٹا (data) (B) اِن لائن (inline)

(C) میتھڈ (method) (D) ایٹریبیوٹس (attributes)

2. ممبر کے افعال کے بارے میں درج ذیل میں سے کون سے بیانات درست ہیں یا غلط؟

i) ایک ممبر فنکشن ڈاٹ آپریٹر کا استعمال کرتے ہوئے براہ راست دوسرے ممبر فنکشن کو کال کر سکتا ہے۔

ii) ممبر فنکشن کلاس کے نجی ڈیٹا تک رسائی حاصل کر سکتا ہے۔

(A) (i) صحیح (ii) صحیح (B) (i) غلط (ii) صحیح

(C) (i) صحیح (ii) غلط (D) (i) غلط (ii) غلط

3. ایک ممبر فنکشن کسی دوسرے ممبر فنکشن کو براہ راست کال کر سکتا ہے، بغیر ڈاٹ آپریٹر کا استعمال کیے بغیر جسے کہتے ہیں

(A) سب فنکشن

(B) سب ممبر

(C) ممبر فنکشن کا نیسٹنگ (nesting)

(D) ممبر فنکشن کی سبلنگ (sibling)

4. کلاس کے اندر وضاحت کردہ ممبر فنکشن ..... افعال کی طرح برتاؤ کرتا ہے

(A) اِن لائن (B) نان ان لائن

(C) آؤٹ لائن (D) ڈیٹا

5. درج ذیل میں سے کون سا رسائی کی وضاحت کرنے والے ڈیٹا کو نادانستہ تبدیلیوں سے بچاتا ہے؟

(A) نجی (B) محفوظ

(C) عوامی (D) عالمی

6.
```
class x
 {
    int y;
    public:
    x(int z){y=z;}
    } x1[4];
int main()
{  x  x2(10);
   return 0;}
```

مذکورہ پروگرام کے لیے کتنے آبجیکٹ بنائے گئے ہیں؟

(A) 10 (B) 14 (C) 5 (D) 2

7. بتائیں کہ کنسٹرکٹر کے بارے میں درج ذیل بیانات درست ہیں یا غلط۔

i) کنسٹرکٹرز کو پرائیویٹ سیکشن میں قرار دیا جائے۔

ii) جب آبجیکٹ بنائے جاتے ہیں تو کنسٹرکٹرس کو خود بخود طلب کیا جاتا ہے۔

(A) (i) صحیح (ii) صحیح (B) (i) صحیح (ii) غلط

(C) (i) غلط (ii) صحیح (D) (i) غلط (ii) غلط

11th_Comp Science_UM_Ch 14.indd 246 19-04-2023 10:06:39

8. مندرجہ ذیل پروٹو ٹائپ کے لیے مندرجہ ذیل کنسٹرکٹر پر عمل درآمد کیا جاتا ہے؟

```
add display(add &);  // add is
a class name
```

(A) ڈیفالٹ کنسٹرکٹر

(B) پیرامیٹرائزڈ (parameterized) کنسٹرکٹر

(C) کاپی کنسٹرکٹر

(D) نان پیرامیٹرائزڈ (non parameterized) کنسٹرکٹر

سیکشن – B

بہت ہی مختصر جوابات دیں

1. ممبران کسے کہتے ہیں؟
2. اسٹرکچر اور کلاس میں فرق کریں حالانکہ دونوں صارف کی وضاحت کردہ ڈیٹا کی قسم ہیں۔
3. oop کے لحاظ سے کلاس اور آبجیکٹ میں کیا فرق ہے؟
4. کنسٹرکٹر (constructor) کی وضاحت کرنا ایک اچھا عمل کیوں سمجھا جاتا ہے؟ حالانکہ کمپائلر خود بخود کنسٹرکٹر بنا سکتا ہے؟
5. ڈسٹرکٹر (destructor) کی اہمیت لکھیں۔

سیکشن-C

مختصر جوابات دیں

1. سنٹیکس کی غلطیوں کو دور کرنے کے بعد درج ذیل پروگرام کو دوبارہ لکھیں، اگر کوئی غلطی ہے توغلطیوں کو انڈر لائن کریں:

```
#include<iostream>
$include<stdio>
class mystud
{       int studid =1001;
        char name[20];
   public
        mystud( )     { }
        void register ( )
        {cin>>stdid;  gets(name);   }
void display ( )
{cout<<studid<<": "<<name<<endl;}
}
int main( )
{ mystud MS;
  register.MS( );
  MS.display( );
}
```

2. مثال کے ساتھ لکھیں کہ آپ آبجیکٹس کو متحرک طور پر کیسے شروع کریں گے؟
3. عوامی رسائی کے تحت کنسٹرکٹر اور ڈسٹرکٹر قرار دینے کے کیا فوائد ہیں؟
4. درج ذیل C++ کوڈ کو دیکھتے ہوئے، سوالات (i) اور (ii) کے جواب دیں۔

```
class TestMeOut
{
 public:
 ~TestMeOut()        //Function 1
  {cout<<"Leaving the exam hall"<<endl;}
  TestMeOut()        //Function 2
  {cout<<"Appearing for exam"<<endl;}
   void MyWork()   //Function 3
  {cout<<"Answering"<<endl;}    };
```

(i) آبجیکٹ اورین ٹیڈ پروگرامنگ میں، فنکشن 1 کو کیا کہا جاتا ہے اور اسے کب کہا/بلایا جاتا ہے؟

(ii) آبجیکٹ اورین ٹیڈ پروگرامنگ میں، فنکشن 2 کو کیا کہا جاتا ہے اور اسے کب کہا/بلایا جاتا ہے؟

سیکشن – D

تفصیل سے بتائیں

1. کنسٹرکٹراور ڈسٹرکٹر کے درمیان فرق کا ذکر کریں۔
2. C++ میں درج ذیل وضاحت کے ساتھ کلاس RESORT کی وضاحت کریں:

```
Private members:
Rno  // Data member to store room number
Name   //Data member to store user name
```


11th_Comp Science_UM_Ch 14.indd 247 19-04-2023 10:06:39

Charges //Data member to store per day charge

Days //Data member to store the number of days

Compute( )/*A function to calculate total amount as Days * Charges and if the total amount exceeds 11000 then total amount is 1.02 * Days *Charges */

Public member:

GetInfo( ) /* Function to Read the information like name , room no, charges and days*/

DispInfo( )/* Function to display all entered details and total amount calculated using COMPUTE function*/

3. درج ذیل کا آوٹ پٹ لکھیے

```
#include<iostream>
using namespace std;
class student
{
        int rno, marks;
        public:
        student(int r,int m)
        { cout<<"Constructor "<<endl;
          rno=r;
          marks=m;
        }
        void printdet()
        {
          marks=marks+30;
          cout<<"Name: Bharathi"<<endl;
          cout<<"Roll no : "<<rno<<"\n";
          cout<<"Marks : "<<marks<<endl;
          }
};
int main()
{
        student s(14,70);
        s.printdet();
        cout<< "Back to Main";
        return 0;
}
```

حوالہ جات:

(1) Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.

(2) The Complete Reference C++ (Forth Edition), Herbert Schildt.Mc.Graw Hills.

11th_Comp Science_UM_Ch 14.indd 248 19-04-2023 10:06:39

پولیمورفزم
(Polymorphism)



سیکھنے کے مقاصد

اس باب کو سیکھنے کے بعد طلباء کے قابل ہوجائیں گے۔

- اوور لوڈنگ کے مقصد کو سمجھنے
- فنکشن، کنسٹرکٹر اور آپریٹر اوورلوڈنگ کا استعمال کرتے ہوئے C++ پروگرام بنانے
- ایسے پروگراموں کو انجام دینے اور ڈیبگ کرنے جس میں پولیمورفزم کا تصور ہو

15.1 تعارف

پولیمورفزم لفظ کا مطلب ہے بہت سی شکلیں (پولی – poly – کئی، مورف – morph – شکلیں) پولیمورفزم ایک پیغام یا فنکشن کی ایک سے زیادہ شکلوں میں ظاہر ہونے کی صلاحیت ہے۔ C++ میں، پولیمورفزم فنکشن اوورلوڈنگ اور آپریٹر اوورلوڈنگ کے ذریعے حاصل کیا جاتا ہے۔ اوور لوڈنگ کی اصطلاح کا مطلب ہے ایک نام جس کے دو یا زیادہ الگ الگ معنی ہوں۔ اس طرح ایک 'اوور لوڈیڈ فنکشن overloaded function –' سے مراد ایک فنکشن ہے جس کے ایک سے زیادہ الگ معنی ہیں۔

15.2 فنکشن اوورلوڈنگ

پیغام یا ڈیٹا کو ایک سے زیادہ شکلوں میں پروسیس کرنے کے فنکشن کی صلاحیت کو فنکشن اوورلوڈنگ کہا جاتا ہے۔ دوسرے لفظوں میں فنکشن اوورلوڈنگ کا مطلب ہے کہ ایک ہی دائرہ کار میں دو یا زیادہ فنکشنس ایک ہی نام کا اشتراک کرتے ہیں لیکن ان کے پیرامیٹرز مختلف ہیں۔ اس صورت حال میں، ایک ہی نام کے حامل فنکشنس کو اوورلوڈ کہا جاتا ہے اور اس عمل کو فنکشن اوورلوڈنگ کہا جاتا ہے۔ فنکشن کے پیرامیٹرز کی تعداد اور اقسام کو فنکشن کا دستخط (function's signature) کہا جاتا ہے۔ جب آپ کسی اوورلوڈیڈ فنکشن کو کال کرتے ہیں، تو کمپائلر استعمال کرنے کے لیے موزوں ترین وضاحت کا تعین کرتا ہے، ان آرگیومنٹ کی اقسام کا موازنہ کرکے جو آپ نے وضاحتوں میں بیان کردہ پیرامیٹر کی اقسام کے ساتھ فنکشن کو کال کرنے کے لیے استعمال کی ہیں۔ انتہائی مناسب اوورلوڈ فنکشن یا آپریٹر کو منتخب کرنے کے عمل کو اوورلوڈ ریزولوشن (overload resolution) کہا جاتا ہے۔

15.2.1 فنکشن اوور لوڈنگ کی ضرورت

کبھی کبھی کسی ایک عمل کے لیے بہت سے مختلف معنی خیز نام تلاش کرنا مشکل ہوتا ہے۔

دائرہ، مثلث اور مستطیل کا رقبہ معلوم کرنے کے لیے صورتحال پر غور کریں جو مندرجہ ذیل فنکشن کا پروٹو ٹائپ دیا گیا ہے۔

```
float area_circle(float radius)                          // دائرے کے رقبے کا حساب لگانے کے لیے
float area_triangle(float half,floatbase,float height)   // مثلث کے رقبے کا حساب لگانے کے لیے
float area_rectangle(float length , float breadth)       // مستطیل کے رقبے کا حساب لگانے کے لیے
```


11th_Comp Science_UM_Ch 15.indd 249 17-04-2023 13:01:47

اسے مندرجہ ذیل طریقے سے سنگل فنکشن ہیڈر کا استعمال کرتے ہوئے دوبارہ لکھا جا سکتا ہے۔

```
float area ( float radius );
float area ( float half, float base, float height );
float area ( float length , float breadth );
```

مثال 15.1 فنکشن اوور لوڈنگ کو ظاہر کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
void print(int i)
 {cout<< " It is integer " << i <<endl;}
void print(double  f)
{  cout<< " It is float " << f <<endl;}
void print(string c)
{  cout<< " It is string " << c <<endl;}
int main() {
        print(10);
        print(10.10);
        print("Ten");
        return 0;
}
```

Output:

```
It is integer 10
 It is float 10.1
It is string Ten
```

ٹپ نوٹس

فنکشن اوور لوڈنگ نہ صرف پولیمورفزم کو نافذ کر رہا ہے بلکہ پروگرام میں موازنہ کی تعداد کو بھی کم کرتا ہے اور پروگرام کو تیزی سے عمل میں لاتا ہے۔ یہ یاد رکھنے والے فنکشن کے ناموں کی تعداد کو کم کرکے پروگرامر کی مدد کرتا ہے۔

15.2.2 فنکشن اوور لوڈنگ کے اصول

1. اوورلوڈ فنکشن اس کے آرگیومنٹس (arguments) یا ڈیٹا کی اقسام کی تعداد میں مختلف ہونا چاہیے۔
2. ایک ہی ڈیٹا کی قسم کو اوور لوڈ کرنے کے لیے اوورلوڈ فنکشنس کی واپسی کی قسم پر غور نہیں کیا جاتا ہے۔
3. اوورلوڈ فنکشنس کے پہلے سے طے شدہ آرگیومنٹس کو فنکشن اوور لوڈنگ میں پیرامیٹر لسٹ کا حصہ نہیں سمجھا جاتا ہے۔

11th_Comp Science_UM_Ch 15.indd 250 17-04-2023 13:01:47

مثال 15.2 فنکشن اوور لوڈنگ کو ظاہر کرنے کے لیے C++ پروگرام

```
#include <iostream>
using namespace std;
long add(long, long);
long add(long,long,long);
float add(float, float);
int main()
{
 long a, b, c,d;
 float e, f, g;
 cout << "Enter three integers\n";
cin >> a >> b>>c;
d=add(a,b,c);          //number of arguments different  but  same data type
cout << "Sum of 3 integers: " << d << endl;
cout << "Enter two integers\n";
cin >> a >> b;
c = add(a, b);         //two arguments with same data type
cout << "Sum of 2 integers: " << c << endl;
cout << "Enter two floating point numbers\n";
 cin >> e >> f;
 g = add(e, f);        //two arguments with different data type
cout << "Sum of floats: " << g << endl;
}
long add(long c, long g)
{
 long sum;
 sum = c + g;
 return sum;
}
float add(float c, float g)
{
 float sum;
 sum = c + g;
 return sum;
}
long add(long c, long g,long h)
{
 long sum;
 sum = c + g+h;
 return sum;
}
```

Output

```
Enter three integers
3 4 5
Sum of 3 integers: 12
Enter two integers
4 6
Sum of 2 integers: 10
Enter two floating point numbers
2.1 3.1
Sum of floats: 5.2
```


11th_Comp Science_UM_Ch 15.indd 251 17-04-2023 13:01:47

15.3 کنسٹرکٹر اوور لوڈنگ
(Constructor overloading)

فنکشن اوور لوڈنگ کو کنسٹرکٹرس کے لیے لاگو کیا جا سکتا ہے، کیونکہ کنسٹرکٹرس کلاسس کے خصوصی فنکشن ہوتے ہیں ۔ ایک کلاس میں مختلف دستخطوں کے ساتھ ایک سے زیادہ کنسٹرکٹر ہو سکتے ہیں۔ کنسٹرکٹر اوور لوڈنگ کلاس کے لیے متعدد قسم کی آبجیکٹس (objects) بنانے کی لچک فراہم کرتی ہے

مثال 15.3 کنسٹرکٹر اوور لوڈنگ

Compiler identifies overloaded constructor by its name and parameters list.

```
#include<iostream>
using namespace std;
class add
{ int num1, num2, sum;
public:
add()
{       cout<<"\n Constructor without parameters.. ";
num1= 0;  num2= 0;  sum = 0; }
add ( int s1, int s2 )
{ cout<<"\n Parameterized constructor... ";
 num1= s1;  num2=s2; sum=0; }
add (add &a)
{ cout<<"\n Copy Constructor ... ";
 num1= a.num1;
 num2=a.num2;
 sum = 0; }
void getdata()
{ cout<<"\nEnter data ... "; cin>>num1>>num2; }
void addition()
{ sum=num1+num2; }
void putdata() {
cout<<"\n The numbers are..";
cout<<num1<<'\t'<<num2;
cout<<"\n The sum of the numbers are.. "<< sum; } };
int main() {
add a, b (10, 20) , c(b);
a.getdata();
a.addition();
b.addition();
c.addition();
cout<<"\n Object a : ";
a.putdata();
cout<<"\n Object b : ";
b.putdata();
cout<<"\n Object c.. ";
c.putdata();
return 0; }
```


11th_Comp Science_UM_Ch 15.indd 252 17-04-2023 13:01:47

Output

```
Constructor without parameters..
 Parameterized constructor...
 Copy Constructor ...
Enter data ...  20 30
 Object a :
 The numbers are..20    30
 The sum of the numbers are.. 50
 Object b :
 The numbers are..10    20
 The sum of the numbers are.. 30
 Object c..
 The numbers are..10    20
 The sum of the numbers are.. 30
```



نوٹ

چونکہ، ایک سے زیادہ کنسٹرکٹرس موجود ہیں، کسی آبجیکٹ کو بناتے وقت کنسٹرکٹر کو آرگیومنٹ بھی پاس کی جانی چاہیے۔

15.4 آپریٹر اوور لوڈنگ

آپریٹر اوور لوڈنگ کی اصطلاح سے مراد عام C++ آپریٹرس جیسے +,++,-,--,=+,=-,.*,<,>کو اضافی فعالیت دینا ہے۔ یہ پولیمورفزم کی ایک قسم بھی ہے جس میں ایک آپریٹر کو اوورلوڈ کیا جاتا ہے تاکہ صارف کو اس کے متعین معنی فراہم کیے جا سکیں۔

مثال کے طور پر '+' آپریٹر کو ڈیٹا کی مختلف اقسام میں اضافہ کرنے کے لیے اوورلوڈ کیا جا سکتا ہے، جیسے کہ انٹیجر (integer)، اسٹرنگ (string) (کن کیٹی نیشن - ن concatenation) وغیرہ۔

تقریباً تمام آپریٹرز کو C++میں اوورلوڈ (overload) کیا جا سکتا ہے۔ تاہم کچھ آپریٹر ہیں جن پر اوورلوڈ نہیں کیا جا سکتا۔ آپریٹر جو اوورلوڈ نہیں ہیں درج ذیل ہیں۔

- اسکوپ آپریٹر (::)
- سائز آف
- ممبر سلیکٹر (.)
- ممبر پوائنٹر سلیکٹر (*)
- ٹرنری آپریٹر (?:)

آپریٹر اوورلوڈنگ سنٹیکس

Keyword **Operator to be overloaded**

```
ReturnType classname :: Operator Operator Symbol (argument list)
{       \\ Function body }
```


11th_Comp Science_UM_Ch 15.indd 253 17-04-2023 13:01:47

15.4.1 آپریٹر اوور لوڈنگ پر پابندیاں

آپریٹر اوور لوڈنگ کو لاگو کرتے وقت ذہن میں رکھنے کے لیے درج ذیل کچھ پابندیاں ہیں۔

1. آپریٹر کی ترجیح اور ہم آہنگی کو تبدیل نہیں کیا جا سکتا۔
2. کوئی نیا آپریٹرس نہیں بنایا جا سکتا، صرف موجودہ آپریٹرس کو اوورلوڈ کیا جا سکتا ہے۔
3. آپریٹر کے طریقہ کار کے معنی کو دوبارہ بیان نہیں کر سکتا۔ آپ انٹیجرس کو شامل کرنے کے طریقے کو تبدیل نہیں کر سکتے۔ آپریٹر کو صرف اضافی فنکشن دیے جا سکتے ہیں۔
4. اوورلوڈیڈ آپریٹرس کے پاس پہلے سے طے شدہ آرگیومنٹس نہیں ہوسکتے ہیں۔
5. جب بائنری آپریٹرس اوورلوڈیڈ ہوتے ہیں، تو بائیں ہاتھ کا آبجیکٹ متعلقہ کلاس کی ایک آبجیکٹ ہونی چاہیے۔

مثال 15.5 بائنری آپریٹر اورلوڈنگ پیچیدہ نمبر کے اضافے میں '+' کا استعمال کرتے ہوئے

```
#include<iostream>
using namespace std;
class complex
{  int real,img;
   public:
void read()
{
cout<<"\nEnter the REAL PART : ";
cin>>real;
cout<<"\nEnter the IMAGINARY PART : ";
cin>>img;
}
complex operator +(complex c2)
{
    complex c3;
    c3.real=real+c2.real;
    c3.img=img+c2.img;
    return c3;
}
    void display()
{  cout<<real<<"+"<<img<<"i";  }
};
int main()
{
    complex c1,c2,c3;
    int choice, cont;
    cout<<"\n\nEnter the First Complex Number";
    c1.read();
    cout<<"\n\nEnter the Second Complex Number";
    c2.read();
    c3=c1+c2;  // binary + overloaded
    cout<<"\n\nSUM = ";
    c3.display();
    return 0;
 }
```

Output

```
Enter the First Complex Number
Enter the REAL PART : 3
Enter the IMAGINARY PART : 4
Enter the Second Complex Number
Enter the REAL PART : 5
Enter the IMAGINARY PART : 8
SUM = 8+12i
```


11th_Comp Science_UM_Ch 15.indd 254 17-04-2023 13:01:47

مثال 15.6 آپریٹر اوور لوڈنگ کا استعمال کرتے ہوئے سٹرنگ کا مجموعہ

```
#include<string.h>
#include<iostream>
using namespace std;
class strings
{
        public:
        char s[20];
        void getstring(char str[])
{
        strcpy(s,str);
  }
        void operator+(strings);
};
void strings::operator+(strings ob)
{
        strcat(s,ob.s);
        cout<<"\nConcatnated String is:"<<s;
}
int main()
{
        strings ob1, ob2;
        char string1[10], string2[10];
        cout<<"\nEnter First String:";
        cin>>string1;
        ob1.getstring(string1);
        cout<<"\nEnter Second String:";
        cin>>string2;
        ob2.getstring(string2);
        //Calling + operator to Join/Concatnate strings
        ob1+ob2;
        return 0;
}
```

Output

```
Enter First String:COMPUTER
Enter Second String:SCIENCE
Concatnated String is:COMPUTERSCIENCE
```


11th_Comp Science_UM_Ch 15.indd 255 17-04-2023 13:01:47

یاد رکھنے کے نکات

- C++ میں، پولیمورفزم فنکشن اوورلوڈنگ اور آپریٹر اوورلوڈنگ کے ذریعے حاصل کیا جاتا ہے۔
- اصطلاح اوور لوڈنگ کا مطلب ہے ایک نام جس کے دو یا زیادہ الگ الگ معنی ہوں۔
- اوورلوڈیڈ فنکشن سے مراد ایک فنکشن ہے جس کے ایک سے زیادہ الگ معنی ہیں۔
- اوور لوڈیڈ فنکشنس کا نام ایک ہی ہے لیکن دستخط مختلف ہیں (آرگیومنٹ کی تعداد اور آرگیومنٹ کی قسم)
- ایک فنکشن کی آرگیومنٹ کی فہرست کو فنکشن دستخط کے طور پر جانا جاتا ہے۔
- دو فنکشن کو اوورلوڈ نہیں کیا جا سکتا جب کہ فرق صرف یہ ہے کہ ایک حوالہ پیرامیٹر لیتا ہے اور دوسرا نارمل، کال بہ قدر پیرامیٹر لیتا ہے(call-by-value parameter)۔
- عام کاموں کے ساتھ ساتھ ممبر کے افعال کو اوورلوڈ کیا جا سکتا ہے۔
- ایک کلاس میں اوورلوڈ کنسٹرکٹرس ہوسکتے ہیں جہاں ڈسٹرکٹرس فنکشن کو اوورلوڈ نہیں کیا جاسکتا۔
- آپریٹر کو خاص معنی دینے کا طریقہ کار آپریٹر اوور لوڈنگ کے نام سے جانا جاتا ہے۔
- آپریٹر اوور لوڈنگ زیادہ تر C++ آپریٹرس کے لیے نئی وضاحتیں فراہم کرتا ہے۔
- یہاں تک کہ صارف کی وضاحت کردہ اقسام (آبجیکٹس) کو بھی اوورلوڈ کیا جا سکتا ہے۔
- اوورلوڈڈ آپریٹر کی وضاحت کلیدی لفظ 'آپریٹر operator' کے بعد آپریٹر کی علامت کے ساتھ دی گئی ہے۔
- ہم درج ذیل کے علاوہ تمام C++ آپریٹرس کو اوورلوڈ کر سکتے ہیں۔
- اسکوپ ریزولوشن آپریٹر (::)، سائز آف ()، کنڈیشنل آپریٹر (?:)، ممبر سلیکشن(.) اور ممبر پوائنٹر سلیکٹر (*) آپریٹر

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. مندرجہ ذیل میں سے کون سا ایک فنکشن ہے جس سے مراد ایک سے زائد معنی ہیں؟
   (A) فنکشن اوورلوڈنگ (B) ممبر اوورلوڈنگ
   (C) آپریٹر اوور لوڈنگ (D) آپریشنس اوور لوڈنگ

2. مندرجہ ذیل میں سے کون سا پروگرام ہے جو موازنہ کی تعداد کو کم کرتا ہے؟
   (A) آپریٹر اوور لوڈنگ (B) آپریشنس اوور لوڈنگ
   (C) فنکشن اوورلوڈنگ (D) ممبر اوورلوڈنگ

3.
```
void dispchar(char ch='$',int size=10)
{
    for(int i=1;i<=size;i++)
    cout<<ch;
}
```


11th_Comp Science_UM_Ch 15.indd 256 17-04-2023 13:01:47

آپ مندرجہ ذیل ان پٹ کے لیے فنکشن dispchar() کو کیسے استعمال کریں گے؟
10 بار $ پرنٹ کرنے کے لئے

dispchar(); (A)　　dispchar(ch,size); (B)

dispchar($,10); (C)　　dispchar('$',10 times); (D)

4. مندرجہ ذیل میں سے کون سا فنکشن اوور لوڈنگ کے حوالے سے درست نہیں ہے؟
(A) اوورلوڈ فنکشنس کو ان کے دستخط میں فرق ہونا چاہیے۔
(B) کسی فنکشن کو اوور لوڈ کرنے کے لیے واپسی کی قسم پر بھی غور کیا جاتا ہے۔
(C) اوور لوڈیڈ فنکشنس کے پہلے سے طے شدہ آرگیومنٹس اوور لوڈنگ کے لیے غور نہیں کیا جاتا ہے۔
(D) ڈسٹرکٹر فنکشن کو اوورلوڈ یڈ نہیں کیا جا سکتا۔

5. مندرجہ ذیل میں سے کون سا فنکشن اوور لوڈنگ کے لیے غلط پروٹو ٹائپ ہے۔

(A) void fun (intx);
void fun (char ch) ;

(B) void fun (intx);
void fun (inty);

(C) void fun (double d);
void fun (char ch);

(D) void fun (double d);
void fun (inty);

سیکشن - B

بہت ہی مختصر جوابات دیں

1. فنکشن اوورلوڈنگ کیا ہے؟
2. ان آپریٹرز کی فہرست بنائیں جنہیں اوورلوڈ نہیں کیا جا سکتا۔
3. کلاس ایڈ ;{add{int x; public: add(int)} کنسٹرکٹر کے لیے آؤٹ لائن کی وضاحت لکھیں۔
4. کیا فنکشن کی واپسی کی قسم کسی فنکشن کو اوور لوڈ کرنے میں مدد کرتی ہے؟
5. کسی فنکشن کو اوور لوڈ کرنے کا کیا فائدہ ہے؟

سیکشن - C

مختصر جوابات دیں

1. فنکشن اوور لوڈنگ کے اصول کیا ہیں؟
2. ایک کمپائلر یہ کیسے طے کرتا ہے کہ جب بہت سے فنکشنس ہوں تو کون سا فنکشن شروع کیا جائے؟ ایک مثال دیں۔
3. آپریٹر اوور لوڈنگ کیا ہے؟ آپریٹرس کی کچھ مثالیں دیں جو اوورلوڈ ہو سکتے ہیں۔
4. کنسٹرکٹر اوور لوڈنگ کے فوائد پر تبادلہ خیال کریں؟
5. کلاس سیل ;{int cost, discount ;public: sale(sale &)} مخصوص کنسٹرکٹر کے لیے ایک نان اِن لائن (non inline) کی وضاحت لکھیں۔

11th_Comp Science_UM_Ch 15.indd 257　17-04-2023 13:01:47

سیکشن – D

تفصیل سے بتائیں

1. آپریٹر اوور لوڈنگ کے کیا اصول ہیں؟
2. درج ذیل کلاس سے گزرنے کے بعد سوال (i) سے (v) کا جواب دیں۔

