



GOVERNMENT OF TAMILNADU

HIGHER SECONDARY FIRST YEAR

COMPUTER SCIENCE

VOLUME-I

Untouchability is Inhuman and a Crime

A publication under Free Textbook Programme of Government of Tamil Nadu

Department of School Education

Government of Tamil Nadu

First Edition - 2018

NOT FOR SALE

Content Creation



State Council of Educational
Research and Training
© SCERT 2018

Printing & Publishing



Tamil Nadu Textbook and Educational
Services Corporation

www.textbooksonline.tn.nic.in

PREFACE

Human civilization achieved the highest peak with the development of computer known as "Computer era". Literate are those who have the knowledge in using the computer whereas others are considered illiterate inspite of the other degrees obtained.

The growth of the nation at present lies in the hands of the youth, hence the content of this book is prepared in such a way so as to attain utmost knowledge considering the future needs of the youth.

- This book does not require prior knowledge in computer Technology
- Each unit comprises of simple activities and demonstrations which can be done by the teacher and also students.
- Technical terminologies are listed in glossary for easy understanding
- The " Do you know?" boxes enrich the knowledge of reader with additional information
- Workshops are introduced to solve the exercises using software applications
- QR codes are used to link supporting additional materials in digital form

HOW TO USE THE BOOK

How to get connected to QR Code?

- o Download the QR code scanner from the google play store/ apple app store into your smartphone
- o Open the QR code scanner application
- o Once the scanner button in the application is clicked, camera opens and then bring it closer to the QR code in the textbook.
- o Once the camera detects the QR code, a URL appears in the screen. Click the URL and go to the content page.



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.

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)

Table of Contents

Chapter No.	Title	Page
UNIT I -FUNDAMENTALS OF COMPUTER AND WORKING WITH A TYPICAL OPERATING SYSTEMS (WINDOWS & LINUX)		
1	Introduction to Computers	1
2	Number Systems	18
3	Computer Organization	51
4	Theoretical Concepts of Operating System	64
5	Working with typical Operating System	
	Part-I Working with Windows	78
	Part-II Working with Linux	104
UNIT II-ALGORITHMIC PROBLEM SOLVING		
6	Specification and Abstraction	116
7	Composition and Decomposition	130
8	Iteration and recursion	147



E - book



Assessment



DIGI links



Learning Objectives

After learning the concepts in this chapter, the students will be able

- To know about Computers
- To learn about various generations of computer
- To understand the basic operations of computers
- To know the components and their functions.
- To know about booting of a computer



Father of Computer

Charles Babbage is considered to be the father of computer, for his invention and the concept of Analytical Engine in 1837. The Analytical Engine contained an Arithmetic Logic Unit (ALU), basic flow control, and integrated memory; which led to the development of first general-purpose computer concept.



Introduction to Computers

1.1 Introduction to Computers





Computers are seen everywhere around us, in all spheres of life, in the field of education, research, travel and tourism, weather forecasting, social networking, e-commerce etc. Computers have now become an indispensable part of our lives. Computers have revolutionized our lives with their accuracy and speed of performing a job, it is truly remarkable. Today, no organization can function without a computer. In fact, various organizations have become paperless. Computers have evolved over the years from a simple calculating device to high speed portable computers.

The growth of computer industry started with the need for performing fast calculations. The manual method of computing was slow and prone to errors. So, attempts were made to develop fast calculating devices, the journey started from the first known calculating device (Abacus) which has led us today to an extremely high speed calculating devices.

1.2 Generations of Computers

Growth in the computer industry is determined by the development in technology.

Based on various stages of development, computers can be categorized into different generations.

SN	Generation	Period	Main Component used	Merits/Demerits
1	First Generation	1942-1955	 Vacuum tubes	<ul style="list-style-type: none"> • Big in size • Consumed more power • Malfunction due to overheat • Machine Language was used
First Generation Computers - ENIAC , EDVAC , UNIVAC 1 ENIAC weighed about 27 tons, size 8 feet × 100 feet × 3 feet and consumed around 150 watts of power				
2	Second Generation	1955-1964	 Transistors	<ul style="list-style-type: none"> • Smaller compared to First Generation • Generated Less Heat • Consumed less power compared to first generation • Punched cards were used • First operating system was developed - Batch Processing and Multiprogramming Operating System • Machine language as well as Assembly language was used.
Second Generation Computers IBM 1401, IBM 1620, UNIVAC 1108				
3	Third Generation	1964-1975	 Integrated Circuits (IC)	<ul style="list-style-type: none"> • Computers were smaller, faster and more reliable • Consumed less power • High Level Languages were used
Third Generation Computers IBM 360 series, Honeywell 6000 series				
4	Fourth Generation	1975-1980	 Microprocessor Very Large Scale Integrated Circuits (VLSI)	<ul style="list-style-type: none"> • Smaller and Faster • Microcomputer series such as IBM and APPLE were developed • Portable Computers were introduced.



5	Fifth Generation	1980 - till date	 <p>Ultra Large Scale Integration (ULSI)</p>	<ul style="list-style-type: none"> • Parallel Processing • Super conductors • Computers size was drastically reduced. • Can recognize Images and Graphics • Introduction of Artificial Intelligence and Expert Systems • Able to solve high complex problems including decision making and logical reasoning
6	Sixth Generation	In future		<ul style="list-style-type: none"> • Parallel and Distributed computing • Computers have become smarter, faster and smaller • Development of robotics • Natural Language Processing • Development of Voice Recognition Software

Table1.1 Generations of computers



The first digital computer

The ENIAC (Electronic Numerical Integrator And Calculator) was invented by J. Presper Eckert and John Mauchly at the University of Pennsylvania and began construction in 1943 and was not completed until 1946. It occupied about 1,800 square feet and used about 18,000 vacuum tubes, weighing almost 50 tons. ENIAC was the first digital computer because it was fully functional.



1.3 Sixth Generation Computing

In the Sixth Generation, computers could be defined as the era of intelligent computers, based on Artificial Neural Networks. One of the most dramatic changes in the sixth generation will be the explosive growth of Wide Area Networking. Natural Language Processing (NLP) is a component of Artificial Intelligence (AI). It provides the ability to develop the computer program to understand human language.



Optical Character Recognition (Optical Grapheme Recognition) engine for the Indus Scripts has been developed using Deep Learning Neural Networks (a sub-field of Artificial Intelligence).

Given photographs, scans, or any image feed of an Indus Valley Civilization artifact, the system will be able to recognize the inscriptions (the symbol/grapheme sequences) from the image. There are totally 417 Symbols/Graphemes/Characters in the Indus Scripts and just 3700+ text inscriptions of data for the machine to learn and attain expert-level status.



1.4. Data and Information

We all know what a computer is? It is an electronic device that processes the input according to the set of instructions provided to it and gives the desired output at a very fast rate. Computers are very versatile as they do a lot of different tasks such as storing data, weather forecasting, booking airlines, railway or movie tickets

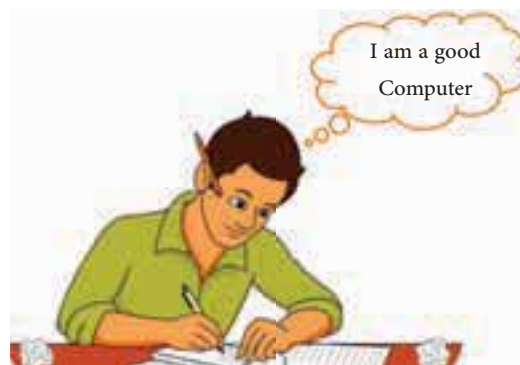
and even playing games.

Computer - man or machine?

Before 19th century, the term “Computer” was referred to humans who performed calculations using Abacus and Slide Rule and not to machine.

The term “computer” is derived from the word “compute” which means to calculate. The person who performs calculation is called as Computer. This term was later given to mechanical device as they began replacing the human computers.

Today's computers are electronic devices that accept data as input, process it, produce output and stores it for future reference.



Data: Data is defined as an un-processed collection of raw facts, suitable for communication, interpretation or processing.

For example, 134, 16 ‘Kavitha’, ‘C’ are data. This will not give any meaningful message.

Information: Information is a collection of facts from which conclusions may be drawn. In simple words we can say

that data is the raw facts that is processed to give meaningful, ordered or structured information. For example Kavitha is 16 years old. This information is about Kavitha and conveys some meaning. This conversion of data into information is called data processing.

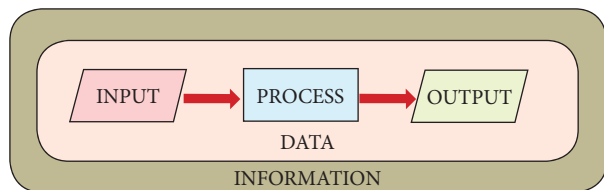


Figure 1.1 Data and Information

“A Computer is an electronic device that takes raw data (unprocessed) as an input from the user and processes it under the control of a set of instructions (called program), produces a result (output), and saves it for future use.”

1.5 Components of a Computer

The computer is the combination of hardware and software. Hardware is the physical component of a computer like motherboard, memory devices, monitor, keyboard etc., while software is the set of programs or instructions. Both hardware and software together make the computer system to function.



Figure 1.2: Computer

Let us first have a look at the functional components of a computer. Every task given to a computer follows an Input- Process- Output Cycle (IPO cycle). It needs certain input, processes that input and produces the desired output. The input unit takes the input, the central processing unit does the processing of data and the output unit produces the output. The memory unit holds the data and instructions during the processing.

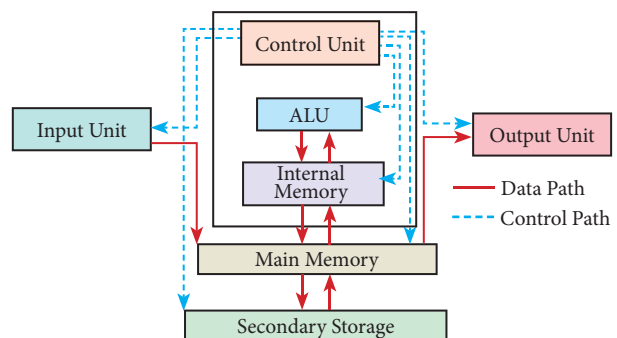


Figure 1.3 components of a computer

1.5.1 Input Unit

Input unit is used to feed any form of data to the computer, which can be stored in the memory unit for further processing. Example: Keyboard, mouse, etc.

1.5.2 Central Processing Unit

CPU is the major component which interprets and executes software instructions. It also control the operation of all other components such as memory, input and output units. It accepts binary data as input, process the data according to the instructions and provide the result as output.

The CPU has three components

which are Control unit, Arithmetic and logic unit (ALU) and Memory unit.

1.5.2.1 Arithmetic and Logic Unit

The ALU is a part of the CPU where various computing functions are performed on data. The ALU performs arithmetic operations such as addition, subtraction, multiplication, division and logical operations. The result of an operation is stored in internal memory of CPU. The logical operations of ALU promote the decision-making ability of a computer.

1.5.2.2 Control Unit

The control unit controls the flow of data between the CPU, memory and I/O devices. It also controls the entire operation of a computer.

1.5.3. Output Unit

An Output Unit is any hardware component that conveys information to users in an understandable form. Example: Monitor, Printer etc.

1.5.4. Memory Unit

The Memory Unit is of two types which are primary memory and secondary memory. The primary memory is used to temporarily store the programs and data when the instructions are ready to execute. The secondary memory is used to store the data permanently.

The Primary Memory is volatile, that is, the content is lost when the power supply is switched off. The Random

Access Memory (RAM) is an example of a main memory. The Secondary memory is non volatile, that is, the content is available even after the power supply is switched off. Hard disk, CD-ROM and DVD ROM are examples of secondary memory.

1.5.5. Input and Output Devices

Input Devices:

(1) **Keyboard:** Keyboard (wired / wireless, virtual) is the most common input device used today. The individual keys for letters, numbers and special characters are collectively known as character keys. This keyboard layout is derived from the keyboard of original typewriter. The data and instructions are given as input to the computer by typing on the keyboard. Apart from alphabet and numeric keys, it also has Function keys for performing different functions. There are different set of keys available in the keyboard such as character keys, modifier keys, system and GUI keys, enter and editing keys, function keys, navigation keys, numeric keypad and lock keys.



Figure 1.4 Keyboard

(2) **Mouse:** Mouse (wired/wireless) is a pointing device used to control the movement of the cursor on the display screen. It can be used to select icons, menus, command buttons or activate something on a computer. Some mouse

actions are move, click, double click, right click, drag and drop.

Different types of mouse available are: Mechanical Mouse, Optical, Laser Mouse, Air Mouse, 3D Mouse, Tactile Mouse, Ergonomic Mouse and Gaming Mouse.






MOST COMMONLY USED MOUSE			
SN	Type of Mouse	Mechanism	Developed and Introduced
1	Mechanical Mouse 	<ul style="list-style-type: none"> • A small ball is kept inside and touches the pad through a hole at the bottom of the mouse. • When the mouse is moved, the ball rolls. • This movement of the ball is converted into signals and sent to the computer. 	Telefunken, German Company, 02/10/1968
2	Optical Mouse 	<ul style="list-style-type: none"> • Measures the motion and acceleration of pointer. • It uses light source instead of ball to judge the motion of the pointer. • Optical mouse has three buttons. • Optical mouse is less sensitive towards surface. 	<ul style="list-style-type: none"> • In 1988, Richard Lyon, Steve Krish independently invented different versions of Optical Mouse.
3	Laser Mouse 	<ul style="list-style-type: none"> • Measures the motion and acceleration of pointer. • Laser Mouse uses Laser Light • Laser Mouse is highly sensitive and able to work on any hard surface. 	

Table 1.2 Commonly used Mouse

Who invented Mouse?

The computer mouse as we know it today was invented and developed by Douglas Engelbart, with the assistance of Bill English, during the 1960's and was patented on November 17, 1970.

(3) **Scanner:** Scanners are used to enter the information directly into the computer's memory. This device works like a Xerox machine. The scanner converts any type of printed or written information including photographs into a digital format, which can be manipulated by the computer.



Figure 1.5 Scanner

(4) **Fingerprint Scanner:** Fingerprint Scanner is a fingerprint recognition device used for computer security, equipped with the fingerprint recognition feature that uses biometric technology. Fingerprint Reader / Scanner is a very safe and convenient device for security instead of using passwords, which is vulnerable to fraud and is hard to remember.



Figure 1.6 Fingerprint Scanner

(5) **Track Ball:** Track ball is similar to the upside-down design of the mouse. The user moves the ball directly, while the device itself remains stationary. The user spins the ball in various directions to navigate the screen movements.



Figure 1.7 Track Ball

(6) **Retinal Scanner:** This performs a retinal scan which is a biometric technique that uses unique patterns on a person's retinal blood vessels.



Figure 1.8 Retinal Scanner

(7) **Light Pen:** A light pen is a pointing device shaped like a pen and is connected to

a monitor. The tip of the light pen contains a light-sensitive element which detects the light from the screen enabling the computer to identify the location of the pen on the screen. Light pens have the advantage of 'drawing' directly onto the screen, but this becomes hard to use, and is also not accurate.



Figure 1.9 Light Pen

(8) Optical Character Reader:

It is a device which detects characters printed or written on a paper with OCR, a user can scan a page from a book. The Computer will recognize the characters in the page as letters and punctuation marks and stores. The Scanned document can be edited using a wordprocessor.



Figure 1.10 Optical Character Reader

(9) Bar Code / QR Code Reader:

A Bar code is a pattern printed in lines of different thickness. The Bar code reader

scans the information on the bar codes transmits to the Computer for further processing. The system gives fast and error free entry of information into the computer.

QR (Quick response) Code: The QR code is the two dimension bar code which can be read by a camera and processed to interpret the image



Figure 1.11 Bar code Reader

(10) Voice Input Systems:

Microphone serves as a voice Input device. It captures the voice data and send it to the Computer. Using the microphone along with speech recognition software can offer a completely new approach to input information into the Computer.



Figure 1.12 Voice input System

(11) Digital Camera:

It captures images / videos directly in the digital form. It uses a CCD (Charge Coupled Device) electronic chip. When light falls on the chip through the lens, it converts light rays into digital format.



Figure 1.13 Digital Camera

(12) Touch Screen: A touch screen is a display device that allows the user to interact with a computer by using the finger. It can be quite useful as an alternative to a mouse or keyboard for navigating a Graphical User Interface (GUI). Touch screens are used on a wide variety of devices such as computers, laptops, monitors, smart phones, tablets, cash registers and information kiosks. Some touch screens use a grid of infrared beams to sense the presence of a finger instead of utilizing touch-sensitive input.



Figure 1.14 Touch Screen

(13) Keyer : A Keyer is a device for signaling by hand, by way of pressing

one or more switches. Modern keyers have a large number of switches but not as many as a full size keyboard. Typically, this number is between 4 and 50. A keyer differs from a keyboard, which has "no board", but the keys are arranged in a cluster.

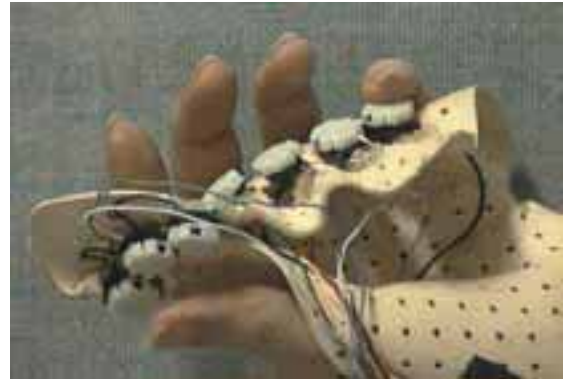


Figure 1.15 Keyer

Output Devices:

(1) Monitor: Monitor is the most commonly used output device to display the information. It looks like a TV. Pictures on a monitor are formed with picture elements called PIXELS. Monitors may either be Monochrome which display text or images in Black and White or can be color, which display results in multiple colors. There are many types of monitors available such as CRT (Cathode Ray Tube), LCD (Liquid Crystal Display) and LED (Light Emitting Diodes). The monitor works with the VGA (Video Graphics Array) card. The video graphics card helps the keyboard to communicate with the screen. It acts as an interface between the computer and display monitor. Usually the recent motherboards incorporate built-in video card.

The first computer monitor was part of the Xerox Alto computer system, which was released on March 1, 1973.



Figure 1.16 Monitor

(2) **Plotter:** Plotter is an output device that is used to produce graphical output on papers. It uses single color or multi color pens to draw pictures.



Figure 1.17 Plotter

(3) **Printers:** Printers are used to print the information on papers. Printers are divided into two main categories:

- Impact Printers
- Non Impact printers

Impact Printers

These printers print with striking of hammers or pins on ribbon. These printers can print on multi-part (using carbon papers) by using mechanical pressure. For example, Dot Matrix printers and Line matrix printers are impact printers.

A Dot matrix printer that prints using a fixed number of pins or wires. Each dot is produced by a tiny metal rod, also called a “wire” or “pin”, which works by the power of a tiny electromagnet or solenoid, either directly or through a set of small levers. It generally prints one line of text at a time. The printing speed of these printers varies from 30 to 1550 CPS (Character Per Second).



Figure 1.18 Impact Printer

Line matrix printers use a fixed print head for printing. Basically, it prints a page-wide line of dots. But it builds up a line of text by printing lines of dots. Line printers are capable of printing much more than 1000 Lines Per Minute, resulting in thousands of pages per hour. These printers also uses mechanical pressure to print on multi-part (using carbon papers).

Non-Impact Printers

These printers do not use striking mechanism for printing. They use electrostatic or laser technology. Quality and speed of these printers are better than Impact printers. For example, Laser printers and Inkjet printers are non-impact printers.

Laser Printers

Laser printers mostly work with similar technology used by photocopiers. It makes a laser beam scan back and

forth across a drum inside the printer, building up a pattern. It can produce very good quality of graphic images. One of the chief characteristics of laser printer is their resolution – how many Dots per inch(DPI). The available resolution range around 1200 dpi. Approximately it can print 100 pages per minute(PPM)



Figure 1.19 Laser Printer

Inkjet Printers:

Inkjet Printers use colour cartridges which combined Magenta, Yellow and Cyan inks to create color tones. A black cartridge is also used for monochrome output. Inkjet printers work by spraying ionised ink at a sheet of paper. The speed of Inkjet printers generally range from 1-20 PPM (Page Per Minute).



Figure 1.20 Inkjet Printer

They use the technology of firing ink by heating it so that it explodes towards the paper in bubbles or by using

piezoelectricity in which tiny electric currents controlled by electronic circuits are used inside the printer to spread ink in jet speed. An Inkjet printer can spread millions of dots of ink at the paper every single second.

Speakers: Speakers produce voice output (audio) . Using speaker along with speech synthesize software, the computer can provide voice output. This has become very common in places like airlines, schools, banks, railway stations, etc..



Figure 1.21 Speakers

Multimedia Projectors:

Multimedia projectors are used to produce computer output on a big screen. These are used to display presentations in meeting halls or in classrooms.



Figure 1.22 Multimedia Projector

1.6 Booting of computer

□ ————— □
An Operating system (OS) is a basic software that makes the computer

to work. When a computer is switched on, there is no information in its RAM. At the same time, in ROM, the pre-written program called POST (Power on Self Test) will be executed first. This program checks if the devices like RAM, keyboard, etc., are connected properly and ready to operate. If these devices are ready, then the BIOS (Basic Input Output System) gets executed. This process is called Booting. Thereafter, a program called “Bootstrap Loader” transfers OS from hard disk into main memory. Now the OS gets loaded (Windows/Linux, etc.,) and will get executed. Booting process is of two types.

1) Cold Booting

2) Warm Booting

Cold Booting: When the system starts from initial state i.e. it is switched on, we call it cold booting or Hard Booting. When the user presses the Power button, the instructions are read from the ROM to initiate the booting process.

Warm Booting: When the system restarts or when Reset button is pressed, we call it Warm Booting or Soft Booting. The system does not start from initial state and so all diagnostic tests need not be carried out in this case. There are chances of data loss and system damage as the data might not have been stored properly.

Points to Remember:

- Computers are seen everywhere around us, in all spheres of life.
- It is an electronic device that processes the input according to the set of instructions provided to it and gives the desired output at a very fast rate.
- Based on various stages of development, computers can be divided into six different generations.
- The computer is the combination of hardware and software.
- Hardware is the physical component of a computer.
- Input unit is used to feed any form of data to the computer.
- CPU interprets and executes software instructions.
- The ALU is a part of the CPU where various computing functions are performed on data.
- The control unit controls the flow of data between the CPU, memory and I/O devices.
- An Output Unit is any hardware component that conveys information to one or more people in user understandable form.
- The Memory Unit is of two kinds which are primary memory and secondary memory.
- Booting Process is of two types – Cold and Warm

Activity



STUDENT ACTIVITY

1. Explain the classification of computers.
2. Give the details of motherboard names, RAM capacity used in the years 1993, 1995, 2005, 2008, 2016.
3. Mention two new input and output devices that are not given in this chapter.

TEACHER ACTIVITY

1. Open a CPU and explain the components of it to students.
2. To connect and disconnect the various components of a computer.

Evaluation



SECTION – A

Choose the correct answer:

1. First generation computers used
 - (a) Vacuum tubes
 - (b) Transistors
 - (c) Integrated circuits
 - (d) Microprocessors
2. Name the volatile memory
 - (a) ROM
 - (b) PROM
 - (c) RAM
 - (d) EPROM
3. Identify the output device
 - (a) Keyboard
 - (b) Memory
 - (c) Monitor
 - (d) Mouse
4. Identify the input device
 - (a) Printer
 - (b) Mouse
 - (c) Plotter
 - (d) Projector
5. Output device is used for printing building plan, flex board, etc.
 - (a) Thermal printer
 - (b) Plotter
 - (c) Dot matrix
 - (d) inkjet printer



6. In ATM machines, which one of the following is used to
- (a) Touch Screen (b) speaker
(c) Monitor (d) Printer
7. When a system restarts which type of booting is used.
- (a) Warm booting (b) Cold booting
(c) Touch boot (d) Real boot.
8. Expand POST
- (a) Post on self Test (b) Power on Software Test
(c) Power on Self Test (d) Power on Self Text
9. Which one of the following is the main memory?
- (a) ROM (b) RAM
(c) Flash drive (d) Hard disk
10. Which generation of computer used IC's?
- (a) First (b) Second
(c) Third (d) Fourth

SECTION-B

Short Answers

1. What is a computer?
2. Distinguish between data and information.
3. What are the components of a CPU?
4. What is the function of an ALU?
5. Write the functions of control unit.
6. What is the function of memory?
7. Differentiate Input and output unit.
8. Distinguish Primary and Secondary memory.

SECTION-C

Explain in Brief

1. What are the characteristics of a computer?
2. Write the applications of computer.
3. What is an input device? Give two examples.
4. Name any three output devices.
5. Differentiate optical and Laser mouse
6. Write shortnote on impact printer
7. Write the characteristics of sixth generation.
8. Write the significant features of monitor.

SECTION - D

Explain in detail

1. Explain the basic components of a computer with a neat diagram.
2. Discuss the various generations of computers.
3. Explain the following
 - a. Inkjet Printer
 - b. Multimedia projector
 - c. Bar code / QR code Reader



References

- (1) Fundamentals of Computers – V. Rajaraman – PHI Publications
- (2) Computer Science text book – NCERT, New Delhi



Internet Resources

- (1) www.wikipedia.org
- (2) <https://www.computerhope.com/jargon/c/computer.htm>



Prepare a comparative study of various computers of past and present with respect to speed, memory, size, power consumption and other features



Computer	It is an electronic device that processes the input according to the set of instructions provided to it and gives the desired output at a very fast rate.
Vacuum tube	Vacuum tubes contain electrodes for controlling electron flow and were used in early computers as a switch or an amplifier.
Transistors	The transistor ("transfer resistance") is made up of semi-conductors. It is a component used to control the amount of current or voltage used for amplification/modulation of an electronic signal.
Punched cards	Punch cards also known as Hollerith cards are paper cards containing several punched or perforated holes that were punched by hand or machine to represent data.
Machine Language	Machine language is a collection of binary digits or bits that the computer reads and interprets.
Assembly language	An assembly language is a low-level programming language.
Integrated Circuits	The IC is a package containing many circuits, pathways, transistors, and other electronic components all working together to perform a particular function or a series of functions.
Microcomputer	Micro computer is used to describe a standard personal computer.
High-level languages	A high-level language is a computer programming language that isn't limited by the computer, designed for a specific job, and is easier to understand.
Natural Language Processing (NLP)	Natural Language Processing is a method used in artificial intelligence to process and derive meaning from the human language.
Robotics	Robot is a term coined by Karel Capek in the 1921 to play RUR (Rossum's Universal Robots). It is used to describe a computerized machine designed to respond to input received manually or from its surroundings.
Nanotechnology	Nanotechnology is an engineering, science, and technology that develops machines or works with one atom or one molecule that is 100 nanometers or smaller.
Bioengineering	A discipline that applies engineering principles of design and analysis to biological systems and biomedical technologies



Learning Objectives

- To know how the computer interprets and stores data in the memory.
- To learn various data representations and binary arithmetic.
- To learn conversion between various Number Systems.

2.1 Introduction

The term data comes from the word **datum**, which means a raw fact. The data is a fact about people, places or some objects.

Example:

Let 'Name', 'Age', 'Class', 'Marks' and 'Subject' be some defined variables. Now, let us assign a value to each of these variables.

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

Figure 2.1 Example for Data

In the above example, the values assigned to the five different variables

Number Systems

are called **data**. When the above data is processed, we get an information "Rajesh is 16 years old, studying in Class XI, has scored 65 marks in Computer Science subject".

2.2 Data Representations

Computer handles data in the form of '0' (Zero) and '1' (One). Any kind of data like number, alphabet, special character should be converted to '0' or '1' which can be understood by the Computer. '0' and '1' that the Computer can understand is called **Machine language**. '0' or '1' are called '**Binary Digits**' (BIT). Therefore, the study of data representation in the computer is important.

- A **bit** is the short form of **Binary digit** which can be '0' or '1'. It is the basic unit of data in computers.
- A **nibble** is a collection of 4 bits (Binary digits).
- A collection of 8 bits is called **Byte**. A byte is considered as the basic unit of measuring the memory size in the computer.
- **Word length** refers to the number of bits processed by a Computer's CPU. For example, a word length can have 8 bits, 16 bits, 32 bits and 64 bits (Present day Computers use 32 bits or 64 bits)

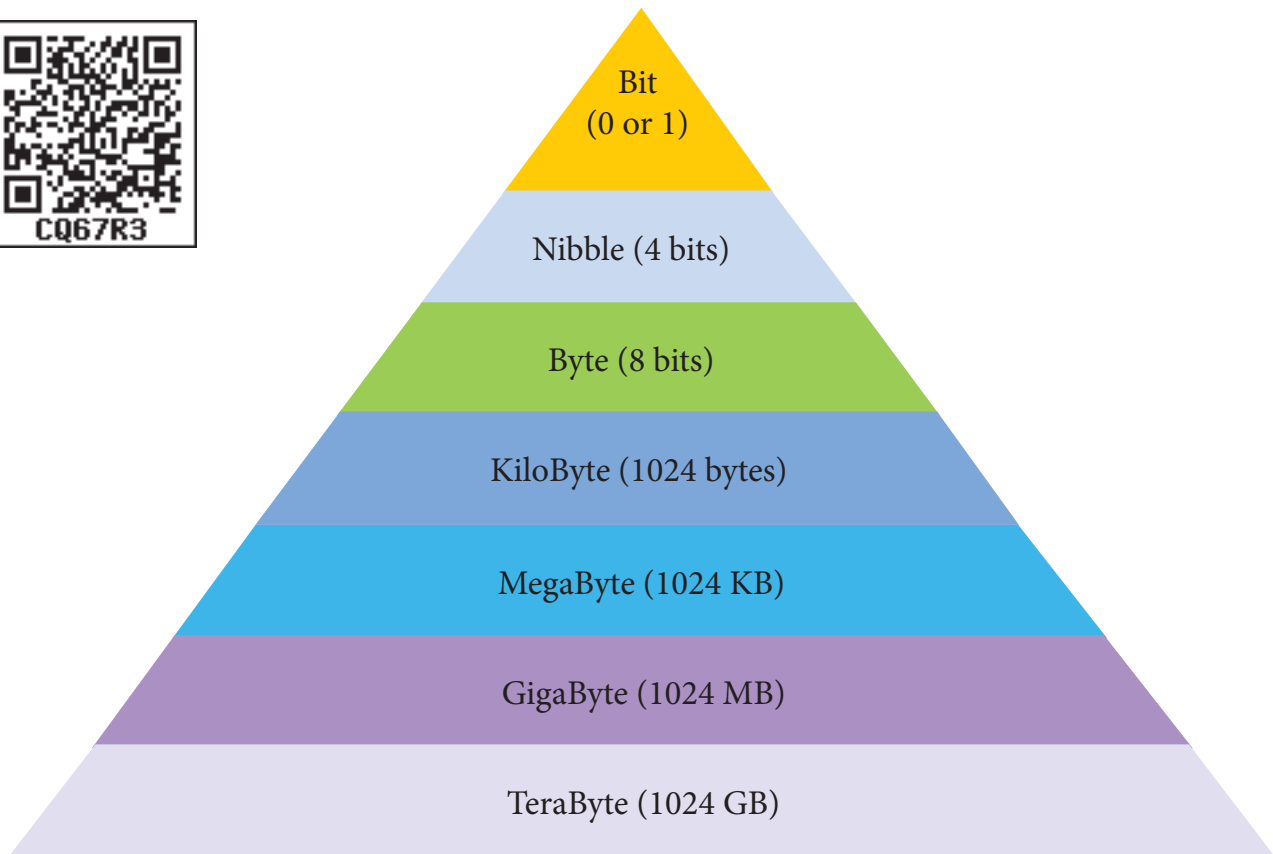


Figure 2.2 Data Representation

Computer memory (Main Memory and Secondary Storage) is normally represented in terms of KiloByte (KB) or MegaByte (MB). In decimal system, 1 Kilo represents 1000, that is, 10^3 . In binary system, 1 KiloByte represents 1024 bytes that is 2^{10} . The following table represents the various memory sizes:

Table 2.1 Memory Size (Read 2^{10} as 2 power 10)

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

Bytes are used to represent characters in a text. Different types of coding schemes are used to represent the character set and numbers. The most commonly used coding scheme is the **American Standard Code for Information Interchange** (ASCII). Each

binary value between 0 and 127 is used to represent a specific character. The ASCII value for (blank space) is 32 and the ASCII value of numeric 0 is 48. The range of ASCII values for lower case alphabets is from 97 to 122 and the range of ASCII values for the upper case alphabets is 65 to 90.



The speed of a computer depends on the number of bits it can process at once. For example, a 64-bit computer can process 64-bit numbers in one operation, while a 32-bit computer break 64-bit numbers down into smaller pieces, making it slower.

2.3 Different Types of Number Systems

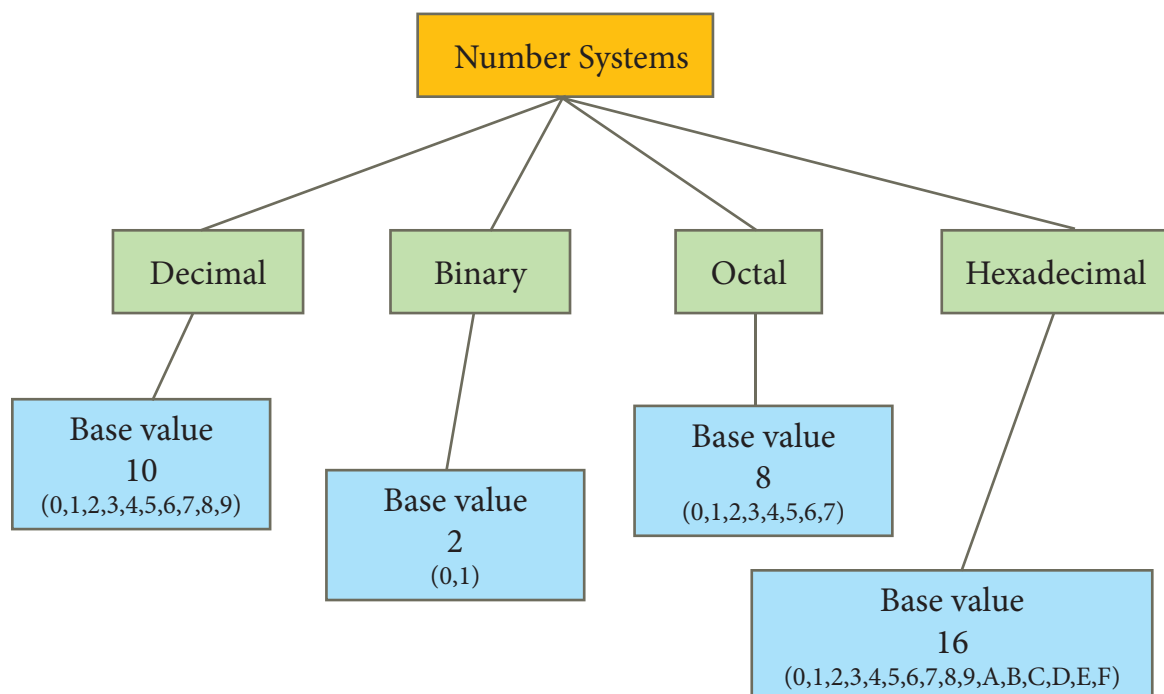


Figure 2.3. Number Systems

A numbering system is a way of representing numbers. The most commonly used numbering system in real life is Decimal number system. Other number systems are Binary, Octal, Hexadecimal number system. Each number system is uniquely identified by its **base value** or **radix**. Radix or base is the count of number of digits in each number system. Radix or base is the general idea behind positional numbering system.

2.3.1 Decimal Number System

It consists of 0,1,2,3,4,5,6,7,8,9(10 digits). It is the oldest and most popular number system used in our day to day life. In the positional number system, each decimal digit is weighted relative to its position in the number. This means that each digit in the number is multiplied by 10 raised to a power corresponding to that digit's position.

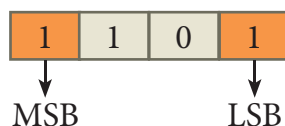
Example

$$\begin{aligned}(123)_{10} &= 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 \\ &= 100 + 20 + 3 \\ &= (123)_{10}\end{aligned}$$

$$\begin{aligned}(547)_8 &= 5 \times 8^2 + 4 \times 8^1 + 7 \times 8^0 \\ &= 5 \times 64 + 4 \times 8 + 7 \times 1 \\ &= 320 + 32 + 7 \\ &= (359)_{10}\end{aligned}$$

2.3.2 Binary Number System

There are only two digits in the Binary system, namely, 0 and 1. The numbers in the binary system are represented to the base 2 and the positional multipliers are the powers of 2. The left most bit in the binary number is called as the **Most Significant Bit (MSB)** and it has the largest positional weight. The right most bit is the **Least Significant Bit (LSB)** and has the smallest positional weight.



Example

The binary sequence $(1101)_2$ has the decimal equivalent:

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

2.3.3 Octal Number System

Octal number system uses digits 0,1,2,3,4,5,6 and 7 (8 digits). Each octal digit has its own positional value or weight as a power of 8.

Example

The Octal sequence $(547)_8$ has the decimal equivalent:

2.3.4 Hexadecimal Number System

A hexadecimal number is represented using base 16. Hexadecimal or Hex numbers are used as a shorthand form of binary sequence. This system is used to represent data in a more compact manner. Since 16 symbols are used, 0 to F, the notation is called hexadecimal. The first 10 symbols are the same as in the decimal system, 0 to 9 and the remaining 6 symbols are taken from the first 6 letters of the alphabet sequence, A to F, where A represents 10, B is 11, C is 12, D is 13, E is 14 and F is 15.

Table 2.2 Binary, Octal, Hexadecimal equivalent of Decimal Numbers

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

Example

The hexadecimal sequence $(25)_{16}$ has the decimal equivalent:

$$\begin{aligned}
 (25)_{16} &= 2 \times 16^1 + 5 \times 16^0 \\
 &= 32 + 5 \\
 &= (37)_{10}
 \end{aligned}$$

Workshop

1. Identify the number system for the following numbers

S. No.	Number	Number system
1	$(1010)_{10}$	Decimal Number system
2	$(1010)_2$	
3	$(989)_{16}$	
4	$(750)_8$	
5	$(926)_{10}$	

2. State whether the following numbers are valid or not. If invalid, give reason.

S.No.	Statement	Yes / No	Reason (If invalid)
1.	786 is an Octal number		
2.	101 is a Binary number		
3.	Radix of Octal number is 7		

2.4 Number System Conversions

2.4.1 Decimal to Binary Conversion

Generally two methods followed.