```
class Book {
int BookCode ; char Bookname[20];float fees;
public:
Book( )                          //Function 1
{ fees=1000;
  BookCode=1;
  strcpy(Bookname,"++C");  }
void display(float C)            //Function 2
{   cout<<BookCode<<":"<<Bookname<<":"<<fees<<endl;   }
~Book( )                         //Function 3
{  cout<<"End of Book Object"<<endl;   }
      Book (int SC,char S[ ],float F)  ;          //Function 4
};
```

(i) مذکورہ پروگرام میں، فنکشن 1 اور فنکشن 4 کو ایک ساتھ ملا کر کیا کہا جاتا ہے؟

(ii) فنکشن 3 کے ذریعہ کون سا تصور بیان کیا گیا ہے؟ اس فنکشن کو کب کہا جاتا ہے / طلب کیا جاتا ہے؟

(iii) فنکشن 3 کا استعمال کیا ہے؟

(iv) فنکشن 1 اور فنکشن 2 کو شروع کرنے کے لیے اسٹیٹ منٹس (statements) کو مین (main) میں لکھیں۔

(v) فنکشن 4 کی وضاحت لکھیں۔

3. درج ذیل پروگرام کا آؤٹ پٹ لکھیں۔

```
include<iostream>
using namespace std;
class Seminar
{ int Time;
public:
Seminar()
 {      Time=30;cout<<"Seminar starts now"<<endl;   }
void Lecture()
{ cout<<"Lectures in the seminar on"<<endl; }
Seminar(int Duration)
{   Time=Duration;cout<<"Welcome to Seminar "<<endl; }
Seminar(Seminar &D)
{   Time=D.Time;cout<<"Recap of Previous Seminar Content "<<endl;}
~Seminar()
```


11th_Comp Science_UM_Ch 15.indd 258 17-04-2023 13:01:47

```
{cout<<"Vote of thanks"<<endl; } };
int main()
{       Seminar s1,s2(2),s3(s2);
        s1.Lecture();
        return 0;
}
```

4. درج ذیل پروگرام کی بنیاد پر سوالات کا جواب دیجیے

```
#include<iostream>
#include<string.h>
using namespace std;
class comp {
public:
char s[10];
void getstring(char str[10])
   { strcpy(s,str);    }
void operator==(comp);
};
void comp::operator==(comp ob)
{ if(strcmp(s,ob.s)==0)
 cout<<"\nStrings are Equal";
else
 cout<<"\nStrings are not Equal"; }
int main()
{ comp ob, ob1;
char string1[10], string2[10];
cout<<"Enter First String:";
cin>>string1;
ob.getstring(string1);
cout<<"\nEnter Second String:";
cin>>string2;
ob1.getstring(string2);
ob==ob1;
return 0; }
```

(i) ان آبجیکٹس (objects) کا ذکر کریں جن کی گنجائش پروگرام کے اختتام تک ہوگی۔

(ii) اس آبجیکٹ کا نام بتائیں جو پروگرام کے درمیان ختم ہو جاتی ہے۔

(iii) اس آپریٹر کا نام لکھیں جو اوور لوڈیڈ (over loaded) ہو اور وہ بیان لکھیں جو اسے طلب کرتا ہے۔

(iv) اوورلوڈیڈ ممبر فنکشن کا پروٹو ٹائپ لکھیں۔

(v) اوورلوڈ یڈ آپریٹر کے لیے کس قسم کے آپرینڈز (operands) استعمال کیے جاتے ہیں؟

(vi) مندرجہ بالا پروگرام میں کون سا کنسٹرکٹر عمل میں لایا جائے گا؟ پروگرام کا آؤٹ پٹ لکھیں۔

کیس اسٹڈی

فرض کریں کہ آپ کے پاس ایک کٹّی بینک (Kitty Bank) ہے جس کی ابتدائی رقم 500 روپے ہے اور آپ کو اس میں کچھ اور رقم شامل کرنی ہوگی۔ 500 روپے کی ابتدائی قیمت کے ساتھ 'رقم – amount' نام کے ڈیٹا ممبر کے ساتھ ایک کلاس 'ڈپازٹ – Deposit' بنائیں۔ اب اس کلاس کے تین کنسٹرکٹرس مندرجہ ذیل بنائیں:

1. بغیر کسی پیرامیٹر (parameter) کے - کٹی بینک میں کوئی رقم شامل نہیں کی جائے گی۔
2. ایک پیرامیٹر ہے، وہ کون سا رقم ہے جو کٹی بینک میں شامل کی جائے گی۔
3. جب بھی رقم شامل کی جائے گی ایک اضافی مساوی رقم خود بخود جمع ہو جائے گی۔

'ڈپازٹ' کا ایک آبجیکٹ بنائیں اور کٹی بینک میں حتمی رقم ظاہر کریں۔

حوالہ جات:

1. Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.
2. The Complete Reference C++ (Forth Edition), Herbert Schildt. Mc.Graw Hills.
3. The C++ Programming Language,BjarneStroustrup

11th_Comp Science_UM_Ch 15.indd 259 17-04-2023 13:01:48

باب 16

انہیریٹینس
(Inheritance)

سیکھنے کے مقاصد



اس باب کی تکمیل کے بعد طلباء کے قابل ہوجائیں گے۔

- انہیریٹنس کے مقصد کو سمجھنے
- انہیریٹنس کا استعمال کرتے ہوئے C++ پروگرام بنانے
- ان پروگراموں کو انجام دینے اور ڈیبگ کرنے جس میں انہیریٹنس کا تصور موجود ہو

16.1 انہیریٹینس کا تعارف

انہیریٹنس آبجیکٹ اورین ٹیڈ پروگرامنگ کی سب سے اہم خصوصیات میں سے ایک ہے۔ آبجیکٹ اورین ٹیڈ پروگرامنگ میں، انہیریٹنس نئی کلاس اور اس کی آبجیکٹس کو موجودہ کلاسس کی خصوصیات کو لینے کے قابل بناتی ہے۔ ایک کلاس جو نئی کلاس بنانے کی بنیاد کے طور پر استعمال ہوتی ہے اسے سوپر کلاس (super class) یا بیس کلاس (base class) کہا جاتا ہے۔ ایک کلاس جو سو پر کلاس سے ملتی ہے اسے سب کلاس (subclass) یا ڈرائیوڈ کلاس (derived class) کہا جاتا ہے۔

16.2 انہیریٹنس کی ضرورت

انہیریٹنس آبجیکٹ اورین ٹیڈ پروگرامنگ کی ایک اہم خصوصیت ہے جو کوڈ کے دوبارہ استعمال کے لیے استعمال ہوتی ہے۔ یہ موجودہ یا بنیادی کلاسوں سے نئی کلاسیں بنانے کا عمل ہے جسے ڈرائیوڈ کلاسس (derived classes) کہتے ہیں۔ انہیریٹنس ہمیں ایک کلاس کے تمام کوڈ (سوائے پرائیویٹ قرار دے کر) دوسری کلاس کو لینے کی اجازت دیتی ہے۔ حاصل ہونے والی کلاس کو بیس کلاس (base class) یا پیرنٹ کلاس (parent class) کہا جاتا ہے اور جو کلاس دوسری کلاس میں ملتی ہے اسے ڈرائیوڈکلاس (derived class) یا چائلڈ کلاس (child class) کہا جاتا ہے۔ ڈرائیوڈکلاس ایک پاور پیکڈ کلاس (power packed class) ہے، کیونکہ یہ اضافی صفات اور طریقے شامل کر سکتی ہے اور اس طرح اس کی فعالیت کو بڑھا سکتی ہے۔

نوٹس

انہیریٹنس کا بنیادی فائدہ ہے۔

- یہ حقیقی دنیا کے تعلقات کی اچھی طرح نمائندگی کرتا ہے۔
- یہ کوڈ کو دوبارہ استعمال کرنے کے قابل بناتا ہے۔
- یہ منتقلی کی حمایت کرتا ہے

16.3 انہیریٹنس کی اقسام

انہیریٹنس کی مختلف قسمیں ہیں مثلاً سنگل انہیریٹنس (single inheritance)، ملٹی پل انہیریٹنس(multiple inheritance)، ملٹی لیول انہیریٹنس (multilevel inheritance)، ہائبرڈ انہیریٹنس (hybrid inheritance) اور ہائیر آرکیکل انہیریٹنس (hierarchical inheritance)۔

1. سنگل انہیریٹنس (Single inheritance)

جب ایک ڈرائیوڈ کلاس صرف ایک بیس کلاس سے ملتی ہے، تو اسے سنگل انہیریٹنس کے نام سے جانا جاتا ہے۔

2. ملٹی پل انہیریٹنس (Multiple inheritance)

جب ایک ڈرائیوڈ کلاس ملٹی پل بیس کلاسس سے ملتی ہے تو اسے ملٹی پل انہیریٹنس کے نام سے جانا جاتا ہے۔

3. ہائیر آرکیکل انہیریٹنس (Hierarchical inheritance)

جب ایک ہی بیس کلاس سے ایک سے زیادہ ڈرائیوڈ کلاسس بنائے جاتے ہیں، تو اسے ہائیر آرکیکل انہیریٹنس کہا جاتا ہے۔

11th_Comp Science_UM_Ch 16.indd 260 17-04-2023 13:29:39

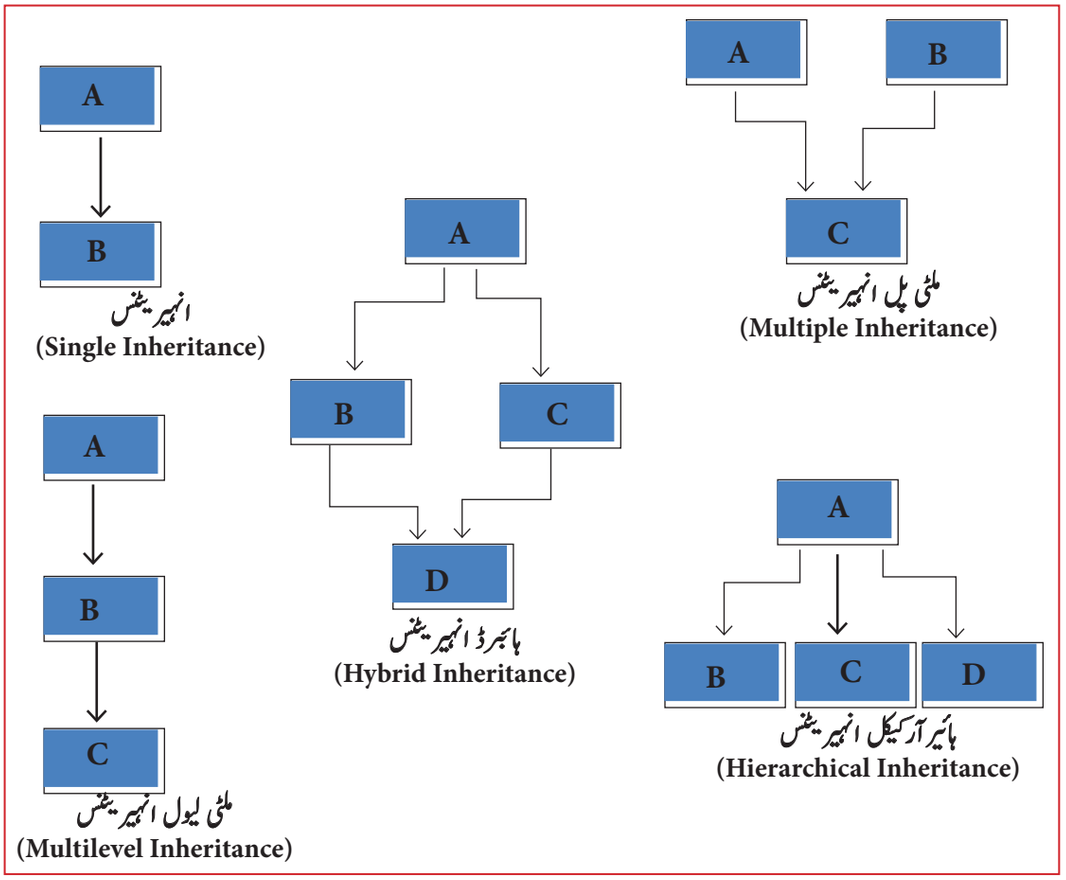
4. ملٹی لیول انہیریٹنس (Multilevel inheritance)

انہیریٹنس کی عبوری نوعیت انہیریٹنس کی اِس شکل سے ظاہر ہوتی ہے۔ جب ایک کلاس کسی کلاس سے اخذ کی جاتی ہے جو ایک ڈرائیوڈکلاس ہے - تو اسے ملٹی لیول انہیریٹنس کہا جاتا ہے۔

5. ہائبرڈ انہیریٹنس (Hybrid inheritance)

جب انہیریٹنس کی ایک سے زیادہ اقسام کا مجموعہ ہو تو اسے ہائبرڈ انہیریٹنس کہا جاتا ہے۔ لہٰذا، یہ ملٹی لیول اور ملٹی پل انہیریٹنس یا ہائیر آرکیکل اور ملٹی لیول انہیریٹنس یا ہائیر آرکیکل، ملٹی لیول اور ملٹی پل انہیریٹنس کا مجموعہ ہو سکتا ہے۔

درج ذیل خاکہ انہیریٹنس کی مختلف اقسام کی نمائندگی کرتا ہے



16.4 ڈرائیوڈ کلاس اور بیس کلاس
(Derived Class and Base class)

ڈرائیوڈ کلاس کی وضاحت کرتے وقت، ڈرائیوڈ کلاس کو اس کلاس کی شناخت کرنی چاہیے جس سے یہ اخذ کیا گیا ہے۔ ڈرائیوڈ کلاس کی وضاحت کے لیے درج ذیل نکات کا مشاہدہ کیا جانا چاہیے۔

i) کلیدی لفظ کلاس استعمال کرنا ہے۔

ii) ڈرائیوڈ کلاس کا نام کلیدی لفظ کلاس کے بعد دیا جانا ہے۔

iii) ایک سنگل کولن (:) (A single colon)

iv) اخذ کی قسم (مرئیت کا طریقہ - visibility mode)، یعنی نجی (Private)، عوامی (Public) یا محفوظ (Protected)۔ اگر کوئی مرئیت کا طریقہ متعین نہیں ہے، تو بذریعہ ڈیفالٹ مرئیت کے طریقہ کو نجی سمجھا جاتا ہے۔

v) بیس کلاس (پیرنٹ کلاس) کا نام، اگر ایک سے زیادہ بیس کلاس ہے، تو اسے کوما (comma) سے الگ کیا جا سکتا ہے۔

11th_Comp Science_UM_Ch 16.indd 261 17-04-2023 13:29:39

class derived_class_name :visibility_mode base_class_name

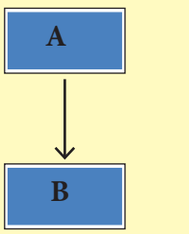
{ // members of derivedclass };

انہیریٹنس کی مختلف شکلوں کی چند مثالیں درج ذیل ہیں۔

16.4.1 سنگل انہیریٹنس

مثال 16.1 سنگل انہیریٹنس

```
# include <iostream>
using namespace std;
class student          //base class
{
private :
char name[20];
int rno;
public:
void acceptname()
{
cout<<"\n Enter roll no and name .. ";
cin>>rno>>name;
}
void displayname()
{
cout<<"\n Roll no :-"<<rno;
cout<<"\n Name :-"<<name<<endl;
}
};
class exam : public student          //derived class with single base class
 {
public:
          int  mark1,mark2,mark3,mark4,mark5,mark6,total;
  void acceptmark()
  {
    cout<<"\n Enter lang,eng,phy,che,csc,mat marks.. ";
    cin>>mark1>>mark2>>mark3>>mark4>>mark5>>mark6;
}
void displaymark()
{
cout<<"\n\t\t Marks Obtained ";
cout<<"\n Language.. "<<mark1;
cout<<"\n English .. "<<mark2;
cout<<"\n Physics .. "<<mark3;
cout<<"\n Chemistry.. "<<mark4;
cout<<"\n Comp.sci.. "<<mark5;
cout<<"\n Maths ..   "<<mark6;
cout<<"\n Total ..   "<<mark1+mark2+mark3+mark4+mark5+mark6;
}
 };
int main()
```



Single inheritance

11th_Comp Science_UM_Ch 16.indd 262 17-04-2023 13:29:39

```
int main()
{
  exam e1;
 e1.acceptname();      //calling base class function using derived class object
 e1.acceptmark();
 e1.displayname();      //calling base class function using derived class object
 e1.displaymark();
 return 0;
}
```

Output:

```
Enter roll no and name .. 1201 KANNAN
Enter lang,eng,phy,che,csc,mat marks.. 95 90 92 96 97 99
Roll no :-1201
Name :-KANNAN
                    Marks Obtained
Language.. 95
English .. 90
Physics .. 92
Chemistry.. 96
Comp.sci.. 97
Maths ..   99
Total ..   569
```

مذکورہ پروگرام میں ڈرائیوڈ کلاس "امتحان" بیس کلاس "طالب علم" کے تمام ممبران کو حاصل ہوتا ہے۔ لیکن اس تک رسائی کا استحقاق صرف بیس کلاس کے غیر نجی ممبروں (non private members) کو حاصل ہے۔

16.4.3 ملٹی لیول انہیریٹنس

مثال 16.2 ملٹی لیول انہیریٹنس

```
# include <iostream>
using namespace std;
class student            //base class
{
private :
char name[20];
int rno;
public:
void acceptname()
{ cout<<"\n Enter roll no and name .. "; cin>>rno>>name;
}
void displayname()
{ cout<<"\n Roll no :-"<<rno;
cout<<"\n Name :-"<<name<<endl;   }};
```


11th_Comp Science_UM_Ch 16.indd 263 17-04-2023 13:29:39

```
class exam : public student          //derived class with single base class
{
        public:
        int  mark1, mark2 ,mark3;
        void acceptmark()
        {       cout<<"\n Enter 3 subject marks.. ";
                cin>>mark1>>mark2>>mark3;     }
void displaymark(){
cout<<"\n\t\t Marks Obtained ";
cout<<"\n Subject1... "<<mark1;
cout<<"\n Subject2... "<<mark2;
cout<<"\n Subject3... "<<mark3;   }   };
class result : public exam
{
int total;
public:
void showresult()
{
total=mark1+mark2+mark3;
cout<<"\nTOTAL MARK SCORED : "<<total;
}
};
int main()
{
        result r1;
r1.acceptname();          //calling base class function using derived class object
r1.acceptmark();          //calling base class function which itself is a derived
                          // class  function using its derived class object
r1.displayname();         //calling base class function using derived class object
r1.displaymark();         /*calling base class function which itself is a derived
                          class  function using its derived class object*/
r1.showresult();          //calling the child class function
return 0;
}
```

Output:

```
Enter roll no and name .. 1201 Lohit Sarathi
 Enter 3 subject marks.. 98 100 100
 Roll no :-1201
 Name :-Lohith Sarathi
            Marks Obtained
Subject1 ...  98
Subject 2 ...  100
Subject 3... 100
TOTAL MARK SCORED    :  298
```


11th_Comp Science_UM_Ch 16.indd 264 17-04-2023 13:29:39

مندرجہ بالا پروگرام میں کلاس "نتیجہ" کلاس "امتحان" سے اخذ کیا گیا ہے جو خود کلاس کے طالب علم سے اخذ کیا گیا ہے۔

نوٹ

ملٹی لیول انہیریٹنس میں انہیریٹنس کی سطح کو رشتہ کے لحاظ سے کسی بھی سطح تک بڑھایا جا سکتا ہے۔ ملٹی لیول انہیریٹنس دادا، باپ اور بچے کے درمیان رشتہ کی طرح ہے۔

بغیر کسی اعلان کے ایک کلاس میں 1 بائٹ سائز ہو گا۔ class x{}; X occupies 1 byte

مذکورہ پروگرام میں ڈرائیوڈ کلاس "نتیجہ" نے کلاس "تفصیل" اور کلاس "امتحان" کی خصوصیات حاصل کی ہیں جو "طالب علم" سے اخذ کی گئی ہیں۔ تو یہ انہیریٹنس ملٹی لیول اور ملٹی انہیریٹنس کا مجموعہ ہے اس لیے اسے ہائبرڈ انہیریٹنس کہا جاتا ہے۔

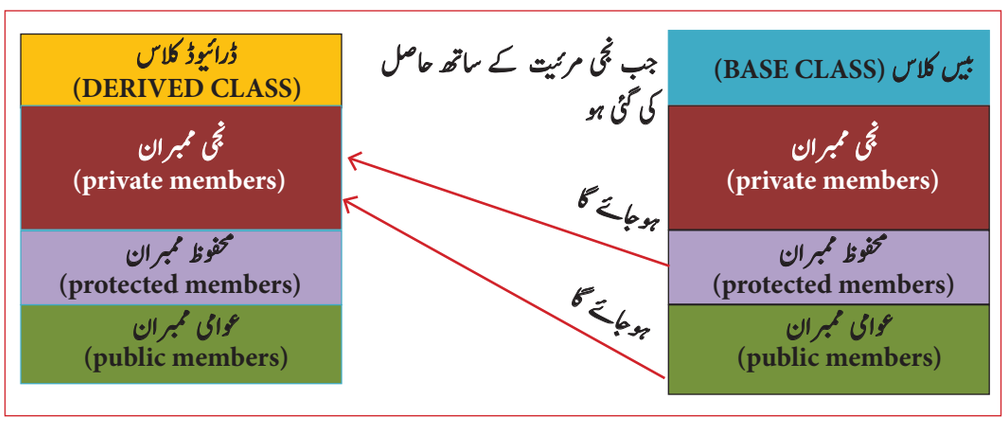
16.5 مرئیت کے طریقے (Visibility Modes)

انہیریٹنس کی ایک اہم خصوصیت یہ جاننا ہے کہ بیس کلاس کا کون سا ممبر ڈرائیوڈ کلاس کے ذریعہ حاصل کیا جائے گا۔ یہ مرئیت کے طریقوں کا استعمال کرکے کیا جاتا ہے۔

ڈرائیوڈ کلاس کے ذریعہ بیس کلاس کی رسائی کو مرئیت کے طریقوں سے کنٹرول کیا جاتا ہے۔ مرئیت کے تین طریقے نجی، محفوظ اور عوامی ہیں۔ ڈیفالٹ مرئیت کا طریقہ نجی ہے۔ اگرچہ مرئیت کے طریقے اور رسائی کی وضاحت کرنے والے (access specifiers) ایک جیسے نظر آتے ہیں، لیکن ان کے درمیان بنیادی فرق یہ ہے کہ رسائی کی وضاحت کرنے والے کلاس میں موجود ممبران کی رسائی کو کنٹرول کرتے ہیں جہاں مرئیت کے طریقے کلاس میں ملنے والے ممبران کی رسائی کو کنٹرول کرتے ہیں۔

16.5.1 نجی مرئیت کا طریقہ (Private visibility mode)

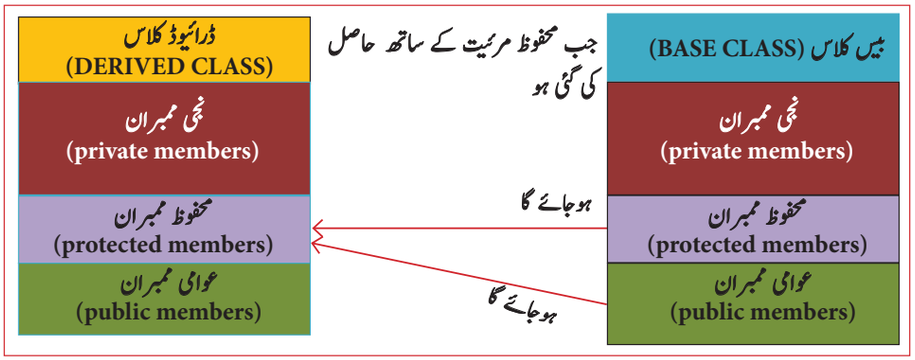
جب ایک بیس کلاس (base class) نجی مرئیت کے طریقہ کے ساتھ ملتی ہے تو بیس کلاس کے عوام اور محفوظ ممبران ڈرائیوڈ کلاس کے 'نجی' ممبر بن جاتے ہیں۔



11th_Comp Science_UM_Ch 16.indd 265 17-04-2023 13:29:39

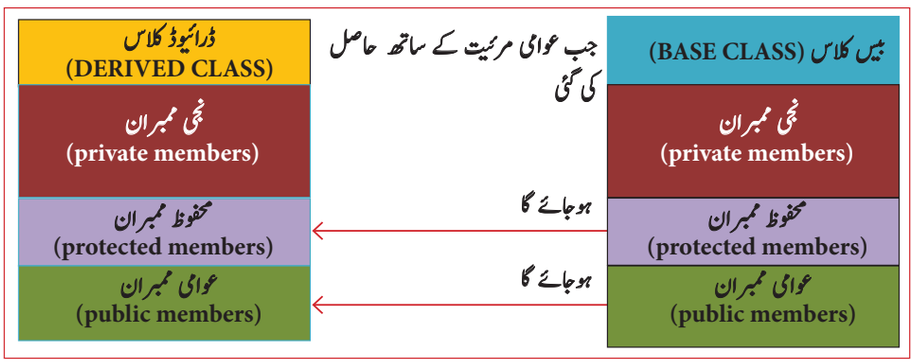
16.5.2 محفوظ مرئیت کا طریقہ (Protected Visibility Mode)

جب بیس کلاس کو محفوظ مرئیت کے طریقہ کے ساتھ ملتا ہے تو بیس کلاس کے محفوظ اور عوامی ممبر ڈرائیوڈ کلاس کے 'محفوظ' ممبر بن جاتے ہیں۔



16.5.3 عوامی مرئیت کا طریقہ (Public Visibility Mode)

جب ایک بیس کلاس عوامی مرئیت کے طریقہ کے ساتھ ملتی ہے تو، بیس کلاس کے محفوظ ممبران کو ڈرائیوڈ کلاس کے محفوظ ممبران کے طور پر ملے گا اور بیس کلاس کے عوامی ممبران کو ڈرائیوڈ کلاس کے عوامی ممبران کے طور پر ملے گا۔



ٹپ نوٹس

جب کلاسس عوامی، محفوظ یا پرائیویٹ کے ساتھ مل چکی ہیں تو بیس کلاس کے پرائیویٹ ممبران نہیں ملتے ہیں، وہ صرف نظر آتے ہیں یعنی ڈرائیوڈ کلاسس میں موجود رہتے ہیں، اور ان تک رسائی حاصل نہیں کی جا سکتی۔

مثال 16.3 مختلف مرئی طریقوں کی اہمیت کی وضاحت

```
//Implementation of Single Inheritance using public visibility mode
#include <iostream>
using namespace std;
class Shape
{    private:
          int count;
   protected:
          int width, height;
   public:
          void setWidth(int w)
          {          width = w;     }
void setHeight(int h)
{ height = h; } };
```


11th_Comp Science_UM_Ch 16.indd 266 17-04-2023 13:29:39

```
class Rectangle: public Shape
{
public:
int getArea()
{
return (width * height);
}
};
int main()
{
Rectangle Rect;
Rect.setWidth(5);
Rect.setHeight(7);
// Print the area of theobject.
cout<< "Total area: "<<Rect.getArea() <<endl;
return 0;
}
```

Output
Total area: 35

مندرجہ ذیل ٹیبل میں انہیریٹنس سے پہلے ہر کلاس کے اندر وضاحت کردہ اراکین شامل ہیں۔

کلاس کے ممبران (MEMBERS of class)	مرئیت کے طریقے (visibility modes)		
	نجی	محفوظ	عوامی
شکل (بیس کلاس)	int count;	int width; int height;	void setWidth(int) void setHeight(int)
مستطیل (صرف اس کے متعین ارکان کے ساتھ ڈرائیوڈ کلاس)			int getArea();

درج ذیل ٹیبل میں انہیریٹنس کے بعد وضاحت کردہ ارکان کی تفصیلات موجود ہیں۔

کلاس کے ممبران (MEMBERS of class)	بیس کلاس کی خصوصیات کے حصول کے لیے عوامی - مرئیت کے طریقے		
	نجی	محفوظ	عوامی
شکل (بیس کلاس)	int count;	int width; int height;	void setWidth(int) void setHeight(int)
مستطیل (ڈرائیوڈ کلاس نے عوامی مرئیت کے ساتھ بیس کلاس کی خصوصیات حاصل کی)	بیس کلاس کے پرائیویٹ ممبران حاصل شدہ کلاس کے ذریعے براہ راست قابل رسائی نہیں ہیں۔	int width; int height;	int getArea(); void setWidth(int) void setHeight(int)

فرض کریں کلاس مستطیل محفوظ مرئیت کے ساتھ اخذ کیا گیا ہے تو کلاس مستطیل کی خصوصیات میں مندرجہ ذیل تبدیل ہوں گی۔

11th_Comp Science_UM_Ch 16.indd 267 17-04-2023 13:29:40

کلاس کے ممبران (MEMBERS of class)	بیس کلاس کی خصوصیات حاصل کرنے کے لیے محفوظ- مرئیت کے طریقے		
	نجی	محفوظ	عوامی
شکل (بیس کلاس)	int count;	int width; int height;	void setWidth(int) void setHeight(int)
مستطیل (ڈرائیوڈ کلاس نے محفوظ مرئیت کے ساتھ بیس کلاس کی خصوصیات حاصل کی)	بیس کلاسس کے پرائیویٹ ممبران حاصل شدہ کلاس کے ذریعے براہ راست قابل رسائی نہیں ہیں۔	int width; int height; void setWidth(int) void setHeight(int)	int getArea();

اگر کلاس مستطیل کو اس کی بیس کلاس شکل سے پرائیویٹ مرئیت کے طریقہ کے ساتھ اخذ کیا گیا ہے تو کلاس مستطیل کی خاصیت حسب ذیل بدل جائے گی۔

کلاس کے ممبران (MEMBERS of class)	بنیادی کلاس کی خصوصیات کے حصول کے لیے نجی- مرئیت کے طریقے		
	نجی	محفوظ	عوامی
شکل (بیس کلاس)	int count;		void setWidth(int) void setHeight(int)
مستطیل (ڈرائیوڈ کلاس نے نجی مرئیت کے ساتھ بیس کلاس کی خصوصیات حاصل کی)	int width; int height; void setWidth(int) void setHeight(int)		int getArea();

جب آپ کسی موجودہ بیس کلاس سے کلاس اخذ کرتے ہیں، تو یہ بیس کلاس کی خصوصیات کو اس کے مرئیت کے طریقہ کی بنیاد پر حاصل کر سکتا ہے۔ لہٰذا ضرورت کے مطابق مناسب مرئیت کا طریقہ دینا چاہیے۔

پرائیویٹ انہیریٹنس کا استعمال اس وقت کرنا چاہیے جب آپ چاہتے ہیں کہ بیس کلاس کی خصوصیات ڈرائیوڈ کلاس کے لیے دستیاب ہوں لیکن ان کلاسس کے لیے نہیں جو ڈرائیوڈ کلاس سے حاصل کی گئی ہیں۔

محفوظ انہیریٹنس کا استعمال اس وقت کیا جانا چاہئے جب بیس کلاس کی خصوصیات صرف ڈرائیوڈ کلاس کے ممبران کے لئے دستیاب ہوں لیکن بیرونی دنیا کے لئے نہیں۔

عوامی انہیریٹنس کا استعمال اس وقت کیا جا سکتا ہے جب بیس کلاس کی خصوصیات ڈرائیوڈ کلاس کے ممبران اور بیرونی دنیا کے لیے بھی دستیاب ہوں۔

16.6 انہیریٹنس اور کنسٹرکٹرس اور ڈسٹرکٹرس
(Inheritance and constructors and destructors)

جب ڈرائیوڈکلاس کا کوئی آبجیکٹ بنایا جاتا ہے تو کمپائلر پہلے بیس کلاس کنسٹرکٹر اور پھر ڈرائیوڈکلاس کے کنسٹرکٹر کو طلب کرتا ہے۔ اس کی وجہ یہ ہے کہ ڈرائیوڈ کلاس بیس کلاس کے ممبران پر مبنی ہے۔ جب ڈرائیوڈ کلاس کا آبجیکٹ ختم ہوجاتا ہے تو پہلے ڈرائیوڈ کلاس ڈسٹرکٹر کو طلب کرتا ہے اس کے بعد بیس کلاس ڈسٹرکٹر طلب کرتا ہے۔

11th_Comp Science_UM_Ch 16.indd 268 17-04-2023 13:29:40

مثال 16.4 کنسٹرکٹرس اور ڈسٹرکٹرس کی ترتیب

```
#include<iostream>
using namespace std;
class base
{
public:
base()
{ cout<<"\nConstructor of base class..."; }
~base()
{ cout<<"\nDestructor of base class.... "; }
};
class derived:public base
{
public :
derived()
{ cout << "\nConstructor of derived ..."; }
~derived()
{ cout << "\nDestructor of derived ..."; }};
class derived1 :public derived
{
public :
derived1()
{ cout << "\nConstructor of derived1 ...";}
~derived1()
{ cout << "\nDestructor of derived1 ...";}
};
int main()
{
derived1 x;
return 0;
}
```

Output:

```
Constructor of base class...
Constructor of derived ...
Constructor of derived1 ...
Destructor of derived1 ...
Destructor of derived ...
Destructor of base class....
```

نوٹس

- کنسٹرکٹرس کو حاصل شدہ کلاس کی ترتیب میں عمل میں لایا جاتا ہے یعنی بیس کنسٹرکٹر سے اخذ کیا جاتا ہے۔ ڈسٹرکٹرس کو برعکس ترتیب میں عمل میں لایا جاتا ہے۔
- ڈرائیوڈ کلاس آبجیکٹ کا سائز = تمام بیس کلاس ڈیٹا ممبرز کا سائز + ڈرائیوڈ کلاس میں تمام ڈیٹا ممبرز کا سائز۔

16.7 ڈرائیوڈ کلاس میں اوور رائیڈنگ / شیڈونگ بیس کلاس فنکشنس

(Overriding / Shadowing Base class functions in derived class)

انہیریٹنس کے معاملے میں ایسے حالات ہوتے ہیں جہاں بیس کلاس کے ممبر فنکشن اور ڈرائیوڈ کلاسس کا نام ایک ہوتا ہے۔ اگر ڈرائیوڈ کلاس آبجیکٹ اوور لوڈیڈ ممبر فنکشن کو طلب کرتا ہے تو یہ کمپائلر کے لیے الجھن کا باعث بنتا ہے کہ کس فنکشن کو طلب کرنا ہے۔ ڈرائیوڈ کلاس ممبر

11th_Comp Science_UM_Ch 16.indd 269 17-04-2023 13:29:40

فنکشن کو بیس کلاس ممبر فنکشن سے زیادہ ترجیح حاصل ہے۔ یہ بیس کلاس کے ممبر فنکشن کو شیڈو کرتا ہے جس کا نام وہی ہے جو ڈرائیوڈ کلاس کے ممبر فنکشن کو ہے۔ اسکوپ ریزولوشن (::) آپریٹر (scope resolution operator) اس مسئلے کو حل کرتا ہے۔

مثال 16.5 ڈرائیوڈ کلاس میں اسکوپ ریزولوشن آپریٹر کا استعمال

```
#include<iostream>
#include<string>
using namespace std;
class Employee
{
   private:
      char name[50];
      int code;
   public:
      void getdata();
      void display();
};
class staff: public Employee
{
   private:
      int ex;
   public:
         void getdata();
      void display();
};
void Employee::display()
{
         cout<<"Name:"<<name<<endl;
         cout<<"Code:"<<code<<endl;
}
void staff::display()
{        Employee :: display();//overriding
         cout<<"Experience:"<<ex<<" Years"<<endl;
}
void Employee::getdata()
{
cout<<"Name:";
cin>>name;
cout<<"\ncode:";
cin>>code;
}
void staff::getdata()
{
Employee::getdata();
cout<<"\nExperience:";
cin>>ex;
int main()
{
         staff s;
         cout<<"Enter data"<<endl;
         s.getdata();
         cout<<endl<<endl<<"\tDisplay Data"<<endl;
         s.display();
         return 0;
}
```

Output

```
Enter data
Name: SUGANYA
Code: 1201
Experience: 30
        Display Data
Name: SUGANYA
Code:1201
Experience:30 Years
```


11th_Comp Science_UM_Ch 16.indd 270 17-04-2023 13:29:40

مذکورہ پروگرام میں getdata() اور display() کی وضاحت بیس اور ڈرائیوڈ کلاس دونوں میں کی گئی ہے۔ لہذا جب ڈرائیوڈ کلاس کا عملہ، ایمپلائی کلاس کی خصوصیات کا وارث ہوتا ہے تو اس کے پاس دو getdata() اور display() ہر ایک ہوگا۔ اخذ کردہ getdata() اور display() کو متعین کردہ getdata() اور display() :: (اسکوپ ریزولوشن) آپریٹر کو بیس کلاس کے نام کے ساتھ بیس کلاس کے ارکان کو دیا جاتا ہے۔

نوٹ

جب ایک ڈرائیوڈ کلاس ممبر فنکشن کا نام اس کے بیس کلاس ممبر فنکشن جیسا ہی ہوتا ہے، توڈرائیوڈ کلاس ممبر فنکشن بیس کلاس میں ملنے والے فنکشن کو سایا کرتا ہے/چھپا دیتا ہے۔ اس صورتحال کو فنکشن اوور رائیڈ نگ (function overriding) کہا جاتا ہے اور اسے بیس کلاس کا نام دے کر حل کیا جاسکتا ہے۔ اس کے بعد :: اور ممبر فنکشن کا نام۔

یاد رکھنے کے نکات

- موجودہ کلاس سے نئی کلاس اخذ کرنے کے طریقہ کار کو انہیریٹنس کہتے ہیں۔
- انہیریٹنس کا سب سے بڑا فائدہ یہ ہے کہ یہ کوڈ کے دوبارہ استعمال میں مدد کرتا ہے۔
- ڈرائیوڈکلاس کو بیس کلاس کی تمام خصوصیات حاصل ہوتی ہیں۔ یہ ایک پاور پیک کلاس ہے، کیونکہ یہ اضافی صفات اور طریقے شامل کر سکتا ہے اور اس طرح اس کی فعالیت کو بڑھا سکتا ہے۔
- انہیریٹنس کی مختلف قسمیں ہیں مثلاً سنگل انہیریٹنس (single inheritance)، ملٹی پل انہیریٹنس(multiple inheritance)، ملٹی لیول انہیریٹنس(multilevel inheritance)، ہائبرڈ انہیریٹنس (hybrid inheritance) اور ہائیر آرکیکل انہیریٹنس (hierarchical inheritance)۔
- جب ایک ڈرائیوڈ کلاس صرف ایک بیس کلاس سے ملتی ہے، تو اسے سنگل انہیریٹنس کے نام سے جانا جاتا ہے۔
- جب ایک ڈرائیوڈ کلاس ملٹی پل بیس کلاسس سے ملتی ہے تو اسے ملٹی پل انہیریٹنس کے نام سے جانا جاتا ہے۔
- انہیریٹنس کی عبوری نوعیت انہیریٹنس کی اِس شکل سے ظاہر ہوتی ہے۔ جب ایک کلاس کسی کلاس سے اخذ کی جاتی ہے جو ایک ڈرائیوڈکلاس ہے - تو اسے ملٹی لیول انہیریٹنس کہا جاتا ہے۔
- جب ایک ہی بیس کلاس سے ایک سے زیادہ ڈرائیوڈ کلاسس بنائے جاتے ہیں، تو اسے ہائیر آرکیکل انہیریٹنس کہا جاتا ہے۔
- جب انہیریٹنس کی ایک سے زیادہ اقسام کا مجموعہ ہو تو اسے ہائبرڈ انہیریٹنس کہا جاتا ہے۔
- ایک سے زیادہ انہیریٹنس میں، بیس کلاسس اس ترتیب سے بنائے جاتے ہیں جس میں وہ ڈرائیوڈ کلاس کے اعلان میں ظاہر ہوتے ہیں۔
- ایک سب کلاس خود کو عوامی، نجی یا محفوظ طریقے سے اخذ کر سکتا ہے۔
- کلاس کے پرائیویٹ ممبر کو حاصل نہیں کیا جاسکتا۔
- عوامی طور پر ڈرائیوڈ کلاس میں، بیس کلاس کے عوامی اراکین عوامی رہتے ہیں اور بیس کلاس کے محفوظ اراکین ڈرائیوڈ کلاس میں محفوظ رہتے ہیں۔
- پرائیویٹ طور پر ڈرائیوڈ کلاس میں، عوام اور بیس کلاس کے محفوظ ممبران ڈرائیوڈ کلاس میں پرائیویٹ ہو جاتے ہیں۔
- جب کلاس کو محفوظ طریقہ پر اخذ کیا جاتا ہے، تو بیس کلاس کے پبلک اور محفوظ ممبران ڈرائیوڈ کلاس میں محفوظ ہو جاتے ہیں۔
- بیس کلاس کے کنسٹرکٹرس اور ڈسٹرکٹرس نہیں ملتے ہیں لیکن ڈرائیوڈ کلاس کے لیے کسی چیز کی تخلیق کے دوران بیس کلاس کے کنسٹرکٹرس کو خود بخود طلب کیا جائے گا۔
- ڈسٹرکٹرس کو برعکس ترتیب میں طلب کیا جاتاہے۔ ڈرائیوڈ کلاسس کے ڈسٹرکٹرس کو پہلے اور بعد میں بیس کلاس کو طلب کیاجاتا ہے۔
- ڈرائیوڈ کلاس آبجیکٹ کا سائز = تمام بیس کلاس ڈیٹا ممبرز کا سائز + ڈرائیوڈ کلاس میں تمام ڈیٹا ممبرز کا سائز۔
- ممبران کی اوور رائیڈنگ کو اسکوپ ریزولوشن آپریٹر (::) کے استعمال سے حل کیا جاتا ہے۔
- یہ پوائنٹر موجودہ آبجیکٹس کے اراکین کا حوالہ دینے کے لیے استعمال کیا جاتا ہے۔

11th_Comp Science_UM_Ch 16.indd 271 17-04-2023 13:29:40

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. درج ذیل میں سے کون سی موجودہ کلاس سے نئی کلاس بنانے کا عمل ہے۔
(A) پولیمورفزم (B) انہیریٹنس
(C) انکیپسولیشن (D) سوپر کلاس

2. مندرجہ ذیل میں سے کس کلاس کا طالب علم بیس کلاس اسکول سے اخذ کرتا ہے۔
(A) اسکول: طالب علم
(B) کلاس کا طالب علم: سرکاری اسکول
(C) طالب علم: سرکاری اسکول
(D) کلاس اسکول: سرکاری طالب علم

3. انہیریٹنس کی وہ قسم جو عبوری فطرت کی عکاسی کرتی ہے
(A) سنگل انہیریٹنس
(B) ملٹی پل انہیریٹنس
(C) ملٹی لیول انہیریٹنس
(D) ہائبرڈ انہیریٹنس

4. جب آپ چاہتے ہیں کہ بیس کلاس کی خصوصیات ڈرائیوڈ کلاس کے لیے دستیاب ہوں، لیکن ان کلاسس کے لیے نہیں جو ڈرائیوڈکلاس سے حاصل کی گئی ہیں تو کون سا مرئیت کا طریقہ (visibility mode) استعمال کرنا چاہیے؟
(A) پرائیویٹ (B) پبلک
(C) پروٹیکٹیڈ (D) یہ تمام

5. انہیریٹنس نئی کلاس بنانے کا عمل ہے، وہ کس سے؟
(A) بیس کلاس (B) ابسٹریکٹ (abstract)
(C) ڈرائیوڈ کلاس (D) فنکشن

6. ایک کلاس ایک کلاس سے ماخوذ ہے جو خود ایک ڈرائیوڈ کلاس ہے، پھر اسے کہا جاتا ہے
(A) ملٹی پل انہیریٹنس
(B) ملٹی لیول انہیریٹنس
(C) سنگل انہیریٹنس
(D) ڈبل انہیریٹنس

7. مندرجہ ذیل میں سے کس کو انہیریٹنس کی ترتیب میں عمل میں لایا جاتا ہے؟
(A) ڈسٹرکٹر (B) ممبر فنکشن
(C) کنسٹرکٹر (D) آبجیکٹ

8. انہیریٹنس کے حوالے سے درج ذیل میں سے کون سا درست ہے؟
(A) بیس کلاس کے پرائیویٹ ممبران پرائیویٹ کے ساتھ ڈرائیوڈ کلاس سے ملے ہوئے ہیں۔
(B) بیس کلاس کے پرائیویٹ ممبران پرائیویٹ رسائی کے ساتھ ڈرائیوڈ کلاس سے ملے ہوئے نہیں ہے۔
(C) بیس کلاس کے عوامی ممبران ملے ہیں لیکن ڈرائیوڈ کلاس کو نظر نہیں آتے۔
(D) بیس کلاس کے محفوظ ممبران ملے ہیں لیکن بیرونی کلاس کو نظر نہیں آتے۔

9. درج ذیل کلاس ڈیکلریشن کی بنیاد پر سوالات کا جواب دیں (9.1 سے 9.4 تک)

```
class vehicle
{ int wheels;
public:
void input_data(float,float);
void output_data();
protected:
int passenger;
 };
class heavy_vehicle : protected vehicle {
int diesel_petrol;
protected:
        int load;
public:
void read_data(float,float)
void write_data(); };
class bus: private heavy_vehicle {
char Ticket[20];
public:
void fetch_data(char);
void display_data(); };
```


11th_Comp Science_UM_Ch 16.indd 272 17-04-2023 13:29:40

9.1 ہیوی_وہیکل کلاس کی بیس کلاس کون سی ہے؟

(a) بس (c) ہیوی_وہیکل

(b) وہیکل (d) دونوں (a) اور (c)

9.2 ڈیٹا ممبر جو فنکشن ڈسپلے ڈیٹا () سے حاصل کیا جا سکتا ہے

(a) پیسنجر (passenger) (b) لوڈ (Load)

(c) ٹکٹ (d) یہ تمام

9.3 ممبر فنکشن جو بس کلاس کے آبجیکٹس کے ذریعے رسائی حاصل کی جاسکتی ہے وہ

(a) input_data(), output_data()

(b) read_data() ,write_data()

(c) fetch_data(), display_data()

(d) یہ تمام

9.4 ممبر فنکشن جو کلاس بس کے ذریعہ عوامی طور پر حاصل کیا جاتا ہے

(a) input_data(), output_data()

(b) read_data(), write_data()

(c) fetch_data(), display_data()

(d) ان میں سے کوئی نہیں

سیکشن - B

بہت ہی مختصر جوابات دیں

1. انہیریٹنس کیا ہے؟
2. بیس کلاس کیا ہے؟
3. ڈرائیوڈ کلاس کو پاور پیکڈ کلاس کیوں کہا جاتا ہے؟
4. کس چیز میں ملٹی لیول اور ملٹی پل انہیریٹنس میں فرق ہے حالانکہ دونوں میں کئی بیس کلاس ہوتے ہیں؟
5. عوامی اور نجی مرئیت کے طریقے میں کیا فرق ہے؟

سیکشن - C

مختصر جوابات دیں

1. نئی کلاس اخذ کرتے وقت کن نکات کو نوٹ کرنا ضروری ہے؟
2. نجی مرئیت کے طریقے میں موجود ممبران اور عوامی مرئیت کا طریقہ میں موجود ممبران میں کیا فرق ہے؟
3. پولیمورفزم اور انہیریٹنس میں کیا فرق ہے حالانکہ کوڈ کے دوبارہ استعمال کے لیے استعمال کیا جاتا ہے؟
4. اوور رائیڈنگ (overriding) سے آپ کا کیا مطلب ہے؟
5. انہیریٹنس میں کنسٹرکٹرس اور ڈسٹرکٹرس کے عمل کے بارے میں کچھ حقائق لکھیں

سیکشن - D

تفصیل سے بتائیں

1. انہیریٹنس کی مختلف اقسام کی وضاحت کریں۔
2. مختلف مرئیت کے طریقے کی وضاحت تصویری نمائندگی کے ذریعے کریں
3. **درج ذیل C++ کوڈ پر غور کریں اور سوالات کے جواب دیں۔**

```
class Personal
{
int Class,Rno;
char Section;
protected:
char Name[20];
public:
personal();
void pentry();
void Pdisplay(); };
class Marks:private Personal
{ float M[5];
protected:
char Grade[5];
public:
Marks();
void Mentry();
void Mdisplay(); };
class Result:public Marks
{
float Total,Agg;
public:
char FinalGrade, Commence[20];
Result();
void Rcalculate();
void Rdisplay();
};
```

3.1 پروگرام میں کس قسم کی انہیریٹنس کو دکھایا گیا ہے؟

3.2 بیس کلاس کے مرئیت کے طریقہ کی وضاحت کریں۔

11th_Comp Science_UM_Ch 16.indd 273 17-04-2023 13:29:40

3.3 جب کلاس رزلٹ کا آبجیکٹ بنایا جاتا ہے تو کنسٹرکٹر/ڈسٹرکٹر کے درخواست کی ترتیب دیں۔

3.4 بیس کلاس (es/) اور ڈرائیوڈ کلاس (es/) کو نام دیں۔

3.5 درج ذیل کلاس کے آبجیکٹ کے زریعہ مشغول ہونے والے بائٹس کی تعداد بتائیں:

(a) پرسنل (b) مارکس

(c) رزلٹ

3.6 کلاس رزلٹ کے آبجیکٹ سے قابل رسائی ڈیٹا ممبرس کے نام لکھیں۔

3.7 کلاس رزلٹ کے آبجیکٹ سے قابل رسائی تمام ممبر فنکشنس کے نام لکھیں۔

3.8 کلاس کے نتائج کے ممبر فنکشنس سے قابل رسائی تمام ممبران کے نام لکھیں۔

4. درج ذیل پروگرام کا آؤٹ پٹ لکھیں۔

```
#include<iostream>
using namespace std;
class A
{       protected:
      int x;
      public:
      void show()
      {cout<<"x = "<<x<<endl;}
      A()
      { cout<<endl<<" I am class A "<<endl;}
      ~A()
      { cout<<endl<<" Bye ";} };
class B : public A
{protected:
  int y;
  public:
  B(int x1, int y1)
  { x = x1;
    y = y1;       }
B()
{   cout<<endl<<" I am class B "<<endl; }
~B()
{   cout<<endl<<" Bye "; }
void show()
{ cout<<"x = "<<x<<endl;
  cout<<"y = "<<y<<endl;    } };
int main()
{A objA;
B objB(30, 20);
objB.show();
return 0;  }
```

5. درج ذیل پروگرام کو ڈیبگ کریں