Method 1: To convert Decimal to Binary “Repeated Division by 2” method can be used. Any Decimal number divided by 2 will leave a remainder of 0 or 1. Repeated division by 2 will leave a sequence of 0s and 1s that become the binary equivalent of the decimal number. Suppose it is required to convert the decimal number N into binary form, dividing N by 2 in the decimal system, we will obtain a quotient N1 and a remainder R1, where R1 can have a value of either 0 or 1. The process is repeated until the quotient becomes 0 or 1. When the quotient is ‘0’ or ‘1’, it is the final remainder value. Write the final answer starting from final remainder value obtained to the first remainder value obtained.

Example

Convert $(65)_{10}$ into its equivalent binary number

2	65	
2	32 - 1	LSB
2	16 - 0	
2	8 - 0	
2	4 - 0	
2	2 - 0	
	1 - 0	MSB

Remainder	
Note :	
65/2	= 32 + 1
32/2	= 16 + 0
16/2	= 8 + 0
8/2	= 4 + 0
4/2	= 2 + 0
2/2	= 1 + 0

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

Method 2 : Sum of Powers of 2.

A decimal number can be converted into a binary number by adding up the powers of 2 and then adding bits as needed to obtain the total value of the number.

- a) Find the largest power of 2 that is smaller than or equal to 65.

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

- b) Set the 64's bit to 1 and subtract 64 from the original number

$$65 - 64 = 1$$

- c) 32 is greater than the remaining total. Therefore, set the 32's bit to 0.
- d) 16 is greater than the remaining total. Therefore, set the 16's bit to 0.
- e) 8 is greater than the remaining total. Therefore, set the 8's bit to 0.
- f) 4 is greater than the remaining total. Therefore, set the 4's bit to 0.
- g) 2 is greater than the remaining total. Therefore, set the 2's bit to 0.
- h) As the remaining value is equivalent to 1's bit, set it to 1.

$$1 - 1 = 0$$

Conversion is complete $65_{10} = (1000001)_2$

Example

The conversion steps can be given as follows:

Given Number : 65

Equivalent or value less than power of 2 is : 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

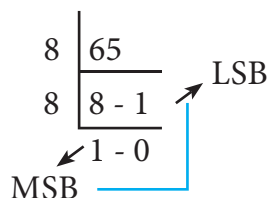
$$65_{10} = (1000001)_2$$

2.4.2 Decimal to Octal Conversion

To convert Decimal to Octal, “**Repeated Division by 8**” method can be used. The method is the same we have learnt in 2.4.1, but in this method, we have to divide the given number by 8.

Example

Convert $(65)_{10}$ into its equivalent Octal number



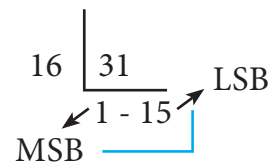
$$(65)_{10} = (101)_8$$

2.4.3 Decimal to Hexadecimal Conversion

To convert Decimal to Hexadecimal, “**Repeated division by 16**” method can be used. The method is the same as we have learnt in 2.4.1, but in this method, we have to divide the given number by 16.

Example

Convert $(31)_{10}$ into its equivalent hexadecimal number.



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

2.4.4 Conversion of fractional Decimal to Binary

The method of **repeated multiplication by 2** has to be used to convert such kind of decimal fractions.

The steps involved in the method of **repeated multiplication by 2**:

- Step 1: Multiply the decimal fraction by 2 and note the integer part. The integer part is either 0 or 1.
- Step 2: Discard the integer part of the previous product. Multiply the fractional part of the previous product by 2. Repeat Step 1 until the same fraction repeats or terminates (0).
- Step 3: The resulting integer part forms a sequence of 0s and 1s that become the binary equivalent of decimal fraction.
- Step 4: The final answer is to be written from first integer part obtained till the last integer part obtained.

	Integer part
$0.2 \times 2 = 0.4$	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)

Note: Fraction repeats, the product is the same as in the first step.

Write the integer parts from top to bottom to obtain the equivalent fractional binary number. Hence $(0.2)_{10} = (0.00110011...)_{2} = (0.00110011)_{2}$

Workshop

3. Convert the following Decimal numbers to its equivalent Binary, Octal, Hexadecimal.

1) 1920

2) 255

3) 126

2.4.5 Binary to Decimal Conversion

To convert Binary to Decimal we can use positional notation method.

Step 1: Write down the Binary digits and list the powers of 2 from right to left (Positional Notation)

Step 2: For each positional notation written for the digit, now write the equivalent weight.

Step 3: Multiply each digit with its corresponding weight

Step 4: Add all the values.

Table 2.3 Positional Notation and Weight

Positional Notation	Weight	Positional Notation	Weight
2^0	1	2^6	64
2^1	2	2^7	128
2^2	4	2^8	256
2^3	8	2^9	512
2^4	16	2^{10}	1024
2^5	32		

Example

Convert $(111011)_2$ into its equivalent decimal number.

Weight	32	16	8	4	2	1
Positional Notation	2^5	2^4	2^3	2^2	2^1	2^0
Given number	1	1	1	0	1	1

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

$$(111011)_2 = (59)_{10}$$

2.4.6 Binary to Octal Conversion

Step 1: Group the given binary number into 3 bits from right to left.

Step 2: You can add preceding 0 to make a group of 3 bits if the left most group has less than 3 bits.

Step 3: Convert equivalent octal value using "2's power positional weight method"

Table 2.4 Octal numbers and their Binary equivalent

Octal	Binary Equivalent
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Example

Convert $(11010110)_2$ into octal equivalent number

Step 1: Group the given number into 3 bits from right to left.

011 010 110

Note: The left most groups have less than 3 bits, so 0 is added to its left to make a group of 3 bits.

Step-2: Find Octal equivalent of each group

011	010	110
$\underbrace{\hspace{1cm}}$	$\underbrace{\hspace{1cm}}$	$\underbrace{\hspace{1cm}}$
3	2	6
$(11010110)_2 = (326)_8$		

2.4.7. Binary to Hexadecimal Conversion

Step 1: Group the given number into 4 bits from right to left.

Step 2: You can add preceding 0's to make a group of 4 bits if the left most group has less than 4 bits.

Step 3: Convert equivalent Hexadecimal value using "2's power positional weight method"

Example

Convert $(1111010110)_2$ into Hexadecimal number

Step 1: Group the given number into 4 bits from right to left.

0011	1101	0110
------	------	------

Note: 0's are added to the left most group to make it a group of 4 bits

0011	1101	0110
$\underbrace{\hspace{1cm}}$	$\underbrace{\hspace{1cm}}$	$\underbrace{\hspace{1cm}}$
3	D	6
$(1111010110)_2 = (3D6)_{16}$		

2.4.8 Conversion of fractional Binary to Decimal equivalent

Follow the steps to convert fractional Binary number to its Decimal equivalent.

Step 1: Convert integral part of Binary to Decimal equivalent using positional notation method (Procedure is same as discussed in 2.4.5)

Step 2: To convert the fractional part of binary to its decimal equivalent.

Step 2.1: Write down the Binary digits in the fractional part

Step 2.2: For all the digits write powers of 2 from left to right starting from 2^{-1} , 2^{-2} , 2^{-3} ,..... 2^{-n} ,

now write the equivalent weight.

Step 2.3: Multiply each digit with its corresponding weight

Step 2.4: Add all the values which you obtained in Step 2.3

4. Convert the given Binary number into its equivalent Decimal, Octal and Hexadecimal number.

- 1) 101110101
- 2) 1011010
- 3) 101011111

Table 2.5 Positional notation and weight

Positional notation	Weight
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

Step 3: To get final answer write the integral part (after conversion), followed by a decimal point(.) and the answer arrived at Step 2.4

Example

Convert the given Binary number $(11.011)_2$ into its decimal equivalent Integer part $(11)_2 = 3$ (Refer **table 2.2**)

2^1	2^0		2^{-1}	2^{-2}	2^{-3}
↑	↑		↑	↑	↑
1	1	.	0	1	1

$$\begin{aligned}
 &3 + . (0 \times 0.5 + 1 \times 0.25 + 1 \times 0.125) \\
 &= 3.375 \\
 &(11.011)_2 = (3.375)_{10}
 \end{aligned}$$

2.4.9. Octal to Decimal Conversion

To convert Octal to Decimal, we can use positional notation method.

1. Write down the Octal digits and list the powers of 8 from right to left (Positional Notation)
2. For each positional notation of the digit write the equivalent weight.
3. Multiply each digit with its corresponding weight
4. Add all the values

Example

Convert $(1265)_8$ to equivalent Decimal number

Weight	512	64	8	1
Positional Notation	8^3	8^2	8^1	8^0
Given number	1	2	6	5

$$\begin{aligned}
 (1265)_8 &= 512 \times 1 + 64 \times 2 + 8 \times 6 + 1 \times 5 \\
 &= 512 + 128 + 48 + 5 \\
 (1265)_8 &= (693)_{10}
 \end{aligned}$$

2.4.10 Octal to Binary Conversion

For each Octal digit in the given number write its 3 digits binary equivalent using positional notation.

Example

Convert $(6213)_8$ to equivalent Binary number

6	2	1	3
↓	↓	↓	↓
110	010	001	011
$(6213)_8 = (110010001011)_2$			

Workshop

5. Convert the following Octal numbers into Binary numbers.

- (A) 472 (B) 145 (C) 347
(D) 6247 (E) 645

2.4.11 Hexadecimal to Decimal Conversion

To convert Hexadecimal to Decimal we can use positional notation method.

1. Write down the Hexadecimal digits and list the powers of 16 from right to left (Positional Notation)
2. For each positional notation written for the digit, now write the equivalent weight.
3. Multiply each digit with its corresponding weight

4. Add all the values to get one final value.

Example

Convert $(25F)_{16}$ into its equivalent Decimal number.

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

$$\begin{aligned}(25F)_{16} &= 2 \times 256 + 5 \times 16 + 15 \times 1 \\ &= 512 + 80 + 15 \\ (25F)_{16} &= (607)_{10}\end{aligned}$$

2.4.12 Hexadecimal to Binary Conversion

Write 4 bits Binary equivalent for each Hexadecimal digit for the given number using positional notation method.

Example

Convert $(8BC)_{16}$ into equivalent Binary number

8	B	C
↓	↓	↓
1000	1011	1100
$(8BC)_{16} = (100010111100)_2$		

6. Convert the following Hexadecimal numbers to Binary numbers

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

2.5 Binary Representation for Signed Numbers

Computers can handle both positive (unsigned) and negative (signed) numbers. The simplest method to represent negative binary numbers is called **Signed Magnitude**. In signed magnitude method, the left most bit is Most Significant Bit (MSB), is called **sign bit or parity bit**.

The numbers are represented in computers in different ways:

- Signed Magnitude representation
- 1's Complement
- 2's Complement

2.5.1 Signed Magnitude representation

The value of the whole numbers can be determined by the sign used before it. If the number has '+' sign or no sign it will be considered as positive. If the number has '-' sign it will be considered as negative.

Example:

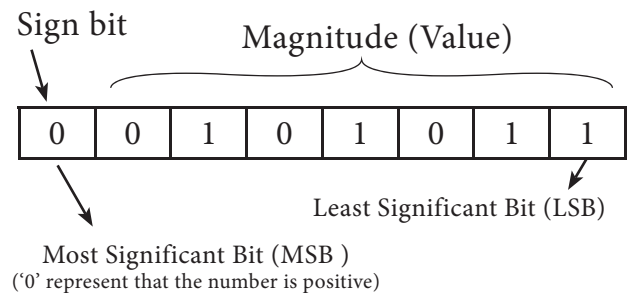
+43 or 43 is a positive number

-43 is a negative number

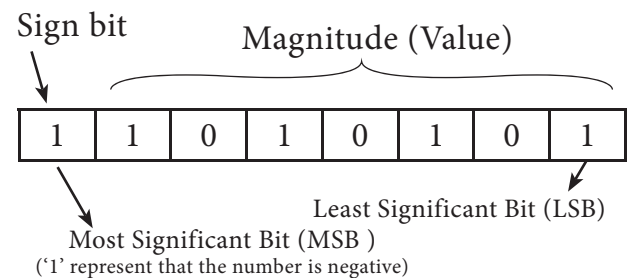
In signed binary representation, the left most bit is considered as sign bit.

If this bit is 0, it is a positive number and if it 1, it is a negative number. Therefore a signed binary number has 8 bits, only 7 bits used for storing values (magnitude) and the 1 bit is used for sign.

+43 is represented in memory as follows:



-43 can be represented in memory as follows.



2.5.2 1's Complement representation

This is an easier approach to represent signed numbers. This is for negative numbers only i.e. the number whose MSB is 1.

The steps to be followed to find 1's complement of a number:

- Step 1: Convert given Decimal number into Binary
- Step 2: Check if the binary number contains 8 bits , if less add 0 at the left most bit, to make it as 8 bits.
- Step 3: Invert all bits (i.e. Change 1 as 0 and 0 as 1)

Example

Find 1's complement for $(-24)_{10}$

Given Number	Binary Number	1's Complement
$(-24)_{10}$	00011000	11100111

2.5.3 2's Complement representation

The 2's-complement method for negative number is as follows:

- Invert all the bits in the binary sequence (i.e., change every 0 to 1 and every 1 to 0 i.e., 1's complement)
- Add 1 to the result to the Least Significant Bit (LSB).

Example

2's Complement represent of $(-24)_{10}$

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

Workshop

7. Write the 1's complement number and 2's complement number for the following decimal numbers:

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

2.6 Binary Arithmetic

As decimal numbers, the binary numbers also permit computations like addition, subtraction, multiplication and division. The following session deals only with binary addition and subtraction.

2.6.1 Binary Addition

The following table is useful when adding two binary numbers.

A	B	SUM (A + B)	Carry
0	0	0	-
0	1	1	-
1	0	1	-
1	1	0	1

In $1 + 1 = 10$, is considered as sum 0 and the 1 as carry bit. This carry bit is added with the previous position of the bit pattern.

Example Add: $1011_2 + 1001_2$

$$\begin{array}{r}
 \text{(Carry Bit)} \rightarrow \quad 1 \quad 1 \\
 \quad \quad \quad 1 \quad 0 \quad 1 \quad 1 \\
 + \quad \quad 1 \quad 0 \quad 0 \quad 1 \\
 \hline
 1 \quad 0 \quad 1 \quad 0 \quad 0
 \end{array}$$

1 0

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

Example Perform Binary addition for the following: $23_{10} + 12_{10}$

Step 1: Convert 23 and 12 into binary form

23_{10}					
2's power	16	8	4	2	1
Binary Number	1	0	1	1	1
$23_{10} = 00010111_2$					

12_{10}				
2's power	8	4	2	1
Binary Number	1	1	0	0
$12_{10} = 00001100_2$				

Step 2: Binary addition of 23 and 12:

Carry Bit →							
$23_{10} = 0$	0	0	1	0	1	1	1
$12_{10} = 0$	0	0	0	1	1	0	0
$35_{10} = 0$	0	1	0	0	0	1	1

2.6.2 Binary Subtraction

The table for Binary Subtraction is as follows:

A	B	Difference (A-B)	Borrow
0	0	0	0
1	0	1	0
1	1	0	0
0	1	1	1

When subtracting 1 from 0, borrow 1 from the next Most Significant Bit, when borrowing from the next Most Significant Bit, if it is 1, replace it with 0. If the next Most Significant Bit is 0, you must borrow from a more significant bit that contains 1 and replace it with 0 and 0s upto that point become 1s.

Example Subtract $1001010_2 - 10100_2$

	0	1	10	0	10	
	1	0	0	1	0	1 0
(-)			1	0	1	0 0
		1	1	0	1	1 0

Example Perform binary addition for the following: $(-21)_{10} + (5)_{10}$

Step 1: Change -21 and 5 into binary form

21_{10}					
2's power	16	8	4	2	1
Binary Number	1	0	1	0	1
$21_{10} = 00010101_2$					

5 ₁₀			
2's power	4	2	1
Binary Number	1	0	1
5 ₁₀ = 00000101 ₂			

Step 2:

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

Step 3:

Binary Addition of -21 and 5 :

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

Workshop

8. Perform the following binary computations:

- (A) $10_{10} + 15_{10}$ (B) $-12_{10} + 5_{10}$
 (C) $14_{10} - 12_{10}$ (D) $(-2_{10}) - (-6_{10})$

2.7 Representing Characters in Memory

As represented in introduction, all the input data given to the computer should be in understandable format. In general, 26 uppercase letters, 26 lowercase letters, 0 to 9 digits and special characters are used in a computer, which is called character set. All these character set are denoted through numbers only. All Characters in the character set needs a common encoding system. There are several encoding systems used for computer. They are

- BCD – Binary Coded Decimal
- EBCDIC – Extended Binary Coded Decimal Interchange Code
- ASCII – American Standard Code for Information Interchange
- Unicode
- ISCII - Indian Standard Code for Information Interchange

2.7.1 Binary Coded Decimal (BCD)

This encoding system is not in the practice right now. This is 2^6 bit encoding system. This can handle $2^6 = 64$ characters only.

2.7.2 American Standard Code for Information Interchange (ASCII)

This is the most popular encoding system recognized by United States. Most of the computers use this system. Remember this encoding system can handle English characters only. This can handle 2^7 bit which means 128 characters.

In this system, each character has individual number (Refer **Appendix**).

The new edition (version) ASCII -8, has 2^8 bits and can handle 256 characters are represented from 0 to 255 unique numbers.

The ASCII code equivalent to the uppercase letter 'A' is 65. The binary representation of ASCII (7 bit) value is 1000001. Also 01000001 in ASCII-8 bit.

2.7.3 Extended Binary Coded Decimal Interchange Code (EBCDIC)

This is similar to ASCII Code with 8 bit representation. This coding system

is formulated by International Business Machine(IBM). The coding system can handle 256 characters. The input code in ASCII can be converted to EBCDIC system and vice - versa.

2.7.4 Indian Standard Code for Information Interchange (ISCII)

ISCII is the system of handling the character of Indian local languages. This as a 8-bit coding system. Therefore it can handle 256 (2^8) characters. This system is formulated by the department of Electronics in India in the year 1986-88 and recognized by Bureau of Indian Standards (BIS). Now this coding system is integrated with Unicode.

2.7.5 Unicode

This coding system is used in most of the modern computers. The popular coding scheme after ASCII is Unicode. ASCII can represent only 256 characters. Therefore English and European Languages alone can be handled by ASCII. Particularly there was a situation, when the languages like Tamil, Malayalam, Kannada and Telugu could not be represented by ASCII. Hence, the Unicode was generated to handle all the coding system of Universal languages. This is 16 bit code and can handle 65536 characters.

Unicode scheme is denoted by hexadecimal numbers. The Unicode table of Tamil, Malayalam, Telugu and Kannada is shown Table 2.6

Table 2.6

Unicode Table of Tamil								Unicode Table of Malayalam									
	0B8	0B9	0BA	0BB	0BC	0BD	0BE	0BF		0D0	0D1	0D2	0D3	0D4	0D5	0D6	0D7
0		ஐ 0B90		ர 0BB0	ீ 0BC0	ஓ 0BD0		ய 0BF0	0	ஃ 0D00	ஹ 0D10	௦ 0D20	௪ 0D30	ி 0D40		ஐ 0D60	ய 0D70
1				ற 0BB1	ு 0BC1			ள 0BF1	1	ஃ 0D01		஡ 0D21	௦ 0D31	ு 0D41		ந 0D61	ற 0D71
2	ஃ 0B82	ஓ 0B92		ல 0BB2	ு 0BC2			த 0BF2	2	ஃ 0D02	ஃ 0D12	஡ 0D22	௦ 0D32	ு 0D42		ஃ 0D62	ஃ 0D72
3	ஃ 0B83	ஓ 0B93	ண 0BA3	ள 0BB3				உ 0BF3	3	ஃ 0D03	ஃ 0D13	ள 0D23	ஐ 0D33	ு 0D43		ஃ 0D63	ஃ 0D73
4		ஓ 0B94	த 0BA4	ழ 0BB4				ம் 0BF4	4		ஃ 0D14	த 0D24	ஃ 0D34	ு 0D44	ஃ 0D54		ஃ 0D74
5	அ 0B85	க 0B95		வ 0BB5				ஹ 0BF5	5	ஃ 0D05	க 0D15	ம 0D25	வ 0D35		ஃ 0D55		ஃ 0D75
6	ஆ 0B86			ஸ 0BB6	ெ 0BC6		௦ 0BE6	பு 0BF6	6	ஃ 0D06	வ 0D16	ஃ 0D26	ஸ 0D36	ெ 0D46	ஃ 0D56	௦ 0D66	ஃ 0D76
7	இ 0B87			ஷ 0BB7	ே 0BC7	ள 0BD7	க 0BE7	ஃ 0BF7	7	ஃ 0D07	ஸ 0D17	ய 0D27	ஃ 0D37	ே 0D47	ஃ 0D57	ஃ 0D67	ஃ 0D77
8	ஈ 0B88		ந 0BA8	ஸ 0BB8	ை 0BC8		உ 0BE8	ஃ 0BF8	8	ஃ 0D08	ஃ 0D18	ந 0D28	ஸ 0D38	ஃ 0D48	ஃ 0D58	ஃ 0D68	ஃ 0D78
9	உ 0B89	ங 0B99	ன 0BA9	ஹ 0BB9			ங 0BE9	ஃ 0BF9	9	ஃ 0D09	ஃ 0D19	ள 0D29	ஃ 0D39		ஃ 0D59	ஃ 0D69	ஃ 0D79
A	ஊ 0B8A	ச 0B9A	ப 0BAA		ொ 0BCA		ச 0BEA	ஃ 0BFA	A	ஃ 0D0A	ஃ 0D1A	ஃ 0D2A	ஃ 0D3A	ஃ 0D4A	ஃ 0D5A	ஃ 0D6A	ஃ 0D7A
B					ோ 0BCB		ஃ 0BEB		B	ஃ 0D0B	ஃ 0D1B	ஃ 0D2B	ஃ 0D3B	ஃ 0D4B	ஃ 0D5B	ஃ 0D6B	ஃ 0D7B
C		ஐ 0B9C			ெ 0BCC		ஃ 0BEC		C	ஃ 0D0C	ஃ 0D1C	ஃ 0D2C	ஃ 0D3C	ஃ 0D4C	ஃ 0D5C	ஃ 0D6C	ஃ 0D7C
D					ஃ 0BCD		எ 0BED		D		ஃ 0D1D	ஃ 0D2D	ஃ 0D3D	ஃ 0D4D	ஃ 0D5D	ஃ 0D6D	ஃ 0D7D
E	எ 0B8E	ஞ 0B9E	ம 0BAE	ா 0BBE			அ 0BEE		E	ஃ 0D0E	ஃ 0D1E	ஃ 0D2E	ஃ 0D3E	ஃ 0D4E	ஃ 0D5E	ஃ 0D6E	ஃ 0D7E
F	ஏ 0B8F	ஃ 0B9F	ய 0BAF	ி 0BBF			ஃ 0BEF		F	ஃ 0D0F	ஃ 0D1F	ஃ 0D2F	ஃ 0D3F	ஃ 0D4F	ஃ 0D5F	ஃ 0D6F	ஃ 0D7F

Table 2.6

Unicode Table of Telugu								Unicode Table of Kannada									
	0C0	0C1	0C2	0C3	0C4	0C5	0C6	0C7		0C8	0C9	0CA	0CB	0CC	0CD	0CE	0CF
0	ం	ఐ	ఠ	ఠ	ీ		ఋ		0	ం	ఐ	ఠ	ఠ	ీ		ఋ	
1	ం		ఠ	ఠ	ం		ఋ		1	ం		ఠ	ఠ	ం		ఋ	ఠ
2	ం	ఐ	ఠ	ల	ం		ఋ		2	ం	ఐ	ఠ	ల	ం		ఋ	ఠ
3	ం	ఐ	ఠ	శ	ం		ఋ		3	ం	ఐ	ఠ	శ	ం		ఋ	
4		ఐ	ఠ	ఠ	ం				4		ఐ	ఠ	ఠ	ం			
5	ఠ	ఠ	ఠ	వ		ం			5	ఠ	ఠ	ఠ	వ		ం		
6	ఠ	ఠ	ఠ	శ	ం	ం	ం		6	ఠ	ఠ	ఠ	త	ం	ం	ం	
7	ఠ	ఠ	ఠ	ష	ం		ం		7	ఠ	ఠ	ఠ	ష	ం		ం	
8	ఠ	ఠ	ఠ	స	ం	ఠ	ఠ	ఠ	8	ఠ	ఠ	ఠ	స	ం		ఠ	
9	ఠ	ఠ		హ		ఠ	ఠ	ఠ	9	ఠ	ఠ		ఠ			ఠ	
A	ఠ	ఠ	ఠ		ం	ఠ	ఠ	ఠ	A	ఠ	ఠ	ఠ		ం		ఠ	
B	ఠ	ఠ	ఠ		ం		ఠ	ఠ	B	ఠ	ఠ	ఠ		ం		ఠ	
C	ఠ	ఠ	ఠ		ం		ఠ	ఠ	C	ఠ	ఠ	ఠ	ం	ం		ఠ	
D		ఠ	ఠ	ఠ	ం		ఠ	ఠ	D		ఠ	ఠ	ఠ	ం		ఠ	
E	ఠ	ఠ	ఠ	ం			ఠ	ఠ	E	ఠ	ఠ	ఠ	ం		ఠ	ఠ	
F	ఠ	ఠ	ఠ	ం			ఠ	ఠ	F	ఠ	ఠ	ఠ	ం			ఠ	

Appendix
American Standard Code for Information Interchange (ASCII)
(Few specific characters only)

Alphabets

Alphabets	Decimal number	Binary number (8 bit)	Octal number	Hexadecimal number
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

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

Numerals

Alphabets	Decimal number	Binary number (8 bit)	Octal number	Hexadecimal number
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

Special Characters

Special symbols	Decimal number	Binary number (8 bit)	Octal number	Hexadecimal number
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

Evaluation



Part I

I Choose the best answer

1. Which refers to the number of bits processed by a computer's CPU?
A) Byte B) Nibble C) Word length D) Bit
2. How many bytes does 1 KiloByte contain?
A) 1000 B) 8 C) 4 D) 1024
3. Expansion for ASCII
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} is referred as
A) Kilo B) Tera C) Peta D) Zetta
5. How many characters can be handled in Binary Coded Decimal System?
A) 64 B) 255 C) 256 D) 128
6. For 1101_2 what is the Hexadecimal equivalent?
A) F B) E C) D D) B
7. What is the 1's complement of 00100110?
A) 00100110 B) 11011001 C) 11010001 D) 00101001
8. Which amongst this is not an Octal number?
A) 645 B) 234 C) 876 D) 123

II Very Short Answers

1. What is data?
2. Write the 1's complement procedure.
3. Convert $(46)_{10}$ into Binary number
4. We cannot find 1's complement for $(28)_{10}$. State reason.
5. List the encoding systems for characters in memory.

III Short Answers

1. What is radix of a number system? Give example
2. Write note on binary number system.
3. Convert $(150)_{10}$ into Binary, then convert that Binary number to Octal
4. Write short note on ISCII
5. Add a) $-22_{10} + 15_{10}$ b) $20_{10} + 25_{10}$


IV Detail Answers

1. a) Write the procedure to convert fractional Decimal to Binary
b) Convert $(98.46)_{10}$ to Binary
2. Find 1's Complement and 2's Complement for the following Decimal number
a) -98 b) -135
3. a) Add $1101010_2 + 101101_2$
b) Subtract $1101011_2 - 111010_2$


Part - II - Boolean Algebra

2.8 Introduction

Boolean algebra is a mathematical discipline that is used for designing digital circuits in a digital computer. It describes the relation between inputs and outputs of a digital circuit. The name Boolean algebra has been given in honor of an English mathematician George Boole who proposed the basic principles of this algebra.



George Boole (1815-1864) was born to a low class family and only received an elementary school education. Despite that, he taught himself highly advanced mathematics and different languages as a teenager without any assistance. He started teaching at age sixteen, and started his own school at age nineteen. By his mid-twenties, he had mastered most of the important mathematical principles in his day.



2.8.1 Binary valued quantities:

Every day we have to make logical decisions:

1. Should I carry Computer Science book every day? Yes / No
2. $8-10 = 10$ is this answer correct? Yes / No

3. Chennai is capital of India? Yes / No

4. What did I say yesterday?

The first three questions thrown above, the answer may be True (Yes) or False (No). But the fourth one, we cannot be answer as True or False. Thus, sentences which can be determined to be True or False are called “Logical Statement” or “Truth Functions”. The results True or False are called “Truth Values”. The truth values depicted by logical constant 1 and 0; 1 means True and 0 means False. The variable which can store these truth values are called “Logical variable” or “Binary valued variables” or “Boolean Variables” as these can store one of the two values of True or False.

2.8.2 Logical Operations:

Boolean algebra makes use of variables and operations (functions). The basic logical operations are AND, OR and NOT, which are symbolically represented by dot (\cdot), plus ($+$), and by over bar / single apostrophe respectively. These symbols are also called as “Logical Operators”.

2.8.3 Truth Table:

A truth table represents all the possible values of logical variable or statements along with all the possible results of given combination of truth values.

2.8.4 AND operator

The AND operator is defined in Boolean algebra by the use of the dot (.) operator. It is similar to multiplication in ordinary algebra. The AND operator combines two or more input variables so that the output is true only if all the inputs are true. The truth table for a 2-input AND operator is shown as follows:

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



The above 2-input AND operation is expressed as: $Y = A \cdot B$

2.8.5 OR operator

The plus sign is used to indicate the OR operator. The OR operator combines two or more input variables so that the output is true if at least one input is true. The truth table for a 2-input OR operator is shown as follows:

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

The above 2-input OR operation is expressed as: $Y = A + B$

2.8.6 NOT operator

The NOT operator has one input and one output. The input is either true or false, and the output is always the opposite, that is, the NOT operator inverts the input. The truth

table for a NOT operator where A is the input variable and Y is the output is shown below:

A	Y
0	1
1	0

The NOT operator is represented algebraically by the Boolean expression: $Y = \overline{A}$

Example:

Consider the Boolean equation:

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

D is equal to 1 (true) if A is 1 or if $(\overline{B} \cdot C)$ is 1, that is, B = 0 and C = 1.

Otherwise D is equal to 0 (false).

The basic logic functions AND, OR, and NOT can also be combined to make other logic operators such as NAND and NOR

2.8.7 NAND operator

The NAND is the combination of NOT and AND. The NAND is generated by inverting the output of an AND operator. The algebraic expression of the NAND function is:

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

The NAND function truth table is shown below:

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

$$A \text{ NAND } B = \text{NOT } (A \text{ AND } B)$$

2.8.8 NOR operator

The NOR is the combination of NOT and OR. The NOR is generated by inverting the output of an OR operator. The algebraic expression of the NOR function is:

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

The NOR function truth table is shown below:

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

$$A \text{ NOR } B = \text{NOT } (A \text{ OR } B)$$

2.9 Basic Logic Gates:

A gate is a basic electronic circuit which operates on one or more signals to produce an output signal. There are three fundamental gates namely AND, OR and NOT. The other logic gates like NAND, NOR, XOR and XNOR are derived gates which are derived from the fundamental gates. NAND and NOR gates are called Universal gates, because the fundamental logic gates can be realized through them.

2.9.1 AND Gate

The AND gate can have two or more input signals and produce an output signal.

The output is "true" only when both inputs are "true", otherwise, the output is "false". In other words the output will be 1 if and only if both inputs are 1; otherwise the output is 0. The output of the AND gate is represented by a variable say C, where A and B are two and if input boolean variables. In boolean algebra, a variable can take either of the values '0' or '1'. The logical symbol of the AND gate is

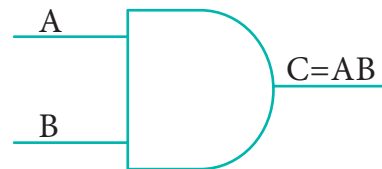


Fig. 2.4 Logic symbol of AND Gate

One way to symbolize the action of an AND gate is by writing the boolean function.

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

In boolean algebra the multiplication sign stands for the AND operation. Therefore, the output of the AND gate is

$$C = A \cdot B \text{ or}$$

simply $C = AB$

Read this as "C equals A AND B". Since there are two input variables here, the truth table has four entries, because there are four possible inputs : 00, 01, 10 and 11.

For instance if both inputs are 0,

$$C = A \cdot B$$

$$= 0 \cdot 0$$

$$= 0$$

The truth table for AND Gate is

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

Table 2.7 Truth Table for AND Gate

2.9.2 OR Gate

The OR gate gets its name from its behaviour like the logical inclusive "OR". The output is "true" if either or both of the inputs are "true". If both inputs are "false" then the output is "false". In other words the output will be 1 if and only if one or both inputs are 1; otherwise, the output is 0. The logical symbol of the OR gate is

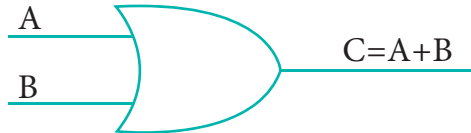


Fig. 2.5 Logic symbol of OR Gate

The OR gate output is

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

We use the + sign to denote the OR function. Therefore,

$$C = A + B$$

Read this as "C equals A OR B".

For instance, if both the inputs are 1

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

The truth table for OR gate is

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

Table 2.8 Truth Table for OR Gate

2.9.3 NOT Gate

The NOT gate, called a logical inverter, has only one input. It reverses the logical state. In other words the output C is always the complement of the input. The logical symbol of the NOT gate is

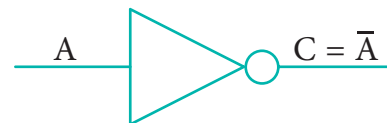


Fig. 2.6 Logic symbol of NOT Gate

The boolean function of NOT gate is

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

In boolean algebra, the overbar stands for NOT operation. Therefore,

$$C = \bar{A}$$

Read this as "C equals NOT A" or "C equals the complement of A".

If A is 0,

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

On the otherhand, if A is 1,

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

The truth table for NOT gate is

Input	Output
A	C
1	0
0	1

Table 2.9 Truth Table for NOT Gate

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

Table 2.10 Truth Table for NOR Gate

2.9.4 NOR Gate

The NOR gate circuit is an OR gate followed by an inverter. Its output is "true" if both inputs are "false". Otherwise, the output is "false". In other words, the only way to get '1' as output is to have both inputs '0'. Otherwise the output is 0. The logic circuit of the NOR gate is

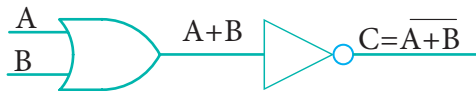


Fig. 2.7 Logic Circuit of NOR Gate



Fig. 2.8 Logic symbol of NOR Gate

The output of NOR gate is

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

Read this as "C equals NOT of A OR B" or "C equals the complement of A OR B".

For example if both the inputs are 0,

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

The truth table for NOR gate is

2.9.5 Bubbled AND Gate

The Logic Circuit of Bubbled AND Gate

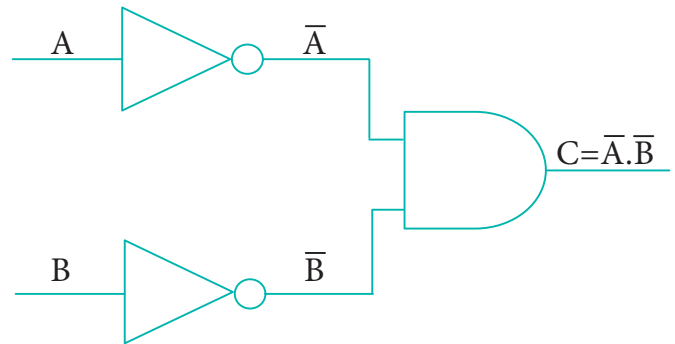


Fig. 2.9 Logic circuit of Bubbled AND Gate

In the above circuit, inverters on the input lines of the AND gate gives the output as

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

This circuit can be redrawn as the bubbles on the inputs, where the bubbles represent inversion.

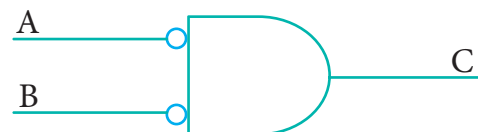


Fig. 2.10 Logic Symbol of Bubbled AND Gate

We refer this as bubbled AND gate. Let us analyse this logic circuit for all input possibilities.

If $A = 0$ and $B = 0$ $C = (\overline{0.0}) = 1.1 = 1$

If $A = 0$ and $B = 1$ $C = (\overline{0.1}) = 1.0 = 0$

If $A = 1$ and $B = 0$ $C = (\overline{1.0}) = 0.1 = 0$

If $A = 1$ and $B = 1$ $C = (\overline{1.1}) = 0.0 = 0$

Here the truth table is

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

You can see that, a bubbled AND gate produces the same output as a NOR gate. So, You can replace each NOR gate by a bubbled AND gate. In other words the circuits are interchangeable.

Therefore

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

Which establishes the De Morgan's first theorem.

2.9.6 NAND Gate

The NAND gate operates an AND gate followed by a NOT gate. It acts in the manner of the logical operation "AND" followed by inversion. The output is "false" if both inputs are "true", otherwise, the output is "true". In other words the output of the NAND gate is 0 if and only if both the inputs are 1, otherwise the output is 1. The logic circuit of NAND gate is

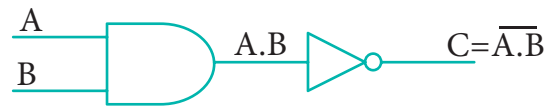


Fig. 2.11 Logic Circuit of NAND Gate

The logical symbol of NAND gate is

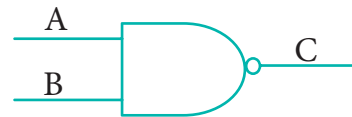


Fig. 2.12 Logic Symbol of NAND Gate

The output of the NAND gate is

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

Read this as "C" equals NOT of A AND B" or "C" equals the complement of A AND B".