```
%include(iostream.h)
#include<conio.h>
class A()
{ public;
int a1,a2:a3;
void getdata[]
{       a1=15;  a2=13; a3=13; }         }
class B:: public A()
{       PUBLIC
        voidfunc()
        { int b1:b2:b3;
          A::getdata[];
          b1=a1;
          b2=a2;
          a3=a3;
          cout<<b1<<'\t'<<b2<<'t\'<<b3; }
void main()
{       B der;
        der1:func(); }
```

کیس اسٹڈی

کلاس اسٹاک (Class Stock) کے لیے کلاس لکھیں۔

ہر اسٹاک میں ایک ڈیٹا ممبر ہوتا ہے جو خالص قیمت رکھتا ہے، اور ایک کنسٹرکٹر جو اس قیمت کا تعین کرتا ہے۔ ہر اسٹاک کا ایک طریقہ ()get_price، ہوتا ہے، جو مجموعی قیمت کا حساب لگاتا ہے اور اسے واپس کرتا ہے (مجموعی قیمت میں VAT کو 21٪ میں شامل ہے)

حوالہ جات:

Object Oriented Programming with C++ (4th Edition), Dr. E. Balagurusamy, Mc.Graw Hills.

11th_Comp Science_UM_Ch 16.indd 274 17-04-2023 13:29:40

باب 17

کمپیوٹر اخلاقیات اور سائبر سیکیورٹی
(Computer Ethics And Cyber Security)

سیکھنے کے مقاصد

اس باب کو سیکھنے کے بعد طلباء کے قابل ہوجائیں گے

- سائبر کرائمز کے بارے میں جاننے
- سائبر دنیا میں رہنما اصولوں اور اخلاقیات کی ضرورت کو سمجھنے
- سائبر کرائمز سے متعلق مسائل کو سمجھنے
- فائر والز (firewalls) اور پراکسی سرورز (proxy servers) کی فعالیت کو جاننے
- انکرپشن (encryption) اور ڈکرپشن (decryption) کے بارے میں جاننے
- آئی ٹی ایکٹ (IT Act) پر علم حاصل کرنے

17.1 تعارف

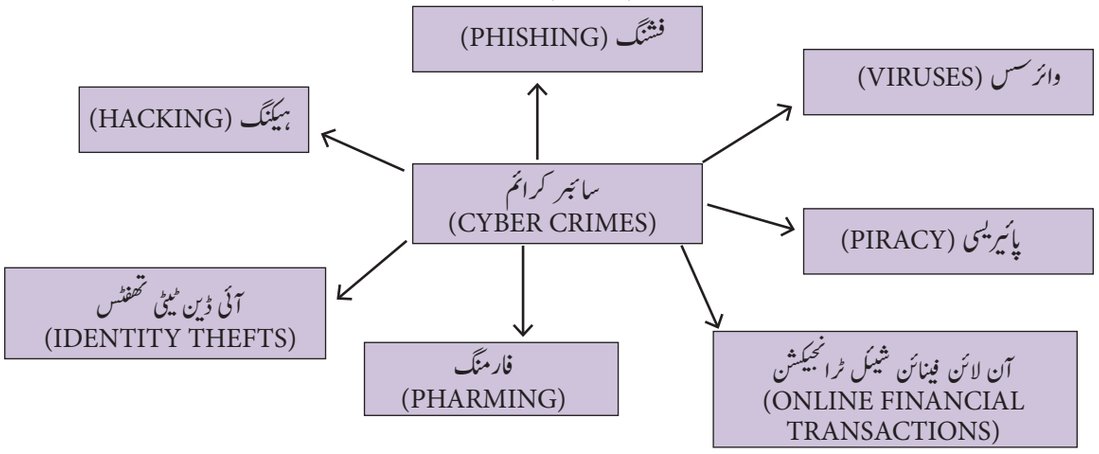
انٹرنیٹ ایک مواصلاتی میڈیا ہے جو آسانی سے قابل رسائی اور سب کے لیے کھلا ہے۔ کمپیوٹر، موبائل فون اور انٹرنیٹ کے ذریعے انفارمیشن ٹیکنالوجی وسیع ہے۔ انفارمیشن ٹیکنالوجی کے غلط استعمال کی بہت گنجائش اور امکانات ہیں۔

عام طور پر کمپیوٹر سسٹم کمزور ہوتے ہیں۔ اس بات کو یقینی بنانے کے لیے کہ قیمتی ڈیٹا غلط ہاتھوں میں نہ جائے اس کا خصوصی خیال رکھنا چاہیے۔ لہذا، ڈیٹا کو محفوظ کرنے کی ضرورت ہے۔

سائبر کرائم ایک ایسا جرم ہے جس میں کمپیوٹر اور نیٹ ورک شامل ہوتا ہے۔ یہ معاشرے کے لیے بڑھتا ہوا خطرہ بنتا جا رہا ہے اور یہ جرائم پیشہ افراد یا ان افراد کے غیر ذمہ دارانہ عمل سے ہوتا ہے جو انٹرنیٹ کے وسیع استعمال کا استحصال کر رہے ہیں۔ یہ انفارمیشن ٹیکنالوجیز کے اخلاقی استعمال کے لیے ایک بڑا چیلنج پیش کرتا ہے۔سائبر کرائم زیادہ تر کاروباری نظاموں کی سالمیت، حفاظت اور بقا کے لیے بھی خطرہ ہیں۔

11th_Comp Science_UM_Ch 17.indd 275 17-04-2023 14:43:21

تصویر 17.1 دنیا بھر میں ہونے والے سائبر جرائم کی اقسام کو پیش کرتا ہے۔



تصویر 17.1 سائبر جرائم کی اقسام

اخلاقیات (ETHICS)

اخلاقیات کا مطلب ہے "کیا غلط اور کیا صحیح"۔ یہ اخلاقی اصولوں کا ایک مجموعہ ہے جو کمپیوٹر استعمال کرنے والے افراد کے رویے پر حکمرانی کرتے ہیں۔ ایک فرد صحیح طرز عمل کی پیروی کرنے کے لیے علم حاصل کرتا ہے، اخلاقیات کا استعمال کرتے ہوئے جسے اخلاقیات بھی کہا جاتا ہے۔ اخلاق معاشرے میں صحیح اور غلط کے عام طور پر قبول شدہ معیارات کا حوالہ دیتے ہیں۔ اسی طرح سائبر دنیا میں بھی کچھ معیارات ہیں جیسے

- پائریٹڈ سافٹ ویئر (Pirated Software) استعمال نہ کریں
- غیر مجاز صارف اکاؤنٹس استعمال نہ کریں
- دوسروں کے پاس ورڈ چوری نہ کریں
- ہیک نہ کریں

کمپیوٹر اخلاقیات میں بنیادی مسائل انٹرنیٹ کے استعمال سے پیدا ہونے والے منظرناموں پر مبنی ہیں جیسے کہ رازداری، کاپی رائٹ والے مواد کی اشاعت، ڈیجیٹل مواد کی غیر مجاز تقسیم اور ویب سائٹس، سافٹ ویئر اور متعلقہ خدمات کے ساتھ صارف کا تعامل۔

کمپیوٹر اخلاقیات

انٹرنیٹ کی مدد سے دنیا اب ایک گلوبل ولیج (Global village) بن چکی ہے۔ انٹرنیٹ افراد کے ساتھ ساتھ مختلف تنظیموں اور کاروباروں کے لیے بھی فائدہ مند ثابت ہوا ہے۔ ای کامرس (e-Commerce) کاروباروں میں بہت مقبول ہوتا جا رہا ہے کیونکہ یہ انہیں کسی بھی دوسرے ذرائع سے زیادہ تیزی سے صارفین کی وسیع رینج تک پہنچنے میں مدد کرتا ہے۔

کمپیوٹر اخلاقیات ان طریقہ کار، اقدار اور طریقوں سے متعلق ہے جو کسی بھی فرد، تنظیم یا ادارے کے اخلاقی اقدار اور عقائد کو نقصان پہنچانے یا ان کی خلاف ورزی کیے بغیر کمپیوٹر ٹیکنالوجی اور اس سے متعلقہ مضامین کو استعمال کرنے کے عمل کو کنٹرول کرتے ہیں۔

اخلاقیات کے رہنما اصول

عام طور پر، کمپیوٹر استعمال کرنے والوں کو درج ذیل ہدایات پر عمل کرنا چاہیے:

1. **ایمانداری:** صارفین کو انٹرنیٹ استعمال کرتے وقت سچا ہونا چاہیے۔
2. **رازداری:** صارفین کو کسی بھی اہم معلومات کو غیر مجاز لوگوں کے ساتھ شیئر نہیں کرنا چاہیے۔
3. **احترام:** ہر صارف کو دوسرے صارفین کی رازداری کا احترام کرنا چاہیے۔
4. **پیشہ ورانہ مہارت:** ہر صارف کو پیشہ ورانہ طرز عمل کو برقرار رکھنا چاہیے۔

11th_Comp Science_UM_Ch 17.indd 276 17-04-2023 14:43:21

5. **قانون کی پابندی :** صارفین کو کمپیوٹر کے استعمال میں سائبر قانون کی سختی سے پابندی کرنی چاہیے۔

6. **ذمہ داری:** ہر صارف کو اپنے اعمال کی ملکیت اور ذمہ داری لینا چاہیے۔

کیا آپ جانتے ہیں؟

اخلاقیات اخلاقی اصولوں کا ایک مجموعہ ہے جو معاشرے میں فرد کے طرز عمل کو کنٹرول کرتے ہیں، اور کمپیوٹر اخلاقیات اخلاقی اصول جو صارفین کے کمپیوٹر کے استعمال کو منظم کرتے ہیں۔

17.2 اخلاقی مسائل

اخلاقی مسئلہ ایک ایسا مسئلہ یا معاملہ ہے جس کے لیے کسی شخص یا تنظیم کو متبادل کے درمیان انتخاب کرنے کی ضرورت ہوتی ہے، جس کا صحیح (اخلاقی) یا غلط (غیر اخلاقی) کے طور پر جائزہ لیا جانا چاہیے۔ معاشرے میں مثبت اثر ڈالنے کے لیے ان مسائل کو حل کیا جانا چاہیے۔

کچھ عمومی اخلاقی مسائل ذیل میں درج ہیں:

- سائبر کرائم (cyber crime)
- سافٹ ویئر پائریسی (Software piracy)
- ہیکنگ (Hacking)
- دھوکہ دہی کے لیے کمپیوٹر کا استعمال
- وائرس کی شکل میں تخریب کاری
- کمپیوٹر کا استعمال کرتے ہوئے جھوٹے دعوے کرنا

سائبر کرائم (Cyber Crime)

سائبر کرائم ایک فکری، وائٹ کالر جرم ہے۔ جو لوگ اس طرح کے جرائم کا ارتکاب کرتے ہیں وہ عام طور پر کمپیوٹر سسٹم کو ذہین طریقے سے استعمال کرتے ہیں۔

مثال کے طور پر - انٹرنیٹ کے ذریعے غیر قانونی رقم کی منتقلی۔

کچھ کمپیوٹر جرائم کی مثالیں اور ان کے افعال ذیل میں ٹیبل 17.1 میں درج ہیں:

11th_Comp Science_UM_Ch 17.indd 277 17-04-2023 14:43:21

ٹیبل 17.1 کمپیوٹر کرائم

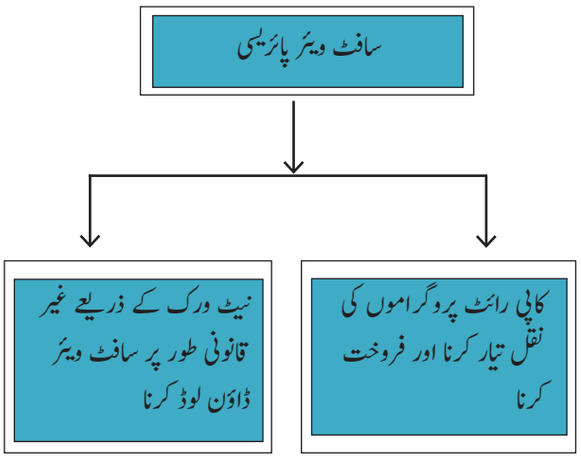
جرم	فنکشن
سائبر دہشت گردی	ہیکنگ، دھمکیاں، اور کسی کاروبار یا شخص کو بلیک میل کرنا۔
سائبر سٹاکنگ (Cyber Stalking)	آن لائن کے ذریعے ہراساں کرنا۔
مالویئر (Malware)	بدنیتی پر مبنی پروگرام جو متعدد افعال انجام دے سکتے ہیں جن میں حساس ڈیٹا کو چوری کرنا، انکرپٹ کرنا یا حذف کرنا، بنیادی کمپیوٹنگ فنکشنز کو تبدیل کرنا یا ہائی جیک کرنا اور صارف کی کمپیوٹر کی سرگرمیوں کی ان کی اجازت کے بغیر نگرانی کرنا شامل ہیں۔
سروس اٹیک کا انکار	جعلی درخواستوں کے ساتھ سسٹم کو اوور لوڈ کرنا تاکہ یہ عام جائز درخواستوں کو پورا نہ کر سکے
دھوکہ	ڈیٹا میں ہیرا پھیری، مثال کے طور پر کسی غیر مجاز اکاؤنٹ میں رقم کی منتقلی کے لیے بینکنگ ریکارڈز کو تبدیل کرنا۔
ہارویسٹنگ (Harvesting)	ایک شخص یا پروگرام غیر قانونی طور پر دوسروں کے اکاؤنٹ (اکاؤنٹس) تک رسائی حاصل کرنے کے لیے جائز صارف سے لاگ ان اور پاس ورڈ کی معلومات اکٹھا کرتا ہے۔
شناخت کی چوری	یہ ایک ایسا جرم ہے جہاں مجرم عام طور پر مالی فائدے کے لیے افراد کی نقالی کرتے ہیں۔
دانشور جائیداد کی چوری	کسی دوسرے شخص یا کمپنی کے ذریعہ تیار کردہ عملی یا تصوراتی معلومات کو چوری کرنا۔
سلامی سلائسنگ (Salami Slicing)	ہر لین دین سے تھوڑی سی رقم چوری کرنا۔
اسکیم (Scam)	لوگوں کو کسی ایسی چیز پر یقین کرنے کے لیے دھوکہ دینا جو سچ نہیں ہے۔
اسپیم (Spam)	انٹرنیٹ صارفین کی ایک بڑی تعداد میں ناپسندیدہ ای میل تقسیم کریں۔
سپوفنگ (Spoofing)	یہ ایک بدنیتی پر مبنی عمل ہے جس میں مواصلت کو نامعلوم ذریعہ سے بھیجا جاتا ہے جس کو وصول کنندہ کے نام سے جانا جاتا ہے۔

سافٹ ویئر کی پائریسی (Software Piracy)

سافٹ ویئر پائریسی اصل میں کسی فرد یا ادارے کے ذریعہ تخلیق کردہ سافٹ ویئر کے کاپی رائٹ کی خلاف ورزی کے بارے میں ہے۔ اس میں کوڈز/ پروگرامس اور دیگر معلومات کو غیر قانونی طور پر چوری کرنا اور غیر مجاز ذرائع سے نقلی کاپیاں بنانا اور ڈیٹا یا تو اپنے فائدے کے لیے یا تجارتی منافع کے لیے استعمال کرنا شامل ہے۔

11th_Comp Science_UM_Ch 17.indd 278 17-04-2023 14:43:21

آسان الفاظ میں، سافٹ ویئر پائریسی "سافٹ ویئر کی غیر مجاز نقل" ہے۔ تصویر 17.2 سافٹ ویئر کی پائریسی کی خاکہ نگاری کی نمائندگی کرتا ہے۔



شکل -17.2 سافٹ ویئر پائریسی کی خاکہ نما نمائندگی

سافٹ ویئر پائریسی کے لیے ایک بالکل مختلف نقطہ نظر کو شیئر ویئر (shareware) کہا جاتا ہے، یہ لوگوں کو سافٹ ویئر کی کاپی کرنے سے روکنے کی کوشش کی فضولیت کو تسلیم کرتا ہے اور اس کے بجائے لوگوں کی ایمانداری پر انحصار کرتا ہے۔ شیئر ویئر پبلشرس صارفین کو پروگرام کی کاپیاں دوستوں اور ساتھیوں کو دینے کی ترغیب دیتے ہیں لیکن اس پروگرام کو باقاعدگی سے استعمال کرنے والے ہر فرد سے پروگرام کے مصنف کو براہ راست رجسٹریشن فیس ادا کرنے کے لیے کہتے ہیں۔ تجارتی پروگرام جو عوام کو غیر قانونی طور پر دستیاب کرائے جاتے ہیں انہیں اکثر وارز (Warez) کہا جاتا ہے۔

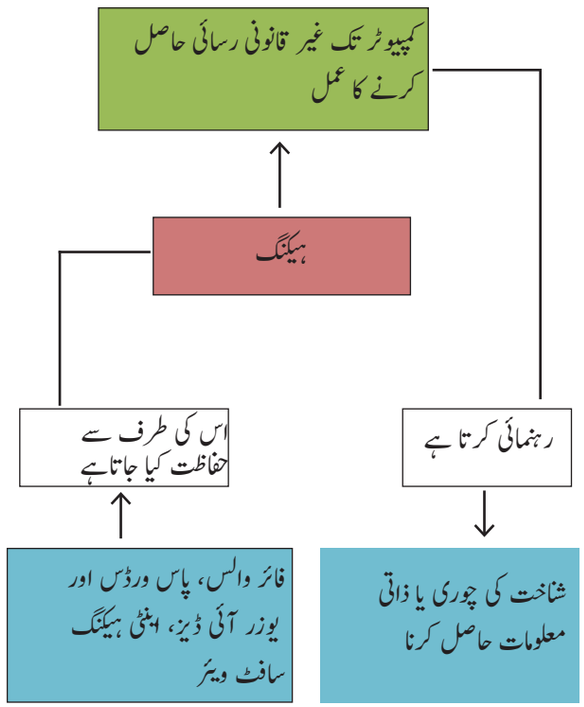
غیر مجاز رسائی (Unauthorised access)

غیر مجاز رسائی اس وقت ہوتی ہے جب کوئی کسی جائز صارف اکاؤنٹ کو توڑ کر ویب سائٹ، پروگرام، سرور، سروس یا دوسرے سسٹم تک رسائی حاصل کرتا ہے۔ مثال کے طور پر، اگر کوئی کسی ایسے اکاؤنٹ کے پاس ورڈ یا صارف نام کا اندازہ لگانے کی کوشش کرتا ہے جو اس تک رسائی حاصل کرنے تک اس کا نہیں تھا، تو اسے غیر مجاز رسائی سمجھا جاتا ہے۔

غیر مجاز رسائی کو روکنے کے لیے، فائر والز (firewalls)، انٹروژن ڈیٹیکشن سسٹم (IDS – Intrusion Detection System)، وائرس (Virus) اور مواد کے اسکینرس (Content Scanners)، پیٹچیس (Patches) اور ہاٹ فکسس (Hot Fixes) کا استعمال کیا جاتا ہے۔

ہیکنگ (Hacking)

مالک کی اجازت یا علم کے بغیر ذاتی ڈیٹا چرانے کے لیے کمپیوٹر سسٹم میں داخل ہونا ہیکنگ ہے (جیسے پاس ورڈ چرانا)۔ یہ کمپیوٹر سسٹم تک غیر مجاز رسائی بھی حاصل کر رہا ہے، اور اس کے مواد کو تبدیل کر رہا ہے۔ یہ کسی مجرمانہ سرگرمی کے تعاقب میں کیا جا سکتا ہے یا یہ ایک مشغلہ ہو سکتا ہے۔ ہیکنگ بے ضرر ہو سکتی ہے اگر ہیکر صرف سسٹمز کے دفاع کو توڑنے کے چیلنج سے لطف اندوز ہو رہا ہو، لیکن ایسی اخلاقی ہیکنگ کو صرف کنٹرول شدہ تجربات کے طور پر استعمال کیا جانا چاہیے۔ تصویر 17.3 ہیکنگ کے خاکہ کی نمائندگی کو ظاہر کرتی ہے۔



تصویر 17.3 ہیکنگ کے خاکہ کے نمائندگی

11th_Comp Science_UM_Ch 17.indd 279 17-04-2023 14:43:21

کریکنگ (Cracking)

کریکنگ وہ جگہ ہے جہاں کوئی پروگرام کے ماخذ میں ترمیم کرتا ہے تاکہ کوڈ کا استحصال یا اس میں ترمیم کی جاسکے۔ کریکر (جسے بلیک ہیٹ (Black hat) یا ڈارک سائیڈ ہیکر (Dark side hacker) بھی کہا جاتا ہے) ایک بدنیتی پر مبنی یا مجرمانہ ہیکر ہے۔

"کریکنگ" کا مطلب ہے ڈیٹا چوری کرنے، کرپٹ کرنے یا غیر قانونی طور پر دیکھنے کے لیے کمپیوٹر سسٹمز میں داخل ہونے کی کوشش کرنا۔

ایک کریکر وہ ہوتا ہے جو کمپیوٹر پروگراموں میں پاس ورڈ یا لائسنس کو نظرانداز کرتے ہوئے، اکثر نیٹ ورک پر کسی اور کے کمپیوٹر سسٹم میں داخل ہوتا ہے۔

وہ کچھ حساس معلومات کی درخواست کرتے ہوئے سرکاری ای میل بھیج سکتے ہیں۔ یہ بینک یا دوسرے سرکاری ادارے کی طرف سے ایک جائز ای میل کی طرح نظر آ سکتا ہے۔

17.3 سائبر سیکیورٹی اور خطرات (Cyber Security and Threats)

سائبر حملے بنیادی طور پر کمپیوٹر سسٹم کو اہم نقصان پہنچانے یا کسی فرد یا تنظیم سے اہم معلومات چرانے کے لیے شروع کیے جاتے ہیں۔ سائبر سیکیورٹی مختلف ٹیکنالوجیز، عمل اور اقدامات کا مجموعہ ہے جو سائبر حملوں کے خطرے کو کم کرتے ہیں اور کمپیوٹر پر مبنی خطرات سے تنظیموں اور افراد کی حفاظت کرتے ہیں۔

سائبر حملوں کی اقسام

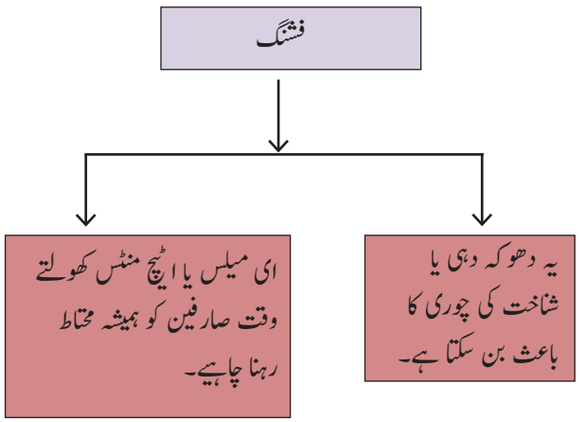
مالویئر (Malware) سافٹ ویئر کی ایک قسم ہے جس کے ذریعے مجرم سافٹ ویئر تک غیر قانونی رسائی حاصل کرتے ہیں اور نقصان پہنچاتے ہیں۔ سائبر حملوں کی مختلف اقسام اور ان کے افعال کو ٹیبل 17.2 میں دیا گیا ہے۔

سائبر سیکیورٹی کے خطرات (Cyber Security Threats)

حالیہ برسوں میں، زیادہ تر افراد اور اداروں کو سیکورٹی کے نظام میں موجود کمزوریوں اور سمجھوتہ شدہ تنظیمی ڈھانچے کی وجہ سے مسائل کا سامنا ہے۔ سائبر سیکیورٹی خطرات کی مختلف اقسام کو ذیل میں درجہ بندی کیا گیا ہے:

فشنگ (Phishing)

فشنگ ایک قسم کا کمپیوٹر جرم ہے جو حملہ کرنے، صارف کا ڈیٹا چوری کرنے کے لیے استعمال ہوتا ہے، بشمول لاگ ان (login) کا نام، پاس ورڈ اور کریڈٹ کارڈ نمبر۔ یہ اس وقت ہوتا ہے جب کوئی حملہ آور ایک ای میلر کو فوری ٹیکسٹ میسج کھولنے کے لیے شکار کو نشانہ بناتا ہے۔ حملہ آور نقصان دہ لنکس یا منسلکات کو تقسیم کرنے کے لیے فشنگ کا استعمال کرتا ہے جو متاثرین سے حساس لاگ ان اسناد کو نکالنے سمیت متعدد افعال انجام دے سکتے ہیں۔

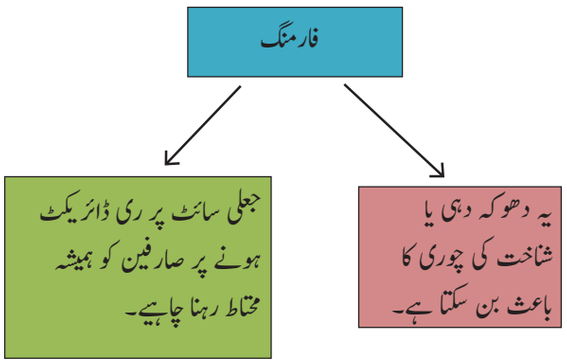


تصویر 17.4 فشنگ کی نمائندگی کا خاکہ

فارمنگ (Pharming)

فارمنگ ایک اسکیمنگ (Scamming) عمل ہے، جس میں نقصان دہ کوڈ ذاتی کمپیوٹر یا سرور پر انسٹال کیا جاتا ہے، جو صارفین کو ان کی معلومات یا اجازت کے بغیر دھوکہ دہی والی ویب سائٹس کی طرف گمراہ کرتا ہے۔ فارمنگ کو "جال کے بغیر فشنگ - phishing without a trap" کہا گیا ہے۔ یہ ایک اور طریقہ ہے کہ ہیکرس انٹرنیٹ پر صارفین کو جوڑ توڑ کرنے کی کوشش کرتے ہیں۔ یہ ایک سائبر حملہ ہے جس کا مقصد ویب سائٹ کی ٹریفک کو جعلی سائٹ پر بھیجنا ہے۔

11th_Comp Science_UM_Ch 17.indd 280 17-04-2023 14:43:21

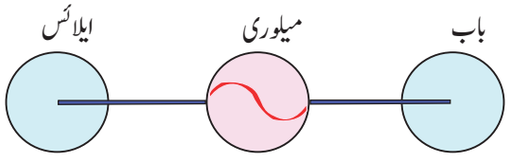


شکل 17.5 فارمنگ کی نمائندگی کا خاکہ

مین اِن دِی مڈل (Man In The Middle (MITM))

مین-ان-دی-مڈل حملہ (MITM ؛ جانوس حملہ بھی) ایک ایسا حملہ ہے جہاں حملہ آور خفیہ طور پر چلاتا ہےاور ممکنہ طور پر دو فریقوں کے درمیان مواصلت کو بدل دیتا ہے جو یہ سمجھتے ہیں کہ وہ براہ راست ایک دوسرے کے ساتھ بات چیت کر رہے ہیں۔

مثال: فرض کریں کہ ایلائس (Alice)، باب (Bob) کے ساتھ بات چیت کرنا چاہتی ہے۔ دریں اثنا، میلوری (Mallory) گفتگو کو سننے اور اختیاری طور پر باب (Bob) کو غلط پیغام پہنچانے کے لیے روکنا چاہتی ہے۔



تصویر 17.6 - مین-ان-دی-مڈل حملے کی ایک مثال

کوکیز (Cookies)

ایک کوکی (جسے HTTP کوکی، ویب کوکی، انٹرنیٹ کوکی، براؤزر کوکی، یا صرف کوکی بھی کہا جاتا ہے) ایک ویب سائٹ سے بھیجے گئے ڈیٹا کا ایک چھوٹا ٹکڑا ہے اور صارف کے ویب براؤزر کے ذریعے صارف کی کمپیوٹر میموری (ہارڈ ڈرائیو) پر محفوظ کیا جاتا ہے جبکہ صارف انٹرنیٹ براؤز کر رہا ہے۔کوکیزویب سائٹس کے لیے ایک قابل بھروسہ طریقہ کار کے طور پر ڈیزائن کیا گیا تھا کہ وہ مکمل معلومات کو یاد رکھیں (جیسے کہ کسی آن لائن اسٹور میں شاپنگ کارٹ میں شامل کردہ آئٹمز) یا صارف کی براؤزنگ سرگرمی (بشمول مخصوص بٹنوں پر کلک کرنا، لاگ ان کرنا وغیرہ) کو ریکارڈ کرنا۔ ان کا استعمال معلومات کے بے ترتیب ٹکڑوں کو یاد رکھنے کے لیے بھی کیا جا سکتا ہے جو صارف نے پہلے فارم فیلڈز جیسے کہ نام، پتے، پاس ورڈ اور کریڈٹ کارڈ نمبرز میں داخل کیے تھے۔ سیکورٹی کے نقطہ نظر سے، اگر کوکی ڈیٹا کو انکرپٹ نہیں کیا جاتا ہے، تو کوئی بھی گمنام صارف (ہیکر) کوکی کی معلومات تک رسائی حاصل کر سکتا ہے اور اس کا غلط استعمال کر سکتا ہے۔

ویب سائٹس عام طور پر درج ذیل وجوہات کے لیے کوکیز استعمال کرتی ہیں:

- اس بارے میں آبادیاتی معلومات اکٹھا کرنے کے لیے کہ کس نے ویب سائٹ کا دورہ کیا ہے۔
- سائٹس اکثر اس معلومات کا استعمال یہ جاننے کے لیے کرتی ہیں کہ وزیٹر (دورہ کرنے والے) کتنی بار سائٹ پر آتے ہیں اور وہ کتنی دیر تک سائٹ پر رہتے ہیں۔
- یہ ویب سائٹ پر صارف کے تجربے کو ذاتی بنانے میں مدد کرتا ہے۔
- کوکیز صارفین کے بارے میں ذاتی معلومات کو ذخیرہ کرنے میں مدد کر سکتی ہیں تاکہ جب کوئی صارف بعد میں سائٹ پر واپس آجائے، تو زیادہ ذاتی نوعیت کا تجربہ فراہم کیا جائے۔

11th_Comp Science_UM_Ch 17.indd 281 17-04-2023 14:43:21

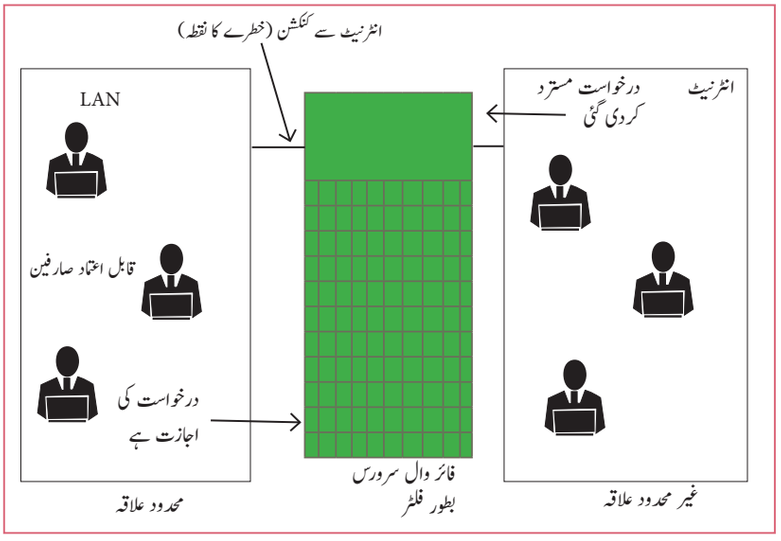
اگر آپ کبھی کسی سائٹ پر واپس آئے اور آپ کا نام پراسرار طور پر اسکرین پر ظاہر ہوتا دکھے، تو اس کی وجہ یہ ہے کہ پچھلے وزٹ پر، آپ نے سائٹ کو اپنا نام دیا تھا اور یہ ایک کوکی میں محفوظ تھا۔ اس کی ایک اچھی مثال یہ ہے کہ جس طرح سے کچھ آن لائن شاپنگ سائٹس صارفین کو ان کی سابقہ خریداریوں کی بنیاد پر سفارشات پیش کریں گی۔ یہ اشتہارات کی نگرانی میں مدد کرتا ہے۔کوکیز کمپیوٹر سسٹم پر بدنیتی سے کام نہیں کرتی ہیں۔ وہ محض ٹیکسٹ فائلیں ہیں جنہیں کسی بھی وقت ڈیلیٹ کیا جا سکتا ہے۔

کوکیز کو وائرس پھیلانے کے لیے استعمال نہیں کیا جا سکتا اور وہ آپ کی ہارڈ ڈرائیو تک رسائی حاصل نہیں کر سکتیں۔ تاہم، کوئی بھی ذاتی معلومات جو آپ کسی ویب سائٹ کو فراہم کرتے ہیں، بشمول کریڈٹ کارڈ کی معلومات، زیادہ تر ممکنہ طور پر کوکی میں محفوظ کی جائے گی جب تک کہ آپ کے براؤزر میں کوکی کی خصوصیت کو واضح طور پر بند نہ کر دیا جائے۔ یہ وہ طریقہ ہے جس میں کوکیز رازداری کے لیے خطرہ ہیں۔

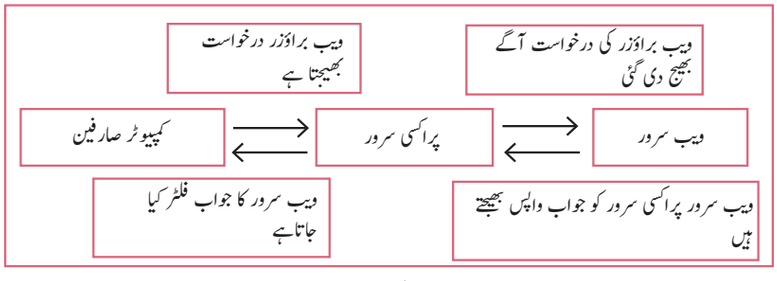
فائر وال اور پراکسی سرورس
(Firewall and Proxy Servers)

فائر وال ایک کمپیوٹر نیٹ ورک سیکیورٹی پر مبنی نظام ہے جو نگرانی کرتا ہے اورپہلے سے طے شدہ حفاظتی اصولوں کی بنیاد پر آنے والے اور جانے والے نیٹ ورک ٹریفک کو کنٹرول کرتا ہے۔ ایک فائر وال عام طور پر ایک قابل اعتماد اندرونی کمپیوٹر نیٹ ورک اور نیٹ ورک کے باہر سپرد کمپیوٹر کے درمیان ایک بلاک قائم کرتا ہے۔ تصویر 17.7 فائر وال سرور کے کام کو ظاہر کرتی ہے۔

ایک پراکسی سرور (Proxy Server) اختتامی صارفین اور ویب سرور کے درمیان ایک ثالث کے طور پر کام کرتا ہے۔ ایک گراہک پراکسی سرور سے جڑتا ہے، کچھ سروس کی درخواست کرتا ہے، جیسے کہ فائل، کنکشن، ویب صفحہ، یا کسی دوسرے سرور سے دستیاب دیگر وسائل۔ پراکسی سرور درخواست کی جانچ کرتا ہے، صداقت کی جانچ کرتا ہے اور اس کی بنیاد پر درخواست کی منظوری دیتا ہے۔ پراکسی سرورس عام طور پر اکثر دیکھے جانے والے سائٹ کے پتوں کو اپنے کیشے (cache) میں رکھتے ہیں جس سے جوابی وقت میں بہتری آتی ہے۔ تصویر 17.8 پراکسی سرور کے کام کو ظاہر کرتی ہے۔



تصویر 17.7 - فائر وال سرور



تصویر 17.8 پراکسی سرور کا کام کرنا

11th_Comp Science_UM_Ch 17.indd 282 17-04-2023 14:43:21

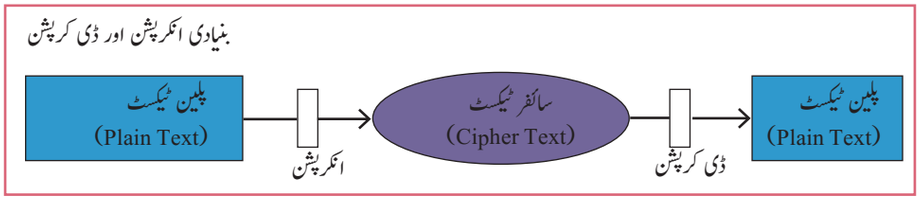
انکرپشن اور ڈی کرپشن (Encryption and Decryption)

انکرپشن اور ڈی کرپشن وہ عمل ہیں جو رازداری کو یقینی بناتے ہیں کہ صرف مجاز افراد ہی معلومات تک رسائی حاصل کر سکتے ہیں۔

انکرپشن سادہ متن کے اعداد و شمار (سادہ متن) کو بے ترتیب (random) اور منڈلا ہوا (mangled) ڈیٹا (جسے سائفر ٹیکسٹ - cipher-text کہتے ہیں) میں ترجمہ کرنے کا عمل ہے۔

ڈکرپشن سائفر ٹیکسٹ کو سادہ متن میں تبدیل کرنے کا الٹا عمل ہے۔ انکرپشن اور ڈکرپشن کرپٹوگرافی (cryptography) کے ذریعے کی جاتی ہے۔ کرپٹوگرافی میں ایک کلید معلومات کا ایک ٹکڑا (پیرامیٹر) ہے جو کرپٹوگرافک الگورتھم (cryptographic algorithm) کے فنکشنل آؤٹ پٹ کا تعین کرتی ہے۔

تصویر 17.9 انکرپشن اور ڈکرپشن کے عمل کو ظاہر کرتی ہے۔



تصویر 17.9 انکرپشن اور ڈکرپشن

انکرپشن کا استعمال فوجوں اور حکومتوں نے خفیہ مواصلات کی سہولت کے لیے کیا ہے۔ اب یہ عام طور پر کئی قسم کے شہری نظاموں میں معلومات کی حفاظت کے لیے استعمال ہوتا ہے۔ یہ مواصلاتی نظام میں ڈیٹا کی حفاظت کے لیے بھی استعمال ہوتا ہے، مثال کے طور پر نیٹ ورکس (جیسے انٹرنیٹ، ای کامرس)، موبائل ٹیلی فون، وائرلیس مائیکروفون، وائرلیس انٹرکام سسٹم، بلوٹوتھ ڈیوائسز اور بینک آٹومیٹک ٹیلر مشینوں کے ذریعے ڈیٹا منتقل کیا جاتا ہے۔ حالیہ برسوں میں مواصلات میں ڈیٹا کو روکنے کی متعدد اطلاعات موصول ہوئی ہیں۔ نیٹ ورکس پر منتقل ہونے پر ڈیٹا کو بھی انکرپٹ کیا جانا چاہیے تاکہ غیر مجاز صارفین کے ذریعے نیٹ ورک ٹریفک سے بچایا جا سکے۔

17.4 انفارمیشن ٹیکنالوجی ایکٹ کا تعارف

اکیسویں صدی میں کمپیوٹر، انٹرنیٹ اور آئی سی ٹی (ICT) یا ای انقلاب (e-revolution) نے لوگوں کا طرز زندگی بدل کر رکھ دیا ہے۔ آج کاغذ پر مبنی کمیونیکیشن کو ای کمیونیکیشن (e-communication) نے بدل دیا ہے۔ اس کے مطابق ہمارے پاس سائبر ورلڈ، ای ٹرانزیکشن (e-transaction)، ای بینکنگ (e-banking)، ای ریٹرن (e-return) اور ای کنٹریکٹس (e-contracts) جیسی نئی اصطلاحات ہیں۔ ای انقلاب کے مثبت پہلو کے علاوہ کمپیوٹر کا منفی پہلو بھی ہے، یعنی انٹرنیٹ اور آئی سی ٹی مجرموں کے ہاتھ میں جو کہ جرم کا ہتھیار بن چکا ہے۔ اس کے مطابق سائبر اسپیس میں سائبر کرائمز کے مسائل سے نمٹنے کے لیے اراکین کا ایک نیا پینل سامنے آیا، یعنی سائبر لاء (cyber law) یا سائبر اسپیس لاء (cyber space law) یا انفارمیشن ٹیکنالوجی لاء (Information Technology law) یا انٹرنیٹ لاء (Internet law)۔

بھارت میں سائبر قانون اور آئی ٹی ایکٹ 2000 (IT Act 2000) کمپیوٹر جرائم کو روکنے کے لیے 2008 میں ترمیم کی جا رہی ہے۔ آئی ٹی ایکٹ 2000 الیکٹرانک ڈیٹا انٹرچینج (EDI-Electronic Data Interchange) اور الیکٹرانک کمیونیکیشن کے دیگر ذرائع سے ہونے والے لین دین کو قانونی شناخت فراہم کرنے کا ایکٹ ہے۔ یہ ہندوستان میں سائبر کرائم اور الیکٹرانک کامرس (ای کامرس - e-commerce) سے متعلق بنیادی قانون ہے۔ ای کامرس الیکٹرانک ڈیٹا کا تبادلہ یا معلومات کی الیکٹرانک فائلنگ ہے۔

11th_Comp Science_UM_Ch 17.indd 283 17-04-2023 14:43:21

کیا آپ جانتے ہیں؟

"سائبر قانون یا انٹرنیٹ قانون ایک اصطلاح ہے جو انٹرنیٹ کے استعمال سے متعلق قانونی مسائل کو سمیٹتی ہے۔

روک تھام (Prevention)

سائبر کرائم کا 25 فیصد حل نہیں ہوا ہے۔ معلومات کی حفاظت کے لیے درج ذیل نکات کو نوٹ کرنا ضروری ہے:

- پیچیدہ پاس ورڈ (Complex Password) ترتیب آپ کی سرفنگ کو محفوظ بنا سکتی ہے۔
- جب انٹرنیٹ استعمال میں نہ ہو تو اسے منقطع کر دیں۔
- اسپام میل یا ای میلز کو نہ کھولیں (Do NOT open) جن میں کوئی ناواقف بھیجنے والا ہو۔
- اینٹی وائرس سافٹ ویئر (Anti-virus software) استعمال کرتے وقت اسے اپنے پاس سب سے نیا رکھیں۔

کیا آپ جانتے ہیں؟

- آگاہی سلامتی کی کلید ہے۔
- معلومات کی حفاظت کاروبار کے جسم میں مدافعتی نظام ہے۔
- ایک چیک (check) جو باؤنس نہیں ہوتا ہے اسے سیکیورٹی چیک (security check) کہتے ہیں۔ باہر آنے سے پہلے روزانہ چیک کیجیے!
- اپنا کام کریں - سیکیورٹی ہوشیار رہیں !!!
- کلک کرنے میں جلدی نہ کریں ... آن لائن خریداری کرتے وقت محتاط رہیں۔
- دوبارہ شروع (restart) کرنا سمارٹ کام ہے۔
- پاس ورڈ ٹوتھ برش (toothbrushes) کی طرح ہوتے ہیں۔ وہ نئے ہونے پر بہترین ہوتے ہیں اور ان کا اشتراک کبھی نہیں کرنا چاہیے۔
- جب آپ اور آپ کا سسٹم الگ ہوجاتا ہے، تو آپ کا سسٹم دن کے لیے سب سے پہلے بند ہونا چاہیے۔
- آپ کا دماغ معلومات کا ذخیرہ ہے، دروازے کو بند رکھیں۔
- _a_ word کا لفظ نہیں ہے بغیر حفاظت، محفوظ اور بے خوف کے۔
- لنک لنک کو نظر انداز کرنے سے روکیں ..... کنیکٹ کرنے سے پہلے سوچیں .....

تشخیص

سیکشن - A

صحیح جواب کا انتخاب کریں

1. مندرجہ ذیل میں سے کون سا طریقہ کار، طریقوں اور اقدار سے متعلق ہے؟
   a) پائریسی b) پروگرام c) وائرس d) کمپیوٹر اخلاقیات

2. عوام کو غیر قانونی طور پر دستیاب تجارتی پروگرام کے نام سے جانا جاتا ہے۔
   a) فری ویئر b) واریز (Warez) c) مفت سافٹ ویئر d) سافٹ ویئر

3. مندرجہ ذیل میں سے کون سا ہے جسے خود کو دہرایا جانے اور خود کو منسلک کرنے کے لیے کمپیوٹر پروگرام کی ضرورت نہیں ہے؟
   a) وائرسس b) وارمس (worms) c) سپائی ویئر d) ٹروجنس

4. مندرجہ ذیل میں سے کون سا ٹریک صارف کسی ویب سائٹ پر جاتا ہے؟
   a) سپائی ویئر b) کوکیز c) وارمس d) ٹروجنس

11th_Comp Science_UM_Ch 17.indd 284 17-04-2023 14:43:22

5. مندرجہ ذیل میں سے کون سا کمپیوٹر سسٹمز بدنیتی پر مبنی پروگرام نہیں ہے؟
a) وارمس (worms) b) ٹروجن c) سپائی ویئر d) کوکیز

6. ایک کمپیوٹر نیٹ ورک سیکیورٹی جو آنے اور جانے والے ٹریفک کی نگرانی اور کنٹرول کرتی ہے۔
a) کوکیز b) وائرس c) فائر وال d) وارمس (worms)

7. سائفر ٹیکسٹ کو سادہ متن میں تبدیل کرنے کے عمل کو کہتے ہیں
a) انکرپشن b) ڈکرپشن c) کلید d) پراکسی سرور

8. ای کامرس کا مطلب ہے۔
a) electronic commerce b) electronic data exchange
c) electric data exchange d) electronic commercialization

9. دوسروں کو ناپسندیدہ ای میل تقسیم کرنا کہلاتا ہے
a) اسکیم b) اسپیم c) فراڈ d) جعل سازی (spoofing)

10. لین دین کے لئے قانونی شناخت کی طرف سے کئے جاتے ہیں
a) الیکٹرانک ڈیٹا انٹرچینج (Electronic Data Interchange) b) الیکٹرانک ڈیٹا ایکسچینج (Electronic Data Exchange)
c) الیکٹرانک ڈیٹا ٹرانسفر (Electronic Data Transfer) d) الیکٹریکل ڈیٹا انٹرچینج (Electrical Data Interchange)

سیکشن - B

بہت ہی مختصر جوابات دیں۔

1. ہارویسٹنگ کیا ہے؟
2. وارز کیا ہیں؟
3. کریکنگ پر ایک مختصر نوٹ لکھیں۔
4. سائبر حملوں کی دو قسمیں لکھیں۔
5. کوکی کیا ہے؟

سیکشن - C

مختصر جوابات دیں

1. فائر والز کا کیا کردار ہے؟
2. انکرپشن اور ڈکرپشن کے بارے میں لکھیں۔
3. پراکسی سرور کے بارے میں وضاحت کریں
4. کسی بھی کمپیوٹر استعمال کرنے والے کو کن رہنما اصولوں پر عمل کرنا چاہیے؟
5. اخلاقی مسائل کیا ہیں؟ کچھ نام بتائیں۔

سیکشن - D

تفصیل سے بیان کریں۔

1. کمپیوٹر کے استعمال سے ہونے والے مختلف جرائم کیا ہیں؟
2. پائریسی کیا ہے؟ پائریسی کی اقسام بتائیں؟ اسے کیسے روکا جا سکتا ہے؟
3. سائبر حملوں کی مختلف اقسام لکھیں۔

حوالہ جات کتابیں :

- Computer Network Security and Cyber Ethics by Joseph MiggaKizza
- "Investigating Cyber Law and Cyber Ethics: Issues, Impacts and Practices: 1" by Alfreda Dudley and James Braman

11th_Comp Science_UM_Ch 17.indd 285 17-04-2023 14:43:22

باب 18

تمل کمپیوٹنگ
(Tamil Computing)

18.1 Introduction

" பிறநாட்டு நல்லறிஞர் சாத்திரங்கள்

தமிழ்மொழியிற் பெயர்த்தல் வேண்டும்;

இறவாத புகழுடைய புதுநூல்கள்

தமிழ்மொழியில் இயற்றல் வேண்டும்;

மறைவாக நமக்குள்ளே பழங்கதைகள்

சொல்வதிலோர் மகிமை இல்லை;

திறமான புலமையெனில் வெளிநாட்டோர்;

அதை வணக்கஞ் செய்தல் வேண்டும்."

– மகாகவி பாரதி

انسانی تہذیب نے بیسویں صدی میں کمپیوٹر کی اختراع سے ترقی کی۔ کمپیوٹر کی ترقی کا آغاز ایک ابتدائی حساب کے آلے کے طور پر ہوا اور اب کمپیوٹر کے بغیر انسانی زندگی کے وجود کی بڑی ترقی کے لیے ایک لازمی جزو بن گیا ہے۔

یہ درست ہے کہ کوئی بھی زبان اس وقت پرانی ہو جائے گی جب اس میں خود کو بدلتی ہوئی ٹیکنالوجیز کے مطابق ڈھالنے کی صلاحیت نہ ہو۔ تمل ہزاروں سالوں سے زندہ زبان ہے۔ جدید ٹیکنالوجی کی ترقی، کلاسیکی تمل کی ترقی کو متاثر نہیں کرتی کیونکہ یہ بڑھتی ہوئی تکنیکی تبدیلیوں کو اپنانے کے لیے تیار ہے۔ ***تمل صرف ایک زبان نہیں ہے، یہ ہماری شناخت، ہماری زندگی اور ہمارا احساس ہے۔***

"எங்கள் வாழ்வும், எங்கள் வளமும் மங்காத தமிழென்று சங்கே முழங்கு" – புரட்சி கவி.

18.2 انٹرنیٹ میں تمل

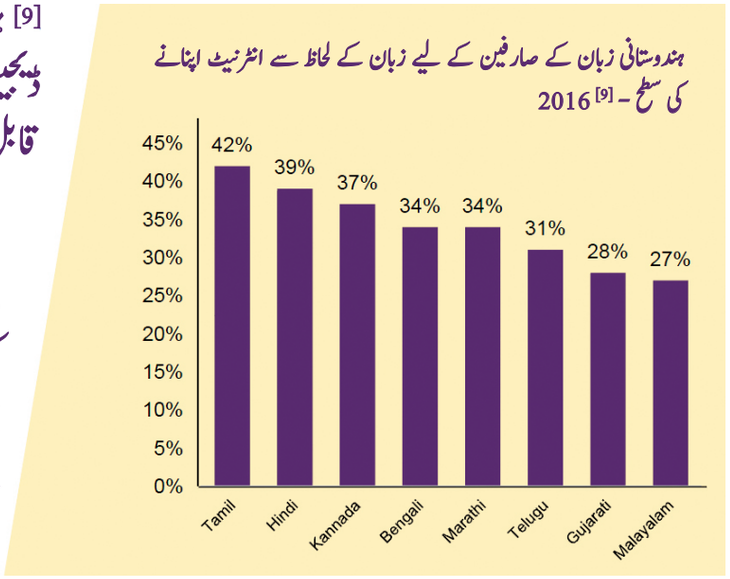
ہم جانتے ہیں کہ انٹرنیٹ آج ہر آدمی کی زندگی میں ایک اہم کردار ادا کرتا ہے۔ انٹرنیٹ ایک بہترین انفارمیشن ٹیکنالوجی ڈیوائس ہے، جس کے ذریعے ہم معلومات حاصل کرتے ہیں۔

2017 میں سنگاپور (Singapore) کی ایک تنظیم KPMG کی طرف سے گوگل (google) کے ساتھ مل کر کی گئی ایک تحقیق میں بتایا گیا کہ، ہندوستان میں سب سے زیادہ استعمال ہونے والی زبانوں میں تمل سب سے اوپر ہے، جہاں 42٪ تمل میں انٹرنیٹ استعمال کر رہے ہیں۔

11th_Comp Science_UM_Ch 18.indd 286 17-04-2023 14:53:54

[9] 68% انٹرنیٹ صارفین مقامی زبان کے ڈیجیٹل مواد کو انگریزی کے مقابلے زیادہ قابل اعتماد سمجھتے ہیں۔

فی الحال تمل ([9] 42%) میں انٹرنیٹ کو اپنانے کی سطح ہندی اور کنڑ کے بعد ہندوستانی زبان کے صارفین میں ہے

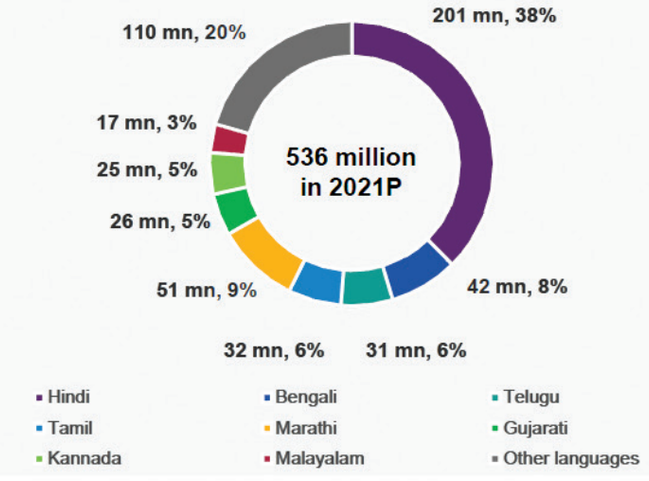


مطالعہ کے مطابق، 2021 تک، ہندوستان میں 74% لوگ تمل کا استعمال کرتے ہوئے انٹرنیٹ تک رسائی حاصل کریں گے اور یہ ہندوستان میں انٹرنیٹ کے استعمال میں سرفہرست ہوگا۔

تمل اور کنڑ بولنے والوں میں ایکو سسٹم کے ہندوستانی زبان کے ساتھ مستقبل میں انٹرنیٹ کو اپنانے کا سب سے زیادہ رجحان ہے

مراٹھی، بنگالی، تمل اور تیلگو انٹرنیٹ استعمال کرنے والے کل ہندستانی زبان کے انٹرنیٹ استعمال کنندگان کی تعداد کا 30% بنتے ہیں۔

2021 تک ہندی انٹرنیٹ استعمال کرنے والوں کی تعداد انگریزی صارفین سے زیادہ ہو جائے گی۔



Language	Adoption propensity
Hindi	54%
Bengali	46%
Telugu	60%
Tamil	74%
Marathi	43%
Gujarati	43%
Kannada	74%
Malayalam	44%
Other languages	NA*

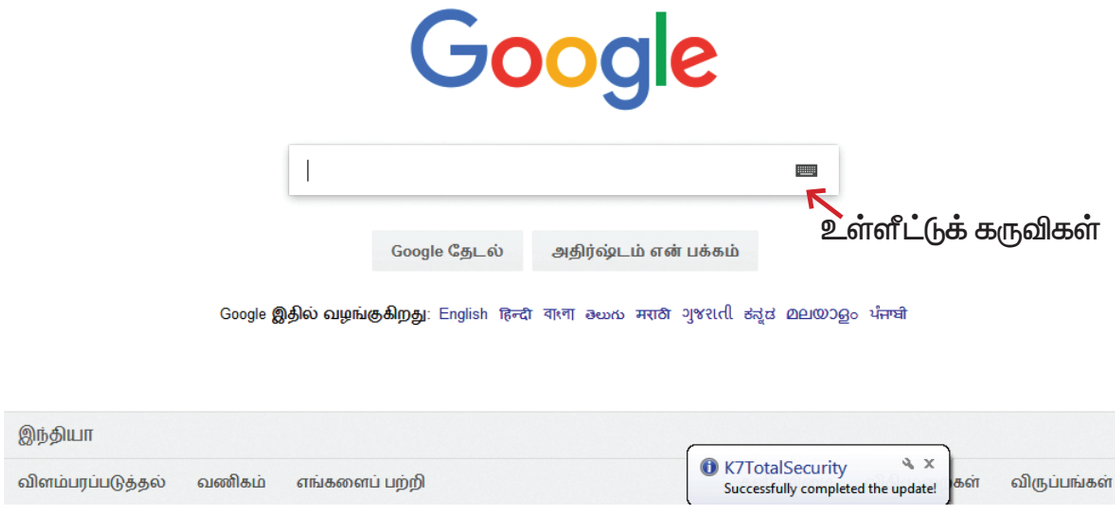
* Propensity to adopt internet in future has been ascertained for select 8 Indian languages

یہ اعداد و شمار (statistical data) تمل میں انٹرنیٹ خدمات کو بہتر بنانے کے لیے کارآمد ثابت ہوں گے۔

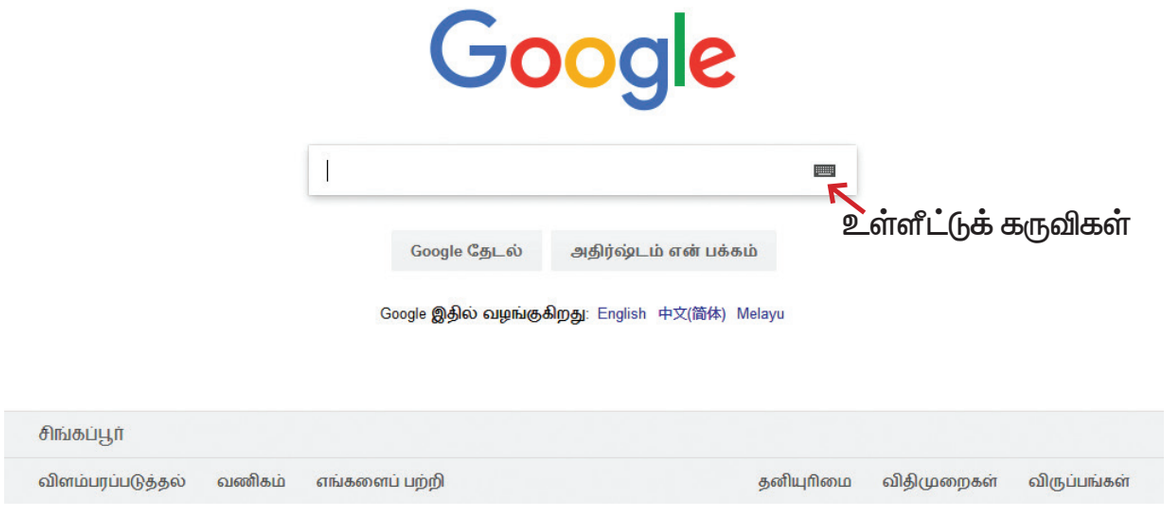
11th_Comp Science_UM_Ch 18.indd 287 17-04-2023 14:53:54

18.3 تمل میں سرچ انجنس

"سرچ انجنس (Search engines)" سائبر اسپیس (Cyber space) سے کسی بھی معلومات کو تلاش کرنے کے لیے استعمال ہوتے ہیں۔ اگرچہ بہت سارے سرچ انجن ہیں، لیکن ان میں سے صرف چند ہی اکثر استعمال میں ہیں۔ ٹاپ ٹین (Top ten) سرچ انجنوں میں گوگل، بنگ (bing) اوریاہو (Yahoo) نے بالترتیب پہلی تین جگہیں لی ہیں۔ گوگل اور بنگ تمل میں تلاش کی سہولت فراہم کرتے ہیں، جس کا مطلب ہے کہ آپ ہر چیز کو تمل کے ذریعے تلاش کر سکتے ہیں۔ گوگل سرچ انجن آپ کو ان بلٹ تمل ورچوئل کی بورڈ (inbuilt Tamil Virtual Keyboard) فراہم کرتا ہے۔

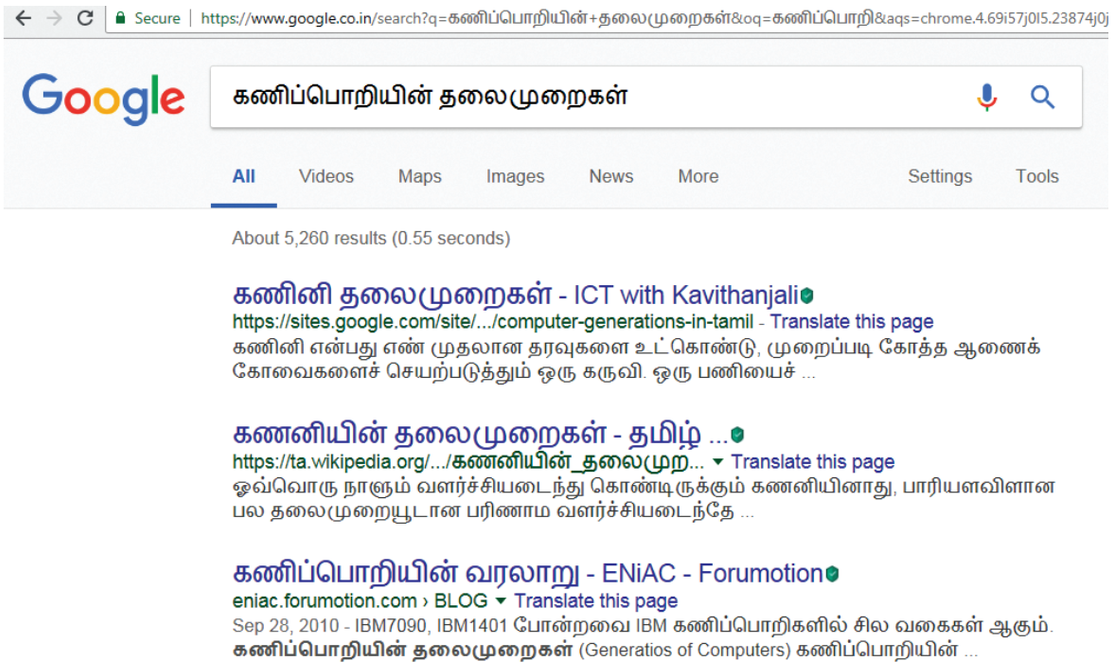


தصویر 18.1 (a) گوگل سرچ انجن (انڈیا)



تصویر 18.1 (b) گوگل سرچ انجن (سنگاپور)

11th_Comp Science_UM_Ch 18.indd 288 17-04-2023 14:53:54



تصویر 18.2 تمل میں تلاش کرنا

18.4 ای گورننس:
(e- Governance)

انٹرنیٹ کے ذریعے سرکاری خدمات حاصل کرنا ای گورننس(e – Governance) کے نام سے جانا جاتا ہے۔ حکومت تمل ناڈو انٹرنیٹ کے ذریعے اپنی خدمات دے رہا ہے۔ دنیا کے کسی بھی کونے سے کوئی بھی حکومت تمل ناڈو سے رابطہ کر سکتا ہے۔ حکومت تمل ناڈو کے ویب پورٹل سے کوئی بھی اہم اعلانات، سرکاری احکامات اور سرکاری فلاحی اسکیمیں حاصل کر سکتا ہے۔



تصویر 18.3 حکومت تمل ناڈو کی سرکاری ویب سائٹ۔

11th_Comp Science_UM_Ch 18.indd 289 17-04-2023 14:53:55

بذریعہ تمل ای گورنینس	ویب اڈریس
حکومت تمل ناڈو کی سرکاری ویب سائٹ	http://www.tn.gov.in/ta
شعبۂ زراعت انجنئرینگ	http://www.aed.tn.gov.in/
شعبۂ ماحولیات	http://www.environment.tn.nic.in/
ڈائریکٹوریٹ امتحانات برائے حکومت	http://www.dge.tn.nic.in/
تمل ناڈو محکمۂ صحت	http://www.tnhealth.org/
شعبۂ تمل ناڈو مائکرو، چھوٹے اور درمیانی انٹرپرائزیز	http://www.msmeonline.tn.gov.in/
شعبۂ دیہی ترقی اور پنچایت راج	http://www.tnrd.gov.in/
محکمۂ اقلیتی بہبود برائے پسماندہ طبقات	http://www.bcmbcmw.tn.gov.in/
محکمۂ جنگلات تمل ناڈو	https://www.forests.tn.gov.in/
محکمۂ اوقاف برائے ہندو مذہب اور خیرات۔	http://www.tnhrce.org/
تمل ناڈو پبلک سروس کمیشن (TNPSC)	http://www.tnpsc.gov.in/tamilversion/index.html
حکومت سری لنکا کی سرکاری ویب سائٹ	https://www.gov.lk/index.php

ہندوستان سے باہر، سری لنکا کی حکومت اپنی تمام خدمات تمل میں سرکاری ویب سائٹ کے ذریعے فراہم کرتی ہے۔

18.5 ای - لائبریری

ای لائبریریاں (e-Libraries) پورٹل یا ای کتابوں کے جمع کرنے کی ویب سائٹ ہیں۔ تمل ای لائبریری خدمات ہزاروں تمل کتابیں بطور ای بکس زیادہ تر مفت میں فراہم کرتی ہیں ۔ یہ تمل لوگوں کے لیے سب سے مفید خدمت ہے جو اپنی آبائی سرزمین سے بہت دور رہتے ہیں۔

تمل ای لائبریری	ویب سائٹ اڈریس
تمل ناڈو اسکول ایجوکیشن، ٹیچر ایجوکیشن ٹریننگ، ٹیکسٹ بک اور وسائل بک	http://www.textbooksonline.tn.nic.in/
تمل ورچول اکیڈمی	http://www.tamilvu.org/library/libindex.htm
کونیمرا پبلک لائبریری	http://connemarapubliclibrarychennai.com/Veettukku_oru_noolagam/index.html
تمل ڈیجیٹل لائبریری	http://tamildigitallibrary.in/
چنئی لائبریری	http://www.chennailibrary.com/
تمل اگم	http://www.thamizhagam.net/parithi/parithi.html

11th_Comp Science_UM_Ch 18.indd 290 17-04-2023 14:53:55

http://www.projectmadurai.org/pmworks.html	پروجیکٹ مدورے
http://www.tamilheritage.org/old/text/ebook/ebook.html	قدیم کتابیں اور مخطوطات
http://www.noolaham.org/wiki/index.php/	نول اہم
http://www.annacentenarylibrary.org/	انّا صدی (Centenary) لائبریری

18.6 تمل ٹائپنگ اور انٹرفیس سافٹ ویئر

تمل زیادہ تر ورڈ پروسیسرس میں دستاویزات ٹائپ کرنے اور انٹرنیٹ پر معلومات تلاش کرنے کے لیے استعمال ہوتا ہے۔ تمل انٹرفیس سافٹ ویئر کا استعمال تمل ٹائپنگ کے مختلف طریقوں میں سے ایک مانوس طریقہ ہے۔ یہ کمپیوٹر اور سمارٹ فون دونوں میں تمل ٹائپ کرنے کا آسان ترین طریقہ ہے۔

18.6.1 مانوس تمل کی بورڈ انٹرفیس:

- E-Kalappai، NHM writer اور Lippikar- مانوس تمل کی بورڈ انٹرفیس سافٹ ویئر ہیں جو تمل ٹائپنگ کے لیے استعمال ہوتے ہیں جو کہ فونیٹکس (phonetics) کا استعمال کرتے ہوئے تمل یونیکوڈ پر کام کرتے ہیں۔
- سیلینم اور پونماڈل (Sellinam and Ponmadal) - مانوس تمل کی بورڈ لے آؤٹ ہیں جو فونیٹکس کا استعمال کرتے ہوئے سمارٹ فون میں اینڈرائیڈ آپریٹنگ سسٹم پر کام کرتے ہیں۔



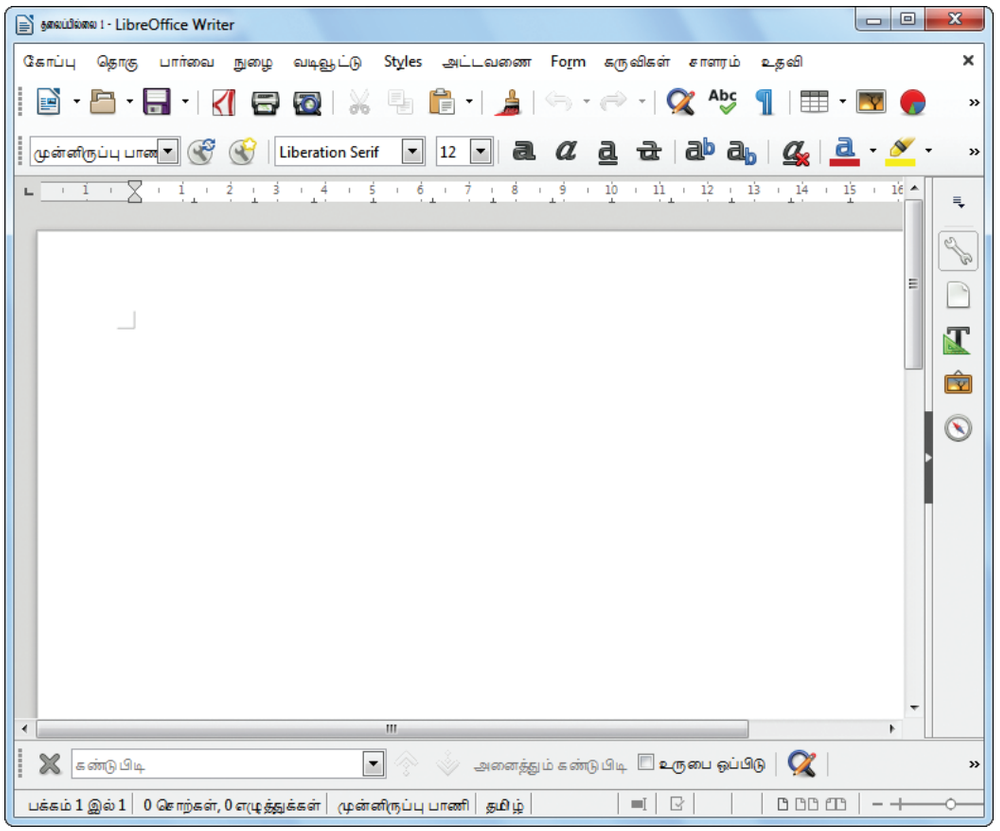
تصویر 18.4 eKalappai کھلنے والی اسکرین

18.7 تمل آفس آٹومیشن ایپلی کیشنز
(Tamil office Automation Applications)

مشہور آفس آٹومیشن سافٹ ویئر جیسے مائیکروسافٹ آفس، اوپن آفس وغیرہ، مکمل تمل انٹرفیس کی سہولت فراہم کرتا ہے۔یہ سافٹ وئیر آپ کے کمپیوٹر میں ڈاؤن لوڈ اور انسٹال کیے جا سکتے ہیں۔ تنصیب کے بعد، آپ کے آفس آٹومیشن سوفٹ ویئر کا ماحول مکمل طور پر تمل میں بدل

11th_Comp Science_UM_Ch 18.indd 291 17-04-2023 14:53:55

جائے گا۔ مینو بارز، آئکن کے نام، ڈائیلاگ باکس تمل میں دکھائے جائیں گے۔ مزید یہ کہ آپ تمل ناموں والی فائلوں کو محفوظ کر سکتے ہیں اور تمل ناموں کے ساتھ فولڈر بنا سکتے ہیں۔



تصویر 18.5 تمل میں لیبرا آفس رائٹر کے ماحول

اس کے علاوہ تمل لیبرا آفس (Tamil Libra Office)، تمل اوپن آفس (Tamil open Office)، اژہگی یونیکوڈ ایڈیٹر (Azhagi Unicode Editor)، پونموزی (Ponmozhi)، مینتھامیز (Menthamiz)، کمبن (Kamban)، وانی (Vani) آفس آٹومیشن سافٹ ویئر ہیں جو خصوصی طور پر تمل کے لیے کام کر رہے ہیں۔ آپ ان ایپلی کیشنز کو مکمل طور پر تمل میں کام کرنے کے لیے استعمال کر سکتے ہیں۔

18.8 تمل ترجمے کی ایپلی کیشنز

تمیزپوری (தமிழ்பொறி) ایک تمل ترجمے کی ایپلی کیشن ہے جس میں 30000 سے زیادہ تمل الفاظ انگریزی الفاظ کے برابر ہیں۔ اس ایپلی کیشن کا استعمال کرتے ہوئے، ہم انگریزی کے چھوٹے چھوٹے جملوں کا تمل میں ترجمہ کر سکتے ہیں۔ گوگل آن لائن ترجمہ کی سہولت بھی دیتا ہے۔

18.9 تمل پروگرامنگ لینگویج

کمپیوٹر اور سمارٹ فونز میں سافٹ ویئر تیار کرنے کے لیے پروگرامنگ زبانیں صرف انگریزی میں دستیاب ہیں۔ پائیتھان (python) پروگرامنگ زبان کی بنیاد پر، پہلی تمل پروگرامنگ لینگویج " Ezhil (எழில்)" کو ڈیزائن کیا گیا ہے۔ اس پروگرامنگ لنگویج کی مدد سے آپ تمل زبان میں سادہ پروگرام لکھ سکتے ہیں۔

11th_Comp Science_UM_Ch 18.indd 292 17-04-2023 14:53:55

18.10 تمل انفارمیشن انٹرچینج کوڈنگ سسٹمز
(Tamil Information interchange Coding Systems)

TSCII (تمل اسکرپٹ کوڈ برائے تبادلۂ معلومات – Tamil Script Code for Information interchange)

کمپیوٹر ڈیٹا اور معلومات کو بائنری سسٹم کے طور پر ہینڈل کرتے ہیں۔ کمپیوٹر ڈیٹا اور معلومات کو سنبھالنے کے لیے ASCII انکوڈنگ سسٹم کا استعمال کرتے ہیں۔ ASCII انکوڈنگ سسٹم صرف انگریزی زبان کو سنبھالنے کے لیے لاگو ہوتا ہے لہذا، TSCII (Tamil Script Code for Information Interchange) ہماری تمل زبان کو سنبھالنے والا پہلا کوڈنگ سسٹم ہے۔ یہ انکوڈنگ اسکیم ICANN کی اکائی IANA (انٹرنیٹ اسائنڈ نمبرس اتھارٹی – Internet assigned Numbers Authority) میں رجسٹرڈ تھی۔

ISCII (انڈین اسکرپٹ کوڈ برائے تبادلۂ معلومات – Indian Script Code for Information Interchange)

یہ انکوڈنگ اسکیموں میں سے ایک ہے جو خاص طور پر تمل سمیت ہندوستانی زبانوں کے لیے بنائی گئی ہے۔ اسے یونیکوڈ کے ساتھ ملایا گیا تھا۔

یونیکوڈ (Unicode):

یونی کوڈ ایک انکوڈنگ سسٹم ہے، جسے تمل سمیت مختلف عالمی زبانوں کو سنبھالنے کے لیے ڈیزائن کیا گیا ہے۔ اس کا پہلا ورژن 1.0.0 ،اکتوبر 1991 میں متعارف کرایا گیا تھا۔ اس اسکیم کو متعارف کرانے کے دوران، تمل سمیت تقریباً 23 زبانوں کو سنبھالا جا سکتا ہے۔ مختلف انکوڈنگ اسکیموں میں سے، یونی کوڈ تامل کو سنبھالنے کے لیے موزوں ہے۔

18.11 تمل آپریٹنگ سسٹم

کمپیوٹر اور سمارٹ فون جیسے الیکٹرانک سسٹم تک رسائی کے لیے آپریٹنگ سسٹم کی ضرورت ہے۔ مائیکروسافٹ ونڈوز پرسنل کمپیوٹرز کے لیے بہت مقبول آپریٹنگ سسٹم ہے۔ لینکس ایک اور مقبول اوپن سورس آپریٹنگ سسٹم ہے۔ آپریٹنگ سسٹم آسانی سے کمپیوٹر تک رسائی کے لیے استعمال ہوتے ہیں۔ ایک آپریٹنگ سسٹم کام کرنے میں آسان ہونا چاہیے اور اس کا ماحول قابل فہم شکل میں ہونا چاہیے۔ اس طرح، کمپیوٹر اور سمارٹ فونز میں استعمال ہونے والے تمام آپریٹنگ سسٹم تمل ماحول میں پیش کیے جاتے ہیں۔

ونڈوز تمل انوائرونمینٹ انٹرفیس (Windows Tamil Environment Interface) انٹرنیٹ سے ڈاؤن لوڈ اور انسٹال کیا جانا چاہیے۔ یہ ونڈو کے تمام عناصر جیسے ٹاسک بار، ڈیسک ٹاپ عناصر، آئیکنس کے نام، تمل میں کمانڈز دکھاتا ہے۔

18.12 تمل کی ترقی کے لیے تنظیم اور منصوبے

تمل ورچوئل اکیڈمی:

انٹرنیٹ کے ذریعے تمل کو پوری دنیا میں پھیلانے کے مقاصد کے ساتھ، تمل ورچوئل یونیورسٹی کا قیام 17 فروری 2001 کو حکومت تمل ناڈو نے کیا تھا۔ تمل ناڈو کے اب یہ تنظیم "Tamil Virtual Academy" کے نام سے کام کرتی ہے۔ یہ کنڈرگارٹن(kindergarten) سے لے کر انڈر گریجویشن کی سطح تک تمل زبان، ثقافت، ورثہ وغیرہ میں مختلف کورسز پیش کرتا ہے۔

ویب سائٹ : Website: http://www.tamilvu.org/index.php

11th_Comp Science_UM_Ch 18.indd 293 17-04-2023 14:53:55

تمل لینگویج کونسل، سنگاپور (Tamil Language Council, Singapore)

سنگاپور کے باشندوں میں تمل کے زیادہ استعمال اور بیداری کو فروغ دینے کے مقاصد کے ساتھ، 2001 میں تمل کی کونسل زبان کو وزارت اطلاعات مواصلات اور فنون، حکومت سنگاپورنے تشکیل دیا تھا۔ اس کونسل کو "**வளர்தமிழ் இயக்கம்**". کہا جاتا ہے۔



ویب سائٹ: http://tamil.org.sg/ta

مدورائی پروجیکٹ (Madurai Project)

پروجیکٹ مدورائی قدیم تمل ادبی کلاسک کے مفت الیکٹرانک ایڈیشن جمع کرنے اور شائع کرنے کے لیے ایک کھلا اور رضاکارانہ اقدام ہے۔ اس کا مطلب ہے کہ یا تو ٹائپ کرنا یا پرانی کتابوں کو اسکین کرنا اور متن کو جمع و محفوظ کرنا تمام مقبول کمپیوٹر پلیٹ فارمز پر استعمال کے لیے سب سے زیادہ آسانی سے قابل رسائی فارمیٹس میں سے ایک ہے۔

1998 میں اس کے آغاز کے بعد سے، جاری کردہ پروجیکٹ مدورائی ای ٹیکسٹ TSCII انکوڈنگ کے مطابق تمل اسکرپٹ کی شکل میں ہیں۔ 2004 کے بعد سے ہم نے تمل یونیکوڈ میں بھی متن جاری کرنا شروع کر دیا ہے ۔

ویب سائٹ: http://www.projectmadurai.org/

تمل ویکیپیڈیا: (Tamil Wikipedia)

ویکیپیڈیا (Wikipedia) ایک اوپن سورس انسائیکلو پیڈیا (Encyclopedia) ہے جہاں کوئی بھی شخص کسی بھی موضوع پر مضمون لکھ سکتا ہے۔ تمل ویکیپیڈیا میں ایک لاکھ سے زیادہ مضامین ہیں۔

ویب سائٹ: https://ta.wikipedia.org/

تمل کو زندہ زبان بنانے کے لیے، یہ ہر تمل باشندوں (Tamilian) کا فرض ہے کہ وہ ٹیکنالوجی کی ترقی میں تمل کو فعال طور پر استعمال کرے۔

11th_Comp Science_UM_Ch 18.indd 294 17-04-2023 14:53:55

یاد رکھنے کے نکات:

- تمل 2016 کے آخر تک ہندوستان میں سب سے زیادہ استعمال ہونے والی علاقائی زبانوں کی فہرست میں سرفہرست ہے، ان میں سے 42٪ انٹرنیٹ استعمال کر رہے ہیں۔
- گوگل اور بنگ تمل میں تلاش کی سہولیات فراہم کرتے ہیں۔
- انٹرنیٹ کے ذریعے سرکاری خدمات حاصل کرنا ای گورننس کے نام سے جانا جاتا ہے۔
- تمل ای-لائبریری خدمات ہزاروں تمل کتابیں زیادہ تر مفت بطور ای بکس فراہم کرتی ہیں ۔
- تمل پوری (தமிழ்பொறி) ایک تمل ترجے کی ایپلی کیشن ہے جس میں 30000 سے زیادہ تمل الفاظ انگریزی الفاظ کے برابر ہیں۔
- پہلی تمل پروگرامنگ زبان ہے "(எழில்)" Ezhil
- یونیکوڈ ایک انکوڈنگ سسٹم ہے، جسے تمل سمیت مختلف عالمی زبانوں کو سنبھالنے کے لیے ڈیزائن کیا گیا ہے۔
- ونڈوز تمل انوائرونمینٹ انٹرفیس (Windows Tamil Environment Interface) انٹرنیٹ سے ڈاؤن لوڈ اور انسٹال ہونا چاہیے۔

تشخیص

بہت ہی مختصر جوابات دیں ۔

1. تمل زبان سے تعاون یافتہ سرچ انجنس کی فہرست بنائیں۔
2. اینڈرائیڈ میں استعمال ہونے والے کی بورڈ لے آؤٹ کیا ہیں؟
3. تمل پروگرامنگ لینگویج کے بارے میں ایک مختصر نوٹ لکھیں۔
4. TSCII کیا ہے؟
5. تمل ورچوئل اکیڈمی پر ایک مختصر نوٹ لکھیں۔

WZ4U0

****

11th_Comp Science_UM_Ch 18.indd 295 17-04-2023 14:53:55



لغات

الفاظ	معانی
پیراڈائم (Paradigm)	پروگرام کے تنظیمی اصول۔
ابسٹریکشن (Abstraction)	ابسٹریکشن سے مراد پس منظر کی تفصیلات کوچھپائے کے بغیر صرف ضروری خصوصیات کو ظاہر کرنا ہے۔
ماڈولائریٹی (Modularity)	ایک ایسا نظام ڈیزائن کرنا جو فنکشنل یونٹس کے سیٹ میں تقسیم ہو۔(نامزد ماڈیولز) جو ایک بڑی ایپلی کیشن میں تشکیل دیے جا سکتے ہیں۔
بیس کلاس (Base Class)	ایک کلاس جس کی خصوصیات دوسری نئی تخلیق شدہ کلاسوں سے ملتی ہیں۔اسے پیرنٹ کلاس بھی کہا جاتا ہے۔
ڈرائیوڈ کلاس (Derived Class)	ایک کلاس جو بیس کلاس کی خصوصیات کو حاصل کرتاہے اسے چائلڈ کلاس یا سب کلاس بھی کہا جاتا ہے۔
کلاس (Class)	کلاس اسی طرح کی آبجیکٹ کے ایک گروپ کی نمائندگی کرتی ہے جو مشترکہ خصوصیات کا اشتراک کرتی ہے۔
آبجیکٹ (Object)	کچھ خصوصیات اور طرز عمل کے ساتھ آبجیکٹ قابل شناخت ہستی ہے۔
انکیپسولیشن (Encapsulation)	انکیپسولیشن میکانزم جس کے ذریعے ڈیٹا اور فنکشن ایک ساتھ ایک ہی یونٹ میں جڑے ہوئے ہیں
انہیریٹنس (Inheritance)	انہیریٹنس کا عمل موجودہ کلاس اور بیس کلاس سے نئی کلاس بنانے کا عمل ہے، جسے ڈرائیوڈ کلاس کہتے ہیں۔
سگنیچر (Signature)	آرگیومنٹ کی تعداد اور آرگیو منٹ کی قسم ۔
پولیمورفیزم (Polymorphism)	بہت سی شکلیں۔
ڈیفالٹ آرگیومنٹ (Default Argument)	ایک قدر کے ساتھ آرگیو منٹ شروع کرنا۔
بیس کلاس: (Base Class :)	ایک کلاس جس سے دوسری کلاس ملتی ہے (جسے سپر کلاس یا پیرنٹ کلاس بھی کہا جاتا ہے)
ڈرائیوڈ کلاس: (Derived Class :)	ایک کلاس جو دوسری کلاس سے خصوصیات حاصل کرتی ہیں۔ (جسے سب کلاس بھی کہا جاتا ہے)
انہیریٹنس (Inheritance)	ایک کلاس کا دوسرے کلاس سے خصوصیات حاصل کرنے کا عمل ۔
انہیری ٹنس (Inheritance) ہائیر آرکی (Hierarchy) (انہیری ٹنس پاتھ)	ایک بیس کلاس اور ڈرائیوڈ کلاس کے درمیان تعلق کو ظاہر کرنے والی زنجیر (جسے ڈیریویشن ہائیر آرک (derivation hierarchy) ی بھی کہا جاتا ہے)
ویزیبلٹی موڈ (Visibility mode)	پبلک (عوامی)، پرائیویٹ (نجی) یا پروٹیکٹیڈ (محفوظ) کی وضاحت کرنے والی جو کلاس میں کسی ممبر کی مرئیت اور دستیابی کو کنٹرول کرتا ہے۔

11th_Comp Science_UM_Ch 18.indd 296 17-04-2023 14:53:55

ولنرایبلٹی (Vulnerability)	حملہ یا نقصان پہنچنے کا امکان۔
ایتھکس(Ethics)	اخلاقی اصول جو کسی شخص کے طرز عمل یا کسی سرگرمی کے انعقاد کو کنٹرول کرتے ہیں۔
سائبر(cyber)	کمپیوٹر، انفارمیشن ٹیکنالوجی کی ثقافت کی خصوصیت اور مجازی حقیقت.
کمپیوٹر کرائم (Computer crime)	کمپیوٹر کرائم کمپیوٹر سسٹم میں ہیرا پھیری کرنا ایک فکری جرم ہے۔
آتھینٹی سٹی (Authenticity)	حقیقی یا حقیقی ہونے کا معیار۔
سبوٹاج (Sabotage)	جان بوجھ کر تباہ، نقصان، یا رکاوٹ پیدا کرنا۔
پرپیٹریٹر(Perpetrator)	وہ شخص جو نقصان دہ، غیر قانونی، یا غیر اخلاقی کام کرتا ہے۔
سافٹ ویئر پائریسی (Software Piracy)	سافٹ ویئر پائریسی سافٹ ویئر کی کاپی رائٹ کی خلاف ورزی ہے جو اصل میں ایک شخص کے ذریعہ بنایا کیا گیا ہے اور غیر قانونی طور پر کسی اور کے ذریعہ استعمال کیا گیا ہے۔
ہیکنگ (Hacking)	ہیکنگ مالک کی اجازت کے بغیر کمپیوٹر سسٹم تک غیر مجاز رسائی حاصل کر نا۔
کریکنگ (Cracking)	کریکنگ کمپیوٹر سسٹم تک غیر مجاز رسائی حاصل کر نا، کسی جرم کا ارتکاب کرنا، جیسے کاپی سے محفوظ پروگرام چلانے کے لیے کوڈ کو چوری کرنا، اس طرح جائز صارفین کی خدمت سے انکار کرنا۔
میلیشی اَس (Malicious)	جان بوجھ کر نقصان پہنچانا۔
فری ویئر (Freeware)	فری ویئر ایک ایسا سافٹ ویئر ہے جو مفت میں دستیاب ہے۔
شیئر ویئر (Shareware)	شیئر ویئر ایک ایسا سافٹ ویئر ہےجو ایک محدود وقت کے لئے (Trial basis) مفت تقسیم کیا جاتا ہے۔
فشنگ (Phishing)	فشنگ ایک اصطلاح ہے جو کسی بدنیتی پر مبنی فرد یا افراد کے گروپ کو بیان کرنے کے لیے استعمال ہوتی ہے جو صارفین کو ای میل بھیج کر یا ایسے ویب صفحات بنا کر دھوکہ دیتے ہیں جو کسی فرد کے آن لائن بینک، کریڈٹ کارڈ، یا دیگر لاگ ان معلومات کو جمع کرنے کے لیے بنائے گئے ہیں۔
دھوکہ دہی (Fraudulent)	بے ایمان، دھوکہ دہی، لوٹ مار، بدعنوان، مجرم، ناجائز ، غیر قانونی۔
گمنام (Anonymous)	بے نام، بغیر نام کے، نامعلوم، غیر متعینہ۔
کوکیز (Cookies)	کوکیز وہ پیغامات ہیں جو ویب سرورس آپ کے ویب براؤزر کو بھیجتے ہیں، جب آپ انٹرنیٹ سائٹس پر جاتے ہیں۔
چھیڑ چھاڑ (Tampering)	نقصان پہنچانے کے لیے مداخلت کرنا۔
امیون (Immune)	کسی خاص انفیکشن (Infection) یا ٹاکسن (Toxin) کے خلاف مزاحم۔
فائر وال (Firewall)	فائر وال ایک نیٹ ورک سیکیورٹی سسٹم ہے جو پہلے سے طے شدہ سیکیورٹی قوانین کی بنیاد پر آنے والے اور جانے والے نیٹ ورک ٹریفک کی نگرانی اور کنٹرول کرتا ہے۔
پراکسی سرور (Proxy Server)	ایک پراکسی سرور مقامی نیٹ ورک اور انٹرنیٹ جیسے بڑے پیمانے پر نیٹ ورک کے درمیان ایک گیٹ وے ہے۔ پراکسی سرورس بڑھتی ہوئی کارکردگی اور سیکیورٹی فراہم کرتے ہیں۔
انکرپشن (Encryption)	انکرپشن ایک پیغام یا معلومات کو انکوڈنگ کرنے کا عمل ہے تاکہ صرف مجاز صارفین ہی اسے ڈی کرپٹ (decrypt) کر سکیں۔
ڈی کرپشن (Decryption)	ڈی کرپشن انکرپٹیڈ ٹیکسٹ (encrypted text) کو دوبارہ عام ٹیکسٹ میں تبدیل کرکے ڈی کوڈ (decode) کرنے کا عمل ہے۔

11th_Comp Science_UM_Ch 18.indd 297 17-04-2023 14:53:55

11 کمپیوٹر سائنس

عملی مشق (Praticals)

ہدایات:

1. پریکٹیکل کلاسس میں C++ سے دس مشقیں کی جاتی ہیں
2. اندرونی انتخاب کے ساتھ C++ سے ایک سوال
3. مارکس کی تقسیم

I. اندرونی تشخیص:	5 مارکس
ریکارڈ بک	5 مارکس

II بیرونی تشخیص:	15 مارکس
(a) C++ پروگرام کوڈنگ	10 مارکس
(b) عمل درآمد اور آؤٹ پٹ	5 مارکس

کل	**20 مارکس**

11th_Computer Science_ Practical_UM.indd 298 20-04-2023 14:45:47

انڈیکس

Sl. No.	سوال نمبر	پروگرام کا نام	صفحہ نمبر
1	CS1	مجموعی تنخواہ	300
2	CS2	فضید	301
3	CS3	پیلینڈروم (Palindrome)	302
4	CS4	نمبر کی تبدیلی	303
5	CS5	فیبونیکی پرائم سیریز (Fibonacci prime series)	305
6	CS6	ارے (array) میں عناصر کو داخل یا حذف کرنا	307
7	CS7	میٹرکس کا باؤنڈری عنصر	310
8	CS8	ABC پبلشرس	312
9	CS9	کلاس کا استعمال کرتے ہوئے ملازم کی تفصیلات	316
10	CS10	طالب علم کی تفصیلات	318

11th_Computer Science_ Practical_UM.indd 299 20-04-2023 14:45:47

CS-1 کسی ملازم کی بنیادی تنخواہ داخل کرنے کے لیے C++ پروگرام لکھیں اور اس کی مجموعی تنخواہ کا حساب درج ذیل کے مطابق کریں:

Basic Salary <25000 : HRA = 20%, DA = 80%
Basic Salary >= 25000 : HRA = 25%, DA = 90%
Basic Salary >= 40000 : HRA = 30%, DA = 95%

کوڈنگ