For example if both the inputs are 1

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

The truth table for NAND gate is

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

Table 2.11 Truth Table for NAND Gate

2.9.7 Bubbled OR Gate

The logic circuit of bubbled OR gate is

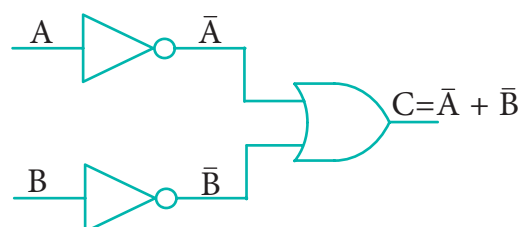


Fig. 2.13 Logic Circuit of Bubbled OR Gate

The output of this circuit can be written as $C = \bar{A} + \bar{B}$

The above circuit can be redrawn as the bubbles on the input, where the bubbles represents the inversion.

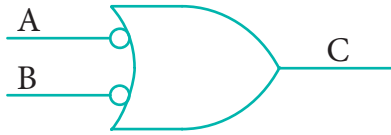


Fig. 2.14 Logic Symbol of Bubbled OR Gate

We refer this as bubbled OR gate. The truth table for the bubbled OR is

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

Table 2.12 Truth Table for Bubbled OR Gate

If we compare the truth tables of the bubbled OR gate with NAND gate, they are identical. So the circuits are interchangeable.

Therefore,

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

Which establishes the De Morgan's second theorem.

2.9.8 XOR Gate

The XOR (exclusive - OR) gate acts in the same way as the logical "either/or." The output is "true" if either, but not both, of the inputs are "true". The output is "false" if both inputs are "false" or if both inputs are "true." Another way of looking at this circuit is to observe that the output is 1 if the inputs

are different, but 0 if the inputs are the same. The logic circuit of XOR gate is

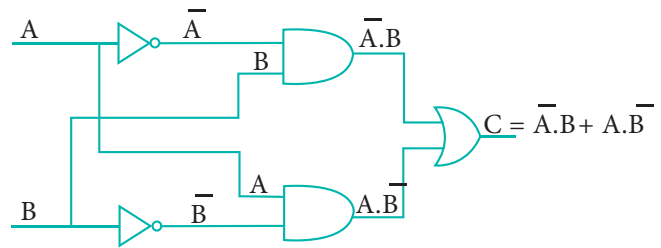


Fig. 2.15 Logic Circuit of XOR Gate

The output of the XOR gate is

The truth table for XOR gate is

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

Table 2.13 Truth Table for XOR Gate

In boolean algebra, exclusive - OR operator \oplus or "encircled plus".

Hence $C = A \oplus B$

The logical symbol of XOR gate is

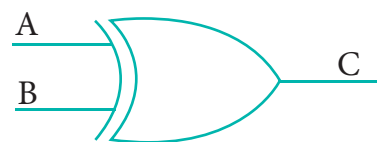


Fig. 2.16 Logic Symbol of XOR Gate

2.9.9 XNOR Gate

The XNOR (exclusive - NOR) gate is a combination XOR gate followed by an inverter. Its output is "true" if the inputs are the same, and "false" if the inputs are different. In simple words, the output is 1 if the input are the same, otherwise the output is 0. The logic circuit of XNOR gate is

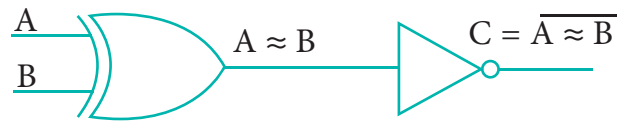


Fig. 2.17 Logic Circuit of XNOR Gate

The output of the XNOR is NOT of XOR

$$\begin{aligned}
 C &= \overline{A \oplus B} \\
 &= \overline{A \cdot B + A \cdot \bar{B}} \\
 &= AB + \bar{A} \bar{B}
 \end{aligned}$$

(Using De Morgan's Theorem)

In boolean algebra, \odot or "included dot" stands for the XNOR.

Therefore, $C = A \odot B$

The logical symbol is

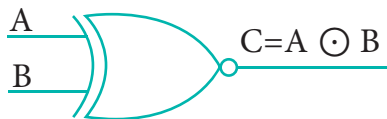


Fig. 2.18 Logic Symbol of XNOR Gate

The truth table for XNOR Gate is

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

Table 2.14 Truth Table for XNOR Gate

Using combination of logic gates, complex operations can be performed. In theory, there is no limit to the number of gates that can be arranged together in a single device. But in practice, there is a limit to the number of gates that can be packed into a given physical space. Arrays of logic gates are found in digital integrated circuits.

Theorems of Boolean Algebra

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



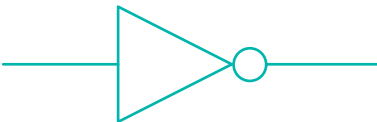


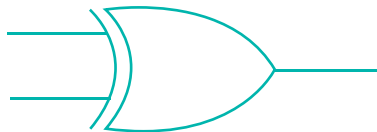
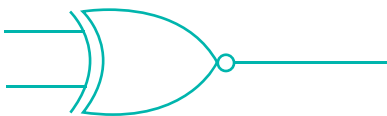
$$A + \overline{A} \cdot B = A + B$$

De Morgan's

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

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

Table 2. 15
Logic Gates and their corresponding Truth Tables

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

Evaluation



Part – I

- Which is a basic electronic circuit which operates on one or more signals?
(A) Boolean algebra (B) Gate
(C) Fundamental gates (D) Derived gates
- Which gate is called as the logical inverter?
(A) AND (B) OR
(C) NOT (D) XNOR
- $A + A = ?$
(A) A (B) 0
(C) 1 (D) A
- NOR is a combination of ?
(A) NOT(OR) (B) NOT(AND)
(C) NOT(NOT) (D) NOT(NOR)
- NAND is called as Gate
(A) Fundamental Gate (B) Derived Gate
(C) Logical Gate (D) Electronic gate



PART – II

- What is Boolean Algebra?
- Write a short note on NAND Gate.
- Draw the truth table for XOR gate.
- Write the associative laws?
- What are derived gates?

Part – III

- Write the truth table of fundamental gates.
- Write a short note on XNOR gate.
- Reason out why the NAND and NOR are called universal gates?
- Give the truth table of XOR gate.
- Write the De Morgan's law.

PART – IV

- Explain the fundamental gates with expression and truth table.
- How AND and OR can be realized using NAND and NOR gate.
- Explain the Derived gates with expression and truth table.

Computer Organization

Learning Objectives

- To know the organization of the computer components and their interconnections.
- To know the processors and their characteristics.
- To know the importance of memory devices and their roles in a computer.
- To explore RAM, ROM and differentiate each of them.
- To know about cache memory and how it improves the performance of a computer
- To know the secondary devices and their usage
- To know about the ports and interfaces so that external devices can be connected



3.1 Introduction

Computer organization deals with the hardware components of a computer system. It includes Input / Output devices, the Central Processing Unit, storage devices and primary memory. It is concerned with how the various components of computer hardware operate. It also deals with how they are interconnected to implement an architectural specification. The term computer organization looks similar to the term computer architecture. But, computer architecture deals with the engineering considerations involved in designing a computer. On the other hand, Computer Organization deals with the hardware components that are transparent to the programmer.

3.2. Basics of Microprocessors

The CPU is the major component of a computer, which performs all tasks. This is realized by the microprocessor which is an Integrated Circuit. Microprocessors were first introduced in early 1970s. The first general purpose microprocessor, 4004 was developed by Intel Inc.

The microprocessor is a programmable multipurpose silicon chip. It is driven by clock pulses. It accepts input as a binary data and after processing, it provides the output data as per the

instructions stored in the memory. A block diagram of a microprocessor based system is shown in Figure 3.1.

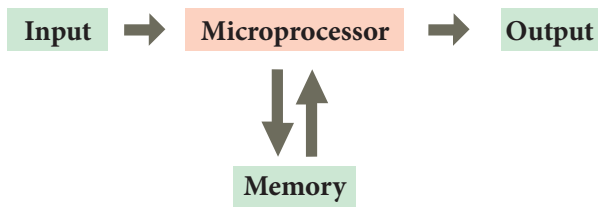


Figure 3.1 A Microprocessor - Based System

The microprocessor is made up of 3 main units. They are:

- **Arithmetic and Logic unit (ALU):** To perform arithmetic and logical instructions based on computer instructions.
- **Control unit:** To control the overall operations of the computer through signals.
- **Registers (Internal Memory):** They are used to hold the instruction and data for the execution of the processor.

The microprocessor is able to communicate with the memory units and the Input / Output devices as in Figure 3.2. The system bus is a bunch of wires which is the collection of address bus, data bus and control bus that serves as communication channels between the Microprocessor and other devices.

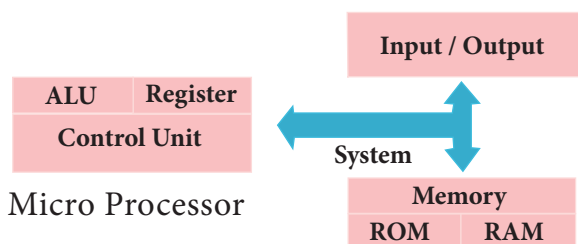


Figure 3.2 Interconnecting the Microprocessor with Other Devices

Characteristics of Microprocessors

A Microprocessor's performance depends on the following characteristics:

- a) Clock speed
- b) Instruction set
- c) Word size

Speed Measurement



Hertz – abbreviated as Hz is the standard unit of measurement used for measuring frequency. Since frequency is measured in cycles per second, one hertz equals one cycle per second.

Hertz is commonly used to measure wave frequencies, such as sound waves, light waves, and radio waves. For example, the average human ear can detect sound waves between 20 and 20,000 Hz. Sound waves close to 20 Hz have a low pitch and are called "bass" frequencies. Sound waves above 5,000 Hz have a high pitch and are called "treble" frequencies.

While hertz can be used to measure wave frequencies, it is also used to measure the speed of computer processors. For example, each CPU is rated at a specific clock speed. This number indicates how many instruction cycles the processor can perform in every second. Since modern processors can perform millions or even billions of instructions per second, clock speeds are typically measured in megahertz or gigahertz.

a) Clock Speed

Every microprocessor has an **internal clock** that regulates the speed at which it executes instructions. The speed at which the microprocessor executes instructions is called the **clock speed**. Clock speed is measured in MHz (Mega Hertz) or in GHz (Giga Hertz).

b) Instruction Set

A command which is given to a computer to perform an operation on data is called an **instruction**. Basic set of machine level instructions that a microprocessor is designed to execute is called as an **instruction set**. This instruction set carries out the following types of operations:

- Data transfer
- Arithmetic operations
- Logical operations
- Control flow
- Input/output

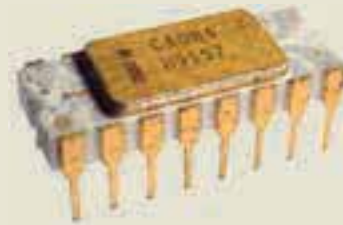
c) Word Size

- The number of bits that can be processed by a processor in a single instruction is called its word size. **Word size** determines the amount of RAM that can be accessed by a microprocessor at one time and the total number of pins on the microprocessor. Total number of input and output pins in turn determines the architecture of the microprocessor.



The first commercial microprocessor, Intel 4004 is a 4 bit processor. It has 4 input pins and 4 output pins. Number of output pins is always equal to the number of input pins. It can process 4 bits at a time. So it is called as a 4 bit processor.

Intel 4004



Produced	From late 1971 to 1981
Manufacturer	Intel Inc.
Clock Speed	740 kHz
Size	10 Micrometer (μm)
Transistors	2300
Data width	4 bits
Package	16 pin

Intel Inc. released many microprocessors like Intel 8085 which is an 8 bit processor, Intel 8086 which is a 16 bit processor and so on. Currently most of the microprocessors use 32 bit or 64 bit architecture.

3.3 Data communication between CPU and memory

The Central Processing Unit(CPU) has a Memory Data Register (MDR) and a Memory Address Register (MAR). The Memory Data Register (MDR) keeps the data which is transferred between the Memory and the CPU. The Program

Counter (PC) is a special register in the CPU which always keeps the address of the next instruction to be executed. The Arithmetic and Logic unit of CPU places the address of the memory to be fetched, into the Memory Address Register.

A bus is a collection of wires used for communication between the internal components of a computer. The address bus is used to point a memory location. A decoder, a digital circuit is used to point to the specific memory location where the **word** can be located. The address register is connected with the address bus, which provides the address of the instruction. A data bus is used to transfer data between the memory and the CPU. The data bus is bidirectional and the address bus is unidirectional. The control bus controls both read and write operations. The read operation fetches data from memory and transfers to MDR. A single control line performs two operations like Read/Write using 1 or 0. Also, the write operation transfers data from the MDR to memory. This organization is shown in Figure 3.3.

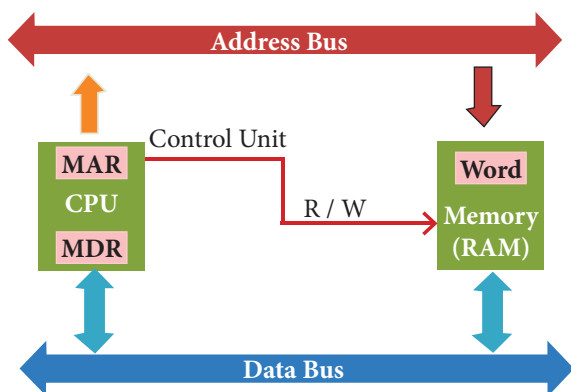


Figure 3.3 Bus connectivity between CPU and Memory

The word in the RAM has the same size (no. of bits) as the Memory Data

Register (MDR). If the processor is an 8-bit processor like Intel 8085, its MDR and the word in the RAM both have 8 bits.

If the size of the MDR is eight bits, which can be connected with a word in the memory which is also eight bits size. The data bus has eight parallel wires to transfer data either from MDR to word or word to MDR based on the control(Read or write). This control line is labeled as R/W , which becomes 1 means READ operation and 0 means WRITE operation. Figure 3.4 shows the content of MDR and the word before the READ operation. Also, Figure 3.5 shows the content of MDR and the word after the READ operation.

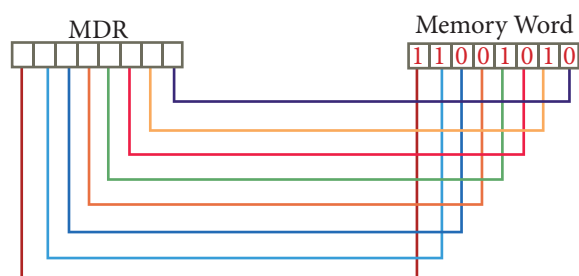


Figure 3.4 Before the read operation

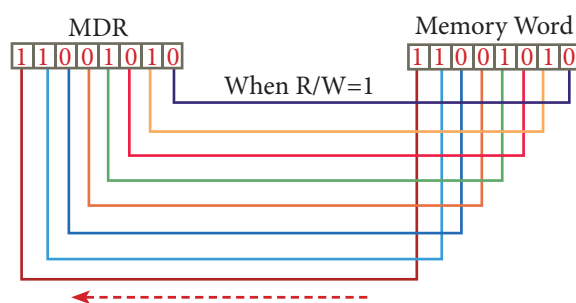


Figure 3.5 After the read operation

The read operation transfers the data(bits) from word to Memory Data Register. The write operation transfers the data(bits) from Memory Data Register to word.



If 5V is applied at one end of a wire, the other end also can receive 5V. In the same way, the buses are wires, and the binary data are voltages (5V as 1 and 0V as 0), and these buses can simply pass the data as voltages from one end to other.

3.4 Types of Microprocessors

Microprocessors can be classified based on the following criteria:

- The width of data that can be processed
- The instruction set

3.4.1 Classification of Microprocessors based on the Data Width

Depending on the data width, microprocessors can process instructions. The microprocessors can be classified as follows:

- 8-bit microprocessor
- 16-bit microprocessor
- 32-bit microprocessor
- 64-bit microprocessor

3.4.2 Classification of Microprocessors based on Instruction Set

The size of the instruction set is another important consideration while categorizing microprocessors. Initially, microprocessors had very small instruction sets because complex

hardware was expensive as well as difficult to build. As technology had developed to overcome these issues, more and more complex instructions were added to increase the functionality of microprocessors. Let us learn more about the two types of microprocessors based on their instruction sets.

3.4.2.1 Reduced Instruction Set Computers (RISC)

RISC stands for **Reduced Instruction Set Computers**. They have a small set of highly optimized instructions. Complex instructions are also implemented using simple instructions, thus reducing the size of the instruction set.

Examples of RISC processors are Pentium IV, Intel P6, AMD K6 and K7.

3.4.2.2 Complex Instruction Set Computers (CISC)

CISC stands for **Complex Instruction Set Computers**. They support hundreds of instructions. Computers supporting CISC can accomplish a wide variety of tasks, making them ideal for personal computers.

Examples of CISC processors are Intel 386 & 486, Pentium, Pentium II and III, and Motorola 68000.

3.5 Memory Devices

A memory is just like a human brain. It is used to store data and instructions. Computer memory is the storage space in the computer, where data

and instructions are stored. There are two types of accessing methods to access (read or write) the memory. They are sequential access and random access. In sequential access, the memory is accessed in an orderly manner from starting to end. But, in random access, any byte of memory can be accessed directly without navigating through previous bytes. Different memory devices are arranged according to the capacity, speed and cost as shown in Figure 3.6.

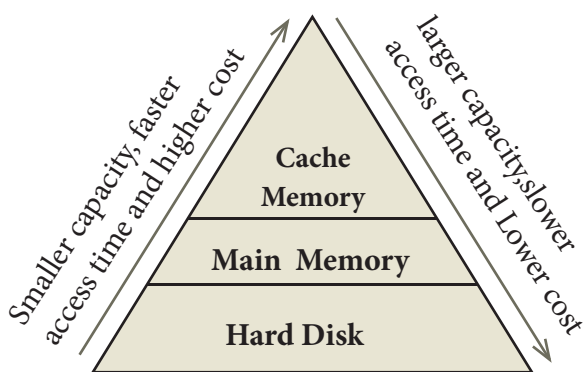


Figure 3.6 Memory Hierarchy

3.5.1 Random-Access Memory (RAM)

The main memory is otherwise called as **Random Access Memory**. This is available in computers in the form of Integrated Circuits (ICs). It is the place in a computer where the Operating System, Application Programs and the data in current use are kept temporarily so that they can be accessed by the computer's processor. The smallest unit of information that can be stored in the memory is called as a bit. The memory can be accessed by a collection of 8 bits which is called as a byte. The bytes are referred by 'B'. If a computer has 1 megabyte of memory, then it can store 10,48,576 bytes (or characters) of information. [Hence 1MB is 1024KB and 1 KB is 1024 Bytes, So $1024 \times 1024 = 10,48,576$ Bytes]

RAM is a volatile memory, which means that the information stored in it is not permanent. As soon as the power is turned off, whatever data that resides in RAM is lost. It allows both read and write operations.

3.5.2 Types of RAM

There are two basic types of RAM

- Dynamic RAM (DRAM)
- Static RAM (SRAM)

These two types differ in the technology they use to hold data. Dynamic RAM being a common type needs to be refreshed frequently. Static RAM needs to be refreshed less often, which makes it faster. Hence, Static RAM is more expensive than Dynamic RAM.

3.5.3 Read Only Memory (ROM)

Read Only Memory refers to special memory in a computer with pre-recorded data at manufacturing time which cannot be modified. The stored programs that start the computer and perform diagnostics are available in ROMs. ROM stores critical programs such as the program that boots the computer. Once the data has been written onto a ROM chip, it cannot be modified or removed and can only be read. ROM retains its contents even when the computer is turned off. So, ROM is called as a non-volatile memory.

3.5.3.1 Programmable Read Only Memory (PROM)

Programmable read only memory is also a non-volatile memory on which data can be written only once. Once a

program has been written onto a PROM, it remains there forever. Unlike the main memory, PROMs retain their contents even when the computer is turned off.

The PROM differs from ROM. PROM is manufactured as a blank memory, whereas a ROM is programmed during the manufacturing process itself. PROM programmer or a PROM burner is used to write data to a PROM chip. The process of programming a PROM is called burning the PROM.

3.5.3.2 Erasable Programmable Read Only Memory (EPROM)

Erasable Programmable Read Only Memory is a special type of memory which serves as a PROM, but the content can be erased using ultraviolet rays. EPROM retains its contents until it is exposed to ultraviolet light. The ultraviolet light clears its contents, making it possible to reprogram the memory.

An EPROM differs from a PROM, PROM can be written only once and cannot be erased. EPROMs are used widely in personal computers because they enable the manufacturer to change the contents of the PROM to replace with updated versions or erase the contents before the computer is delivered.



Figure 3.7 Erasable Programmable Read Only Memory



Most of the EPROM chips have a transparent area at the top surface which is covered by stickers. If it gets removed, the ultraviolet light in the sunlight may erase the contents.

3.5.3.3 Electrically Erasable Programmable Read Only Memory (EEPROM)

Electrically Erasable Programmable Read Only Memory is a special type of PROM that can be erased by exposing it to an electrical charge. Like other types of PROM, EEPROM retains its contents even when the power is turned off. Comparing with all other types of ROM, EEPROM is slower in performance.

3.5.4 Cache Memory

The cache memory is a very high speed and expensive memory, which is used to speed up the memory retrieval process. Due to its higher cost, the CPU comes with a smaller size of cache memory compared with the size of the main memory. Without cache memory, every time the CPU requests the data, it has to be fetched from the main memory which will consume more time. The idea of introducing a cache is that, this extremely fast memory would store data that is frequently accessed and if possible, the data that is closer to it. This helps to achieve the fast response time, Where response Time, (Access Time) refers to how quickly the memory can respond to a read / write request. Figure 3.8 shows the arrangement of cache memory between the CPU and the main memory.



Figure 3.8 Cache Memory Arrangement

3.6 Secondary Storage Devices

A computer generally has limited amount of main memory which is expensive and volatile. To store data and programs permanently, secondary storage devices are used. Secondary storage devices serve as a supportive storage to main memory and they are non-volatile in nature, secondary storage is also called as Backup storage

3.6.1 Hard Disks

Hard disk is a magnetic disk on which you can store data. The hard disk has the stacked arrangement of disks accessed by a pair of heads for each of the disks. The hard disks come with a single or double sided disk.

3.6.2 Compact Disc (CD)

A CD or CD-ROM is made from 1.2 millimeters thick, polycarbonate plastic material. A thin layer of aluminum or gold is applied to the surface. CD data is represented as tiny indentations known as "pits", encoded in a spiral track moulded into the top of the polycarbonate layer. The areas between pits are known as "lands". A motor within the CD player rotates the disk. The capacity of an ordinary CD-ROM is 700MB.



Fig 3.9 Compact Disc

3.6.3 Digital Versatile Disc (DVD)

A **DVD (Digital Versatile Disc or Digital Video Disc)** is an optical disc capable of storing up to 4.7 GB of data, more than six times what a CD can hold. DVDs are often used to store movies at a better quality. Like CDs, DVDs are read with a laser.

The disc can have one or two sides, and one or two layers of data per side; the number of sides and layers determines how much it can hold. A 12 cm diameter disc with single sided, single layer has 4.7 GB capacity, whereas the single sided, double layer has 8.5 GB capacity. The 8 cm DVD has 1.5 GB capacity. The capacity of a DVD-ROM can be visually determined by noting the number of data sides of the disc. Double-layered sides are usually gold-coloured, while single-layered sides are usually silver-coloured, like a CD.



Fig 3.10 Digital Versatile Disc

3.6.4 Flash Memory Devices

Flash memory is an electronic (solid-state) non-volatile computer

storage medium that can be electrically erased and reprogrammed. They are either EEPROM or EPROM. Examples for Flash memories are pendrives, memory cards etc. Flash memories can be used in personal computers, Personal Digital Assistants (PDA), digital audio players, digital cameras and mobile phones. Flash memory offers fast access times. The time taken to read or write a character in memory is called access time. The capacity of the flash memories vary from 1 Gigabytes (GB) to 2 Terabytes (TB). A sample of flash memory is shown in Figure 3.11.



Figure 3.11 Flash Memory

3.6.5 Blu-Ray Disc

Blu-Ray Disc is a high-density optical disc similar to DVD. Blu-ray is the type of disc used for PlayStation games and for playing High-Definition (HD) movies. A double-layer Blu-Ray disc can store up to 50GB (gigabytes) of data. This is more than 5 times the capacity of a DVD, and above 70 times of a CD. The format was developed to enable recording, rewriting and playback of high-definition video, as well as storing large amount of data. DVD uses a red laser to read and write data. But, Blu-ray uses a blue-violet laser to write. Hence, it is called as Blu-Ray.



Fig 3.12 Blu- Ray Disc

3.7 Ports and Interfaces



The Motherboard of a computer has many I/O sockets that are connected to the ports and interfaces found on the rear side of a computer (Figure 3.13). The external devices can be connected to the ports and interfaces. The various types of ports are given below:

Serial Port: To connect the external devices, found in old computers.

Parallel Port: To connect the printers, found in old computers.

USB Ports: To connect external devices like cameras, scanners, mobile phones, external hard disks and printers to the computer.

USB 3.0 is the third major version of the Universal Serial Bus (USB) standard to connect computers with other electronic gadgets as shown in Figure 3.13. USB 3.0 can transfer data up to 5 Giga byte/second. USB3.1 and USB 3.2 are also released.



Figure 3.13 USB 3.0 Ports

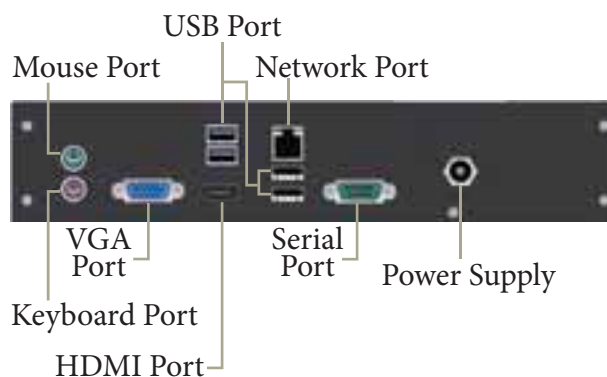


Fig 3.14 Ports and Interfaces

VGA Connector: To connect a monitor or any display device like LCD projector.

Audio Plugs: To connect sound speakers, microphone and headphones.

PS/2 Port: To connect mouse and keyboard to PC.

SCSI Port: To connect the hard disk drives and network connectors.

High Definition Multimedia Interface (HDMI)

High-Definition Multimedia Interface is an audio/video interface which transfers the uncompressed video and audio data from a video controller, to a compatible computer monitor, LCD projector, digital television etc.



Figure 3.15 HDMI Ports

Activity



Student Activity

- Identify the components of a computer
- Connecting external devices like printer/LCD projector.

Teacher Activity

- Show the components of a computer
- Display different ROM ICs
- Display the flash memory
- Demonstrate various ports and their usage

Evaluation



Part – I

Choose the correct answer

1. Which of the following is said to be the brain of a computer?
(a) Input devices (b) Output devices
(c) Memory device (d) Microprocessor
2. Which of the following is not the part of a microprocessor unit?
(a) ALU (b) Control unit
(c) Cache memory (d) register
3. How many bits constitute a word?
(a) 8 (b) 16
(c) 32 (d) determined by the processor used.
4. Which of the following device identifies the location when address is placed in the memory address register?
(a) Locator (b) encoder
(c) decoder (d) multiplexer
5. Which of the following is a CISC processor?
(a) Intel P6 (b) AMD K6 (c) Pentium III (d) Pentium IV
6. Which is the fastest memory?
(a) Hard disk (b) Main memory (c) Cache memory (d) Blue-Ray disk
7. How many memory locations are identified by a processor with 8 bits address bus at a time?
(a) 28 (b) 1024 (c) 256 (d) 8000
8. What is the capacity of 12cm diameter DVD with single sided and single layer?
(a) 4.7 GB (b) 5.5 GB (c) 7.8GB (d) 2.2 GB

9. What is the smallest size of data represented in a CD?
(a) blocks (b) sectors (c) pits (d) tracks
10. Display devices are connected to the computer through.
(a) USB port (b) Ps/2 port (c) SCSI port (d) VGA connector

Part – II

- (1) What are the parameters which influence the characteristics of a microprocessor?
- (2) What is an instruction?
- (3) What is a program counter?
- (4) What is HDMI?
- (5) Which source is used to erase the content of a EPROM?

Part – III

- (1) Differentiate Computer Organization from Computer Architecture.
- (2) Classify the microprocessor based on the size of the data.
- (3) Write down the classifications of microprocessors based on the instruction set.
- (4) Differentiate PROM and EPROM.
- (5) Write down the interfaces and ports available in a computer.
- (6) Differentiate CD and DVD
- (7) How will you differentiate a flash memory and an EEPROM?

Part – IV

- (1) Explain the characteristics of a microprocessor.
- (2) How the read and write operations are performed by a processor? Explain.
- (3) Arrange the memory devices in ascending order based on the access time.
- (4) Explain the types of ROM.



Computer hardware	The physical parts or components of a computer, such as the CPU, mother board, monitor, keyboard, etc.
Intel	Intel Corporation is an American multinational corporation and technology company involving in hardware manufacturing, especially mother board and processors
Silicon chip	Silicon chip is an integrated , set of electronic circuits on one small flat piece of semiconductor material, silicon.
Multipurpose	Multipurpose is several purpose
Address bus	Address bus is a collection of wires that carry the address as bits
Data bus	Data bus is a collection of wires to carry data in bits
Control bus	Control bus is a control line/collection of wires to control the operations/functions
Arithmetic operations	Arithmetic operations are the mathematical operations on data like add, subtract etc
Data Transfer	Data Transfer means moving data from one component to another
Logical operations	Logical operations are the operations on binary/Boolean data like AND, OR , NOT
Bidirectional	Bidirectional means both the directions/ways
Unidirectional	Unidirectional means only one direction
Access time	Access time is the time delay or latency between a request to an electronic system, and the access being completed or the requested data returned

Theoretical concepts of Operating System



Learning objectives

- ✓ To know the concept of Operating Systems and their types.
- ✓ To acquire the basic Knowledge of Operating Systems and its functions.

4.1 Introduction to Software

A software is set of instructions that perform specific task. It interacts basically with the hardware to generate the desired output.

4.1.1 Types of Software

Software is classified into two types:

- 1) Application Software
- 2) System Software



Application Software:

Application software is a set of programs to perform specific task. For example MS-word is an application software to create text document and VLC player is familiar application software to play audio, video files and many more.

System Software:

System software is a type of computer program that is designed to run the computer's hardware and application programs. For example Operating System and Language Processor

4.2 Introduction to Operating System (OS):

An Operating System (OS) is a system software which serves as an interface between a user and a computer.

This controls input, output and other peripheral devices such as disk drives, printers and electronic gadgets. The functions of an Operating System include file management, memory management, process management and device management and many more.



Figure: 4.1 Operating System

Without an Operating System, a computer cannot effectively manage all the resources. When a computer is switched on, the operating system is loaded in to

the memory automatically. A user cannot communicate directly with the computer hardware, unless an operating system is loaded.

Some of the popular Operating Systems used in personal computers and laptops are **Windows**, **UNIX** and **Linux**. The mobile devices mostly use Android and iOS as mobile **OS**.

4.2.1 Need for Operating System

Operating System has become essential to enable the users to design applications without the knowledge of the computer's internal structure of hardware. Operating System manages all the Software and Hardware. Most of the time there are many different computer programmes running at the same time, they all need to access the Computers, CPU, Memory and Storage. The need of Operating System is basically - an interface between the user and hardware.

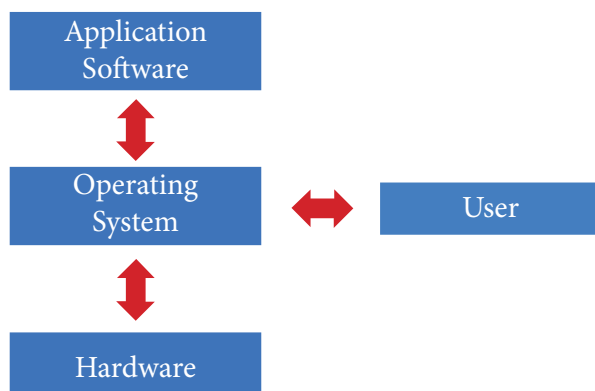


Figure: 4.2 Interaction of Operating system and user

Operating System works as translator, while it translates the user

request into machine language(Binary language), processes it and then sends it back to Operating System. Operating System converts processed information into user readable form

Uses of Operating Systems

The following are few uses of Operating System

The main use of Operating System is

- ▲ to ensure that a computer can be used to extract what the user wants it do.
- ▲ Easy interaction between the users and computers.
- ▲ Starting computer operation automatically when power is turned on (Booting).
- ▲ Controlling Input and Output Devices
- ▲ Manage the utilization of main memory.
- ▲ Providing security to user programs.

4.3 Types of Operating System

Operating System are classified into the following types depending on their processing capabilities.

4.3.1 Single User Operating Systems

An operating system allows only a single user to perform a task at a time. It is called as a Single user and single Task operating system. For a user, a task is a function such as printing a document, writing a file to disk, editing a file or

downloading a file etc. MS-DOS is an example for a single user and single task Operating System.

4.3.2 Multi-user Operating Systems

It is used in computers and laptops that allow same data and applications to be accessed by multiple users at the same time. The users can also communicate with each other. Windows, Linux and UNIX are examples for multi-user Operating System.



Build a cheap computer with raspbion OS and a Raspberry Pi. raspbion OS is a platform that's designed to teach how to build a computer, what every part of a circuit board does, and finally how to code apps or games. The platform is available in pre-designed kits

4.4 Key features of the Operating System

The various key features are given below

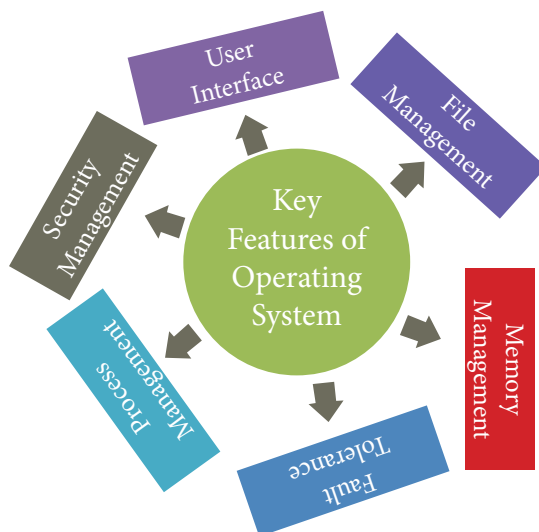


Figure: 4.3 Key Features of the Operating System

4.4.1 User Interface (UI)

User interface is one of the significant feature in Operating System. The only way that user can make interaction with a computer. If the computer interface is not user-friendly, the user slowly reduces the computer usage from their normal life. This is the main reason for the key success of GUI (Graphical User Interface) based Operating System. The GUI is a window based system with a pointing device to direct I/O, choose from menus, make selections and a keyboard to enter text. Its vibrant colours attract the user very easily. Beginners are impressed by the help and pop up window message boxes. Icons are playing vital role of the particular application.

Now Linux distribution is also available as GUI based Operating System. The following points are considered when User Interface is designed for an application.

1. The user interface should enable the user to retain this expertise for a longer time.
2. The user interface should also satisfy the customer based on their needs.
3. The user interface should save user's precious time. Create graphical elements like Menus, Window, Tabs, Icons and reduce typing work will be an added advantage of the Operating System.
4. The ultimate aim of any product is to satisfy the customer. The User Interface is also to satisfy the customer.

5. The user interface should reduce number of errors committed by the user with a little practice the user should be in a position to avoid errors (Error Log File)

4.4.2 Memory Management

Memory Management is the process of controlling and coordinating computer's main memory and assigning memory block (space) to various running programs to optimize overall computer performance. The Memory management involves the allocation of specific memory blocks to individual programs based on user demands. At the application level, memory management ensures the availability of adequate memory for each running program at all times.

The objective of Memory Management process is to improve both the utilization of the CPU and the speed of the computer's response to its users via main memory. For these reasons the computers must keep several programs in main memory that associates with many different Memory Management schemes.

The Operating System is responsible for the following activities in connection with memory management:

- Keeping track of which portion of memory are currently being used and who is using them.
- Determining which processes (or parts of processes) and data to move in and out of memory.

- Allocation and de-allocation of memory blocks as needed by the program in main memory. (Garbage Collection)

4.4.3 Process management

Process management is function that includes creating and deleting processes and providing mechanisms for processes to communicate and synchronize with each other.

A process is the unit of work (program) in a computer. A word-processing program being run by an individual user on a computer is a process. A system task, such as sending output to a printer or screen, can also be called as a Process.

A computer consists of a collection of processes, they are classified as two categories:

- Operating System processes which is executed by system code
- User Processes which is execute by user code

All these processes can potentially execute concurrently on a single CPU.

A process needs certain resources including CPU time, memory, files and I/O devices to finish its task.

The Operating System is responsible for the following activities associated with the process management:

- Scheduling processes and threads on the CPUs

- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication

The following algorithms are mainly used to allocate the job (process) to the processor.

1. FIFO
2. SJF
3. Round Robin
4. Based on Priority

FIFO (First In First Out)Scheduling:

This algorithm is based on queuing technique. Assume that a student is standing in a queue to get grade sheet from his/her teacher. The other student who stands first in the queue gets his/her grade sheet first and leaves from the queue. Followed by the next student in the queue gets it collected and so on. This is the basic logic of the FIFO algorithm.

Technically, the process that enters the queue first is executed first by the CPU, followed by the next and so on. The processes are executed in the order of the queue.

SJF (Shortest Job First)Scheduling:

This algorithm works based on the size of the job being executed by the CPU.

Consider two jobs A and B.

- 1) A = 6 kilo bytes
- 2) B = 9 kilo bytes

First the job “A” will be assigned and then job “B” gets its turn.

Round RobinScheduling

The Round Robin (RR) scheduling algorithm is designed especially for time sharing systems. Jobs (processes) are assigned and processor time in a circular method. For example take three jobs A, B, C. First the job A is assigned to CPU then job B and job C and then again A, B and C and so on.

Based On Priority

The given job (process) is assigned based on a Priority. The job which has higher priority is more important than other jobs. Take two jobs A and B. Let the priority of A be 5 and priority B be 7.

Job B is assigned to the processor before job A.

4.4.4 Security Management

The major challenge in computer and software industry is to protect user’s legitimate data from hackers. The Operating System provides three levels of securities to the user end. They are

- (1) File access level
- (2) System level
- (3) Network level

In order to access the files created by other people, you should have the access permission. Permissions can either

be granted by the creator of the file or by the administrator of the system.

System level security is offered by the password in a multi-user environment.

Both windows and Linux offer the password facility.

Network security is an indefinable one. So people from all over the world try to provide such a security.

All the above levels of security features are provided only by the Operating System.

4.4.5 Fault Tolerance

The Operating Systems should be robust. When there is a fault, the Operating System should not crash, instead the Operating System have fault tolerance capabilities and retain the existing state of system.

4.4.6 File Management

File management is an important function of OS which handles the data storage techniques. The operating System manages the files, folders and directory systems on a computer. Any type of data in a computer is stored in the form of files and directories/folders through File Allocation Table (FAT). The FAT stores general information about files like filename, type (text or binary), size, starting address and access mode (sequential/indexed / indexed-sequential/ direct/relative). The file manager of the operating system helps to create, edit, copy, allocate memory to the files and also

updates the FAT. The OS also takes care of the files that are opened with proper access rights to read or edit them. There are few other file management techniques available like Next Generation File System (NTFS) and ext2(Linux).

4.4.7 Multi-Processing

This is a one of the features of Operating System. It has two or more processors for a single running process (job). Processing takes place in parallel is known as parallel processing. Each processor works on different parts of the same task or on two or more different tasks. Since the execution takes place in parallel, this feature is used for high speed execution which increases the power of computing.

4.4.8 Time-sharing

This is a one of the features of Operating Systems. It allows execution of multiple tasks or processes concurrently. For each task a fixed time is allocated. This division of time is called Time- sharing. The processor switches rapidly between various processes after a time is elapsed or the process is completed.

For example assume that there are three processes called P1, P2, P3 and time allocated for each process 30, 40, 50 minutes respectively. If the process P1 completes within 20 minutes then processor takes the next process P2 for the execution. If the process P2 could not complete within 40 minutes, then the current process P2 will be paused and switch over to the next process P3.

4.4.9 Distributed Operating Systems

This feature takes care of the data and application that are stored and processed on multiple physical locations across the world over the digital network (internet/intranet). The Distributed Operating System is used to access shared data and files that reside in any machine around the world. The user can handle the data from different locations. The users can access as if it is available on their own computer.

The advantages of distributed Operating System are as follows:

- A user at one location can make use of all the resources available at another location over the network.
- Many computer resources can be added easily in the network
- Improves the interaction with the customers and clients.
- Reduces the load on the host computer.



Figure: 4.4 Distributed Operating Systems

4.5 Prominent Operating Systems

Prominent OS are as follows:

- UNIX
- Microsoft Windows
- Linux
- iOS
- Android

Modern operating systems use a Graphical User Interface (GUI). A GUI lets you use your mouse to click icons, buttons, menus and everything is clearly displayed on the screen using a combination of graphics and text elements.

OS can be either proprietary with a commercial license or can be open source.

Each Operating System's GUI has a different look and feel, so if you switch to a different Operating System, it may seem unfamiliar at first. However, modern Operating Systems are designed to be ease of use and most of the basic principles are the same.



Figure: 4.5 Various Operating Systems

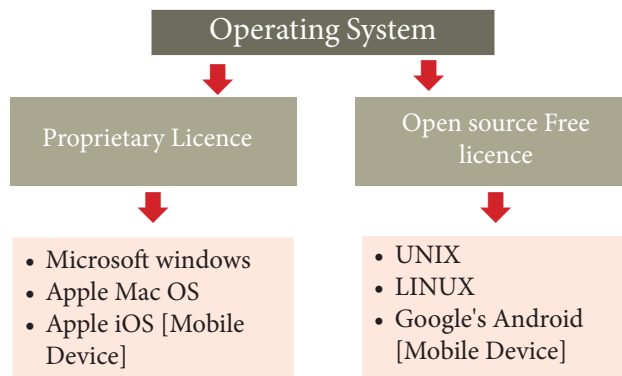


Figure: 4.6 Classification of Operating Systems according to availability

4.5.1 UNIX

UNIX is a family of multitasking, multi-user operating systems that derive originally from AT&T Bell Labs, where the development began in the 1970s by Ken Thompson and Dennis Ritchie.

4.5.2 Linux

Linux is a family of open-source operating systems. It can be modified and distributed by anyone around the world. This is different from proprietary software like Windows, which can only be modified by the company that owns it. The main advantage of Linux operating system is that it is open source. There are many versions and their updates. Most of the servers run on Linux because it is easy to customize.



Figure: 4.7 LINUX Ubuntu Operating System

There are a few different distributions of Linux, like Ubuntu, Mint, Fedora, RedHat, Debian, Google's Android, Chrome OS, and Chromium OS which are popular among users.

The Linux operating system was originated in 1991, as a project of “Linus Torvalds” from a university student of Finland. He posted information about his project on a news group for computer students and programmers. He received support and assistance from a large pool of volunteers who succeeded in creating a complete and functional Operating System. Linux is similar to the UNIX operating system.

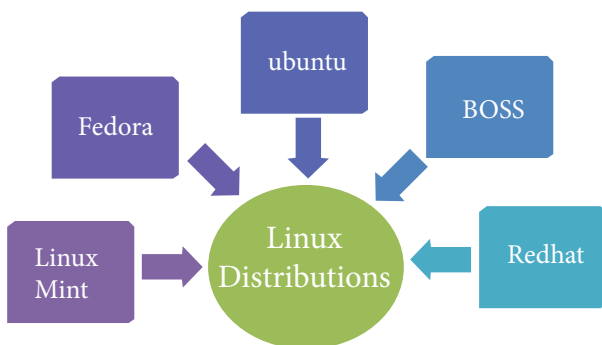


Figure: 4.8 Linux Distributions



Unix and the C programming language were developed by AT&T and distributed to government and academic institutions, which led to both being ported to a wider variety of machine families than any other operating system.

4.5.3 Microsoft Windows

Microsoft Windows is a family of proprietary operating systems designed by Microsoft Corporation and primarily targeted to Intel and AMD architecture based computers.



ReactOS is a Windows-alternative open source operating system, which is being developed on the principles of Windows - without using any of Microsoft's code.

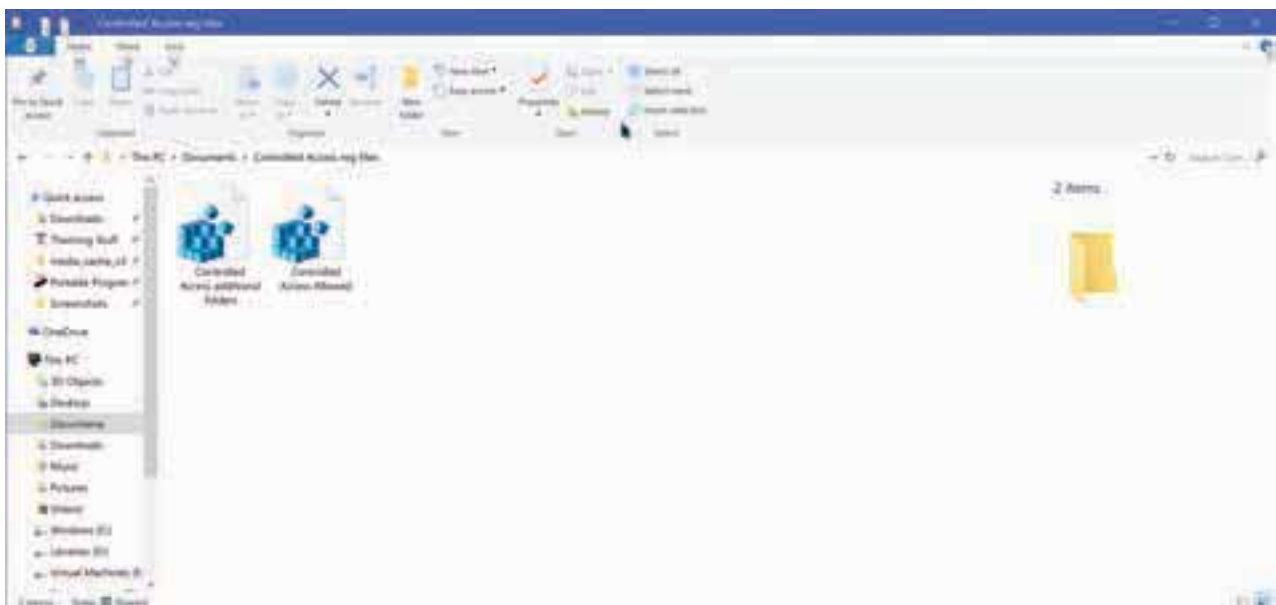


Figure: 4.9 Microsoft WindowsFolder Screen



Figure: 4.10 Microsoft Windows Home Screen

4.5.4 Operating systems for mobile devices

Mobile devices such as phones, tablets and MP3 players are different from desktop and laptop computers and hence they need special Operating Systems. Examples of mobile Operating Systems are Apple iOS and Google Android. The iOS running on an iPad is shown in Figure 4.12.



Figure: 4.11 Various Mobile Devices

Operating systems for mobile devices generally are not as fully featured as those made for desktop and laptop computers and they are not able to run all software.

4.5.5 Android

Android is a mobile operating system developed by Google, based on Linux and designed primarily for touch screen mobile devices such as smart phones and tablets. Google has further developed Android TV for televisions, Android Auto for cars and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronic gadgets.



Figure: 4.12 Android Mobile Open Source versions

4.5.6 iOS - iPhone OS

iOS (formerly iPhone OS) is a mobile Operating System created and developed by Apple Inc., exclusively for its hardware. It is the Operating System that presently powers many of the company's mobile devices, including the iPhone, iPad and iPod Touch. It is the second most popular mobile Operating System globally after Android.



Figure: 4.13 iOS - iPhone Home Screen



Student Activity

Activity 1: Draw a line between the operating system logo and the correct description.

A command-line operating system is an example of Open Source software development and Free Operating System



Is an Operating System that is very popular in universities, companies, bigenterprises etc



A popular Operating System for mobile phone technology which is not linked with Apple products.

UNIX



Used with Apple computers and works well with cloud computing.



Designed to be used for the Apple iPhone



The most popular GUI Operating System for personal computers.

Activity 2: Discuss and provide the suitable answers for the questions below.

One of the functions of an Operating System is multi-tasking

- 1) Explain one reason why multi-tasking is needed in an operating system
- 2) State two other function of an Operating System

Teacher Activity:

1. Install two different Operating Systems in a single computer.
2. Create a virtual Operating System using virtualization software.

Evaluation



Part I

- 1) Operating system is a
 - A)Application Software
 - B) Hardware
 - C)System Software
 - D)Component
- 2) Identify the usage of Operating Systems
 - A)Easy interaction between the human and computer
 - B)Controlling input & output Devices
 - C)Managing use of main memory
 - D)All the above
- 3) Which of the following is not a function of an Operating System?
 - A)Process Management
 - B)Memory Management
 - C)Security management
 - D)Complier Environment
- 4) Which of the following OS is a Commercially licensed Operating system?
 - A)Windows
 - B)UBUNTU
 - C)FEDORA
 - D)REDHAT
- 5) Which of the following Operating systems support Mobile Devices?
 - A)Windows 7
 - B)Linux
 - C)BOSS
 - D)iOS
- 6) File Management manages
 - A)Files
 - B)Folders
 - C)Directory systems
 - D)All the Above
- 7) Interactive Operating System provides
 - A)Graphics User Interface (GUI)
 - B)Data Distribution
 - C)Security Management
 - D)Real Time Processing
- 8) Android is a
 - A)Mobile Operating system
 - B)Open Source
 - C)Developed by Google
 - D)All the above



- 9) Which of the following refers to Android operating system's version?
A)JELLY BEAN B)UBUNTU C)OS/2 D)MITTIKA

Part II

- 1) What are the advantages of memory management in Operating System?
- 2) What is the multi-user Operating system?
- 3) What is a GUI?
- 4) List out different distributions of Linux operating system.
- 5) What are the security management features available in Operating System ?
- 6) What is multi-processing?
- 7) What are the different Operating Systems used in computer?

Part III

- 8) What are the advantages and disadvantages of Time-sharing features?
- 9) Explain and List out examples of mobile operating system.
- 10) What are the differences between Windows and Linux Operating system?
- 11) Explain the process management algorithms in Operating System.

Part IV

- 12) Explain the concept of a Distributed Operating System.
- 13) Explain the main purpose of an operating system.
- 14) Explain advantages and disadvantages of open source operating systems.

References

- 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

Working with typical operating systems

Part - I : Working with Windows



Learning Objectives

After learning the concepts in this chapter, the students will be able

- To know the concepts of Operating System.
- To know the versions of the windows operating system.
- To know the concepts like desktop and the elements of window.
- To explore the document window.
- To compare the different types of icons.
- To explore the windows directory structure.
- To practice creating files and folders in specific drives.
- To manage the files and folders.
- To know the procedure to start and shutdown the computer.



5.1. Introduction to Operating System

An Operating System (OS) is a system software (Figure 5.1) that enables the hardware to communicate and operate with other software. It also acts as an interface between the user and the hardware and controls the overall execution of the computer.

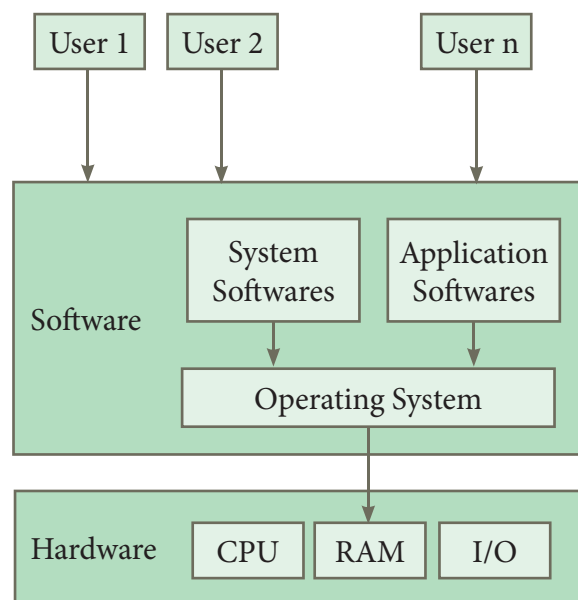


Figure 5.1. Overview of an Operating System

Following are some of the important functions of an Operating System as discussed in the previous chapter:

- Memory Management
- Process Management
- Device Management
- File Management

- Security Management
- Control overall system performance
- Error detecting aids
- Coordination between other software and users

The most popular Operating System are as follows:

- Windows Series - for desktop and laptop computers.
- Android - for smart phones.
- iOS - for Apple phones, i-Pad and i-Pod.
- Linux - Open source Operating System for desktop and server.

5.2. Introduction to Windows Operating System

Every computer needs an Operating System to function. Microsoft Windows is one of the most popular Graphical User Interface (GUI). Multiple applications can execute simultaneously in Windows, and this is known as **“Multitasking”**.




Windows Operating System uses both Keyboard and mouse as input devices. Mouse is used to interact with Windows by clicking its icons. Keyboard is used to enter alphabets, numerals and special characters.








Some of the functions of Windows Operating System are:

- Access applications (programs) on the computer (word processing, games, spread sheets, calculators and so on).
- Load any new program on the computer.
- Manage hardware such as printers, scanners, mouse, digital cameras etc.,
- File management activities (For example creating, modifying, saving, deleting files and folders).
- Change computer settings such as colour scheme, screen savers of your monitor, etc.

With reference to the Table 5.1, let us see the versions of Windows Operating System.

5.3. Various versions of Windows

Versions	Logo	Year	Specific features
Windows 1.x		1985	<ul style="list-style-type: none"> • Introduction of GUI in 16-bit processor • Mouse was introduced as an input device.
Windows 2.x		1987	<ul style="list-style-type: none"> • Supports to minimize or maximize windows. • Control panel feature was introduced with various system settings and customising options.
Windows 3.x		1992	<ul style="list-style-type: none"> • Introduced the concept of multitasking. • Supported 256 colours which brought a more modern, colourful look to the interface.

Windows 95		1995	<ul style="list-style-type: none"> • Introduced Start button, the taskbar, Windows Explorer and Start menu. • Introduced 32 - bit processor and focused more on multitasking.
Windows 98		1998	<ul style="list-style-type: none"> • Integration of the Web browser (Internet Explorer) with the Operating System. • DOS gaming began to disappear as Windows based games improved. • Plug and play feature was introduced.
Windows NT			<ul style="list-style-type: none"> • Designed to act as servers in network.
Windows Me		2000	<ul style="list-style-type: none"> • It introduced automated system diagnostics and recovery tools.
Windows 2000		2000	<ul style="list-style-type: none"> • Served as an Operating System for business desktop and laptop systems. • Four versions of Windows 2000 were released: Professional (for business desktop and laptop systems), Server (both a Web server and an office server), Advanced Server (for line-of-business applications) and Data Centre Server (for high-traffic computer networks).
Windows XP		2001	<ul style="list-style-type: none"> • Introduced 64-bit Processor. • Improved Windows appearance with themes and offered a stable version.
Windows Vista		2006	<ul style="list-style-type: none"> • Updated the look and feel of Windows.




Windows 7		2009	<ul style="list-style-type: none"> • Booting time was improved, introduced new user interfaces like Aero Peek, pinning programs to taskbar, handwriting recognition etc. and Internet Explorer 8.
Windows 8		2012	<ul style="list-style-type: none"> • Windows 8 is faster than previous versions of Windows. • Start button was removed. • Windows 8 takes better advantage of multi-core processing, solid state drives (SSD), touch screens and other alternate input methods. • Served as common platform for mobile and computer.
Windows 10		2015	<ul style="list-style-type: none"> • Start Button was added again. • Multiple desktop. • Central Notification Center for App notification and quick actions. • Cortana voice activated personal assistant.

Table 5.1 Versions of Windows Operating System.

5.4. Handling the mouse

Before learning Window Operating System, you should know more about mouse and its actions.

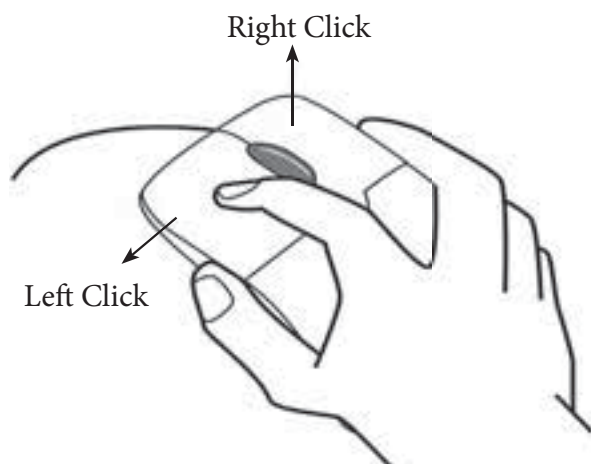


Figure 5.2. Mouse actions

The following are the mouse actions:

Action	Reaction
Point to an item	Move the mouse pointer over the item.
Click	Point to the item on the screen, press and release the left mouse button.
Right click	Point to the item on the screen, press and release the right mouse button. Clicking the right mouse button displays a pop up menu with various options.
Double-click	Point to the item on the screen, quickly press twice the left mouse button.
Drag and drop	Point to an item then hold the left mouse button as you move the pointer press and you have reached the desired position, release the mouse button.

5.5. Windows Desktop

The opening screen of Windows is called “Desktop”.

The desktop of your computer may look different from what is seen in Figure 5.3.

This is because Windows allows you to change the appearance of the desktop.

In Figure 5.3, the desktop shows the Start button, Taskbar, Notification Area and date and time.

5.5.1. The Icons

Icon is a graphic symbol representing the window elements like files, folders, shortcuts etc., Icons play a vital role in GUI based applications.

5.5.1.1. Standard Icons

The icons which are available on desktop by default while installing Windows OS are called standard icons. The standard icons available in all Windows OS are My Computer, Documents and Recycle Bin.

DO YOU KNOW? You can move to the Desktop any time by pressing the Winkey + D or using Aero Peek while working in any application. You can see Figure 5.4 to know where Aero peek lies in the Taskbar.



Figure 5.3. Microsoft Windows 7 Desktop



Figure 5.4. Aero peek button

5.5.1.2. Shortcut Icons:

Shortcut icons can be created for any application or file or folder. By double clicking the icon, the related application or file or folder will open. This represents the shortcut to open a particular application. (Figure5.5)

5.5.1.3. Disk drive icons:

The disk drive icons graphically represent five disk drive options. (i) Hard disk (ii) CD-ROM/DVD Drive (iii) Pen drive (iv) Other removable storage such as mobile, smart phone, tablet etc., (v) Network drives if your system is connected with other system. (Figure 5.6)



Figure 5.5.The types of Icons

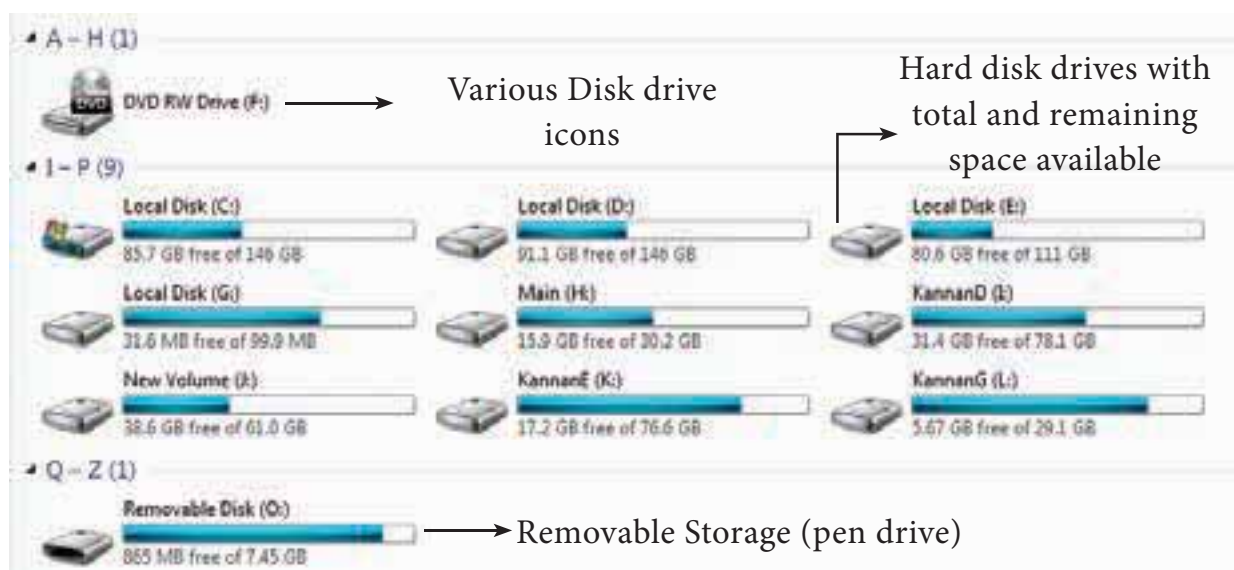


Figure 5.6.Disk drive Icons

5.6. The Window

Window is a typical rectangular area in an application or a document. It is an area on the screen that displays information for a specific program.

5.7. Application Window

It is an area on a computer screen with defined boundaries, and within which information is displayed. Such windows can be resized, maximized, minimized, placed side by side, overlap, and so on.

An Application Window contains an open application i.e. current application such as Word or Paint. When two or more windows are open, only one of them is active and the rest are inactive. Figures 5.7 and 5.8 display the Application Window of OpenOffice Writer and the appearance of the Multiple Windows opened (overlapped) in the Desktop.

5.8. Document Window

A document window is a section of the screen used to display the contents of a document. Figure 5.9 is an example of a document window.

Note

When you open any application, such as OpenOffice Writer, OpenOffice Impress or OpenOffice Calc etc., you will find two Windows on the screen. The larger Window is called the Application Window. This Window helps the user to communicate with the Application program. The smaller window, which is inside the Application Window, is called the Document window. This Window is used for typing, editing, drawing, and formatting the text and graphics.

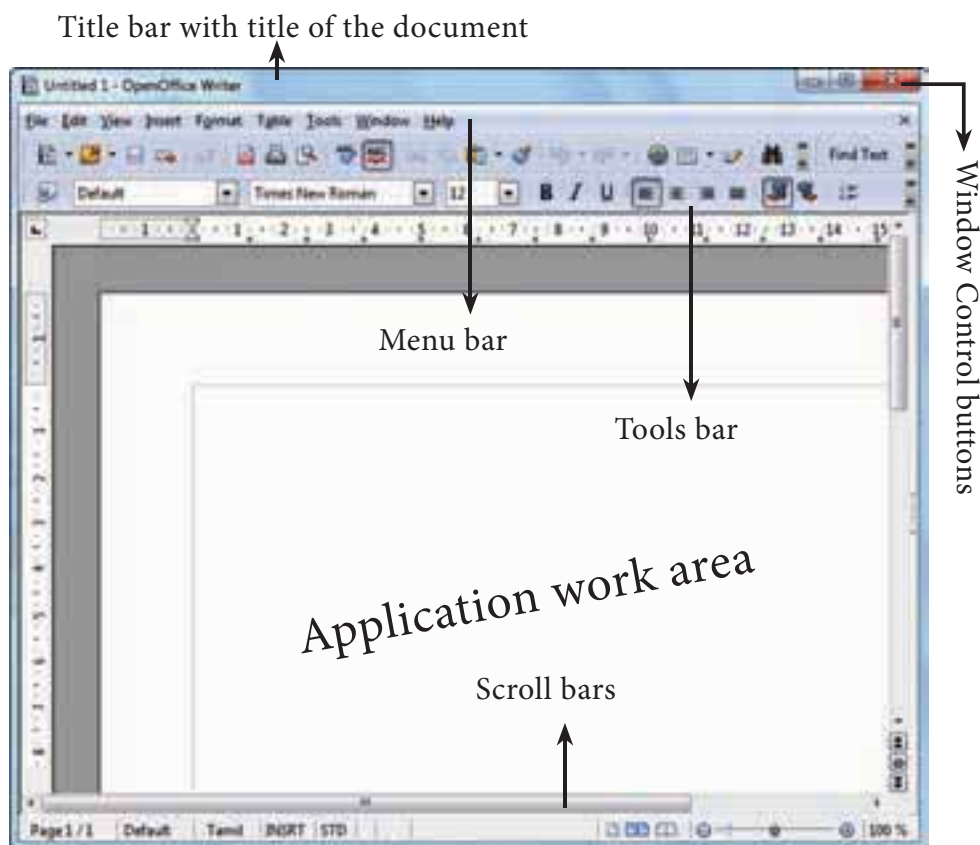


Figure 5.7. Application Window



Figure 5.8. Multiple Windows opened in Desktop

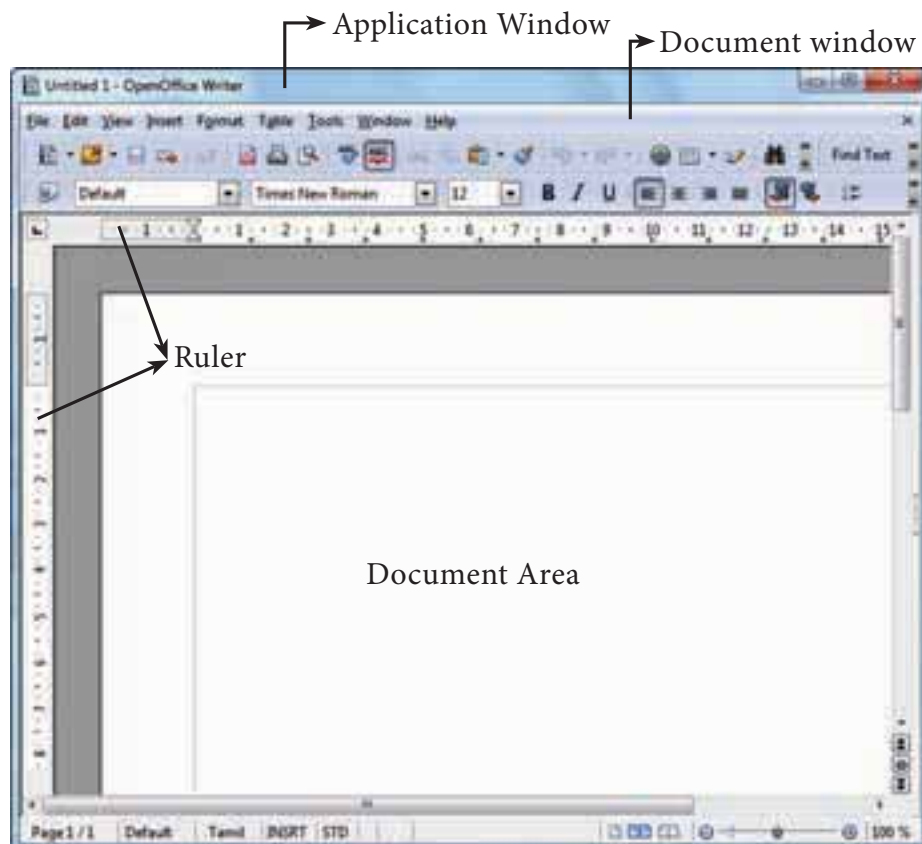


Figure 5.9. Document Window

5.9. Elements of a window

Figure 5.10 helps to understand the elements of a window.

5.9.1. Title Bar – The title bar will display the name of the application and the name of the document opened. It will also contain minimize, maximize and close button.

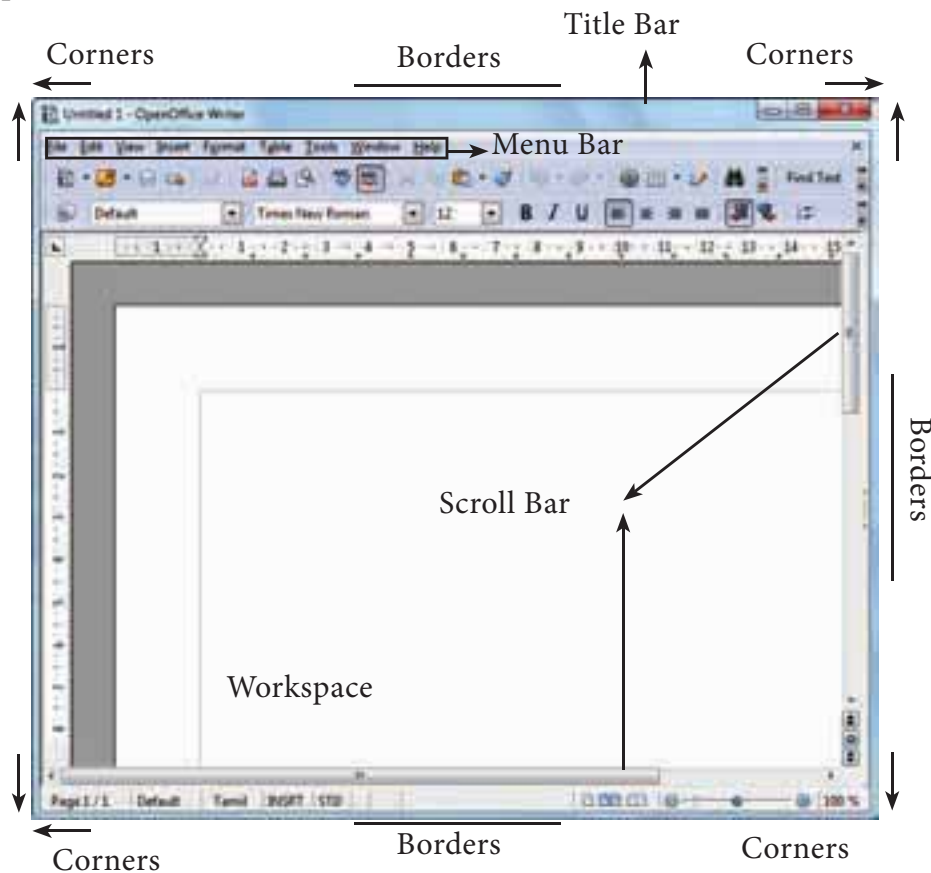


Figure 5.10 The elements of a window.

5.9.2 Menu Bar

The menu bar is seen under the title bar. Menus in the menu bar can be accessed by pressing Alt key and the letter that appears underlined in the menu title. Additionally, pressing Alt or F10 brings the focus on the first menu of the menu bar.

In Windows 7, in the absence of the menu bar, click **Organize** and from the drop down menu, click the **Layout** option and select the desired item from that list.

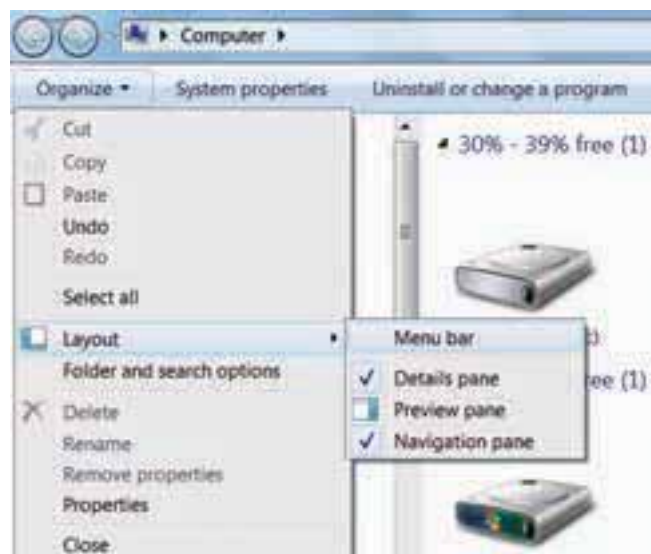


Figure 5.11. To display Menu Bar

Figure 5.11 helps to understand how to make menu bar visible in its absence.

5.9.3. The Workspace

The workspace is the area in the document window to enter or type the text of your document. Figure 5.10 Shows the workspace area in the document window.

5.9.4. Scroll bars - The scroll bars are used to scroll the workspace horizontally or vertically. Figure 5.10 shows the Scroll bars.

5.9.5. Corners and borders

The corners and borders of the window helps to drag and resize the windows. The mouse pointer changes to a double headed

arrow when positioned over a border or a corner. Drag the border or corner in the direction indicated by the double headed arrow to the desired size as shown in Figure 5.10. The window can be resized by dragging the corners diagonally across the screen.

5.10. Explore the Computer

5.10.1. Start Menu

In the lower left-hand corner of the windows screen is the Start button. When you click on the button, the Start menu will appear. Using the start menu, you can start any application.

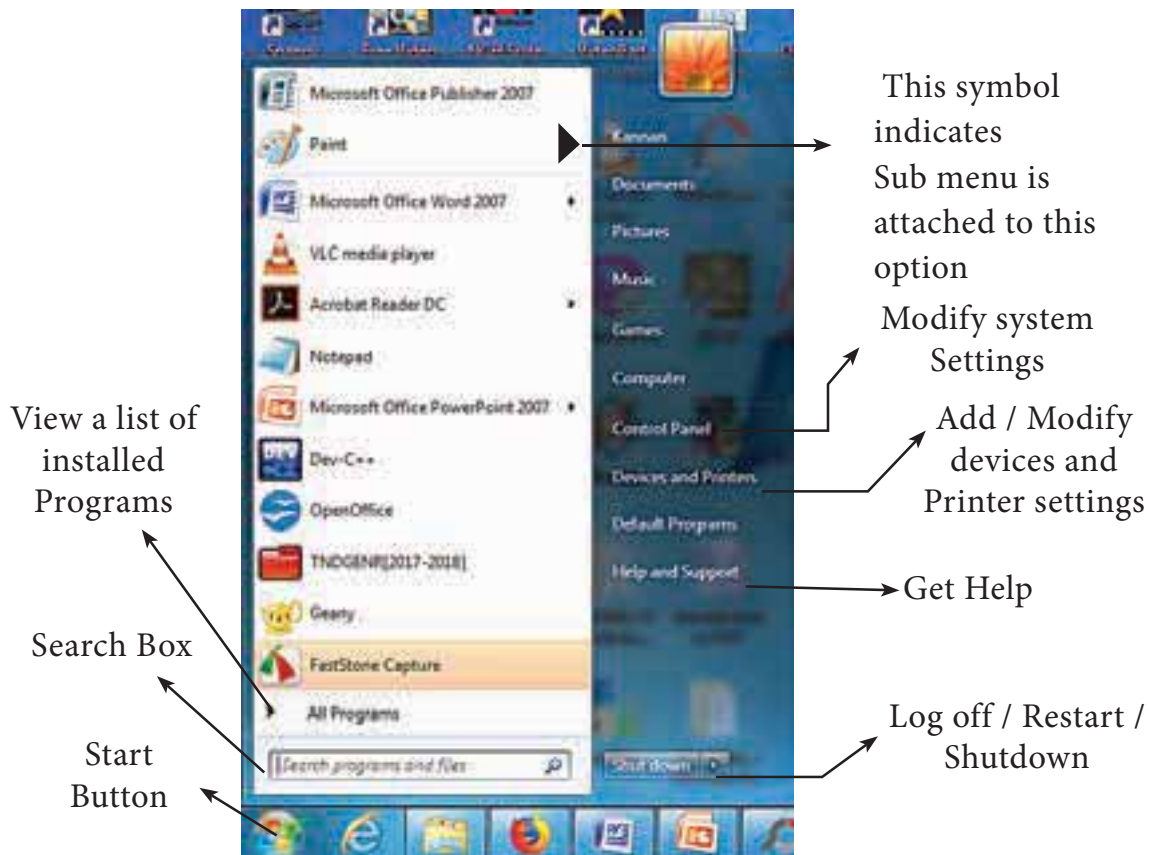


Figure 5.12 - Start Menu

Taskbar

At the bottom of the screen is a horizontal bar called the taskbar. This bar contains (from left to right) the Start button, shortcuts to various programs, minimized programs and in the extreme right corner you can see the system tray which consist of volume

control, network, date and time etc. Next to the Start button is the quick Launch Toolbar which contains task for frequently used applications.