```
#include <iostream>
using namespace std;
int main()
{
float basic, gross, da,hra;
cout<<"Enter basic salary of an employee: ";
cin>>basic;
if (basic <25000)
{
da = basic *80/100;
hra= basic *20/100;
}
else if (basic >=25000 && basic<40000)
{
da = basic *90/100;
hra= basic *25/100;
}
else if (basic>=40000)
{
da = basic *95/100;
hra= basic *30/100;
}
gross= basic +hra+ da;
cout<< "\n\t Basic Pay ....."<< basic<<endl;
cout<< "\t Dearness Allowance ....." << da <<endl;
cout<< "\t House Rent Allowance....."<< hra <<endl;
  cout<< "\t ----------------------------------------"<<endl;
  cout<< "\t Gross Salary....."<<gross <<endl;
```


11th_Computer Science_ Practical_UM.indd 300 20-04-2023 14:45:47

```
cout<< "\t ----------------------------------------"<<endl;
return 0;
}
```

آوٹ پٹ

```
Enter basic salary of an employee: 25000
                Basic Pay       : 25000
        Dearness Allowance      : 22500
    House Rent Allowance        :   6250
                        ------------------------
            Gross Salary        : 53750
                        ------------------------
```

فہید – CS2

CS-2 ایک طالب علم کا فیصد چیک کرنے کے لیے ایک C++ پروگرام لکھیں اور سوئچ کیس کا استعمال کرتے ہوئے اسکور کیے گئے ڈویژن (ممتاز، فرسٹ، سکینڈ، تھرڈ اور فیل) دکھائیں۔

Percentage	Division
>=80	Distinction
>=60 and <80	First division
>=50 and <60	Second Division
>=40 and <50	Third Division
<40	Fail

کوڈنگ

```
#include <iostream>
using namespace std;
int main()
{
float percent;
int x;
cout<<"Enter your percentage: ";
cin>>percent;
cout<<"You scored "<<percent<<"%"<<endl;
   x = percent/10;
switch (x)
{
case 10:
case 9:
```


11th_Computer Science_ Practical_UM.indd 301 20-04-2023 14:45:47

```
case 8:
cout<<"You have passed with Distinction";
break;
case 7:
case 6:
cout<<"You have passed with First division";
break;
case 5:
cout<<"You have passed with Second division";
break;
case 4:
cout<<"You have passed with Third division";
break;
default:
cout<<"Sorry: You have failed";
}
return 0;
}
```

آوٹ پٹ 1

```
Enter your percentage: 79
You scored 79%
You have passed with First division
```

آوٹ پٹ 2

```
Enter your percentage: 39
You scored 39%
Sorry: You have failed
```

CS3 - پیلنڈروم (Palindrome)

CS-3 کوئی بھی نمبر درج کرنے کے لیے C++ پروگرام لکھیں اور وائل لوپ کا استعمال کرتے ہوئے چیک کریں کہ آیا نمبر پلینڈروم ہے یا نہیں ۔

کوڈنگ

```
#include <iostream>
using namespace std;
int main()
{
int n,num, digit, rev =0;
cout<<"Enter a positive number: ";
```


11th_Computer Science_ Practical_UM.indd 302 20-04-2023 14:45:47

```
cin>>num;
    n =num;
while (num)
{
digit=num%10;
rev=(rev *10)+ digit;
num=num/10;
}
cout<<" The reverse of the number is: "<< rev <<endl;
if (n == rev)
cout<<" The number is a palindrome";
else
cout<<" The number is not a palindrome";
return 0;
}
```

آوٹ پٹ 1

```
Enter a positive number to reverse: 1234
 The reverse of the number is: 4321
 The number is not a palindrome
```

آوٹ پٹ 2

```
Enter a positive number to reverse: 1221
The reverse of the number is: 1221
The number is a palindrome
```

CS4 - نمبر کی تبدیلی

CS-4 ڈو وائل لوپ (do while loop) کا استعمال کرتے ہوئے درج ذیل مینو پر مبنی C++ پروگرام بنائیں.

1. ایک اعشاریہ کو بائنری نمبر میں تبدیل کریں
2. ایک بائنری نمبر کو اعشاریہ میں تبدیل کریں
3. باہر نکلیں۔

انتخاب پر منحصر ہے کہ قیمت کو قبول کریں اور نتیجہ ظاہر کریں ۔ پروگرام کو اس وقت تک جاری رہنا چاہئے جب تک کہ صارف تیسرے آپشن کا انتخاب نہ کرلے۔

کوڈنگ

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
```