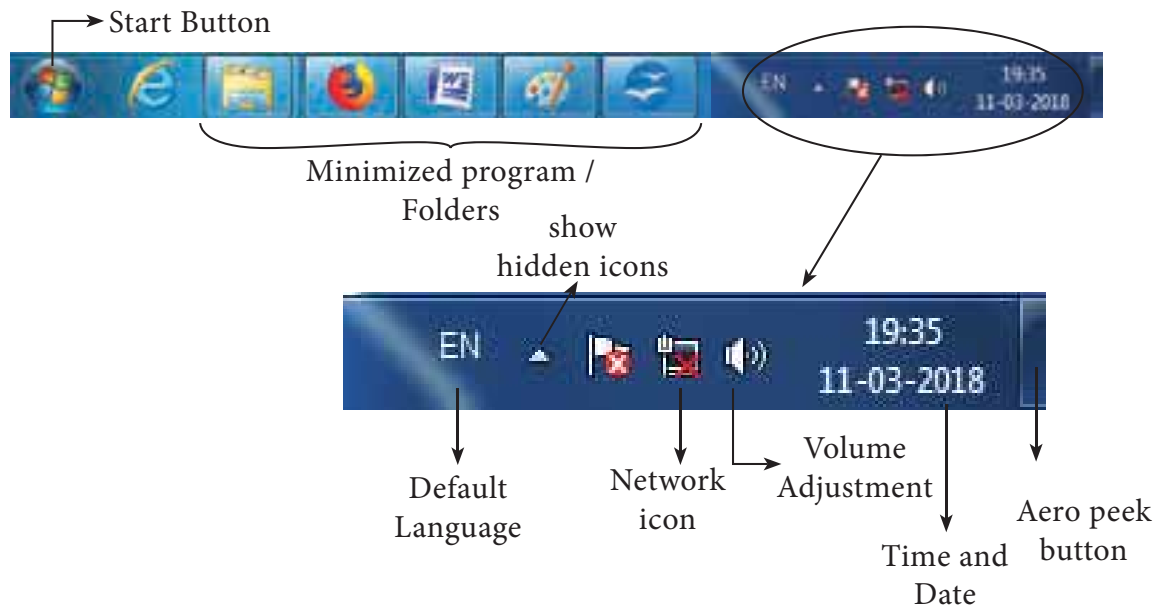


Figure 5.13.Taskbar

5.10.2. Computer Icon

By clicking this icon, the user can see the disk drivers mounted in the system. In windows XP, Vista, this icon is called "My computer" in Windows 8 and 10, it is called "This PC". The functionality of computer icon remains the same in all versions of windows as shown in Figure 5.14.

5.10.3. Starting and Closing Applications

Most of the applications installed on your computer are available through the start menu. Depending on the system setup, the applications in the Start menu varies. To start an application:

1. Click the Start button and then point to All Programs. The Program menu appears. (Figure 5.15)
2. Point to the group that contains the application you want to start, and then click the application name.



Figure 5.14. Computer icon in versions of Windows OS

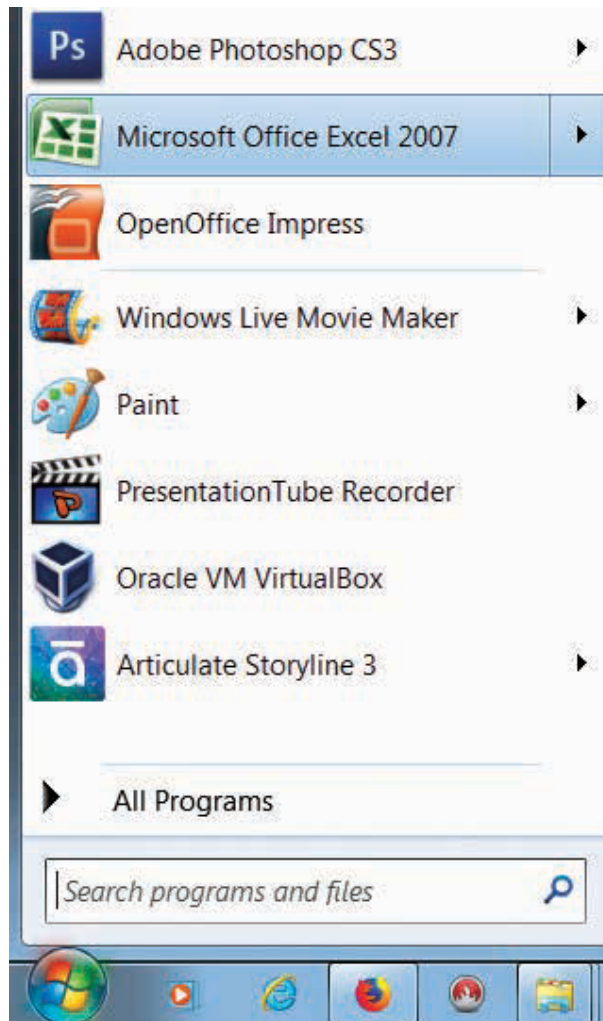


Figure 5. 15.Starting a applicatioin using Start menu

3. You can also open an application by clicking Run on the Start menu, and the name of the application. (Figure 5.16)

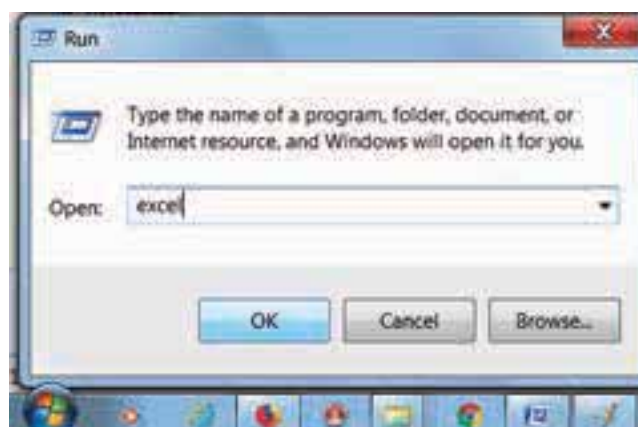


Figure 5.16.Starting a program using Run option

4. To quit a application, click the Close button in the upper right corner of the application window. (Figure 5.17)

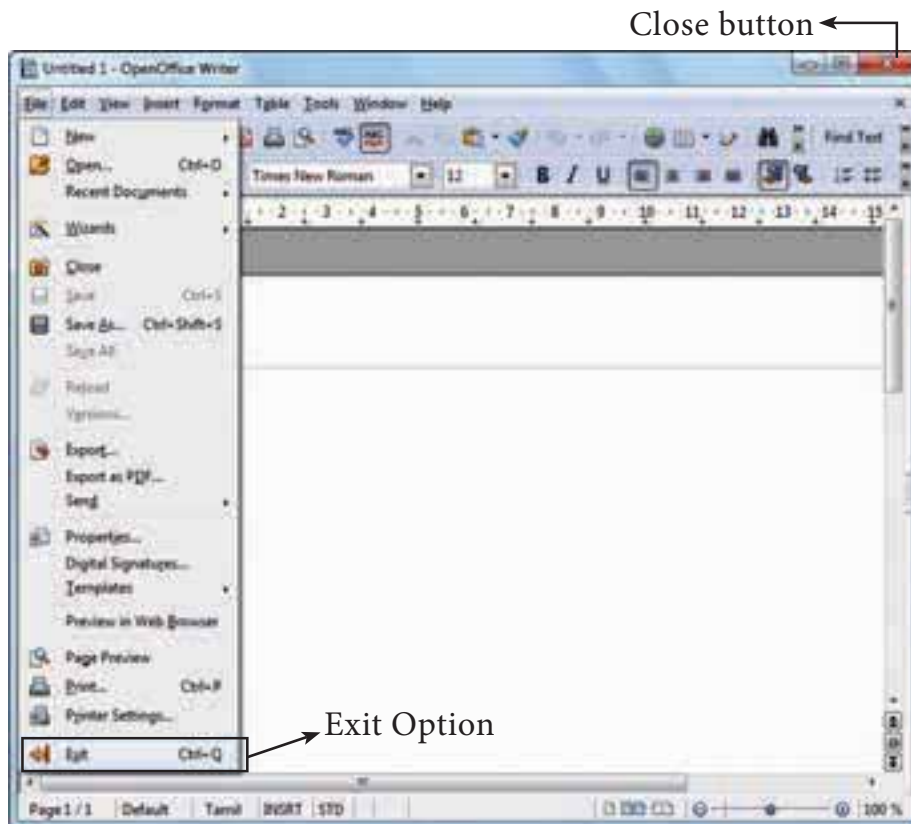


Figure 5.17. Closing the application using Close button and Exit option

5. You can also quit an application by clicking on File → Exit and File → Close option in Windows 7. (Figure 5.17)

Workshop

1. ♦ Start the application Wordpad using Start menu and Run option.
- ♦ Close the Wordpad application using File menu.

5.11. Managing Files and Folders

In Windows 7, you can organize your documents and programs in the form of files and folders. You can move, copy, rename, delete and search the files and folders.

5.11.1. Creating files and Folders

5.11.1.1 Creating Folders

You can store your files in many locations – on the hard disk or in other devices. To better organise your files, you can store them in folders.

There are two ways in which you can create a new folder:

Method I:

Step 1: Open **Computer** Icon.

Step 2: Open any drive where you want to create a new folder. (For example select D:)

Step 3: Click on File → New → Folder.

Step 4: A new folder is created with the default name “New folder”. (Figure 5.19)

Step 5: Type in the folder name and press Enter key. (Figure 5.20 shows the newly created Folder named “Test Folder”).

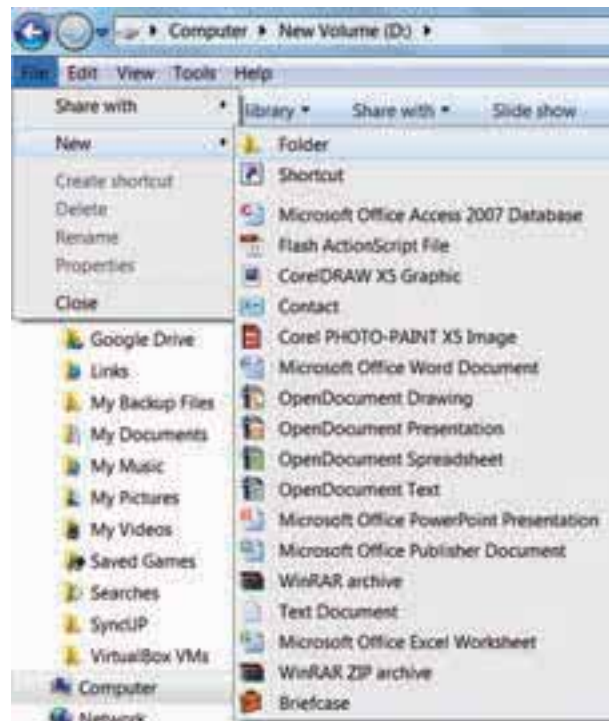


Figure 5.18. Creating a Folder using File menu

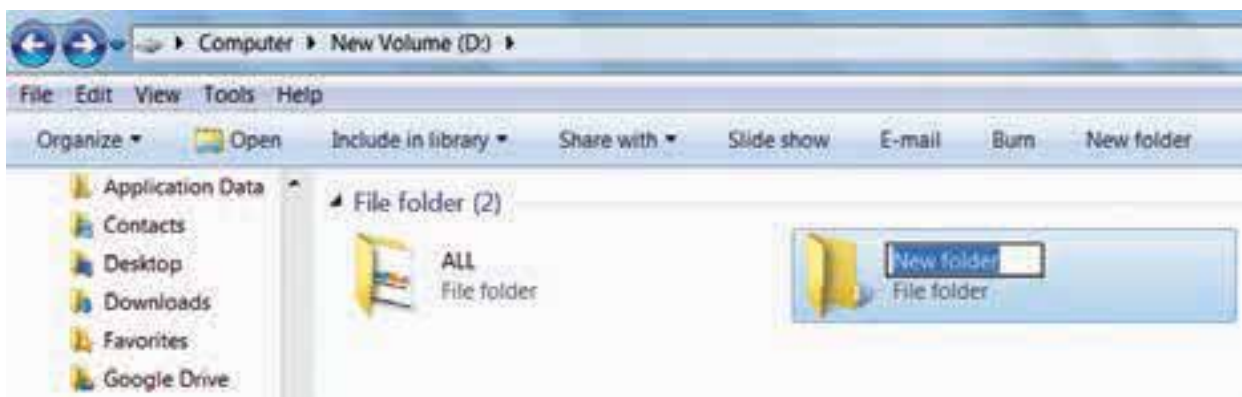


Figure 5.19. New Folder created with the default name

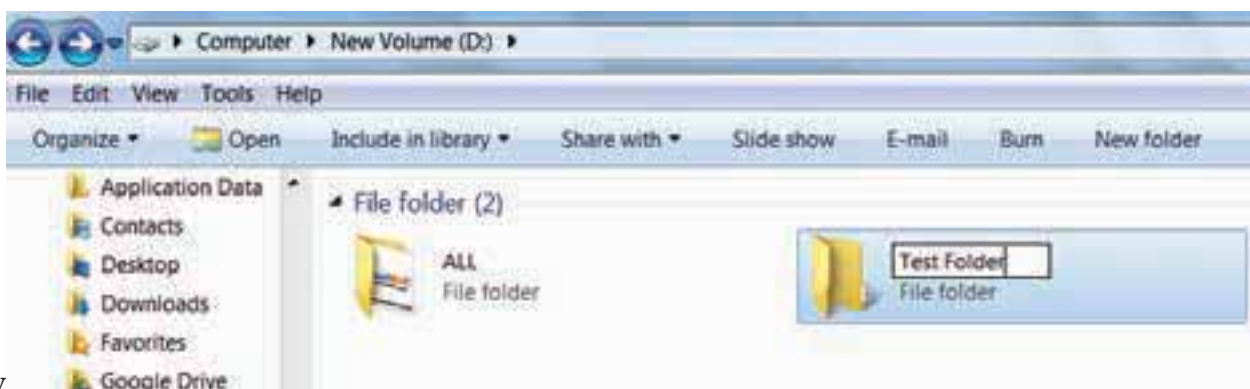


Figure 5.20. Renaming the new Folder

Method II:

In order to create a folder in the desktop:

Step 1: In the Desktop, right click → New → Folder. (Figure 5.21 Shown the procedure)

Step 2: A Folder appears with the default name “New folder” and it will be highlighted as shown in the Figure 5.22.

Step 3: Type the name you want and press Enter Key.

Step 4: The name of the folder will change.

Workshop

2. Create a Folder in My Documents with your name using any one of the methods discussed.

5.11.1.2 Creating Files (Wordpad)

Wordpad is an in-built word processor application in Windows OS to create and manipulate text documents.

In order to create files in wordpad you need to follow the steps given below.

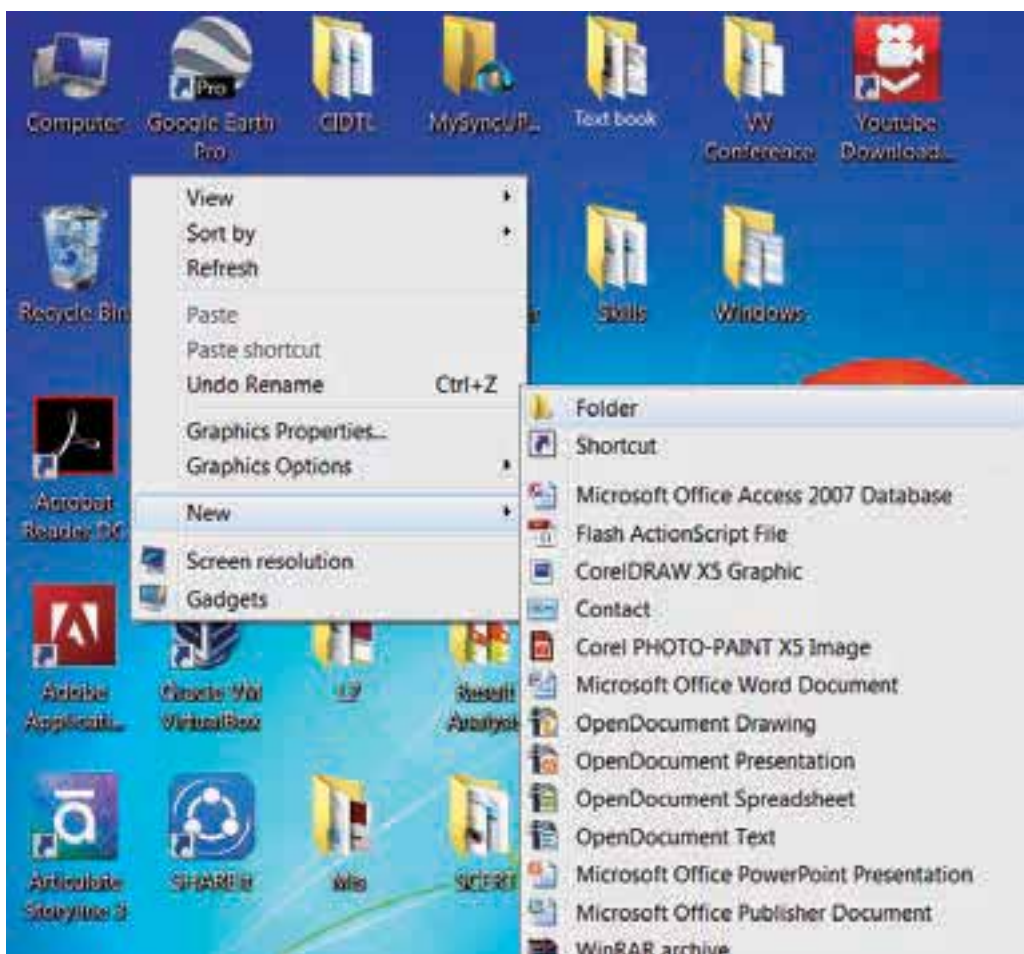


Figure 5.21. Creating a folder in the desktop

1. Click Start → All Programs → Accessories → Wordpad or Run → type Wordpad, click OK. Wordpad window will be opened as shown in Figure 5.23.
2. Type the contents in the workspace and save the file using File → Save or Ctrl + S.
3. Save As dialog box will be opened.
4. In the dialog box, select the location where you want to save the file by using **look in** drop down list box.
5. Type the name of the file in the **file name** text box.
6. Click save button.



Figure 5.22 New folder icon on the desktop

Workshop

3. Open the Wordpad application and save it under a folder created with your name in My Documents.

5.11.2. Finding Files and Folders

You can use the **search** box on the **Start** menu to quickly search a particular folder or file in the computer or in a specific drive.

To find a file or folder:

1. Click the **Start** button, the **search** box appears at the bottom of the start menu.
2. Type the name of the file or the folder you want to search. Even if you give the part of the file or folder name, it will display the list of files or folders starting with the specified name. (Figure 5.24)
3. The files or the folders with the specified names will appear, if you click that file, it will directly open that file or the folder.

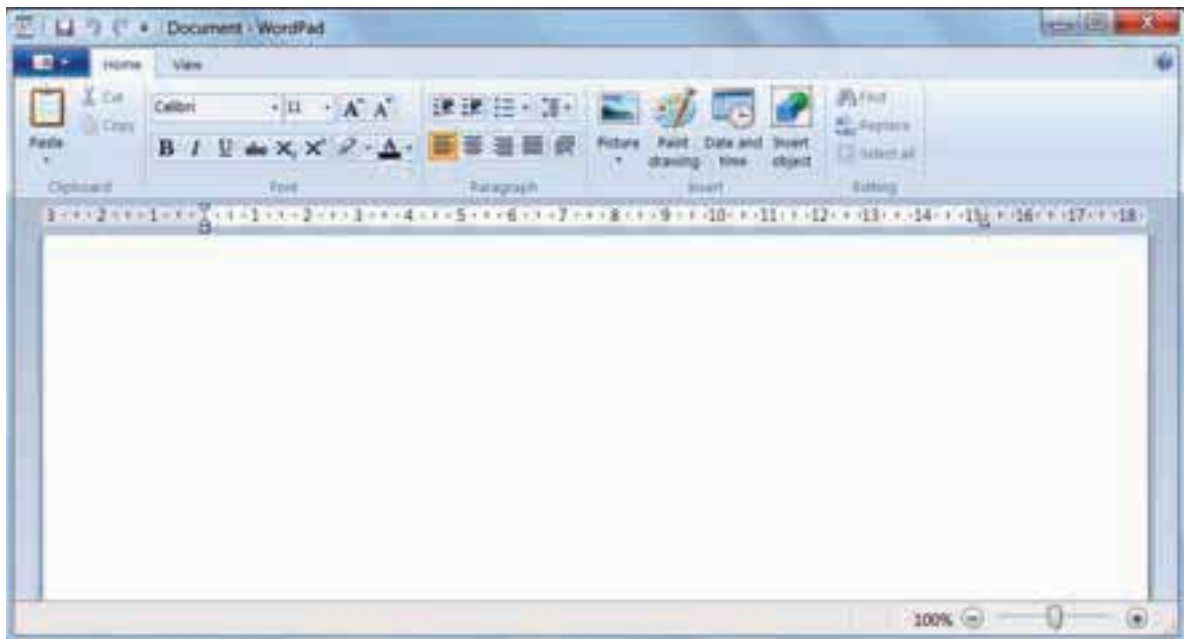


Figure 5.23. Wordpad - Word Processor application

4. There is another option called “**See more results**” which appears above the **search** box.
5. If you click it, it will lead you to a **Search Results** dialog box where you can click and open that file or the folder.

Searching Files or folders using Computer icon

1. Click **Computer Icon** from desktop or from **Start menu**.
2. The Computer disk drive screen will appear and at the top right corner of that screen, there is a **search** box option. (Figure 5.25)
3. Type the name of the file or the folder you want to search. Even if you give the part of the file or folder name, it will display the list of files or folders starting with the specified name.

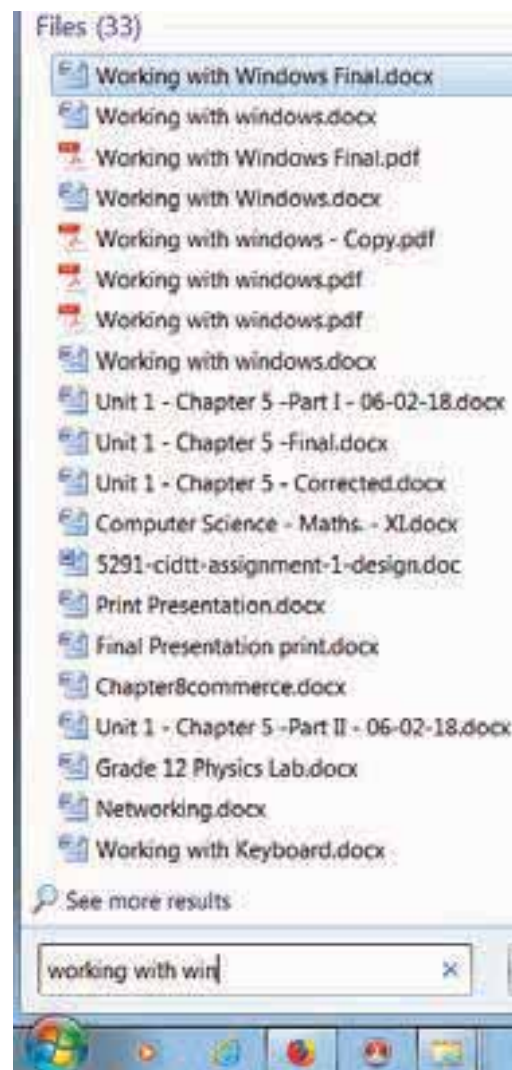


Figure 5.24. Finding a File/Folder using Start button

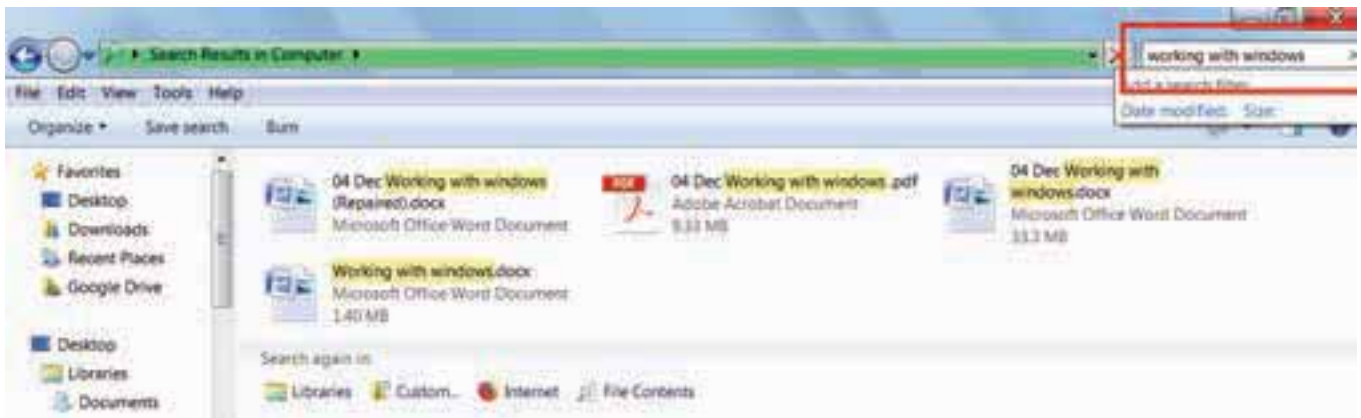


Figure 5.25. Finding a File/Folder in the Computer icon screen

4. Just click and open that file or the folder.

Workshop

4. Find the file created in Workshop-3 using the above procedure

5.11.3. Opening existing Files or Folders

The most common way of opening a file or a Folder is to double click on it.

5.11.4. Renaming Files or Folders

There are number of ways to rename files or folders. You can rename using the File menu, left mouse button or right mouse button.

Method 1

Using the FILE Menu

1. Select the File or Folder you wish to Rename.
2. Click File→ Rename.
3. Type in the new name.
4. To finalise the renaming operation, press Enter as in Figure 5.26.

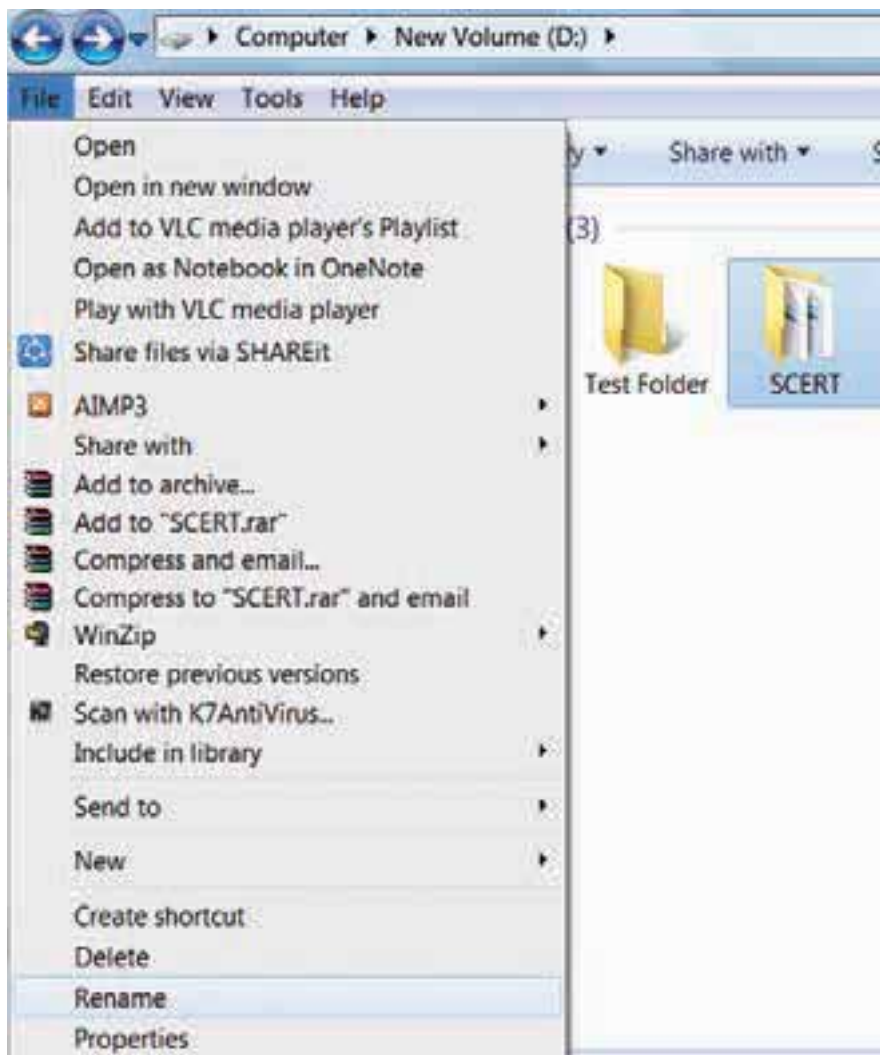


Figure 5.26. Renaming File/Folders using the File menu

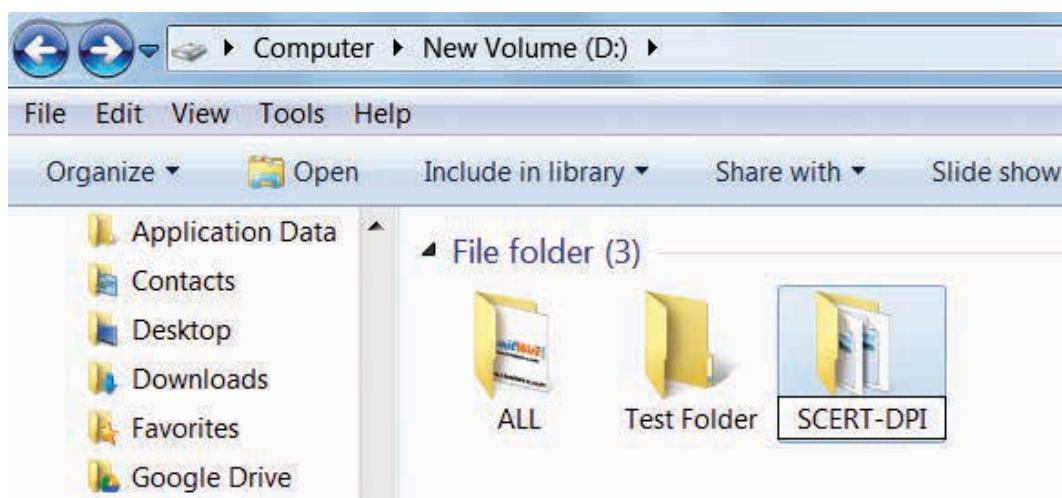


Figure 5.27. Folder renamed

Figure 5.27, you can see that the folder is renamed as SCERT-DPI from SCERT.

Method 2

Using the Right Mouse Button

1. Select the file or folder you wish to rename.
2. Click the right mouse button over the file or folder. (Figure 5.28)
3. Select Rename from the pop-up menu.
4. Type in the new name.
5. To finalise the renaming operation, press Enter.
6. Figure 5.29. Shows that the folder "New Folder" is renamed as C++.



Figure 5.28. Renaming File/Folders using the Right Mouse Button



Figure 5.29. New Folder is renamed as C++

Method 3

Using the Left Mouse Button

1. Select the file or folder you wish to rename.
2. Press F2 or click over the file or folder. A surrounding rectangle will appear around the name.
3. Type in the new name.
4. To finalise the renaming operation, press Enter.

Workshop

5. Rename the file created by you using the File menu, left mouse button or right mouse button.

5.11.5. Moving/Copying Files and Folders

You can move your files or folders to other areas using variety of methods.

Moving Files and Folders

Method I-CUT and PASTE

To move a file or folder, first select the file or folder and then choose one of the following:

- Click on the **Edit** → **Cut** or **Ctrl + X**
Or right **click** → **cut** from the pop-up menu.
- To move the file(s) or folder(s) in the new location, navigate to the new location and paste it using Click **Edit** → **Paste** from edit menu or **Ctrl + V** using keyboard.
- Or Right **click** → **Paste** from the pop-up menu. The file will be pasted in the new location.

Method II – Drag and Drop

In the disk drive window, we have two panes called left and right panes. In the left pane, the files or folders are displayed like a tree structure. In the right pane, the files inside the specific folders in the left pane are displayed with various options.

- In the right pane of the Disk drive window, select the file or folder you want to move.
- Click and drag the selected file or folder from the right pane, to the folder list on the left pane.
- Release the mouse button when the target folder is highlighted (active).
- Your file or folder will now appear in the new area. Figure 5.30 shows how

to move files or folders using drag and drop method.

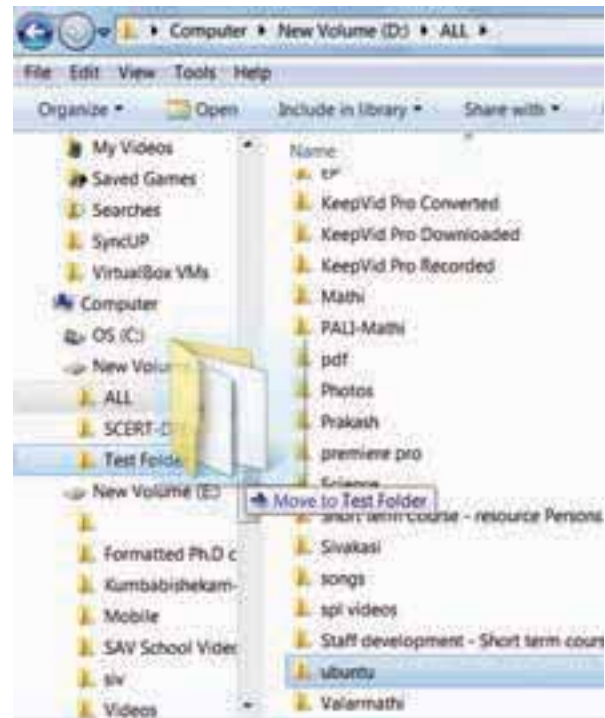


Figure 5.30. Moving the File/Folder using drag and drop

Copying Files and Folders

There are variety of ways to copy files and folders:

Method I - COPY and PASTE

To copy a file or folder, first select the file or folder and then choose one of the following:

- Click **Edit** → **Copy** or **Ctrl + C** or right **click** → **Copy** from the pop-up menu.
- To paste the file(s) or folder(s) in the new location, navigate to the target location then do one of the following:
- Click **Edit** → **Paste** or **Ctrl + V**.
- Or **Right click** → **Paste** from the pop-up menu.

Method II – Drag and Drop

- In the RIGHT pane, select the file or folder you want to copy.
- Click and drag the selected file and/or folder to the folder list on the left, and drop it where you want to copy the file and/or folder.
- Your file(s) and folder(s) will now appear in the new area.

Note

If you want to select multiple files or folders, use **Ctrl + Click**.



Figure 5.31. Selecting Computer option from Start menu

5.11.6. Copying Files and Folders to removable disk

There are several methods of transferring files to or from a removable disk.

- Copy and Paste
- Send To

METHOD I - Copy and Paste

- Plug the USB flash drive directly into an available USB port.
- If the USB flash drive or external drive folder does NOT open automatically, follow these steps:
- Click Start→Computer. (Figure 5.31)

- Double-click on the Removable Disk associated with the USB flash drive. (Figure 5.32)



Figure 5.32. Double Clicking Removable Disk

- Navigate to the folders in your computer containing files you want to transfer.

Right-click on the file you want to copy, then select **Copy**. (Figure 5.33)

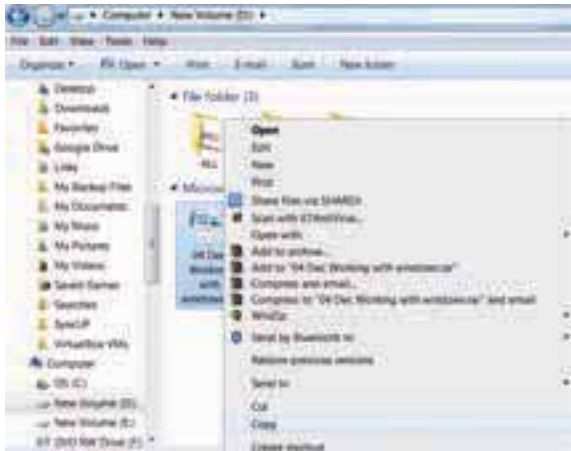


Figure 5.33. Copying File using right click

- Return to the Removable Disk window, right-click within the window, then select **Paste**. (Figure 5.34)

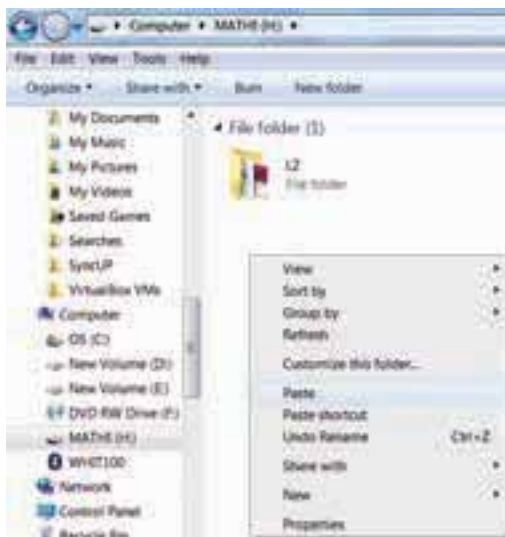


Figure 5.34. Pasting File using right click

METHOD II - Send To

- Plug the USB flash drive directly into an available USB port.
- Navigate to the folders in your computer containing files you want to transfer.
- Right-click on the file you want to transfer to your removable disk.
- Click **Send To** and select the Removable Disk associated with the USB flash drive. (Figure 5.35)

Workshop

6. ♦ Move the file created by you in My Documents to Drive D:.
- ♦ Copy the file created by you from drive D: to a removable disk.

5.11.7. Deleting Files and Folders

- When you delete a file or folder, it will move into the Recycle Bin.

To delete a file or folder:

Select the file or folder you wish to delete.



Figure 5.35. Copying File using Send to option

1. Right-click the file or folder, select **Delete** option from the pop-up menu or Click **File** → **Delete** or press **Delete** key from the keyboard.
2. The file will be deleted and moved to the Recycle bin.

Workshop

7. Delete the file created by you after duplicating the same under My Documents.

Note

To permanently delete a file or folder (i.e. to avoid sending a file or folder to the Recycle Bin), hold down the SHIFT key, and press **delete** on the keyboard.

Recycle Bin

Recycle bin is a special folder to keep the files or folders deleted by the user, which means you still have an opportunity to recover them. The user cannot access the files or folders available in the Recycle bin without restoring it. To restore file or folder from the Recycle Bin

- Open Recycle bin.
- Right click on a file or folder to be restored and select **Restore** option from the pop-up menu.
- To restore multiple files or folders, select **Restore all items**.
- To delete all files in the Recycle bin, select **Empty the Recycle Bin**.

5.12. Creating Shortcuts on the Desktop

Shortcuts to your most often used folders and files may be created and placed on the Desktop to help automate your work.

- Select the file or folder that you wish to have as a shortcut on the Desktop.
- Right click on the file or folder.
- Select **Send to** from the shortcut menu, then select Desktop (create shortcut) from the sub-menu.
- A shortcut for the file or folder will now appear on your desktop and you can open it from the desktop in the same way as any other icon. Figure 5.36.

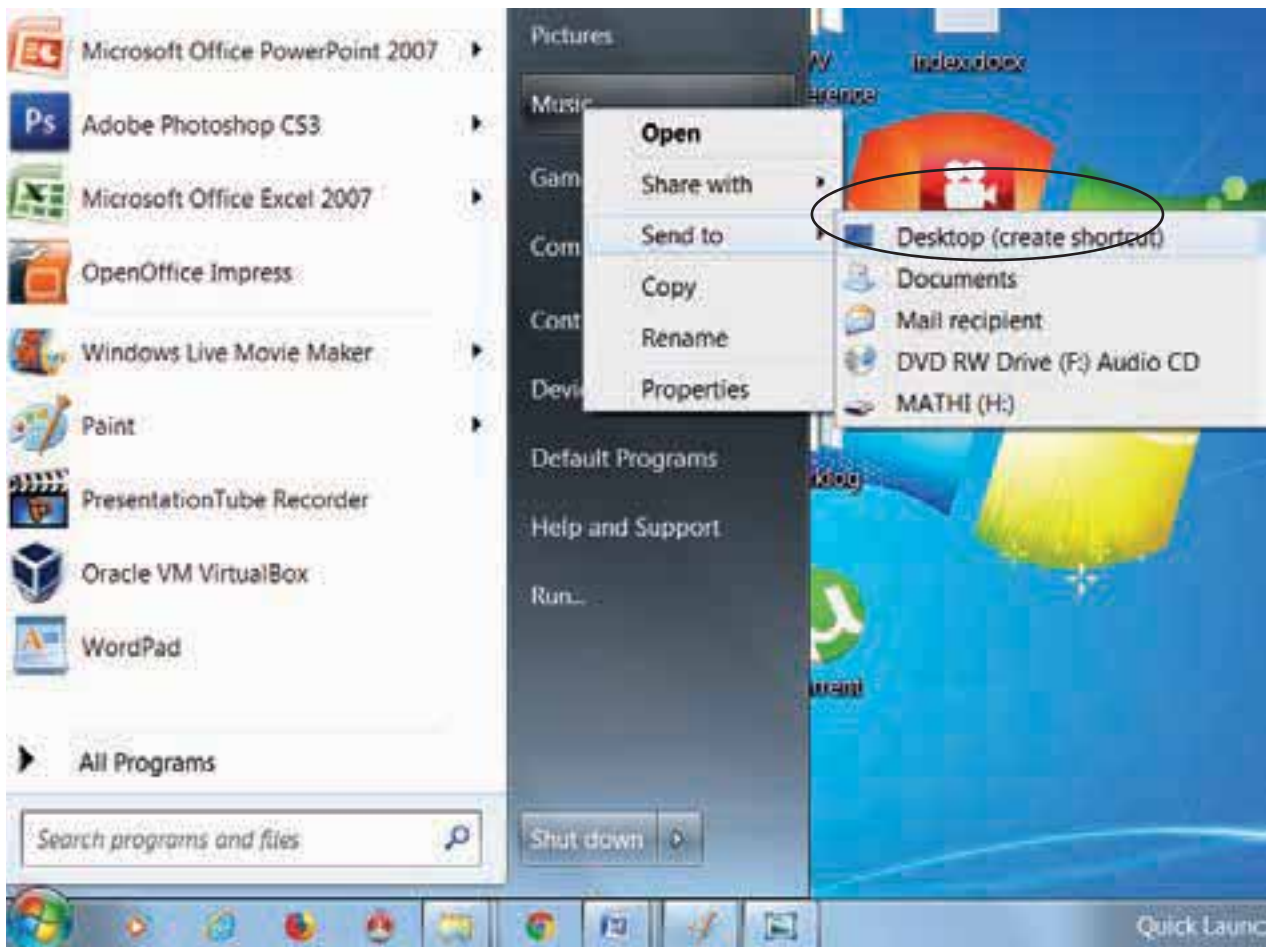


Figure 5.36 Creating Shortcut

5.13. Shutting down or Logging off a Computer

Once you have closed all open applications, you can either log off your computer or shut down the computer.

Log Off

To Log off/Shut down the computer:

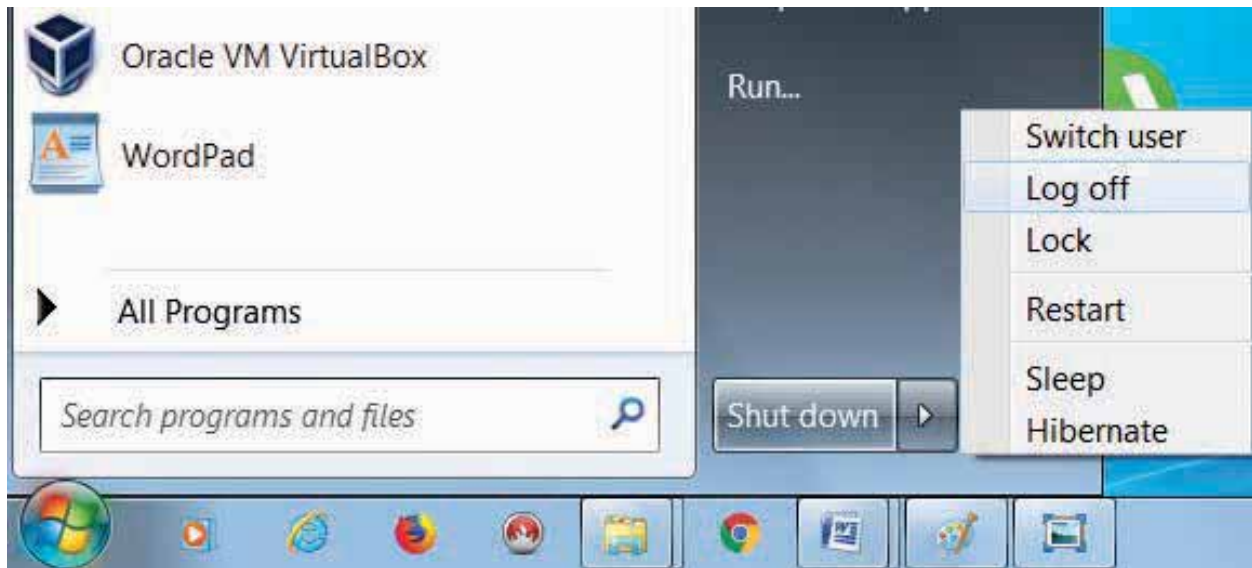


Figure 5.37. Log off option

- Click **start** → **log off** (click the arrow next to Shut down) or **Start** → **Shutdown** . (Figure 5.37.)
- If you have any open programs, then you will be asked to close them or windows will Force shut down, you will lose any un-saved information if you do this.
- **Switch User:** Switch to another user account on the computer without closing your open programs and Windows processes.
- **Log Off:** Switch to another user account on the computer after closing all your open programs and Windows processes.
- **Lock:** Lock the computer while you're away from it.
- **Restart:** Reboot the computer. (This option is often required as part of installing new software or Windows update.)
- **Sleep:** Puts the computer into a low-power mode that retains all running programs and open Windows in computer memory for a super-quick restart.
- **Hibernate** (found only on laptop computers): Puts the computer into a low-power mode after saving all running programs and open Wwindows on the machine's hard drive for a quick restart.

Part - II : Working with Linux (Ubuntu)

**Learning Objectives**

- To compare Windows Operating System with Ubuntu.
- To differentiate the Window elements or icons from Ubuntu Launcher.
- To explore how to copy, delete and rename files or folders in Ubuntu.
- To explore the differences in Windows 7, Windows 8 and Windows 10.
- To compare the Windows elements between Windows 7, Windows 8 and Windows 10.
- To format Files and folders in Windows 7, Windows 8 and Windows 10.

5.14. Open Source Operating System

Open Source refers to a program or software in which the source code is available in the web to the general public free of cost.

Open Source code is typically created as a collaborative effort in which programmers continuously improve upon the source code in the web and share the changes within the community.

5.15. Linux

Linux is one of the popular Open Source versions of the UNIX Operating System. It is Open Source as its source code is freely available.

The most popular Linux server distributors are:

- Ubuntu Linux
- Linux Mint
- Arch Linux
- Deepin
- Fedora
- Debian
- CentOS

**5.16. Ubuntu**

Ubuntu is a Linux-based operating system. It is designed for computers, smartphones, and network servers. The system is developed by a UK based company called Canonical Ltd.

Ubuntu was conceived in 2004 by Mark Shuttleworth, a successful South African entrepreneur, and his company Canonical Ltd.

5.16.1. Significant features of Ubuntu

- The desktop version of Ubuntu supports all normal software like Windows such as Firefox, Chrome, VLC, etc.
- It supports the office suite called LibreOffice.
- Ubuntu has in-built email software called Thunderbird, which gives the user access to email such as Exchange, Gmail, Hotmail, etc.
- There are free applications for users to view and edit photos, to manage and share videos.
- It is easy to find content on Ubuntu with the smart searching facility.

- The best feature is, it is a free operating system and is backed by a huge open source community.

5.17. Ubuntu desktop

There are many similarities between Ubuntu and other operating systems, such as Microsoft Windows, Apple. This is because they are all based on the concept of a Graphical User Interface (GUI).

The following are the names of the icons in the Ubuntu OS.

- Search your Computer
- Files
- Firefox Webbrowser

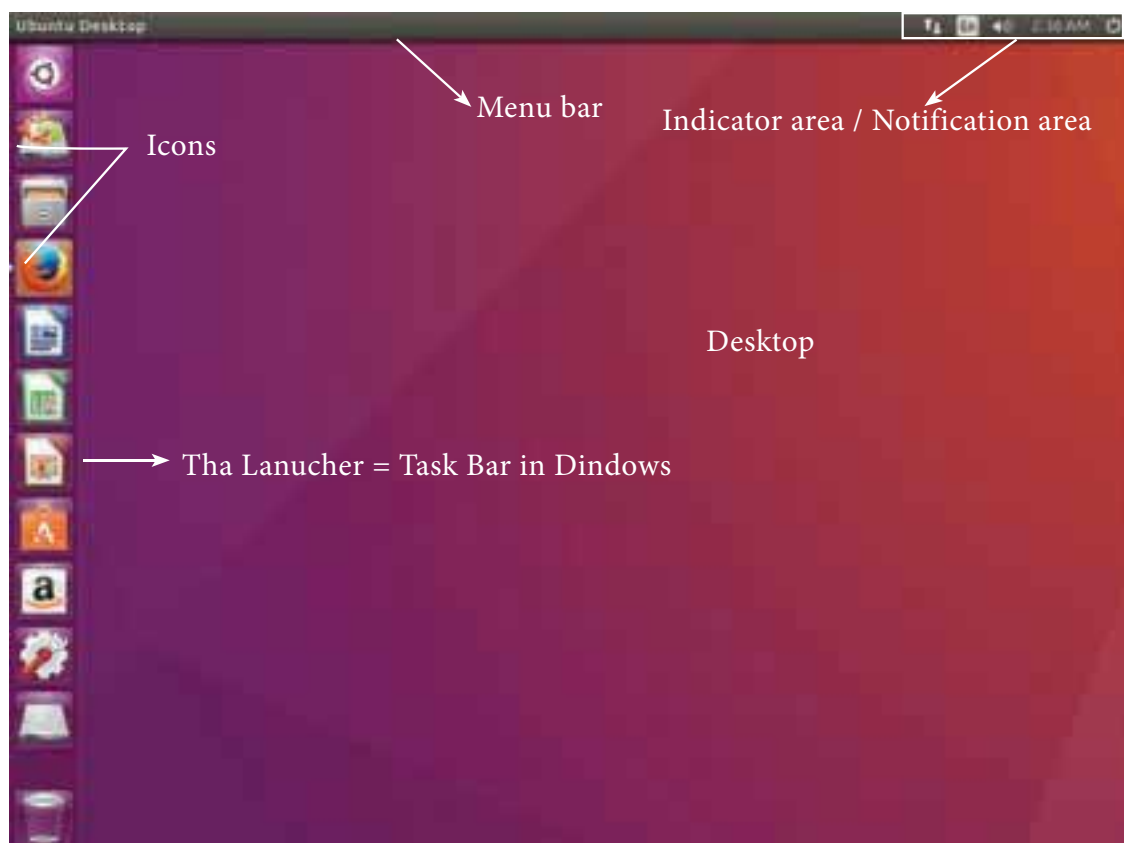


Figure 5.38. The Ubuntu 16.04 default desktop.

- LibreOffice Writer
- LibreOffice Calc
- LibreOffice Impress
- Ubuntu Software
- Amazon
- System Settings
- Trash

Figure 5.40 explains the icons in the Ubuntu operating system and their equivalent ones in the Windows operating system.

Menu bar The menu bar is located at the top of the screen. The menu bar incorporates common functions used in Ubuntu. The frequently used icons in the menu bar are found on the right. The most common indicators in the Menu bar are located in the indicator or notification area (Figure 5.39)



Figure 5.39 Indicators in the Menu bar

Network indicator - This manages network connections, allowing you to connect to a wired or wireless network.

Text entry settings - This shows the current keyboard layout (such as En, Fr, Ku, and so on) . If more than one keyboard layout is shown, it allows you to select a keyboard layout out of those choices. The keyboard indicator menu contains the following menu items: Character Map, Keyboard Layout Chart, and Text Entry Settings.

Messaging indicator- This incorporates your social applications. From here, you can access instant messenger and email clients.

Sound indicator - This provides an easy way to adjust the volume as well as access your music player.

Clock - This displays the current time and provides a link to your calendar and time and date settings.

Session indicator - This is a link to the system settings, Ubuntu Help, and session options (like locking your computer, user/guest session, logging out of a session, restarting the computer, or shutting down completely).

Title bar - The title bar shows the name of the currently selected directory. It also contains the Close, Minimize, and Maximize buttons.

Toolbar - The toolbar displays your directory browsing history (using two arrow buttons), your location in the file system, a search button, and options for your current directory view.

5.18. The desktop background

Below the menu bar at the top of the screen is an image covering the entire desktop. This is the default desktop background, or wallpaper, belonging to the default **Ubuntu 16.04** theme known as **Ambiance**. (Figure 5.38)



Figure 5.40 Ubuntu Desktop elements

5.19. The Launcher (Equivalent to Task bar)



The vertical bar of icons on the left side of the desktop is called the Launcher. The Launcher provides easy access to applications, mounted devices, and the Trash. All current applications on your system will place an icon in the Launcher. (Figure 5.40)

5.20. Elements of Ubuntu



5.20.1. Search your Computer Icon

This icon is equal to search button in Windows OS. Here, you have to give the name of the File or Folder for searching them. (Figure 5.40)

5.20.2. Files

This icon is equivalent to My Computer icon. From here, you can directly go to Desktop, Documents and so on. (Figure 5.40)

5.20.3.Firefox Web Browser

By clicking this icon, you can directly browse the internet. This is equivalent to clicking the Web Browser in Task bar in Windows. (Figure 5.40)

5.20.4.LibreOffice Writer

This icon will directly take you to document preparation

application like MS Word in Windows. (Figure 5.41)

5.20.5.Libre Office Calc

This icon will open LibreOffice Calc application. It is similar to MS Excel in Windows. (Figure 5.42)

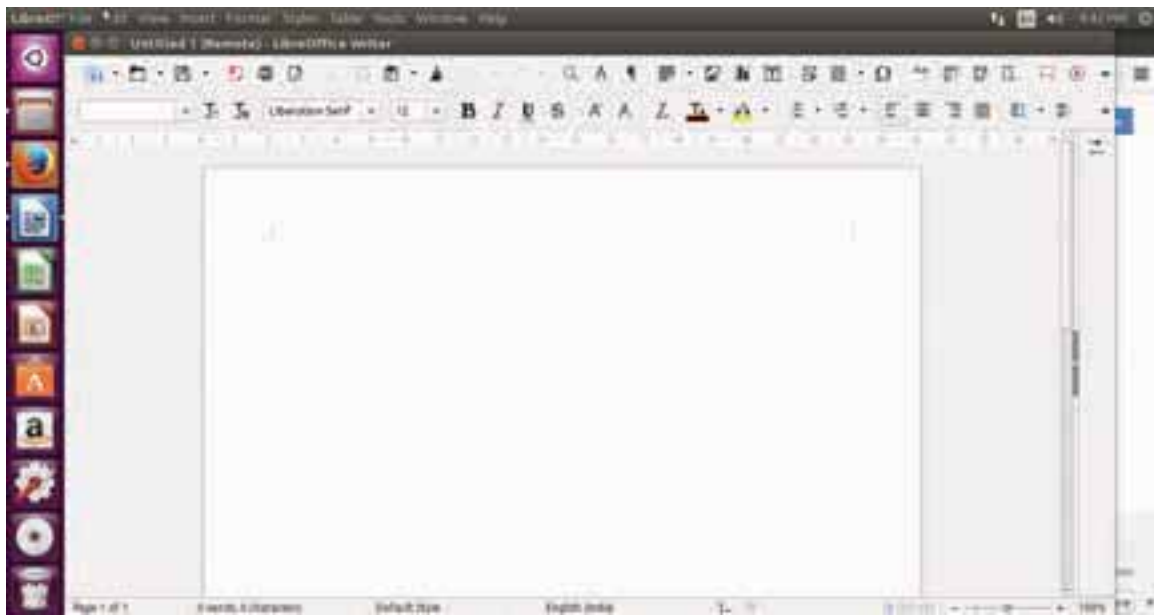


Figure 5.41 LibreOffice Writer Window

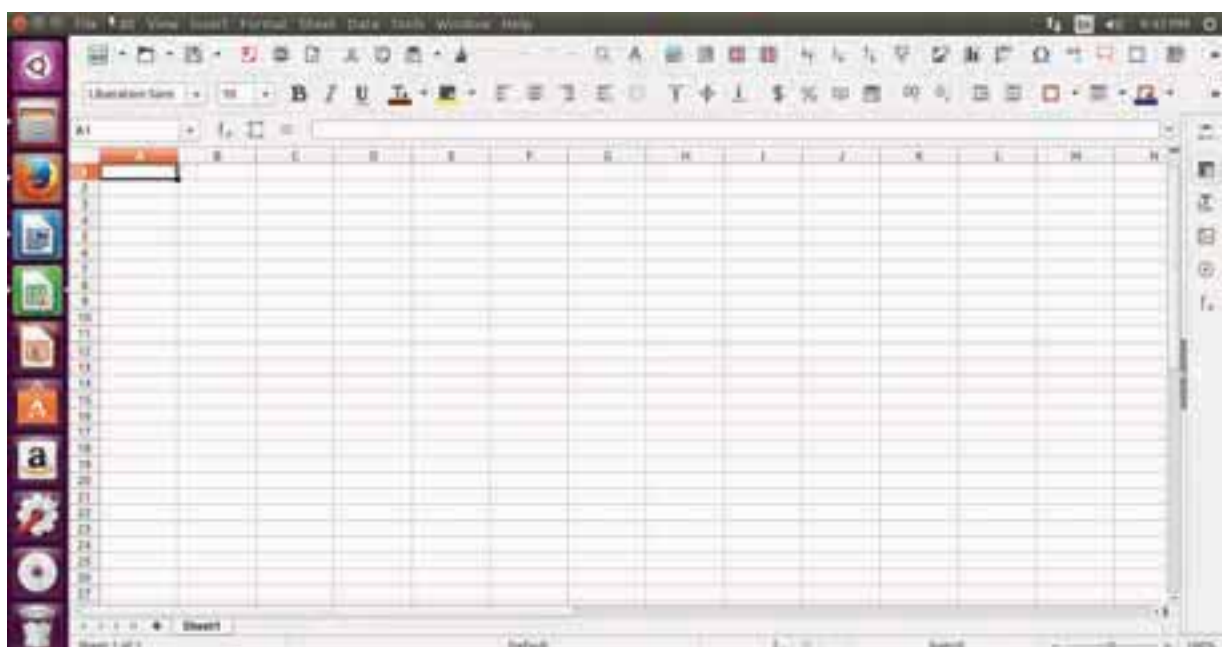


Figure 5.42 LibreOffice Calc Window

5.20.6 LibreOffice Impress

By clicking this icon, you can open LibreOffice Impress to prepare any presentations in Ubuntu like MS PowerPoint. (Figure 5.43)

5.20.7. Ubuntu Software Icon

This icon will let you add any additional applications you want. This

can be done by clicking the Update option at the top right corner of that screen. (Figure 5.40)

5.20.8. Online Shopping icon

Using this icon user can buy and sell any products online. (Figure 5.40)

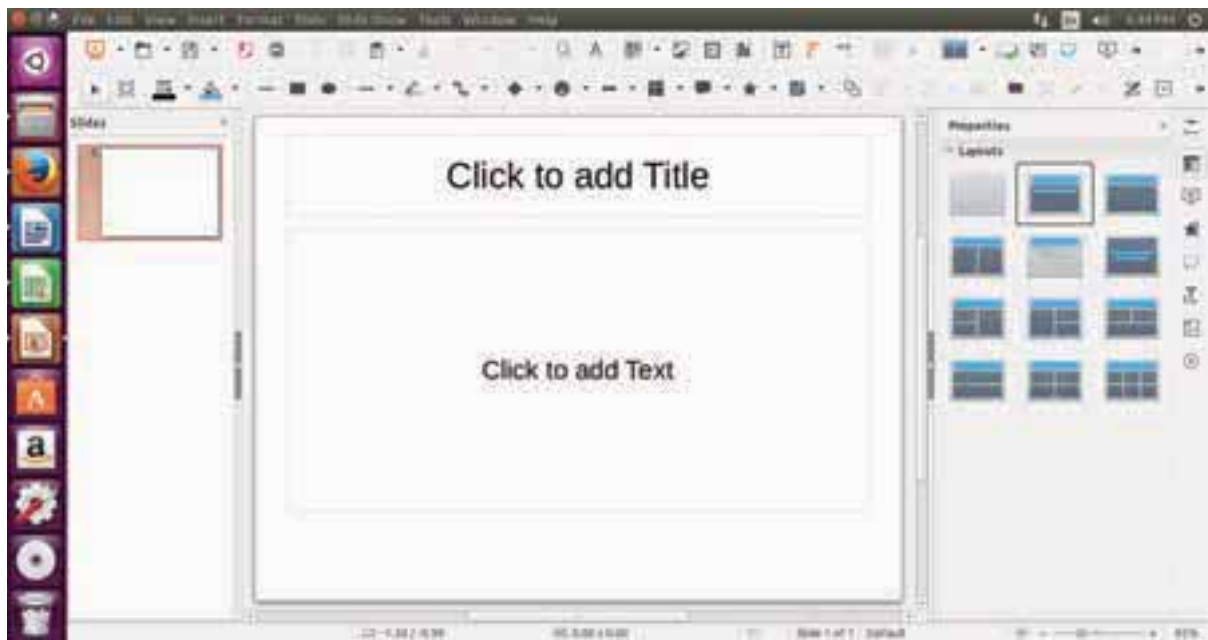


Figure 5.43 LibreOffice Impress Window

5.20.9. System Settings Icons



Figure 5.44 System Settings Icons

This icon is similar to the Control panel in the Windows Operating System. But here, you need to authenticate the changes by giving your password. You cannot simply change as you do in Windows. (Figure 5.44)

5.20.10 Trash

This icon is the equivalent of Recycle bin of windows OS. All the deleted Files and Folders are moved here. (Figure 5.40)

5.21 Creating, Deleting Files/Folders

Similar to Windows OS, you can create, delete the files and folders with the same procedure by clicking Files icon. Figure 5.46 shows the method of creating File or Folder by right clicking in the Desktop. A new File or new Folder can also be created by using File menu (Figure 5.47)

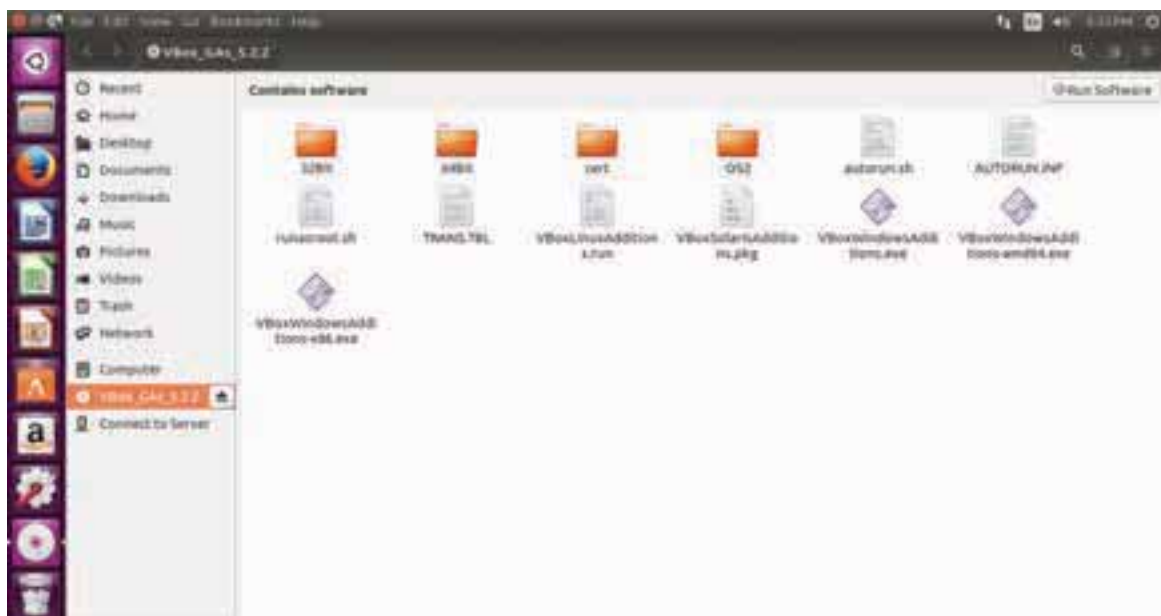


Figure 5.45 VBOX_GAs_5.2.2 Icons



Figure 5.46 Creating a File or Folder by right clicking



Figure 5.47 Creating a File or Folder by using File Menu

Deleting a File/Folder

A file / folder created by you can be moved to trash by using right click or by using menu. (Figure 5.48 & Figure 5.49)

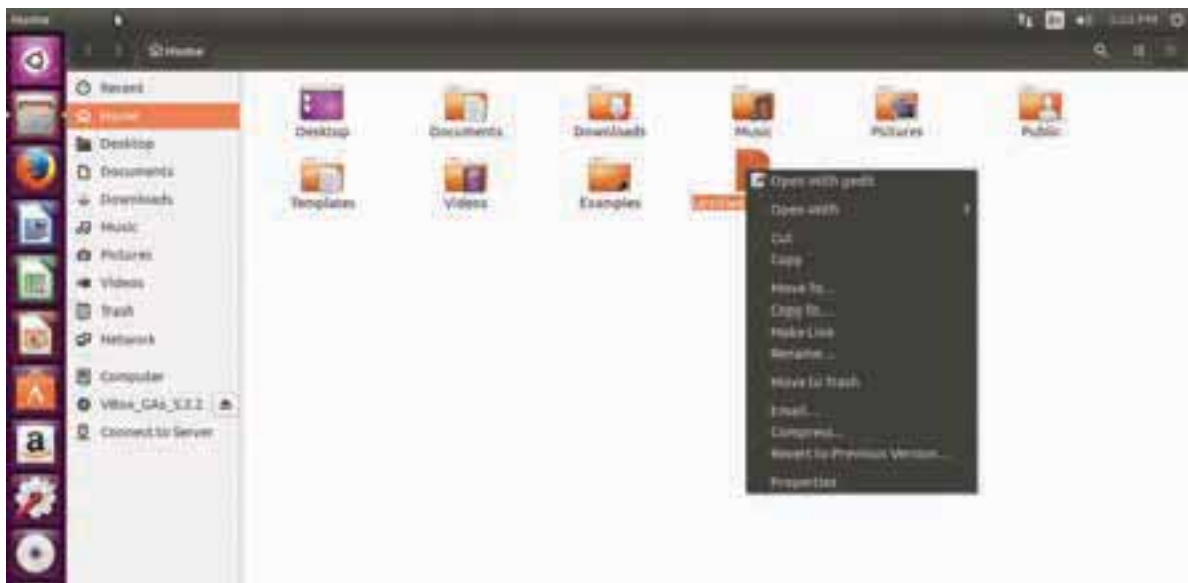


Figure 5.48 Deleting a File or Folder by right clicking



Figure 5.49 Deleting a File or Folder by using Edit menu

5.22 Shutting down Ubuntu using Session options

When you have finished working on your computer, you can choose to Log Out, Suspend or Shut down through the Session Indicator on the far right side of the top panel.

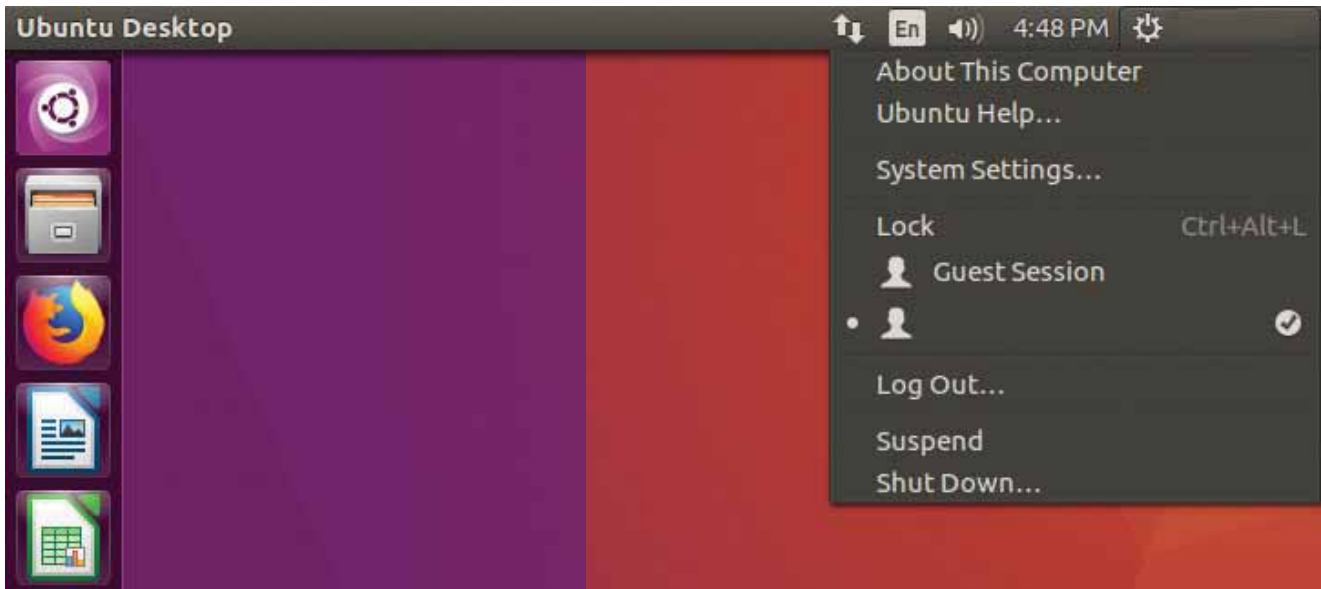


Figure 5.50 Session Options



Student Activity

1. Create files and folders using Windows and Ubuntu and compare them.
2. Prepare an Assignment on the topic “Popular Linux distributions”.
3. Customise few applications using Ubuntu. Write the procedure.
4. Create a File/Folder in Windows 7, Windows 8 and Windows 10. Prepare a report on the differences you face while creating the same.
5. Prepare a table on the difference in views of Windows 7, Windows 8 and Windows 10 operating system.

Teacher Activity

The teacher can adopt the following methodologies to incorporate inside the classroom.

1. Laboratory method – the teacher can take the entire class to the computer lab and demonstrate the concept using projector.
2. Demonstration using laptop and projector – The teacher can demonstrate the same inside the classroom using a laptop and projector.

Evaluation



PART I

Choose the Best Answer

1. From the options given below, choose the operations managed by the operating system.
 - a. Memory
 - b. Processor
 - c. I/O devices
 - d. all of the above
2. Which is the default folder for many Windows Applications to save your file?
 - a. My Document
 - b. My Pictures
 - c. Documents and Settings
 - d. My Computer
3. Under which of the following OS, the option Shift + Delete – permanently deletes a file or folder?
 - a. Windows 7
 - b. Windows 8
 - c. Windows 10
 - d. None of the OS
4. What is the meaning of "Hibernate" in Windows XP/Windows 7?
 - a. Restart the Computer in safe mode
 - b. Restart the Computer in hibernate mode
 - c. Shutdown the Computer terminating all the running applications
 - d. Shutdown the Computer without closing the running applications
5. Which of the following OS is not based on Linux?
 - a. Ubuntu
 - b. Redhat
 - c. CentOS
 - d. BSD

6. Which of the following in Ubuntu OS is used to view the options for the devices installed?
a. Settings b. Files c. Dash d. VBOX_GAs_5.2.2
7. Identify the default email client in Ubuntu.
a. Thunderbird b. Firefox c. Internet Explorer d. Chrome
8. Which is the default application for spreadsheets in Ubuntu? This is available in the software launcher.
a. LibreOffice Writer b. LibreOffice Calc
c. LibreOffice Impress d. LibreOffice Spreadsheet
9. Which is the default browser for Ubuntu?
a. Firefox b. Internet Explorer
c. Chrome d. Thunderbird
10. Where will you select the option to log out, suspend, restart, or shut down from the desktop of Ubuntu OS?
a. Session Indicator b. Launcher c. Files d. Search

PART II

1. Differentiate cut and copy options.
2. What is the use of a file extension?
3. Differentiate Files and Folders.
4. Differentiate Save and save As option.
5. What is Open Source?
6. What are the advantages of open source?
7. Mention the different server distributions in Linux OS.
8. How will you log off from Ubuntu OS?

PART III

1. Analyse: Why the drives are segregated?
2. If you are working on multiple files at a time, sometimes the system may hang. What is the reason behind it. How can you reduce it?
3. Are drives such as hard drive and floppy drives represented with drive letters? If so why, if not why?
4. Write the specific use of Cortana.
5. List out the major differences between Windows and Ubuntu OS.
6. Are there any difficulties you face while using Ubuntu? If so, mention it with reasons.
7. Differentiate Thunderbird and Firefox in Ubuntu OS.
8. Differentiate Save, Save As and Save a Copy in Ubuntu OS.

PART IV

1. Explain the versions of Windows Operating System.
2. Draw and compare the icon equivalence in Windows and Ubuntu.
3. Complete the following matrix

Navigational method	Located on	Ideally suited for
Start button	Task bar	
	Desktop	Exploring your disk drives and using system tools
Windows Explorer		Seeing hierarchy of all computer contents and resources in one window.
Quick Launch		

4. Observe the figure and mark all the window elements. Identify the version of the Windows OS.
5. Write the procedure to create, rename, delete and save a file in Ubuntu OS. Compare it with Windows OS.



Operating System (OS)	System software that enables the hardware to communicate and operate with other software.
Mouse	Handheld hardware input device that control a cursor in a GUI and can move and select text, icons, files, and folders.
Windows	Familiar operating system developed by Microsoft corp.
Desktop	Opening screen of windows operating system.
Icon	Tiny image represent a command.
Folder	Container of files
Linux	An operating system.
Ubuntu	A flavour of Linux operating System.
Firefox	One of the familiar web browser.
LibreOffice	Office automation tool available with Ubuntu by default.
Trash	A special folder contains deleted files.

Specification and Abstraction



Learning Objectives

After learning the concepts in this chapter, the students will be able

- To understand the concept of algorithmic problem solving.
- To apply the knowledge of algorithmic technique in problem solving.

A number of processes performed in our daily life follow the step-by-step execution of a sequence of instructions. Getting ready to school in the morning, drawing "kolams", cooking a dish, adding two numbers are examples of processes. Processes are generated by executing algorithms. In this chapter, we will see how algorithms are specified and how elements of a process are abstracted in algorithms.

6.1 Algorithms

An algorithm is a sequence of instructions to accomplish a task or solve a problem. An instruction describes an action. When the instructions are executed, a process evolves, which accomplishes the intended task or solves the given problem. We can compare an algorithm to a recipe, and the resulting process to cooking.

We are interested in executing our algorithms in a computer. A computer can only execute instructions in a programming language. Instructions of a computer are also known as statements. Therefore, ultimately, algorithms must

be expressed using statements of a programming language.

A problem is specified by given input data, desired output data and a relation between the input and the output data. An algorithm starts execution with the input data, executes the statements, and finishes execution with the output data. When it finishes execution, the specified relation between the input data and the output data should be satisfied. Only then has the algorithm solved the given problem.



G Polya was a Hungarian mathematician. He made fundamental contributions to combinatorics, number theory, numerical analysis and probability theory. He is also noted for his work in heuristics and mathematics education, identifying systematic methods of problem solving to further discovery and invention in mathematics for students and teachers. In "How to Solve It", he suggests the following steps when solving a mathematical problem: 1. Understand the problem. 2. Devise a plan. 3. Carry out the plan. 4. Review your work.



An algorithm is a step by step sequence of statements intended to solve a problem. When executed with input data, it generates a computational process, and ends

with output data, satisfying the specified relation between the input data and the output data.

Example 6.1. Add two numbers: To add two numbers, we proceed by adding the right most digits of the two numbers, then the next right most digits together with carry that resulted from the previous (right) position, and so on. The computational process for adding 2586 and 9237 is illustrated in Table 6.1.

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

Table 6.1: The process for adding two numbers

6.2 Algorithmic Problems

There are some principles and techniques for constructing algorithms. We usually say that a problem is algorithmic in nature when its solution involves the construction of an algorithm. Some types of problems can be immediately recognized as algorithmic.

Example 6.2. Consider the well-known Goat, grass and wolf problem: A farmer wishes to take a goat, a grass bundle and a wolf across a river. However, his boat can take only one of them at a time. So several trips are necessary across the river. Moreover, the goat should not be left alone with the grass (otherwise, the goat would eat the grass), and the wolf should not be left alone with the goat (otherwise, the wolf would eat the goat). How

can the farmer achieve the task? Initially, we assume that all the four are at the same side of the river, and finally, all the four must be in the opposite side. The farmer must be in the boat when crossing the river. A solution consists of a sequence of instructions indicating who or what should cross. Therefore, this is an algorithmic problem. Instructions may be like

Let the farmer cross with the wolf.

or

Let the farmer cross alone.