11th_Computer Science_ Practical_UM.indd 303 20-04-2023 14:45:47

```
    int dec,d,i,temp,ch;
    long int bin;
do
{
    dec=bin=d=i=0;
    cout<<"\n\n\t\tMENU\n1. Decimal to Binary number\n2.Binary to Decimal number\
    n3.Exit\n";
    cout <<"Enter your choice(1/2/3)";
    cin>>ch;
    switch (ch)
    {
            case 1:  cout << "Enter a decimal number: ";  cin >> dec;
                    temp=dec;
    while (dec!=0)
    {
    d = dec%2;
    bin += d * pow(10,i);
    dec /= 2;
    i++;
}
    cout << temp << " in decimal = " << bin << " in binary" << endl ;break;
    case 2: cout << "Enter a binary number: ";    cin >> bin;
            temp=bin;
            while (bin!=0)
    {
    d = bin%10;
    dec += d*pow(2,i);
    bin /= 10;
    i++;
}
    cout << temp << " in binary = " <<dec << " in decimal";
    break;
    case 3: break;
    default : cout<<"Invalid choice";
 }
} while (ch!=3);
return 0;
}
```

آوٹ پٹ 1

```
    MENU
1.Decimal to Binary number
```


11th_Computer Science_ Practical_UM.indd 304 20-04-2023 14:45:47

2.Binary to Decimal number
3.Exit
Enter your choice(1/2/3)1
Enter a decimal number: 23
23 in decimal = 10111 in binary
MENU
1.Decimal to Binary number
2.Binary to Decimal number
3.Exit
Enter your choice(1/2/3)2
Enter a binary number: 11001
11001 in binary = 25 in decimal
MENU
1.Decimal to Binary number
2.Binary to Decimal number
3.Exit
Enter your choice(1/2/3)3

آوٹ پٹ 2

MENU
1.Decimal to Binary number
2.Binary to Decimal number
3.Exit
Enter your choice(1/2/3)4
Invalid choice
MENU
1.Decimal to Binary number
2.Binary to Decimal number
3.Exit
Enter your choice(1/2/3)3

CS5 – فیبو نیکی پرائم سیریز (Fibonacci Prime Series)

CS-5 فیبو نیکی سیریز کو n شرائط تک تیار کرنے کے لیے صارف کے طے شدہ فنکشن کا استعمال کرتے ہوئے C++ پروگرام لکھیں اور اگر ہر اصطلاح پرائم(Prime) یا کمپوزٹ (Composite) ہو تو پرنٹ کریں۔

کوڈنگ

```
#include <iostream>
#include <stdlib.h>
```


11th_Computer Science_ Practical_UM.indd 305 20-04-2023 14:45:47

```
using namespace std;
void Primechk (int a )
{ int j;
    if ( a == 0 || a == 1 )
   { cout<< "  NEITHER PRIME NOR COMPOSITE ";}
    else
{
   for (j = 2 ; j<a; j++)
   {         if (a%j==0)
   { cout<< "\tCOMPOSITE" ;
   break ;
}
   }
    if ( a==j )
    cout<< "\tPRIME" ;
   }
}
   void  fibo ( int n )
   { int a = -1 , b = 1 ,c=0 ;
   for ( int i = 1 ; i <= n ; i++)
   {
            cout<<endl;
   c = a + b ;
   cout<<c;
   Primechk(c);
   a = b;
   b = c ;
    }
   }
 int main ()
 {
   int n ;
   cout << " ENTER THE NUMBER OF REQUIRED FIBO TERMS " ;
   cin >> n ;
   cout<< "\n\tFIBONACCI SERIES\n " ;
   fibo (n) ;
return 0;
  }
```


11th_Computer Science_ Practical_UM.indd 306 20-04-2023 14:45:47

آؤٹ پٹ

```
ENTER THE NUMBER OF TERMS 10
   FIBONACCI SERIES
0  NEITHER PRIME NOR COMPOSITE
1  NEITHER PRIME NOR COMPOSITE
1  NEITHER PRIME NOR COMPOSITE
2       PRIME
3       PRIME
5       PRIME
8       COMPOSITE
13      PRIME
21      COMPOSITE
34      COMPOSITE
```

CS6 - ارے(array) میں عناصر کو داخل یا حذف کرنا

CS-6 انٹیجرس کی ایک ہی جہت والے ارے میں عناصر کو داخل اور حذف کرنے کے لیے مینو سے چلنے والا C++ پروگرام لکھیں اور داخل کرنے یا حذف کرنے کے بعد ارے (array) پرنٹ کریں۔

کوڈنگ

```
#include<iostream>
using namespace std;
int a[20],b[20],c[40];
int m,n,p,val,i,j,key,pos,temp;
/*Function Prototype*/
void display();
void insert();
void del();
int main()
{
int choice;
cout<<"\nEnter the size of the array elements:\t";
cin>>n;
cout<<"\nEnter the elements for the array:\n";
for (i=0;i<n;i++)
```


11th_Computer Science_ Practical_UM.indd 307 20-04-2023 14:45:47

```
{
cin>>a[i];
}
do {
cout<<"\n\n--------Menu----------\n";
cout<<"1.Insert\n";
cout<<"2.Delete\n";
cout<<"3.Exit\n";
cout<<"----------------------";
cout<<"\nEnter your choice:\t";
cin>>choice;
switch (choice)
{
   case 1: insert();
   break;
   case 2: del();
   break;
   case 3:break;
   default :cout<<"\nInvalid choice:\n";
}
} while (choice!=3);
return 0;
}
void display()//displaying an array elements
{
   int i;
   cout<<"\nThe array elements are:\n";
   for(i=0;i<n;i++)
{
   cout<<a[i]<<" ";
}
}//end of display()
   void insert()//inserting an element in to an array
{
   cout<<"\nEnter the position for the new element:\t";
   cin>>pos;
```


11th_Computer Science_ Practical_UM.indd 308 20-04-2023 14:45:47

```
    cout<<"\nEnter the element to be inserted :\t";
    cin>>val;
    for (i=n; i>=pos-1; i--)
{
    a[i+1]=a[i];
}
a[pos-1]=val;
       n=n+1;
display();
}//end of insert()
void del()//deleting an array element
{
    cout<<"\n Enter the position of the element to be deleted:\t";
    cin>> pos;
    val= a [pos];
    for (i= pos;i<n-1;i++)
{
    a[i]=a[i+1];
}
    n=n-1;
cout<<"\nThe deleted element is = "<<val;
display();
}//end of delete()
```

آؤٹ پٹ

```
Enter the size of the array elements:   5
Enter the elements for the array:
1
2
3
4
5
--------Menu-----------
1.Insert
2.Delete
3.Exit
-----------------------
```


11th_Computer Science_ Practical_UM.indd 309 20-04-2023 14:45:47

```
Enter your choice:      1
Enter the position for the new element: 3
Enter the element to be inserted :      26
The array elements are:
1  2  26  3  4  5
--------Menu-----------
1.Insert
2.Delete
3.Exit
----------------------
Enter your choice:      2
Enter the position of the element to be deleted:        2
The deleted element is = 2
The array elements are:
1 3 26 4 5
--------Menu-----------
1.Insert
2.Delete
3.Exit
--------------------------
Enter your choice:      3
--------------------------
```

CS 7 - میٹرکس کا باؤنڈری عنصر

CS-7 میٹرکس کے باؤنڈری عناصر کو پرنٹ کرنے کے لیے C++ پروگرام لکھیں۔

کوڈنگ

```
#include <iostream>
using namespace std;
void printBoundary (int a[][10], int m, int n)
{
for (int i = 0; i < m; i++) {
for (int j = 0; j < n; j++)
{
if (i==0|| j==0||i==m-1||j==n-1)
```


11th_Computer Science_ Practical_UM.indd 310 20-04-2023 14:45:47

```
cout<<a[i][j]<<" ";
else
cout<<"  ";
}
cout <<endl ;
}
}
// Driver code
int main()
{
   int a[10][10] ,i,j,m,n;
   cout<<"Enter more than 3 number of rows and columns"<<endl;
   cin>>m>>n;
            for (i=0;i<m;i++)
   {
            for (j=0;j<n;j++)
   {
            cout<<"enter the value for array["<<i+1<<"]"<<"["<<j+1<<"] :";
            cin>>a[i][j];
    }
}
system("cls");
   cout<<"\n\nOriginal Array\n";
   for (i=0;i<m;i++)
{
   for (j=0;j<n;j++)
   {
            cout<<a[i][j]<<" ";
    }
   cout<<endl;
}
   cout<<"\n\n The Boundry element\n";
   printBoundary(a, m, n);
return 0;
}
```

آؤٹ پٹ

Enter more than 3 number of rows and columns

4 4

11th_Computer Science_ Practical_UM.indd 311 20-04-2023 14:45:47

enter the value for array[1][1] :1
enter the value for array[1][2] :2
enter the value for array[1][3] :3
enter the value for array[1][4] :4
enter the value for array[2][1] :5
enter the value for array[2][2] :6
enter the value for array[2][3] :7
enter the value for array[2][4] :8
enter the value for array[3][1] :9
enter the value for array[3][2] :0
enter the value for array[3][3] :1
enter the value for array[3][4] :2
enter the value for array[4][1] :3
enter the value for array[4][2] :4
enter the value for array[4][3] :5
enter the value for array[4][4] :6

Original Array

1 2 3 4
5 6 7 8
9 0 1 2
3 4 5 6

The Boundary element

1 2 3 4
5 8
9 2
3 4 5 6

CS-8 درج ذیل وضاحتوں کے ساتھ C++ میں پبلشر نامی کلاس کی وضاحت کریں۔

نجی اراکین

بک نمبر انٹیجر

ٹائٹل 20 حروف

11th_Computer Science_ Practical_UM.indd 312 20-04-2023 14:45:47

مصنف 10 حروف

قیمت فلوٹ

کل اماونٹ فلوٹ

کاپیوں کی تعداد اور قیمت کا حساب لگانے اور کل رقم واپس کرنے کے لیے ()Calculate نامی ممبر فنکشن کی وضاحت کریں۔

عوامی اراکین

- تمام ڈیٹا ممبرس کو شروع کرنے کے لیے ایک ڈیفالٹ کنسٹرکٹر فنکشن۔ کتاب کا نمبر خود بخود 1001 سے تیار ہونا چاہیے۔
- ()Readdata فنکشن عنوان، مصنف اور قیمت کے لیے اقدار کو قبول کرنے کے لیے صارف سے کاپیوں کی تعداد حاصل کریں اور ()Calculate کی درخواست کریں۔
- ()Display data فنکشن تمام ڈیٹا ممبرس کو درج ذیل فارمیٹ میں دکھاتا ہے۔

```
                    ABC PUBLISHERS
                    ~~~~~~~~~~~~~~~~~~
                        INVOICE
                        ~~~~~~~~~
==================================
Book Number         :
Title           :
Author Name         :
Price Per Book:
Total Amount        :
==================================
```

کوڈنگ

```
#include<iostream>
#include<stdlib.h>
using namespace std;
int id=1001;
class Publisher
{
int Bookno;
char Title[20];
char Author [10];
float Price;
float Totamt;
float calculate (int);
public:
Publisher()
```


11th_Computer Science_ Practical_UM.indd 313 20-04-2023 14:45:47

```
{Bookno=id;
Title[0]='\0';
Author[0]='\0';
     Price=0;
Totamt=0;
id++;
}
void Readdata();
void Displaydata();
};
void Publisher::Readdata()
{
int nocopies;
cout<<"Enter the Title name ";cin>>Title;
cout<<"Enter the Author name ";cin>>Author;
cout<<"Enter the Price ";cin>>Price;
cout<<"Enter the Number of copies ";cin>>nocopies;
Totamt=calculate(nocopies);
}
float Publisher::calculate(int x)
{
   return x*Price;
}
void Publisher::Displaydata()
{
cout<<"\n\t\tABC PUBLISHERS\n";
cout<<"\t\t~~~~~~~~~~~~~~~\n";
cout<<"\t\t  INVOICE\n";
cout<<"\t\t  ~~~~~~~\n";
cout<<"\n==================================\n";
cout<<" Book Number : "<<Bookno<<endl;
cout<<"Title                    : "<<Title<<endl;
cout<<"Author Name   : "<<Author<<endl;
cout<<"Price Per Book : "<<Price<<endl;
cout<<"Total Amount  : "<<Totamt<<endl;
cout<<"\n==================================\n";
```


11th_Computer Science_ Practical_UM.indd 314 20-04-2023 14:45:47

```
}
int main()
{
   int n,i;
   Publisher p[10];
   cout<<"Enter the number of object to be created";cin>>n;
   for (i=0;i<n;i++)
   p[i].Readdata();
   for (i=0;i<n;i++)
   p[i].Displaydata();
   return 0;
}
```

آؤٹ پٹ

```
Enter the number of object to be created2
Enter the Title name C++Programming
Enter the Author name Balaguru
Enter the Price 500
Enter the Number of copies 3
Enter the Title name CoreJava
Enter the Author name Xavier
Enter the Price 250
Enter the Number of copies 5
                                    ABC PUBLISHERS
                                    ~~~~~~~~~~~~~~~
                                        INVOICE
                                        ~~~~~~~
==================================
Book Number    : 1001
Title                  : C++Programming
Author Name            : Balaguru
Price Per Book  : 500
Total Amount    : 1500
==================================
          ABC PUBLISHERS
          ~~~~~~~~~~~~~~~
          INVOICE
```


11th_Computer Science_ Practical_UM.indd 315 20-04-2023 14:45:47

~~~~~~~

==========

Book Number : 1002

Title : CoreJava

Author Name : Xavier

Price Per Book : 250

Total Amount : 1250

==========

CS9 – کلاس کا استعمال کرتے ہوئے ملازم کی تفصیلات

CS-9 عوام میں درج ذیل اراکین پر مشتمل کلاس ملازم بنانے کے لیے C++ پروگرام بنائیں۔

(Public members) عوامی اراکین

eno integer

name 20 characters

des 20 characters

member Function

void get() to accept values for all data members

Declare the derived class called Salary which contain the following details.

Public members

bp, hra, da, pf, np float

member Function

void get1() to accept values for bp,hra,da and pf and invoke calculate()

calculate() calculate the np by adding bp,hra,da subtracting pf

display() Display all the details

Create the derived class object and read the number of employees.Call the function get(),get1() for each employee and display the details

کوڈنگ

```
#include<iostream>
using namespace std;
class emp{
public:
int eno;
char name[20], des[20];
void get(){
```


11th_Computer Science_ Practical_UM.indd 316 20-04-2023 14:45:47
~~~~~~~

```
cout<<"Enter the employee number:";
cin>>eno;
cout<<"Enter the employee name:";
cin>>name;
cout<<"Enter the designation:";
cin>>des;
}
};
class salary :public emp
{
float bp,hra, da,pf,np;
public:
void get1()
{
cout<<"Enter the basic pay:";
cin>>bp;
cout<<"Enter the HouseRent Allowance:";
cin>>hra;
cout<<"Enter the Dearness Allowance :";
cin>>da;
cout<<"Enter the Provident Fund:";
cin>>pf;
}
void calculate()
{
np=bp+hra+ da -pf;
}
void display()
   {
cout<<eno<<"\t"<<name<<"\t"<<des<<"\t"<<bp<<"\t"<<hra<<"\t"<<da<<"\t"<<pf<<"\
t"<<np<<"\n";
}
};
int main(){
int i, n;
char ch;
salary s[10];
cout<<"Enter the number of employee:";
cin>>n;
for (i =0; i < n; i++){
```


11th_Computer Science_ Practical_UM.indd 317 20-04-2023 14:45:48

```
s[i].get();
s[i].get1();
s[i].calculate();
}
   cout<<"\n\t\t\tEmployee Details\n";
   cout<<"\ne_no \t e_name\t des \t bp \t hra \t da \t pf \t np \n";
   for (i =0; i < n; i++){
   s[i].display();
}
return 0;
}
```

آؤٹ پٹ

```
Enter the number of employee:2
Enter the employee number:1201
Enter the employee name:Ramkumar
Enter the designation:Engineer
Enter the basic pay:50000
Enter the House Rent Allowance:10000
Enter the Dearness Allowance :5000
Enter the Provident Fund:1000
Enter the employee number:1202
Enter the employee name:Viswanathan
Enter the designation:Engineer-Tech
Enter the basic pay:40000
Enter the House Rent Allowance:9000
Enter the Dearness Allowance :4500
Enter the Provident  Fund:1000
                    Employee Details
e_noe_name                  des                 bp      hra         da      pf      np
1201    Ramkumar    Engineer        50000   10000       5000    1000    64000
1202    Viswanathan Engineer-Tech       40000   9000        4500    1000    52500
```

CS10 – طالب علم کی تفصیلات

CS-10 درج ذیل تفصیلات کے ساتھ اسٹوڈنٹ نامی کلاس بنانے کے لیے ایک C++ پروگرام لکھیں۔

Protected member

Rno integer

Public members

11th_Computer Science_ Practical_UM.indd 318 20-04-2023 14:45:48

void Readno(int); to accept roll number and assign to Rno
void Writeno(); To display Rno.

کلاس ٹیسٹ عام طور پر طلباء کی کلاس سے اخذ کیا گیا ہے جس میں درج ذیل تفصیلات شامل ہیں۔

Protected member
Mark1 float
Mark2 float
Public members
void Readmark(float,float); To accept mark1 and mark2
void Writemark(); To display the marks

درج ذیل تفصیل کے ساتھ اسپورٹس کے نام سے ایک کلاس بنائیں

Protected members
score integer
Public members
void Readscore(int); To accept the score
void Writescore(); To display the score

کلاس کا نتیجہ عوامی طور پر ٹیسٹ اور اسپورٹس کلاس سے اخذ کیا گیا ہے جس میں درج ذیل تفصیلات شامل ہیں۔

Private member
Total float
Public member
void display() assign the sum of mark1 ,mark2,score in total.
invokeWriteno(),Writemark() and Writescore() .Display the total also.

کوڈنگ

```
#include<iostream>
using namespace std;
class Student
{
   protected:
   int Rno;
   public:
   void Readno(int r)
   {
   Rno=r;
   }
   void Writeno()
   {
            cout<<"\nRoll no : "<<Rno;
   }
};
```


11th_Computer Science_ Practical_UM.indd 319 20-04-2023 14:45:48

```
class Test :public Student
{
   protected:
   float Mark1,Mark2;
   public:
   void Readmark (float m1,float m2)
   {
            Mark1=m1;
            Mark2=m2;
   }
void Writemark()
{
cout<<"\n\n\tMarks Obtained\n ";
cout<<"\n Mark1       : "<<Mark1;
cout<<"\n Mark2       : "<<Mark2;
}
};
class Sports
{
protected:
int score;// score = Sports mark
public:
void  Readscore (int s)
{
score=s;
}
void  Writescore()
   {
cout<<"\n Sports Score : "<<score;
   }
};
class Result :public Test,public Sports
{
int Total;
public:
void display()
{
   Total = Mark1 + Mark2 + score;
Writeno();
Writemark();
```


11th_Computer Science_ Practical_UM.indd 320 20-04-2023 14:45:48

```
Writescore();
cout<<"\n\n Total Marks Obtained    : "<< Total<<endl;
}
};
int main()
{
   Result stud1;
stud1.Readno(1201);
stud1.Readmark(93.5,95);
stud1.Readscore(80);
cout<<"\n\t\t\t HYBRID INHERITANCE PROGRAM\n";
stud1.display();
return 0;
}
```

آؤٹ پٹ

```
HYBRID INHERITANCE PROGRAM
Roll no : 1201
      Marks Obtained
 Mark1              : 93.5
 Mark2              : 95
 Sports Score       : 80
 Total Marks Obtained    : 268
```


11th_Computer Science_ Practical_UM.indd 321 20-04-2023 14:45:48

Class 11 - Computer Science - Urdu Medium

Text Book Reviewers and Translators - ریاضی برائے بارھویں جماعت

Academic Co ordinator & Head

Dr. Mrs. S. Shameem,
Deputy Director, SCERT,
Chennai.

Reviewer & Co ordinator

Dr. Mohd. Tayyab Ali
Assistant Professor of Urdu,
The New College, Chennai – 600014

Reviewers and Translators

Mr. Syed Shabbeer Hussain
Assistant Professor of Urdu,
The New College, Chennai – 600014

Mr. Sajid Hussain
Assistant Professor of Urdu,
The New College, Chennai – 600014

Mrs. Rehana Yasmine
Former PG Assistant (Computer Science),
Anjuman Matriculation Higher Secondary School,
Chennai – 600017.

Mrs. Mahmooda Tamkeen
Former Assistant Professor (Computer Science),
Aalim Muhammad Salegh College of Engineering,
Avadi, Chennai - 55

Mr. Mohammed Shoaib Salik
Computer Analyst,
Tata Consultancy Services, Chennai - 97

Mr. Syed Tanveer Hussain
Computer Software engineer,
Wipro, Chennai

Dr. Md. Nayeemur Rahman
Vice Chairman, Tamil Nadu State Urdu Academy
Department of Higher Education
Government of Tamil Nadu

Ghani Mohammed Azhar
General Secretay, Vaniyambadi Muslim Educational Society,
Vaniyambadi

C. Khaiser Ahmed
Secretary and Corresspondent,
Islamiah Women's Arts and Science College, Vaniyambadi.

Patel Mohammed Yousuff
Vice Presient, Vainyambadi Muslim Educationals Society,
Vaniyambadi.

N. Mohammed Nayeem
President, Vaniyambadi Urdu Academy, Vaniyambadi.

Syed Akbar Zahid,
Secretary, Vaniyambadi Urdu Academy, Vaniyambadi.

Prof. P. Tanveer Ahmed
H.O.D., Department of Arabic and Urdu,
Islamiah College (Autonomous), Vaniyambadi.

Dr. M. Sayeeduddin
Assistant Professor, Department of Urdu,
Islamiah College (Autonomous), Vaniyambadi.

Professor Salma
H.O.D, Department of Computer Science,
Islamiah Women's Arts and Science College, Vaniyambadi.

Professor Firza Afreen
Assistant Professor of Computer Science,
Islamiah Women's Arts and Science College, Vaniyambadi.

Layout

B. Yogesh, Chennai

V. Vinoth Kumar, Chennai

R. Manohar, Chennai

Typing

Sajid Hussain
Hammad Graphics, Chennai

Inhouse -QC

V. Sridhar

Coordinator

Ramesh Munisamy

EMIS Technology Team

R.M. Satheesh
State Coordinator Technical,
TN EMIS, Samagra Shiksha.

K.P. Sathya Narayana
IT Consultant,
TN EMIS, Samagra Shikaha.

R. Arun Maruthi Selvan
Technical Project Consultant,
TN EMIS, Samagra Shiksha.

تعلیمی سربراہ اور معاون

ڈاکٹر یس شمیم صاحبہ
ڈپٹی ڈائریکٹر، ایس سی ای آر ٹی، چنئی۔

نظر ثانی اور معاون

ڈاکٹر محمد طیب علی
اسسٹنٹ پروفیسر،شعبہ اردو،نیو کالج، چنئی

مترجمین

سید شبیر حسین
اسسٹنٹ پروفیسر،شعبہ اردو،نیو کالج، چنئی

ساجد حسین
اسسٹنٹ پروفیسر،شعبہ اردو،نیو کالج، چنئی

ریحانہ یاسمین
سابق پی جی اسسٹنٹ،کمپیوٹر سائنس،انجمن میٹریکولیشن، ہائر سکنڈری اسکول، چنئی

محمودہ تمکین
سابق اسسٹنٹ پروفیسر، کمپیوٹر سائنس، عالم محمد صالح انجینئرنگ کالج، چنئی

محمد شعیب سالک
کمپیوٹر انالائسس،ٹاٹا کنسلٹنسی سرویسس،چنئی

سعید تنویر حسین
کمپیوٹر سافٹ انجینئرنگ ، وپرو، چنئی

ڈاکٹر ایم ڈی نعیم الرحمن
وائس چیئرمین ،
تمل ناڈو اسٹیٹ اردو اکیڈمی
محکمہ اعلیٰ تعلیم ، حکومت تمل ناڈو

غنی محمد ازہر
جنرل سکریٹری ، وانمباری مسلم ایجوکیشنل سوسائٹی ، وانمباڑی

سی۔ قیصر احمد
سکریٹری و کرسپانڈنٹ، اسلامیہ ویمنس آرٹس اور سائنس کالج ، وانمباڑی

پٹیل محمد یوسف
نائب صدر، وانمباڑی مسلم ایجوکیشنل سوسائٹی ، والمباڑی

این۔ محمد نعیم صدر
وانمباڑی اردو اکیڈمی، وانمباری

سید اکبر زاہد سکریٹری
والمباڑی اردو اکیڈمی، وانمباڑی

پروفیسر پی۔ تنویر احمد
صدر، شعبہ عربی و اردو، اسلامیہ کالج (خود مختار) وانمباڑی

ڈاکٹر ایم ۔ سعیدالدین
اسسٹنٹ پروفیسر شعبہ اردو، اسلامیہ کالج (خود مختار) وانمباڑی

پروفیسر سلمہ
صدر شعبہ کمپیوٹر سائنس، اسلامیہ ویمنس آرٹس اور سائنس کالج ، وانمباڑی

فرزا آفرین
اسسٹنٹ پروفیسر، شعبہ کمپیوٹر سائنس، اسلامیہ ویمنس آرٹس اور سائنس کالج ، وانمباڑی

This book has been printed on 80 G.S.M. Elegant Maplitho Paper.

Printed by Web Offset at :

11th_Std_Computer Science_Translator_UM.indd 322 18-04-2023 15:38:12