However, some algorithmic problems do not require us to construct algorithms. Instead, an algorithm is provided and we are required to prove some of its properties.

Example 6.3. Consider The Chameleons of Chromeland problem: On the island of Chromeland there are three different types of chameleons: red chameleons, green chameleons, and blue chameleons. Whenever two chameleons of different colors meet, they both change color to the third color. For which number of red, green and blue chameleons is it possible to arrange a series of meetings that results in all the chameleons displaying the same color? This is an algorithmic problem, because there is an algorithm to arrange meetings between chameleons. Using some properties of the algorithm, we can find out for which initial number of chameleons, the goal is possible.

6.3 Building Blocks of Algorithms

We construct algorithms using basic building blocks such as

- Data
- Variables
- Control flow
- Functions

6.3.1 Data

Algorithms take input data, process the data, and produce output data. Computers provide instructions to perform operations on data. For example, there are instructions for doing arithmetic operations on numbers, such as add, subtract, multiply and divide. There are different kinds of data such as numbers and text.

6.3.2 Variables

Variables are named boxes for storing data. When we do operations on data, we need to store the results in variables. The data stored in a variable is also known as the value of the variable. We can store a value in a variable or change the value of variable, using an assignment statement.

Computational processes in the real-world have state. As a process evolves, the state changes. How do we represent the state of a process and the change of state, in an algorithm? The state of a process can be represented by a set of variables in an algorithm. The state at any point of execution is simply the values of the variables at that point. As the values of the variables are changed, the state changes.

Example 6.4. State: A traffic signal may be in one of the three states: green, amber, or red. The state is changed to allow a smooth flow of traffic. The state may be

represented by a single variable signal which can have one of the three values: green, amber, or red.

6.3.3 Control flow

An algorithm is a sequence of statements. However, after executing a statement, the next statement executed need not be the next statement in the algorithm. The statement to be executed next may depend on the state of the process. Thus, the order in which the statements are executed may differ from the order in which they are written in the algorithm. This order of execution of statements is known as the control flow.

There are three important control flow statements to alter the control flow depending on the state.

- In sequential control flow, a sequence of statements are executed one after another in the same order as they are written.
- In alternative control flow, a condition of the state is tested, and if the condition is true, one statement is executed; if the condition is false, an alternative statement is executed.
- In iterative control flow, a condition of the state is tested, and if the condition is true, a statement is executed. The two steps of testing the condition and executing the statement are repeated until the condition becomes false.

6.3.4 Functions

Algorithms can become very complex. The variables of an algorithm

and dependencies among the variables may be too many. Then, it is difficult to build algorithms correctly. In such situations, we break an algorithm into parts, construct each part separately, and then integrate the parts to the complete algorithm.

The parts of an algorithm are known as functions. A function is like a sub algorithm. It takes an input, and produces an output, satisfying a desired input output relation.

Example 6.5. Suppose we want to calculate the surface area of a cylinder of radius r and height h .

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

We can identify two functions, one for calculating the area of a circle and the other for the circumference of the circle. If we abstract the two functions as `circle_area(r)` and `circle_circumference(r)`, then `cylinder_area(r, h)` can be solved as

$$\text{cylinder_area}(r, h) = 2 \times \text{circle_area}(r) + \text{circle_circumference}(r) \times h$$

6.4 Algorithm Design Techniques

There are a few basic principles and techniques for designing algorithms.

1. **Specification:** The first step in problem solving is to state the problem precisely. A problem is specified in terms of the input given and the output desired. The specification must also state the properties of the given input, and the relation between the input and the output.

2. **Abstraction:** A problem can involve a lot of details. Several of these details are unnecessary for solving the problem. Only a few details are essential. Ignoring or hiding unnecessary details and modeling an entity only by its essential properties is known as abstraction. For example, when we represent the state of a process, we select only the variables essential to the problem and ignore inessential details.

3. **Composition:** An algorithm is composed of assignment and control flow statements. A control flow statement tests a condition of the state and, depending on the value of the condition, decides the next statement to be executed.

4. **Decomposition:** We divide the main algorithm into functions. We construct each function independently of the main algorithm and other functions. Finally, we construct the main algorithm using the functions. When we use the functions, it is enough to know the specification of the function. It is not necessary to know how the function is implemented.

6.5 Specification

To solve a problem, first we must state the problem clearly and precisely. A problem is specified by the given input and the desired output. To design an algorithm for solving a problem, we should know the properties of the given input and the properties of the desired output. The goal of the algorithm is to establish the relation between the input and the desired output.

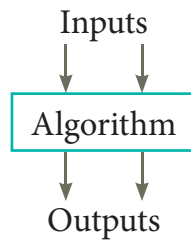


Figure 6.1: Input-output relation

An algorithm is specified by the properties of the given input and the relation between the input and the desired output. In simple words, specification of an algorithm is the desired input-output relation.

The inputs and outputs are passed between an algorithm and the user through variables. The values of the variables when the algorithm starts is known as the initial state, and the values of the variables when the algorithm finishes is known as the final state.

Let P be the required property of the inputs and Q the property of the desired outputs. Then the algorithm S is specified as

1. `algorithm_name (inputs)`
2. `-- inputs : P`
3. `-- outputs: Q`

This specification means that if the algorithm starts with inputs satisfying P , then it will finish with the outputs satisfying Q .

A double dash `--` indicates that the rest of the line is a comment. Comments are statements which are used to annotate a program for the human readers and not executed by the computer. Comments at crucial points of flow are useful, and even necessary, to understand the algorithm. In

our algorithmic notation, we use double dashes (`--`) to start a comment line. (In C++, a double slash `//` indicates that the rest of the line is a comment).

Example 6.6. Write the specification of an algorithm to compute the quotient and remainder after dividing an integer A by another integer B . For example,

divide (22, 5) = 4, 2

divide (15, 3) = 5 , 0

Let A and B be the input variables. We will store the quotient in a variable q and the remainder in a variable r . So q and r are the output variables.

What are the properties of the inputs A and B ?

1. A should be an integer. Remainder is meaningful only for integer division, and
2. B should not be 0, since division by 0 is not allowed.

We will specify the properties of the inputs as

— inputs: A is an integer and $B \neq 0$

What is the desired relation between the inputs A and B , and the outputs q and r ?

1. The two outputs q (quotient) and r (remainder) should satisfy the property

$$A = q \times B + r, \text{ and}$$

2. The remainder r should be less than the divisor B ,

$$0 \leq r < B$$

Combining these requirements, we will specify the desired input-output relation as

— **outputs:** $A = q \times B + r$ and $0 < r < B$.

The comment that starts with — inputs: actually is the property of the given inputs. The comment that starts with — outputs: is the desired relation between the inputs and the outputs. The specification of the algorithm is

1. **divide** (A , B)

2. -- **inputs:** A is an integer and $B \neq 0$

3. -- **outputs :** $A = q \times B + r$ and $0 \leq r < B$

Specification format: We can write the specification in a standard three part format:

- The name of the algorithm and the inputs.
- Input: the property of the inputs.
- Output: the desired input-output relation.

The first part is the name of the algorithm and the inputs. The second part is the property of the inputs. It is written as a comment which starts with — inputs: The third part is the desired input-output relation. It is written as a comment which starts with — outputs:. The input and output can be written using English and mathematical notation.

Example 6.7. Write the specification of an algorithm for computing the square root of a number.

1. Let us name the algorithm square_root.
2. It takes the number as the input. Let us name the input n. n should not be negative.
3. It produces the square root of n as the output. Let us name the output y. Then n should be the square of y.

Now the specification of the algorithm is

square_root(n)

-- **inputs:** n is a real number, $n \geq 0$.

-- **outputs:** y is a real number such that $y^2 = n$.

6.5.1 Specification as contract

Specification of an algorithm serves as a contract between the designer of the algorithm and the users of the algorithm, because it defines the rights and responsibilities of the designer and the user.

Ensuring that the inputs satisfy the required properties is the responsibility of the user, but the right of the designer. The desired input-output relation is the responsibility of the designer and the right of the user. Importantly, if the user fails to satisfy the properties of the inputs, the designer is free from his obligation to satisfy the desired input-output relation.

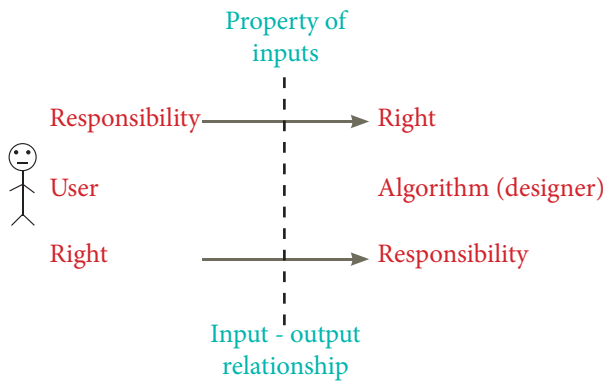


Figure 6.2: Input property and the input-output relation as rights and responsibilities

Example 6.8. Consider the specification of the algorithm `square_root`.

`square_root(n)`

-- **inputs:** `n` is a real number, $n \geq 0$.

-- **outputs :** `y` is a real number such that $y^2 = n$.

The algorithm designer can assume that the given number is non-negative, and construct the algorithm. The user can expect the output to be the square root of the given number.

The output could be the negative square root of the given number. The specification did not commit that the output is the positive square root. If the user passes a negative number as the input, then the output need not be the square root of the number.

6.6 Abstraction

To ride a bicycle, it is sufficient to understand the functioning of the pedal, handlebar, brakes and bell. As a rider, we model a bicycle by these four features. A bicycle has a lot more details, which the

rider can ignore. Those details are irrelevant for the purpose of riding a bicycles.

A problem can involve a lot of details. Several of these details are irrelevant for solving the problem. Only a few details are essential. Abstraction is the process of ignoring or hiding irrelevant details and modeling a problem only by its essential features. In our everyday life, we use abstractions unconsciously to handle complexity. Abstraction is the most effective mental tool used for managing complexity. If we do not abstract a problem adequately, we may deal with unnecessary details and complicate the solution.

Example 6.9. A map is an abstraction of the things we find on the ground. We do not represent every detail on the ground. The map-maker picks out the details that we need to know. Different maps are drawn for different purposes and so use different abstractions, i.e., they hide or represent different features. A road map is designed for drivers. They do not usually worry about hills so most hills are ignored on a road map. A walker's map is not interested in whether a road is a one-way street, so such details are ignored.

Example 6.10. In medicine, different specialists work with different abstractions of human body. An orthopaedician works with the abstraction of skeletal system, while a gastroenterologist works with digestive system. A physiotherapist abstracts the human body by its muscular system.

We use abstraction in a variety of ways while constructing algorithms — in the specification of problems, representing state

by variables, and decomposing an algorithm to functions. An algorithm designer has to be trained to recognize which features are essential to solve the problem, and which details are unnecessary. If we include unnecessary details, it makes the problem and its solution over-complicated.

Specification abstracts a problem by the properties of the inputs and the desired input-output relation. We recognize the properties essential for solving the problem, and ignore the unnecessary details.

State: In algorithms, the state of a computation is abstracted by a set of variables.

Functions: When an algorithm is very complex, we can decompose it into functions and abstract each function by its specification.

6.6.1 State

State is a basic and important abstraction. Computational processes have state. A computational process starts with an initial state. As actions are performed, its state changes. It ends with a final state. State of a process is abstracted by a set of variables in the algorithm. The state at any point of execution is simply the values of the variables at that point.

Example 6.11. Chocolate Bars: A rectangular chocolate bar is divided into squares by horizontal and vertical grooves. We wish to break the bar into individual squares.

To start with, we have the whole of the bar as a single piece. A cut is made by

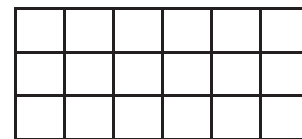
choosing a piece and breaking it along one of its grooves. Thus a cut divides a piece into two pieces. How many cuts are needed to break the bar into its individual squares?

In this example, we will abstract the essential variables of the problem. We solve the problem in Example 8.6.

Essential variables: The number of pieces and the number of cuts are the essential variables of the problem. We will represent them by two variables, p and c , respectively. Thus, the state of the process is abstracted by two variables p and c .

Irrelevant details:

1. The problem could be cutting a chocolate bar into individual pieces or cutting a sheet of postage stamps into individual stamps. It is irrelevant. The problem is simply cutting a grid of squares into individual squares.



2. The sequence of cuts that have been made and the shapes and sizes of the resulting pieces are irrelevant too. From p and c , we cannot reconstruct the sizes of the individual pieces. But, that is irrelevant to solving the problem.

Example 6.12. Consider Example 1.2, Goat, grass and wolf problem. In this example,

we will write a specification of the problem. We will solve it in Example 2.1. The problem involves four individuals, and each is at one of the two sides of the river. This means that

we can represent the state by four variables, and each of them has one of the two values. Let us name the variables farmer, goat, grass and wolf, and their possible values L and R. A value of L means "at the left side". A value of R means "at the right side". Since the boat is always with the farmer, it is important to not introduce a variable to represent its position.

In the initial state, all four variables farmer, goat, grass, wolf have the value L.

farmer, goat, grass, wolf = L, L, L, L

In the final state, all four variables should have the value R.

farmer, goat, grass, wolf = R, R, R, R

The specification of the problem is

cross_river

-- **inputs: farmer, goat, grass, wolf = L, L, L, L**

-- **outputs: farmer, goat, grass, wolf = R, R, R, R**

subject to the two constraints that

1. the goat cannot be left alone with the grass:

if goat = grass then farmer = goat

2. the goat cannot be left alone with the wolf:

if goat = wolf then farmer = goat

6.6.2 Assignment statement

Variables are named boxes to store values. Assignment statement is used to store a

value in a variable. It is written with the variable on the left side of the assignment operator and a value on the right side.

variable := value

When this assignment is executed, the value on the right side is stored in the variable on the left side. The assignment

m := 2

stores value 2 in variable m.

m
2

If the variable already has a value stored in it, assignment changes its value to the value on the right side. The old value of the variable is lost.

The right side of an assignment can be an expression.

variable := expression

In this case, the expression is evaluated and the value of the expression is stored in the variable. If the variable occurs in the expression, the current value of the variable is used in evaluating the expression, and then the variable is updated. For example, the assignment

m := m + 3

evaluates the expression $m + 3$ using the current value of m.

m + 3

= 2 + 3

= 5

and stores the value 5 in the variable m.

m
5

The two sides of an assignment statement are separated by the symbol `:=`, known as assignment operator, and read as "becomes" or "is assigned". The assignment statement

v := e

is read as v "becomes" e. Note that assignment operator is not equality operator¹. The meanings of `v := e` and `v = e` are different. Assignment does not state a mathematical equality of a variable, but changes the value of a variable. The assignment `m := m + 3` does not state that m is equal to `m + 3`. Rather, it changes the value of the variable m to the value of the expression `m + 3`.

An assignment statement can change the values of multiple variables simultaneously. In that case, the number of variables on the left side should match the number of expressions on the right side. For example, if we wish to assign to three variables v1, v2 and v3, we need 3 expressions, say, e1, e2, e3.

v1, v2, v3 := e1, e2, e3

The left side is a comma-separated list of variables. The right side is a comma-separated list of expressions. To execute

¹Unfortunately, several programming languages, including C++, use the symbol `=` as assignment operator, and therefore, another symbol, `==` as equality operator.

an assignment statement, first evaluate all the expressions on the right side using the current values of the variables, and then store them in the corresponding variables on the left side.

Example 6.13. What are the values of variables m and n after the assignments in

line (1) and line (3)?

1. **m, n := 2, 5**

2. **-- m, n = ?, ?**

3. **m,n:=m+3,n-1**

4. **-- m, n = ?, ?**

The assignment in line (1) stores 2 in variable m, and 5 in variable n.

m	n		
<table border="1"><tr><td>2</td></tr></table>	2	<table border="1"><tr><td>5</td></tr></table>	5
2			
5			

The assignment in line (3) evaluates the expressions `m + 3` and `n - 1` using the current values of m and n as

m + 3, n - 1

=2+3,5-1

= 5,4

and stores the values 5 and 4 in the variables m and n, respectively.

m	n
5	4

1. **m, n := 2,5**

2. -- $m, n = 2, 5$

3. $m, n := m + 3, n - 1$

4. -- $m, n = 2 + 3, 5 - 1 = 5, 4$

Values of the variables after the two assignments are shown in in line (2) and line(4).

Example 6.14. In Example 6.11, we abstracted the state of the process by two

variables p and c . The next step is to model the process of cutting the chocolate bar. When we make a single cut of a piece, the number of pieces (p) and the number of cuts (c) both increase by 1. We can model it by an assignment statement.

$$p, c := p + 1, c + 1$$

which is read as p and c "become" $p + 1$ and $c + 1$, respectively.

Points to Remember:

- A programming language provides basic statements and a notation for composing compound statements.
- An algorithm is a step-by-step sequence of statements to solve a problem.
- As an algorithm is executed, a process evolves which solves the problem.
- Algorithmic problem solving involves construction of algorithms as well as proving properties of algorithms.
- The specification of an algorithm consists of the name of the algorithm (together with its inputs), the input property, and the desired input-output relation.
- Specification of an algorithm is a contract between the designer and users of the algorithm.
- Abstraction is the process of hiding or ignoring the details irrelevant to the task so as to model a problem only by its essential features.
- Specification abstracts a problem by the essential variables of the problem.
- The values of the variables in an algorithm define the state of the process.
- Assignment statement changes the values of variables, and hence the state.



Part I Choose the Best Answer

1. Which of the following activities is algorithmic in nature?
(a) Assemble a bicycle. (b) Describe a bicycle.
(c) Label the parts of a bicycle. (d) Explain how a bicycle works.
2. Which of the following activities is not algorithmic in nature?
(a) Multiply two numbers. (b) Draw a kolam.
(c) Walk in the park. (d) Braid the hair.
3. Omitting details inessential to the task and representing only the essential features of the task is known as
(a) specification (b) abstraction (c) composition (d) decomposition
4. Stating the input property and the as :-output relation a problem is known
(a) specification (b) statement (c) algorithm (d) definition
5. Ensuring the input-output relation is
(a) the responsibility of the algorithm and the right of the user.
(b) the responsibility of the user and the right of the algorithm.
(c) the responsibility of the algorithm but not the right of the user.
(d) the responsibility of both the user and the algorithm.
6. If $i = 5$ before the assignment $i := i-1$ after the assignment, the value of i is
(a) 5 (b) 4 (c) 3 (d) 2
7. If $0 < i$ before the assignment $i := i-1$ after the assignment, we can conclude that
(a) $0 < i$ (b) $0 \leq i$ (c) $i = 0$ (d) $0 \geq i$

Part II Very Short Answers

1. Define an algorithm.
2. Distinguish between an algorithm and a process.
3. Initially,
farmer, goat, grass, wolf = L, L, L, L

and the farmer crosses the river with goat. Model the action with an assignment statement.

4. Specify a function to find the minimum of two numbers.
5. If $\sqrt{2} = 1.414$, and the `square_root()` function returns -1.414, does it violate the following specification?

-- square_root (x)

-- inputs: x is a real number , $x \geq 0$

-- outputs: y is a real number such that $y^2=x$

Part III Short Answers

1. When do you say that a problem is algorithmic in nature?
2. What is the format of the specification of an algorithm?
3. What is abstraction?
4. How is state represented in algorithms?
5. What is the form and meaning of assignment statement?
6. What is the difference between assignment operator and equality operator?

Part IV Explain

1. Write the specification of an algorithm hypotenuse whose inputs are the lengths of the two shorter sides of a right angled triangle, and the output is the length of the third side.
2. Suppose you want to solve the quadratic equation $ax^2 + bx + c = 0$ by an algorithm.

quadratic_solve (a, b, c)

-- inputs : ?

-- outputs: ?

You intend to use the formula and you are prepared to handle only real number roots. Write a suitable specification.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

3. Exchange the contents: Given two glasses marked A and B. Glass A is full of apple drink and glass B is full of grape drink. For exchanging the contents of glasses A and B, represent the state by suitable variables, and write the specification of the algorithm.

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

Composition and Decomposition



Learning Objectives

After learning the concepts in this chapter, the students will be able

- To know the notations used in algorithmic techniques.
- To understand Composition and Decomposition in algorithmic techniques.

In Chapter 1, we saw that algorithms are composed of statements. Statements can be grouped into compound statements, giving rise to a hierarchical structure of algorithms. Decomposition is one of the elementary problem-solving techniques. An algorithm may be broken into parts, expressing only high level details. Then, each part may be refined into smaller parts, expressing finer details, or each part may be abstracted as a function.

7.1 Notations for Algorithms

We need a notation to represent algorithms. There are mainly three different notations for representing algorithms.

- A programming language is a notation for expressing algorithms to be executed by computers.
- Pseudo code is a notation similar to programming languages. Algorithms expressed in pseudo code are not intended to be executed by computers, but for communication among people.

- Flowchart is a diagrammatic notation for representing algorithms. They give a visual intuition of the flow of control, when the algorithm is executed.

7.1.1 Programming language

A programming language is a notation for expressing algorithms so that a computer can execute the algorithm. An algorithm expressed in a programming language is called a program. C, C++ and Python are examples of programming languages. Programming language is formal. Programs must obey the grammar of the programming language exactly. Even punctuation symbols must be exact. They do not allow the informal style of natural languages such as English or Tamil. There is a translator which translates the program into instructions executable by the computer. If our program has grammatical errors, this translator will not be able to do the translation.



7.1.2 Pseudo-code

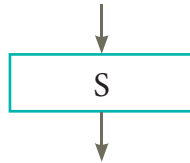
Pseudo code is a mix of programming-language-like constructs and plain English. This notation is not formal nor exact. It uses the same building blocks as programs, such as variables and control flow. But, it allows the use of natural English for statements and conditions. An algorithm expressed as pseudo code is not for computers to execute directly, but for human readers

to understand. Therefore, there is no need to follow the rules of the grammar of a programming language. However, even pseudo code must be rigorous and correct. Pseudo code is the most widely used notation to represent algorithms.

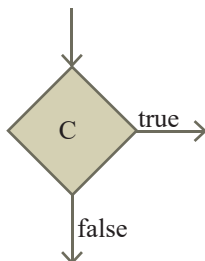
7.1.3 Flowcharts

Flowchart is a diagrammatic notation for representing algorithms. They show the control flow of algorithms using diagrams in a visual manner. In flowcharts, rectangular boxes represent simple statements, diamond-shaped boxes represent conditions, and arrows describe how the control flows during the execution of the algorithm. A flowchart is a collection of boxes containing statements and conditions which are connected by arrows showing the order in which the boxes are to be executed.

1. A statement is contained in a rectangular box with a single outgoing arrow, which points to the box to be executed next.



2. A condition is contained in a diamond-shaped box with two outgoing arrows, labeled true and false. The true arrow points to the box to be executed next if the condition is true, and the false arrow points to the box to be executed next if the condition is false.



3. Parallelogram boxes represent inputs given and outputs produced.



4. Special boxes marked Start and the End are used to indicate the start and the end of an execution:



The flowchart of an algorithm to compute the quotient and remainder after dividing an integer A by another integer B is shown in Figure 7.1, illustrating the different boxes such as input, output, condition, and assignment, and the control flow between the boxes. The algorithm is explained in Example 7.4.

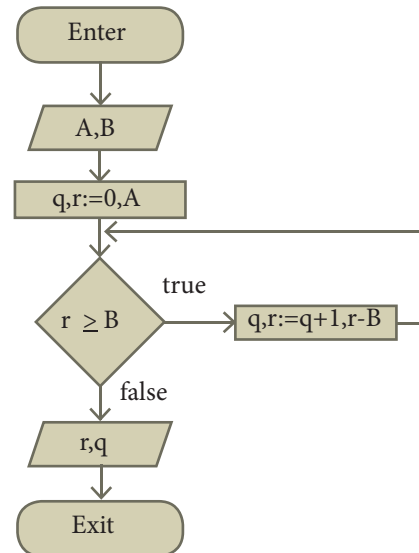


Figure 7.1: Flowchart for integer division

Flowcharts also have disadvantages.

- (1) Flowcharts are less compact than representation of algorithms in

programming language or pseudo code. (2) They obscure the basic hierarchical structure of the algorithms. (3) Alternative statements and loops are disciplined control flow structures. Flowcharts do not restrict us to disciplined control flow structures.

7.2 Composition

A statement is a phrase that commands the computer to do an action. We have already seen assignment statement. It is a simple statement, used to change the values of variables. Statements may be composed of other statements, leading to a hierarchical structure of algorithms. Statements composed of other statements are known as compound statements.

Control flow statements are compound statements. They are used to alter the control flow of the process depending on the state of the process. There are three important control flow statements:

- Sequential
- Alternative
- Iterative

When a control flow statement is executed, the state of the process is tested, and depending on the result, a statement is selected for execution.

7.2.1 Sequential statement

A sequential statement is composed

of a sequence of statements. The statements in the sequence are executed one after another, in the same order as they are written in the algorithm, and the control flow is said to be sequential. Let S1 and S2 be statements. A sequential statement composed of S1 and S2 is written as

S1

S2

In order to execute the sequential statement, first do S1 and then do S2.

The sequential statement given above can be represented in a flowchart as shown in Figure 7.2. The arrow from S1 to S2 indicates that S1 is executed, and after that, S2 is executed.

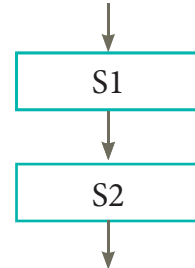


Figure 7.2: Sequential control flow

Let the input property be P, and the input-output relation be Q, for a problem. If statement S solves the problem, it is written as

1. -- P
2. S
3. -- Q

If we decompose the problem into two components, we need to compose S as a

sequence of two statements S1 and S2 such that the input-output relation of S1, say R, is the input property of S2.

1. -- P
2. S1
3. -- R
4. S2
5. -- Q

Example 7.1. Let us solve the Farmer, Goat, Grass, and Wolf problem of Example 6.12. We decided to represent the state of the process by four variables farmer, goat, grass, and wolf, representing the sides of the farmer, goat, grass and wolf, respectively. In the initial state, all four variables have the value L (Left side). In the final state, all four variables should have the value R (Right side). The goal is

to construct a statement S so as to move from the initial state to the final state.

1. -- **farmer, goat, grass, wolf = L, L, L, L**
2. S
3. -- **farmer , goat , grass , wolf = R, R, R, R**

We have to compose S as a sequence of assignment statements such that in none of the intermediate states

1. goat and wolf have the same value but farmer has the opposite value, or
2. goat and grass have the same value but farmer has the opposite value. Subject to these constraints, a sequence of assignments and the state after each assignment are shown in Figure 7.3.

1. -- farmer, goat, grass, wolf = L, L, L, L
2. **farmer, goat := R, R**
3. -- farmer , goat , grass , wolf = R, R, L, L
4. **farmer := L**
5. farmer, goat, grass, wolf = L, R, L, L
6. **farmer, grass := R, R**
7. -- farmer , goat , grass , wolf = R, R, R, L
8. **farmer, goat := L, L**
9. -- farmer, goat, grass, wolf = L, L, R, L
10. **farmer, wolf := R, R**
11. -- farmer , goat , grass , wolf = R, L, R, R
12. **farmer := L**
13. -- farmer , goat , grass , wolf = L, L, R, R
14. **farmer , goat := R, R**
15. -- farmer , goat , grass , wolf = R, R, R, R

Figure 7.3: Sequence of assignments for goat, grass and wolf problem

Other than lines (1) and (15), in line (7), goat and grass have the same value, but farmer also has the same value as they. In line (9), goat and wolf have the same value, but farmer also has the same value as they. Thus, the sequence has achieved the goal state, without violating the constraints.

7.2.2 Alternative statement

A condition is a phrase that describes a test of the state. If C is a condition and both

$S1$ and $S2$ are statements, then

if C

$S1$

else

$S2$

is a statement, called an alternative statement, that describes the following action:

1. Test whether C is true or false.
2. If C is true, then do $S1$; otherwise do $S2$.

In pseudo code, the two alternatives $S1$ and $S2$ are indicated by indenting them from the keywords `if` and `else`, respectively. Alternative control flow is depicted in the flowchart of Figure 2.4. Condition C has two outgoing arrows, labeled `true` and `false`. The `true` arrow points to the $S1$ box. The `false` arrow points to the $S2$ box. Arrows out of $S1$ and $S2$ point to the same box, the box after the alternative statement.

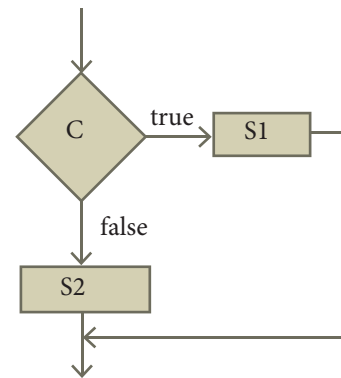


Figure 7.4: Alternative control flow

Conditional statement: Sometimes we need to execute a statement only if a condition is true and do nothing if the condition is false. This is equivalent to the alternative statement in which the `else`-clause is empty. This variant of alternative statement is called a conditional statement. If C is a condition and S is a statement, then

if C

S

is a statement, called a conditional statement, that describes the following action:

1. Test whether C is true or false.
2. If C is true then do S ; otherwise do nothing.

The conditional control flow is depicted in the flowchart of Figure 2.5.

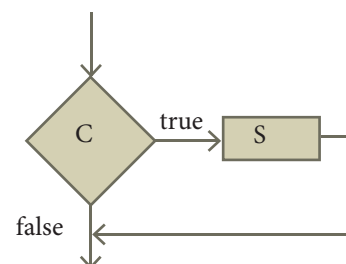


Figure 7.5: Conditional control flow

Example 7.2. Minimum of two numbers:
Given two numbers a and b , we want to find the minimum of the two using the alternative statement. Let us store the minimum in a variable named `result`. Let $a \downarrow b$ denote the minimum of a and b (for instance, $4 \downarrow 2 = 2$, $-5 \downarrow 6 = -5$). Then, the specification of algorithm minimum is

```

minimum(a, b)

-- input  $s : a, b$ 

-- outputs:  $result = a \downarrow b$ 

```

Algorithm minimum can be defined as

```

1.  minimum(a, b)
2.      --  $a, b$ 
3.      if  $a < b$ 
4.          result :=  $a$ 
5.      else
6.          result =  $b$ 
7.      --  $result = a \downarrow b$ 

```

7.2.3 Case analysis

Alternative statement analyses the problem into two cases. Case analysis statement generalizes it to multiple cases. Case analysis splits the problem into an exhaustive set of disjoint cases. For each case, the problem is solved independently. If $C1$, $C2$, and $C3$ are conditions, and $S1$, $S2$, $S3$ and $S4$ are statements, a 4-case analysis statement has the form,

```

1. case  $C1$ 

```

```

2.     $S1$ 
3. case  $C2$ 
4.     $S2$ 
5. case  $C3$ 
6.     $S3$ 
7. else
8.     $S4$ 

```

The conditions $C1$, $C2$, and $C3$ are evaluated in turn. For the first condition that evaluates to true, the corresponding statement is executed, and the case analysis statement ends. If none of the conditions evaluates to true, then the default case $S4$ is executed.

1. The cases are exhaustive: at least one of the cases is true. If all conditions are false, the default case is true.
2. The cases are disjoint: only one of the cases is true. Though it is possible for more than one condition to be true, the case analysis always executes only one case, the first one that is true. If the three conditions are disjoint, then the four cases are (1) $C1$, (2) $C2$, (3) $C3$, (4) (not $C1$) and (not $C2$) and (not $C3$).

Example 7.3. We want an algorithm that compares two numbers and produces the result as

$$\text{compare}(a, b) = \begin{cases} 1 & \text{if } a < b \\ 0 & \text{if } a = b \\ 1 & \text{if } a > b \end{cases}$$

We can split the state into an exhaustive set of 3 disjoint cases: $a < b$, $a = b$, and $a > b$. Then we can define `compare()` using a case analysis.

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 Iterative statement

An iterative process executes the same action repeatedly, subject to a condition C . If C is a condition and S is a statement, then

while C

S

is a statement, called an iterative statement, that describes the following action:

1. Test whether C is true or false.
2. If C is true, then do S and go back to step 1; otherwise do nothing.

The iterative statement is commonly known as a loop. These two steps, testing

C and executing S , are repeated until C becomes false. When C becomes false, the loop ends, and the control flows to the statement next to the iterative statement. The condition C and the statement S are called the loop condition and the loop body, respectively. Testing the loop condition and executing the loop body once is called an iteration. not C is known as the termination condition.

Iterative control flow is depicted in the flowchart of Figure 7.6. Condition C has two outgoing arrows, true and false. The true arrow points to S box. If C is true, S box is executed and control flows back to C box. The false arrow points to the box after the iterative statement (dotted box). If C is false, the loop ends and the control flows to the next box after the loop.

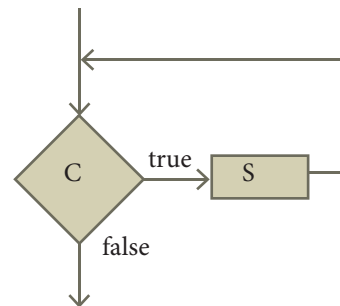


Figure 7.6: Iterative control flow

Example 7.4. Construct an iterative algorithm to compute the quotient and remainder after dividing an integer A by another integer B .

We formulated the specification of the algorithm in Example 6.6 as

divide (A , B)

-- inputs: A is an integer and $B \neq 0$

-- **outputs : q and r such that $A = q \times B + r$ and**

-- **$0 \leq r < B$**

Now we can construct an iterative algorithm that satisfies the specification.

divide (A , B)

-- **inputs: A is an integer and $B \neq 0$**

-- **outputs : q and r such that $A = q \times B + r$ and**

-- **$0 < r < B$**

q := 0, A

while $r \geq B$

q, r := q + 1, r - B

The algorithm is presented as a flowchart in Figure 7.1.

We can execute the algorithm step-by-step for a test input, say, $(A, B) = (22, 5)$. Each row of Table 7.1 shows one iteration — the evaluation of the expressions and the values of the variables at the end of an iteration. Note that the evaluation of the expression uses the values of the variables from the previous row. Output variables q and r change their values in each iteration. Input variables A and B do not change their values. Iteration 0 shows the values just before the loop starts. At the end of iteration 4, condition $(r \geq B) = (2 \geq 5)$ is false, and hence the loop ends with $(q, r) = (4, 2)$.

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		

Table 7.1: Step by step execution of divide (22, 5)

Example 7.5. In the Chameleons of Chromeland problem of Example 1.3, suppose two types of chameleons are equal in number. Construct an algorithm that arranges meetings between these two types so that they change their color to the third type. In the end, all should display the same color.

Let us represent the number of chameleons of each type by variables a, b and c, and their initial values by A, B and C, respectively. Let $a = b$ be the input property. The input-output relation is $a = b = 0$ and $c = A+B+C$. Let us name the algorithm monochromatize. The algorithm can be specified as

monochromatize(a, b, c)

-- **inputs: $a=A, b=B, c=C, a=b$**

-- **outputs : $a = b = 0, c = A+B+C$**

In each iterative step, two chameleons of the two types (equal in number) meet and change their colors to the third one. For example, if $A, B, C = 4, 4, 6$, then the series of meetings will result in

iteration	a	b	c
0	4	4	6
1	3	3	8
2	2	2	10
3	1	1	12
4	0	0	14

Table 7.2: Series of meetings between two types of chameleons equal in number.

In each meeting, a and b each decreases by 1, and c increases by 2. The solution can be expressed as an iterative algorithm.

monochromatize(a, b, c)

-- 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

The algorithm is depicted in the flowchart of Figure 7.7.

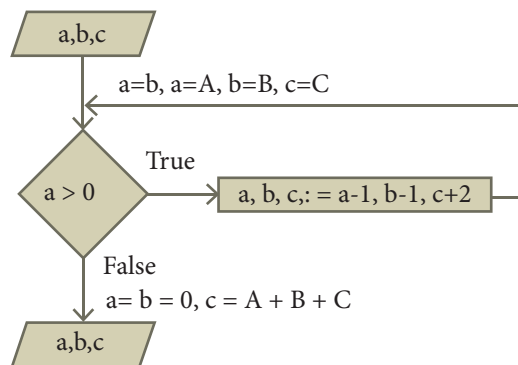


Figure 7.7: Algorithm monochromatize

7.3 Decomposition

Problem decomposition is one of the elementary problem-solving techniques. It involves breaking down a problem into

smaller and more manageable problems, and combining the solutions of the smaller problems to solve the original problem. Often, problems have structure. We can exploit the structure of the problem and break it into smaller problems. Then, the smaller problems can be further broken until they become sufficiently small to be solved by other simpler means. Their solutions are then combined together to construct a solution to the original problem.

7.3.1 Refinement

After decomposing a problem into smaller subproblems, the next step is either to refine the subproblem or to abstract the subproblem.

1. Each subproblem can be expanded into more detailed steps. Each step can be further expanded to still finer steps, and so on. This is known as refinement.
2. We can also abstract the subproblem. We specify each subproblem by its input property and the input-output relation. While solving the main problem, we only need to know the specification of the subproblems. We do not need to know how the subproblems are solved.

Example 7.6. Consider a school goer's action in the morning. The action can be written as

- 1 Get ready for school

We can decompose this action into smaller, more manageable action steps which she takes in sequence:

- 1 Eat breakfast

2 Put on clothes

3 Leave home

We have refined one action into a detailed sequence of actions. However, each of these actions can be expanded into a sequence of actions at a more detailed level, and this expansion can be repeated. The action "Eat breakfast" can be expanded as

1 -- Eat breakfast

2 Eat idlis

3 Eat eggs

4 Eat bananas

The action "Put on clothes" can be expanded as

1 -- Put on clothes

2 Put on blue dress

3 Put on socks and shoes

4 Wear ID card

and "Leave home" expanded as

1 -- Leave home

2 Take the bicycle out

3 Ride the bicycle away

Thus, the entire action of "Get ready for school" has been refined as

1 -- Eat breakfast

2 Eat idlis

3 Eat eggs

4 Eat bananas

5

6 -- Put on clothes

7 Put on blue dress

8 Put on socks and shoes

9 Wear ID card

10

11 -- Leave home

12 Take the bicycle out

13 Ride the bicycle away

Refinement is not always a sequence of actions. What the student does may depend upon the environment. How she eats breakfast depends upon how hungry she is and what is on the table; what clothes she puts on depends upon the day of the week. We can refine the behaviour which depends on environment, using conditional and iterative statements.

1 -- Eat breakfast

2 if hungry and idlis on the table

3 Eat idlis

4 if hungry and eggs on the table

5 Eat eggs

6 if hungry and bananas on the table

7 Eat bananas

8

8 -- Put on clothes

10 if Wednesday

11 Put on blue dress

12 else

13 Put on white dress

14 Put on socks and shoes

15 Wear the ID card

16

17 -- Leave home

18 Take the bicycle out

19 Ride the bicycle away

The action "Eat idlis" can be further refined as an iterative action:

1 -- Eat idlis

2 Put idlis on the plate

3 Add chutney

4 while idlis in plate

5 Eat a bite of idli

How "Get ready for school" is refined in successive levels is illustrated in Figure 2.8.

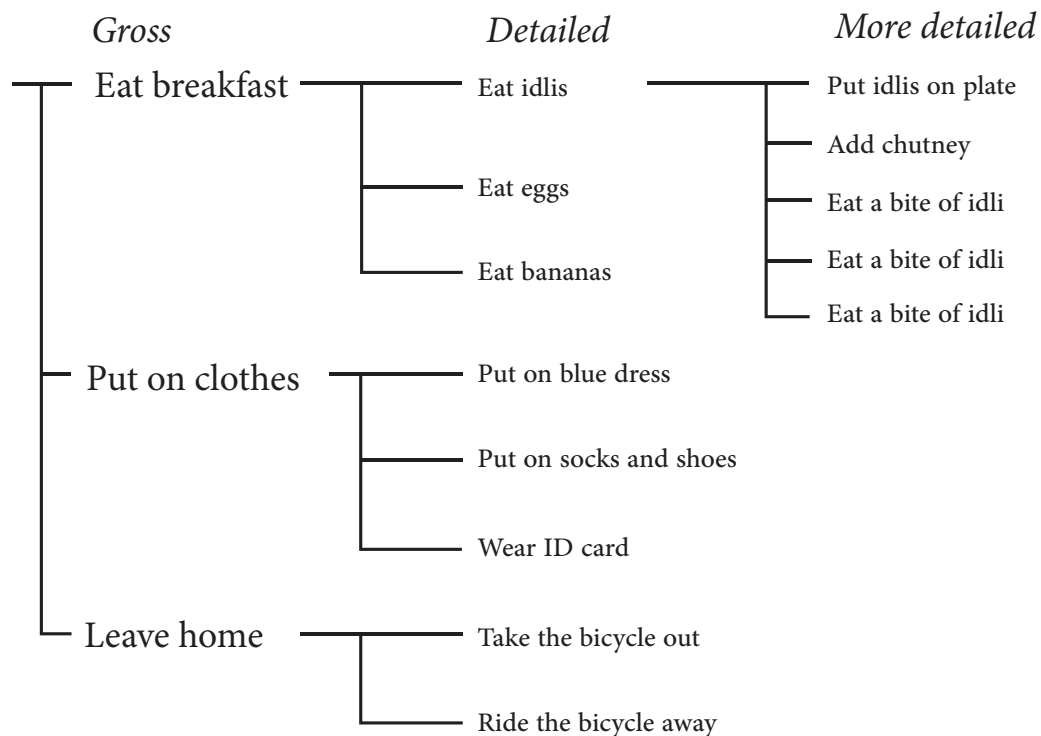


Figure 7.8: Refinement at various levels of details

The action "Eat breakfast" is depicted in a flowchart shown in Figure 2.9.

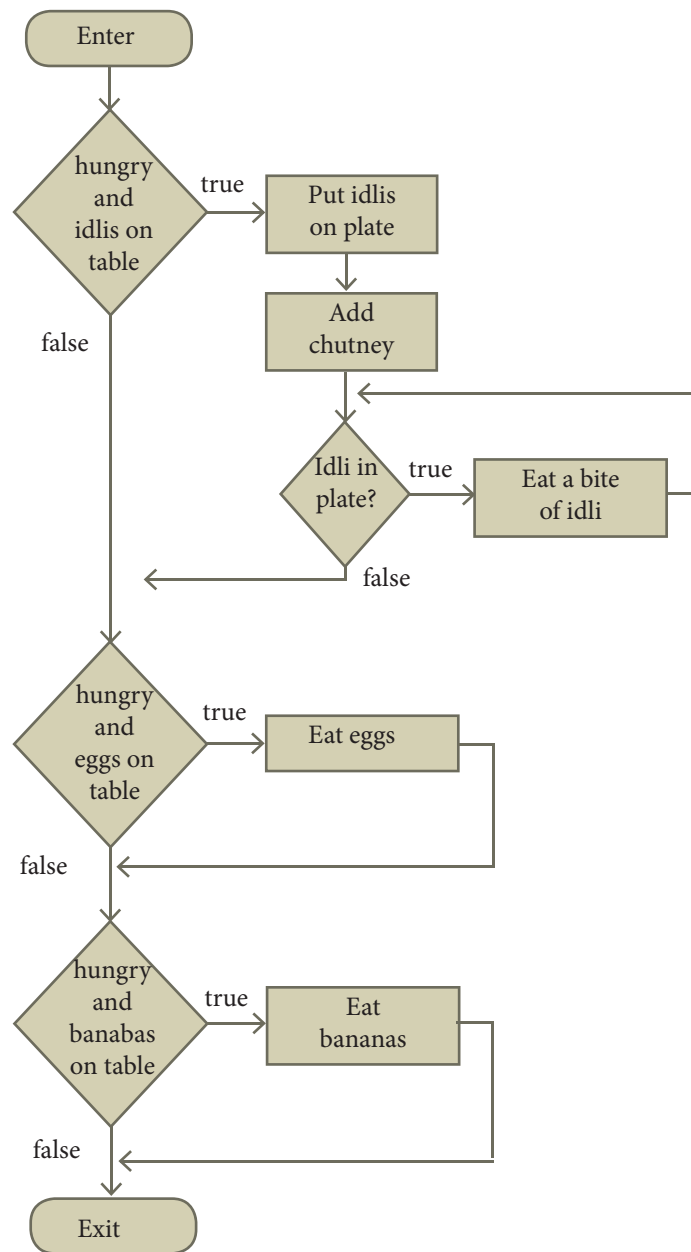


Figure 7.9: Flowchart for Eat breakfast

Note that the flowchart does not show the hierarchical structure of refinement.

7.3.2 Functions

After an algorithmic problem is decomposed into subproblems, we can abstract the subproblems as functions. A function is like a sub-algorithm. Similar to an algorithm, a function is specified by the input property, and the desired input-output relation.

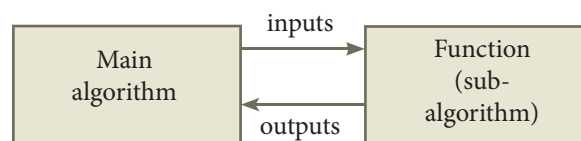


Figure 7.10: Function definition

To use a function in the main algorithm, the user need to know only the specification of the function — the function name, the input property, and the input-output relation. The user must ensure that the inputs passed to the function will satisfy the specified property and can assume that the outputs from the function satisfy the input-output relation. Thus, users of the function need only to know what the function does, and not how it is done by the function. The function can be used as a "black box" in solving other problems.

Ultimately, someone implements the function using an algorithm. However, users of the function need not know about the algorithm used to implement the function. It is hidden from the users. There is no need for the users to know how the function is implemented in order to use it.

An algorithm used to implement a function may maintain its own variables. These variables are local to the function in the sense that they are not visible to the user of the function. Consequently, the user has fewer variables to maintain in the main algorithm, reducing the clutter of the main algorithm.

Example 7.7. Consider the problem of testing whether a triangle is right-angled, given its three sides a , b , c , where c is the longest side. The triangle is right-angled, if

$$c^2 = a^2 + b^2$$

We can identify a subproblem of squaring a number. Suppose we have a function `square()`, specified as

```
square(y)

-- inputs : y

-- outputs : y2
```

we can use this function three times to test whether a triangle is right-angled. `square()` is a "black box" — we need not know how the function computes the square. We only need to know its specification.

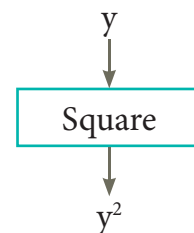


Figure 7.11: square function

```

1  right_angled(a, b, c)
2  -- inputs:  $c \geq a, c \geq b$ 
3  -- outputs: result = true if  $c^2 = a^2 + b^2$ ;
4  -- result = false, otherwise
5  if square (c) = square (a) + square (b)
6    result := true
7  else
8    result := false
```

Points to Remember

- Compound statements are composed of sequential, alternative and iterative control flow statements.
- The value of a condition is true or false, depending on the values of the variables.
- Alternative statement selects and executes exactly one of the two statements, depending on the value of the condition.
- Conditional statement is executed only if the condition is true. Otherwise, nothing is done.
- Iterative statement repeatedly evaluates a condition and executes a statement as long as the condition is true.
- Programming language, pseudo code, and flowchart are notations for expressing algorithms.
- Decomposition breaks down a problem into smaller subproblems and combine their solutions to solve the original problem.
- A function is an abstraction of a subproblem, and specified by its input property, and its input-output relation.
- Users of function need to know only what the function does, and not how it is done.
- In refinement, starting at a high level, each statement is repeatedly expanded into more detailed statements in the subsequent levels.

Evaluation



Part I

1. Suppose $u, v = 10, 5$ before the assignment. What are the values of u and v after the sequence of assignments?

1 $u := v$

2 $v := u$

(a) $u, v = 5, 5$

(b) $u, v = 5, 10$

(c) $u, v = 10, 5$

(d) $u, v = 10, 10$



2. Which of the following properties is true after the assignment (at line 3?)

1 $-- i+j = 0$

2 $i, j := i+1, j-1$

3 $-- ?$

(a) $i+j > 0$

(b) $i+j < 0$

(c) $i+j = 0$

(d) $i = j$

3. If C1 is false and C2 is true, the compound statement

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. If C is false just before the loop, the control flows through

1 S1

2 while C

3 S2

4 S3

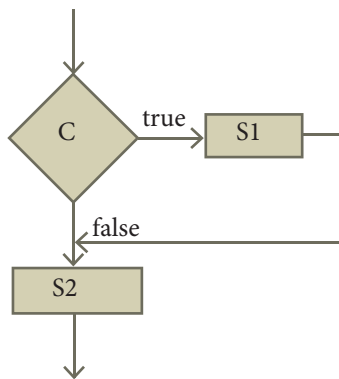
(a) S1 ; S3

(b) S1 ; S2 ; S3

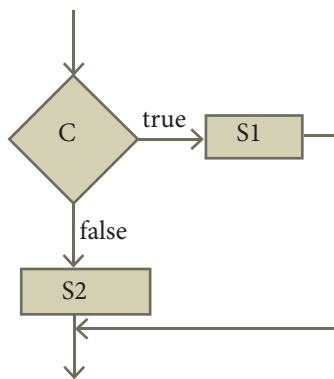
(c) S1 ; S2 ; S2 ; S3

(d) S1 ; S2 ; S2 ; S2 ; S3

5. If C is true, S1 is executed in both the flowcharts, but S2 is executed in



(1)



(2)

(a) (1) only

(b) (2) only

(c) both (1) and (2)

(d) neither (1) nor (2)

6. How many times the loop is iterated?

$i := 0$

while $i \neq 5$

$i := i + 1$

- (a) 4 (b) 5 (c) 6 (d) 0

Part II

1. Distinguish between a condition and a statement.
2. Draw a flowchart for conditional statement.
3. Both conditional statement and iterative statement have a condition and a statement. How do they differ?
4. What is the difference between an algorithm and a program?
5. Why is function an abstraction?
6. How do we refine a statement?

Part III

1. For the given two flowcharts write the pseudo code.
2. If C is false in line 2, trace the control flow in this algorithm.

1 S1

2 -- C is false

3 if C

4 S2

5 else

6 S3

7 S4

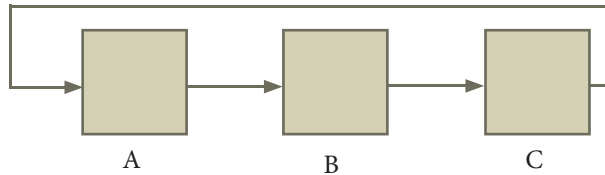
3. What is case analysis?
4. Draw a flowchart for -3case analysis using alternative statements.
5. Define a function to double a number in two different ways: (1) $n + n$, (2) $2 \times n$

Part IV

1. Exchange the contents: Given two glasses marked A and B. Glass A is full of apple drink

and glass B is full of grape drink. Write the specification for exchanging the contents of glasses A and B, and write a sequence of assignments to satisfy the specification.

2. Circulate the contents: Write the specification and construct an algorithm to circulate the contents of the variables A, B and C as shown below: The arrows indicate that B gets the value of A, C gets the value of B and A gets the value of C.



3. Decanting problem. You are given three bottles of capacities 5, 8, and 3 litres. The 8L bottle is filled with oil, while the other two are empty. Divide the oil in 8L bottle into two equal quantities. Represent the state of the process by appropriate variables. What are the initial and final states of the process? Model the decanting of oil from one bottle to another by assignment. Write a sequence of assignments to achieve the final state.
4. Trace the step-by-step execution of the algorithm for factorial(4).

factorial(n)

-- inputs : n is an integer , $n \geq 0$

-- outputs : $f = n!$

f, i := 1, 1

while i ≤ n

f, i := f × i, i+1

Iteration and recursion



Learning Objectives

After learning the concepts in this chapter, the students will be able

- To know the concepts of variants and invariants used in algorithmic techniques.
- Apply algorithmic techniques in iteration and recursion process.



There are several problems which can be solved by doing the same action repeatedly. Both iteration and recursion are algorithm design techniques to execute the same action repeatedly. What is the use of repeating the same action again and again? Even though the action is the same, the state in which the action is executed is not the same. Each time we execute the action, the state changes. Therefore, the same action is repeatedly executed, but in different states. The state changes in such a way that the process progresses to achieve the desired input-output relation.

Iteration: In iteration, the loop body is repeatedly executed as long as the loop condition is true. Each time the loop body is executed, the variables are updated. However, there is also a property of the variables which remains unchanged by the execution of the loop body. This unchanging property is called the loop invariant. Loop invariant is the key to construct and to reason about iterative algorithms.

Recursion: Recursion is another algorithm design technique, closely related to iteration,



E W Dijkstra was one of the most influential pioneers of Computing Science. He made fundamental contributions in diverse areas such as programming language design, operating systems, and program design. He coined the phrase "structured programming" which helped lay the foundations for the discipline of software engineering. In 1972, he was awarded ACM Turing Award, considered the highest distinction in computer science. Dijkstra is attributed to have said "Computer science is no more about computers than astronomy is about telescopes."



but more powerful. Using recursion, we solve a problem with a given input, by solving the same problem with a part of the input, and constructing a solution to the original problem from the solution to the partial input.

8.1 Invariants

Example 8.1. Suppose the following assignment is executed with $(u, v) = (20, 15)$. We can annotate before and after the assignment.

-- before: $u, v = 20, 15$

$u, v := u+5, v-5$

-- after: $u, v = 25, 10$

After assignment $(u, v) = (25, 10)$. But what do you observe about the value

of the function $u + v$?

before: $u + v = 20 + 15 = 35$

after: $u + v = 25 + 10 = 35$

The assignment has not changed the value of $u + v$. We say that $u + v$ is an invariant of the assignment. We can annotate before and after the assignment with the invariant expression.

-- **before:** $u + v = 35$

u, v := u + 5, v - 5

-- **after :** $u + v = 35$

We can say, $u + v$ is an invariant: it is 35 before and after. Or we can say $u + v = 35$ is an invariant: it is true before and after.

Example 8.2. If we execute the following assignment with $(p, c = 10, 9)$, after the assignment, $(u, v) = (11, 10)$.

-- **before :** $p, c = 10, 9$

p, c := p + 1, c + 1

-- **after:** $p, c = 11, 10$

Can you discover an invariant? What is the value of $p - c$ before and after?

before: $p - c = 10 - 9 = 1$

after: $p - c = 11 - 10 = 1$

We find that $p - c = 1$ is an invariant.

In general, if an expression of the variables has the same value before and

after an assignment, it is an invariant of the assignment. Let $P(u, v)$ be an expression involving variables u and v . $P(u, v)[u, v := e1, e2]$ is obtained from $P(u, v)$ by replacing u by $e1$ and v by $e2$ simultaneously. $P(u, v)$ is an invariant of assignment $u, v := e1, e2$ if

$$P(u, v) [u, v := e1, e2] = P(u, v)$$

Example 8.3. Show that $p - c$ is an invariant of the assignment

p, c := p + 1, c + 1

Let $P(p, c) = p - c$. Then

$$P(p, c) [p, c := p + 1, c + 1]$$

$$= p - c [p, c := p + 1, c + 1]$$

$$= (p + 1) - (c + 1)$$

$$= p - c$$

$$= P(p, c)$$

Since $(p - c)[p, c := p + 1, c + 1] = p - c$, $p - c$ is an invariant of the assignment $p, c := p + 1, c + 1$.

Example 8.4. Consider two variables m and n under the assignment

m, n := m + 3, n - 1

Is the expression $m + 3n$ an invariant?

Let $P(m, n) = m + 3n$. Then

$$P(m, n) [m, n := m + 3, n - 1]$$

$$= m + 3n [m, n := m + 3, n - 1]$$

$$= (m + 3) + 3(n - 1)$$

$$= m + 3 + 3n - 3$$

$$= m + 3n$$

$$= P(m, n)$$

Since $(m + 3n) [m, n := m + 3, n - 1] = m + 3n$, $m + 3n$ is an invariant of the assignment $m, n := m + 3, n - 1$.

8.2 Loop invariant

In a loop, if L is an invariant of the loop body B , then L is known as a loop invariant.

while C

-- L

B

-- L

The loop invariant is true before the loop body and after the loop body, each time. Since L is true at the start of the first iteration, L is true at the start of the loop also (just before the loop). Since L is true at the end of the last iteration, L is true when the loop ends also (just after the loop). Thus, if L is a loop variant, then it is true at four important points in the algorithm, as annotated in the algorithm and shown in Figure 3.1.

1. at the start of the loop (just before the loop)
2. at the start of each iteration (before loop body)
3. at the end of each iteration (after loop body)

4. at the end of the loop (just after the loop)

1. -- L , start of loop

while

C

2. -- L , start of iteration

B

3. -- L , end of iteration

4. -- L , end of loop

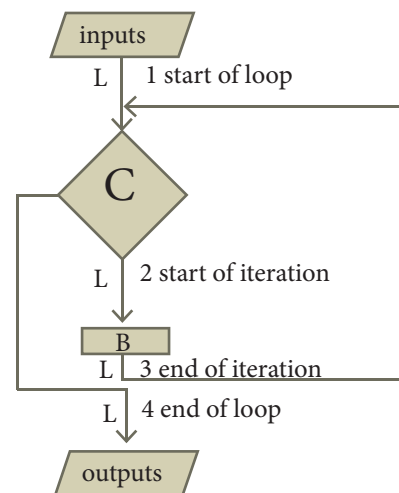


Figure 8.1: The points where the loop invariant is true

To construct a loop,

1. Establish the loop invariant at the start of the loop.
2. The loop body should so update the variables as to progress toward the end, and maintain the loop invariant, at the same time.
3. When the loop ends, the termination condition and the loop invariant should establish the input-output relation.

8.3 Invariants — Examples

The loop invariant is true in four crucial points in a loop. Using the loop invariant, we can construct the loop and reason about the properties of the variables at these points.

Example 8.5. Design an iterative algorithm to compute a^n . Let us name the algorithm `power(a, n)`. For example,

`power(10, 4) = 10000`

`power(5, 3) = 125`

`power(2, 5) = 32`

Algorithm `power(a, n)` computes a^n by multiplying a cumulatively n times.

$$a^n = \underbrace{a \times a \times \dots \times a}_{n \text{ times}}$$

The specification and the loop invariant are shown as comments.

`power(a, n)`

-- inputs: n is a positive integer

-- outputs: $p = a^n$

`p, i := 1, 0`

`while i \neq n`

-- loop invariant: $p = a^i$

`p, i := p \times a, i+1`

The step by step execution of `power(2, 5)` is shown in Table 8.1. Each row shows the values of the two variables p and

i at the end of an iteration, and how they are calculated. We see that $p = a^i$ is true at the start of the loop, and remains true in each row. Therefore, it is a loop invariant.

iteration	p	p \times a	i	i+1	a^i
0	1		0		2^0
1	2	1 \times 2	1	0 + 1	2^1
2	4	2 \times 2	2	1 + 1	2^2
3	8	4 \times 2	3	2 + 1	2^3
4	16	8 \times 2	4	3 + 1	2^4
5	32	16 \times 2	5	4+1	2^5

Table 8.1: Trace of `power(2, 5)`

When the loop ends, $p = a^i$ is still true, but $i = 5$. Therefore, $p = a^5$. In general, when the loop ends, $p = a^n$. Thus, we have verified that `power(a, n)` satisfies its specification.

Example 8.6. Recall the Chocolate bar problem of Example 1.11. How many cuts are needed to break the bar into its individual squares?

We decided to represent the number of pieces and the number of cuts by variables p and c respectively. Whenever a cut is made, the number of cuts increases by one and the number of pieces also increases by one. We decided to model it by an assignment.

`p, c := p + 1, c+1`

The process of cutting the bar can be modeled by a loop. We start with one piece and zero cuts, $p = 1$ and $c = 0$. Let n be the number of individual squares. When the number of pieces p equals the number

of individual squares n , the process ends.

$p, c := 1, 0$

while $p \neq n$

$p, c := p + 1, c + 1$

We have observed (in Example 8.2) that $p - c$ is an invariant of the assignment $p, c := p + 1, c + 1$. Let $p - c = k$, where k is a constant. The points in the algorithm where $p - c = k$ is true are shown in the algorithm below, and in the flowchart of Figure 8.2.

$p, c := 1, 0$

1. -- $p - c = k$

while $p \neq n$

2. -- $p - c = k$

$p, c := p + 1, c + 1$

3. -- $p - c = k$

4. -- $p - c = k, p = n$

The loop invariant $p - c = k$ is True at the start of the loop (line 1). Moreover, at the start of the loop, $p - c = 1$. Therefore, $k = 1$, and the loop invariant is $p - c = 1$

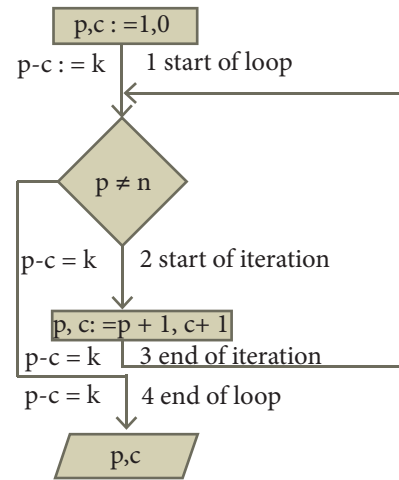


Figure 8.2: The points where the loop invariant is true

When the loop ends (line 4), the loop invariant is still true ($p - c = 1$). Moreover, the loop condition is false ($p = n$). From $p - c = 1$ and $p = n$,

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

When the process ends, the number of cuts is one less than the number of squares.

Example 8.7. There are 6 equally spaced trees and 6 sparrows sitting on these trees, one sparrow on each tree. If a sparrow flies from one tree to another, then at the same time, another sparrow flies from its tree to some other tree the same distance away, but in the opposite direction. Is it possible for all the sparrows to gather on one tree?

Let us index the trees from 1 to 6. The index of a sparrow is the index of the tree it is currently sitting on. A pair of sparrows flying can be modeled as an iterative step of a loop. When a sparrow at tree i flies to tree $i + d$, another sparrow at tree j flies to tree $j - d$. Thus, after each iterative step, the sum S of the indices of the sparrows remains invariant. Moreover, a loop invariant is true at the start and at the end of the loop.

At the start of the loop, the value of the invariant is

$$S = 1 + 2 + 3 + 4 + 5 + 6 = 21$$

When the loop ends, the loop invariant has the same value. However, when the loop ends, if all the sparrows were on the same tree, say k , then $S = 6k$.

$S = 21,$	loop invariant at the start of the loop
$S = 6k,$	loop invariant at end of the loop
$6k = 21,$	loop invariant has the same value at the start and the end
21 is a multiple of 6	

It is not possible — 21 is not a multiple of 6. The desired final values of the sparrow indices is not possible with the loop invariant. Therefore, all the sparrows cannot gather on one tree.

Example 8.8. Consider the Chameleons of Chromeland of Example 6.3. There are 13 red, 15 green, and 17 blue chameleons on Chromeland. When two chameleons of different colors meet they both change their color to the third one (for example,

if a red and a green meet, both become blue). Is it possible to arrange meetings that result in all chameleons displaying blue color?

Let r , g , and b be the numbers of red, green and blue chameleons. We can model the meetings of two types as an iterative process. A meeting changes (r, g, b) into $(r-1, g-1, b+2)$ or $(r-1, g+2, b-1)$ or $(r+2, g-1, b-1)$. Consider, for example, the meeting of a red and a green chameleon.

$$r, g, b := r-1, g-1, b+2$$

The difference in the numbers of any two types either do not change or changes by 3. This is an invariant.

$$r - 1 - (g - 1) = r - g$$

$$r - 1 - (b + 2) = (r - b) - 3$$

$$g - 1 - (b + 2) = (g - b) - 3$$

This is true for all three cases. If any two types differ in number by a multiple of 3 at the start of the iterative process, the difference can be reduced in steps of 3, to 0, when the iterative process ends. However, at the start,

$$r - g = 13 - 15 = -2$$

$$g - b = 15 - 17 = -2$$

$$g - r = 17 - 13 = 4$$

No two colors differ in number by a multiple of 3. Therefore, all the chameleons cannot be changed to a single color.

Example 8.9. Jar of marbles: You are given a jar full of two kinds of marbles, white and black, and asked to play this game. Randomly select two marbles from the jar. If they are the same color, throw them out, but put another black marble in (you may assume that you have an endless supply of spare marbles). If they are different colors, place the white one back into the jar and throw the black one away. If you knew the original numbers of white and black marbles, what is the color of the last marble in the jar?

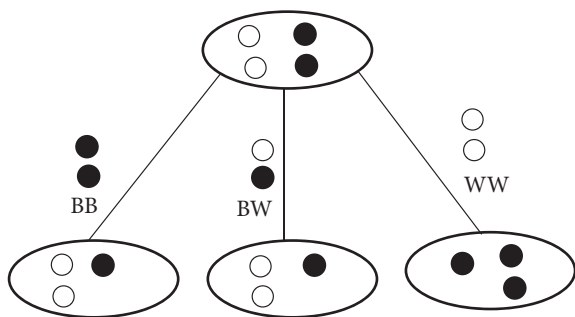


Figure 8.3: State changes in the jar marbles

The number of white and black marbles in the jar can be represented by two variables w and b . In each iterative step, b and w change depending on the colors of the two marbles taken out: Black Black, Black White or White White. It is illustrated in Figure 8.3 and annotated in the algorithm below.

```

1 while at least two marbles in jar
2   --  $b, w$ 
3   take out any two marbles
4   case both are black -- BB
5     throw away both the marbles
6     put a black marble back
7     --  $b = b' - 1, w = w', b + w = b' + w' - 1$ 
8   case both are white -- WW
9     throw away both the marbles
```

```

10    put a black marble back
11    --  $b = b' + 1, w = w' - 2, b + w = b' + w' - 1$ 
12  else
13    throw away the black one
14    put the white one back
15    --  $b = b' - 1, w = w', b + w = b' + w' - 1$ 
```

For each case, how b , w and $b+w$ change is shown in the algorithm, where b' and w' are values of the variables before taking out two marbles. Notice the way w changes. Either it does not change, or decreases by 2. This means that the parity of w , whether it is odd or even, does not change. The parity of w is invariant.

Suppose, at the start of the game, w is even. When the game ends, w is still even. Moreover, only one marble is left, $w+b = 1$.

```

1   $w + b = 1$       end of the loop
2   $w = 0$  or  $w = 1$   from 1
3   $w$  is even        loop invariant
4   $w = 0$             from 2,3
5   $b = 1$             from 1,4
```

Last marble must be black. Similarly, if at the start of the game, there is an odd number of whites, the last marble must be white.

One last question: do we ever reach a state with only one marble? Yes, because the total number of marbles $b+w$ always decreases by one at each step, it will eventually become 1.

8.4 Recursion

Recursion is an algorithm design technique, closely related to induction. It

is similar to iteration, but more powerful. Using recursion, we can solve a problem with a given input, by solving the instances of the problem with a part of the input.

Example 8.10. Customers are waiting in a line at a counter. The man at the counter wants to know how many customers are waiting in the line.

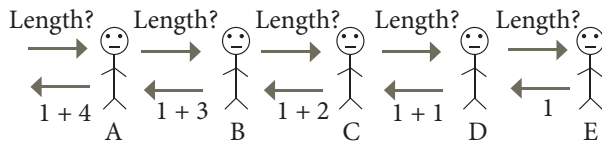


Figure 8.4: Length of a line

Instead of counting the length himself, he asks customer A for the length of the line with him at the head, customer A asks customer B for the length of the line with customer B at the head, and so on. When the query reaches the last customer in the line, E, since there is no one behind him, he replies 1 to D who asked him. D replies $1+1 = 2$ to C, C replies $1+2 = 3$ to B, B replies $1+3 = 4$ to A, and A replies $1+4 = 5$ to the man in the counter

8.4.1 Recursive process

Example 8.10 illustrates a recursive process. Let us represent the sequence of 5 customers A, B, C, D and E as

$[A,B,C,D,E]$

The problem is to calculate the length of the sequence $[A,B,C,D,E]$. Let us name our solver length. If we pass a sequence as input, the solver length should output the length of the sequence.

$\text{length } [A,B,C,D,E] = 5$

Solver length breaks the sequence $[A,B,C,D,E]$ into its first customer and the rest of the sequence.

$\text{first } [A,B,C,D,E] = A$

$\text{rest } [A,B,C,D,E] = [B,C,D,E]$

To solve a problem recursively, solver length passes the reduced sequence $[B,C,D,E]$ as input to a sub-solver, which is another instance of length. The solver assumes that the sub-solver outputs the length of $[B,C,D,E]$, adds 1, and outputs it as the length of $[A,B,C,D,E]$.

$\text{length } [A,B,C,D,E] = 1 + \text{length } [B,C,D,E]$

Each solver

1. receives an input,
2. passes an input of reduced size to a sub-solver,
3. receives the solution to the reduced input from the sub-solver, and produces the solution for the given input

as illustrated in Figure 8.5.

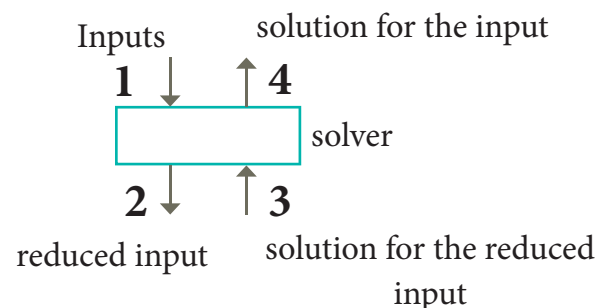


Figure 8.5: One instance of a solver in a recursive process

Figure 8.6 shows the input received and the solution produced by each

solver for Example 8.10. Each solver reduces the size of the input by one and passes it on to a sub-solver, resulting in 5 solvers. This continues until the input received by a solver is small enough to output the solution directly. The last solver received [E] as the input. Since [E] is small enough, the solver outputs the length of [E] as 1 immediately, and the recursion stops.

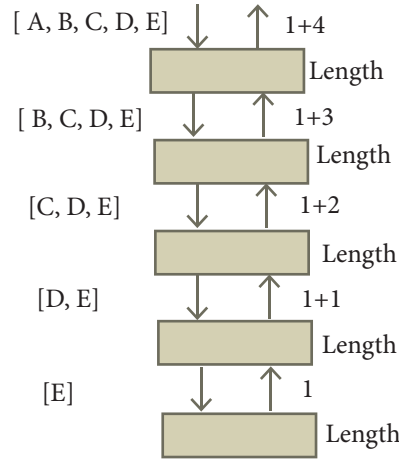


Figure 8.6: Recursive process with solvers and sub-solvers

The recursive process for length [A,B,C,D,E] is shown in Figure 8.7.

$$\begin{aligned}
 1 & \quad \text{length [A,B,C,D,E]} \\
 2 & \quad = 1 + 1 \text{ length [B,C,D,E]} \\
 3 & \quad = 1 + 1 + \text{length [C,D,E]} \\
 4 & \quad = 1 + 1 + 1 + \text{length [D,E]} \\
 5 & \quad = 1 + 1 + 1 + 1 + \text{length [E]} \\
 6 & \quad = 1 + 1 + 1 + 1 + 1 \\
 7 & \quad = 1 + 1 + 1 + 2 \\
 8 & \quad = 1 + 1 + 3 \\
 9 & \quad = 1 + 4 \\
 10 & \quad = 5
 \end{aligned}$$

Figure 8.7: Recursive process for computing the length of a sequence

8.4.2 Recursive problem solving

Each solver should test the size of the input. If the size is small enough, the solver should output the solution to the problem directly. If the size is not small enough, the solver should reduce the size of the input and call a sub-solver to solve the problem with the reduced input. For Example 8.10, solver's algorithm can be expressed as

$$\text{length of sequence} = \begin{cases} 1 & \text{if sequence has only one customer} \\ 1 + \text{length of tail,} & \text{otherwise} \end{cases}$$

To solve a problem recursively, the solver reduces the problem to sub-problems, and calls another instance of the solver, known as sub-solver, to solve the sub-problem. The input size to a sub-problem is smaller than the input size to the original problem. When the solver calls a sub-solver, it is known as recursive call. The magic of recursion allows the solver to assume that the sub-solver (recursive call) outputs the solution to the sub-problem. Then, from the solution to the sub-problem, the solver constructs the solution to the given problem.

As the sub-solvers go on reducing the problem into sub-problems of smaller sizes, eventually the sub-problem becomes small enough to be solved directly, without recursion. Therefore, a recursive solver has two cases:

1. **Base case:** The problem size is small enough to be solved directly. Output the solution. There must be at least one base case.
2. **Recursion step:** The problem size is not small enough. Deconstruct the problem into a sub-problem, strictly smaller in size than the given problem. Call a sub-solver to solve the sub-problem. Assume that the sub-solver outputs the solution to the sub-problem. Construct the solution to the given problem.

This outline of recursive problem solving technique is shown below.

```
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
```

2 × 16

Whenever we solve a problem using recursion, we have to ensure these two cases: In the recursion step, the size of the input to the recursive call is strictly smaller than the size of the given input, and there is at least one base case.

8.4.3 Recursion — Examples

Example 8.11. The recursive algorithm for length of a sequence can be written as

```
length (s)
-- inputs : s
-- outputs : length of s
if s has one customer -- base case
1
else
1 + length(tail(s)) -- recursion step
```

Example 8.12. Design a recursive algorithm to compute a^n . We constructed an iterative algorithm to compute a^n in

Example 8.5. a^n can be defined recursively as

$$a^n = \begin{cases} 1 & \text{if } n = 0 \\ a \times a^{n-1} & \text{otherwise} \end{cases}$$

The recursive definition can be expressed as a recursive solver for computing $\text{power}(a, n)$.

$\text{power}(a, n)$

-- inputs: n is an integer, $n \geq 0$

-- outputs: a^n

if $n = 0$ -- base case

1

else -- recursion step

$a \times \text{power}(a, n-1)$

The recursive process with solvers for calculating $\text{power}(2, 5)$ is shown in Figure 8.8.

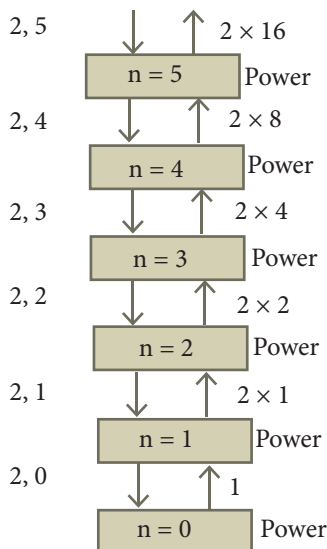


Figure 8.8: Recursive process with solvers for calculating $\text{power}(2, 5)$

The recursive process resulting from $\text{power}(2, 5)$ is shown in Figure 8.9.

```
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
```

Figure 8.9: Recursive process for $\text{power}(2, 5)$

Example 8.13. A corner-covered board is a board of $2^n \times 2^n$ squares in which the square at one corner is covered with a single square tile. A triominoe is a L-shaped tile formed with three adjacent squares (see Figure 8.10). Cover the corner-covered board with the L-shaped triominoes without overlap. Triominoes can be rotated as needed.

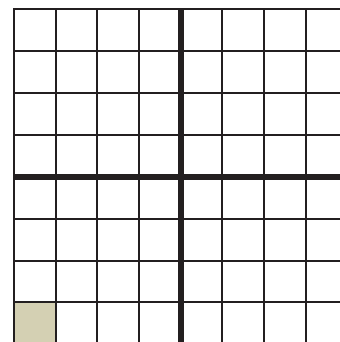
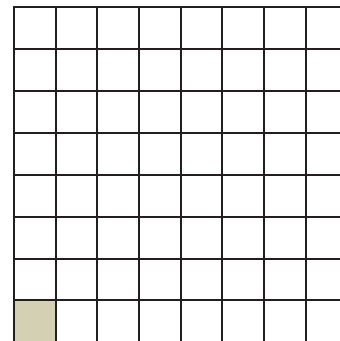


Figure 8.10: Corner-covered board and triominoe

The size of the problem is n (board of size $2^n \times 2^n$). We can solve the problem by recursion. The base case is $n = 1$. It is a 2×2 corner-covered board. We can cover it with one triominoe and solve the problem. In the recursion step, divide the corner-covered board of size $2^n \times 2^n$ into 4 sub-boards, each of size $2^{n-1} \times 2^{n-1}$, by drawing horizontal and vertical lines through the centre of the board. Place a triominoe at the center of the entire board so as to not cover the corner-covered sub-board, as shown in the left-most board of Figure 8.11. Now, we have four corner-covered boards, each of size $2^{n-1} \times 2^{n-1}$.

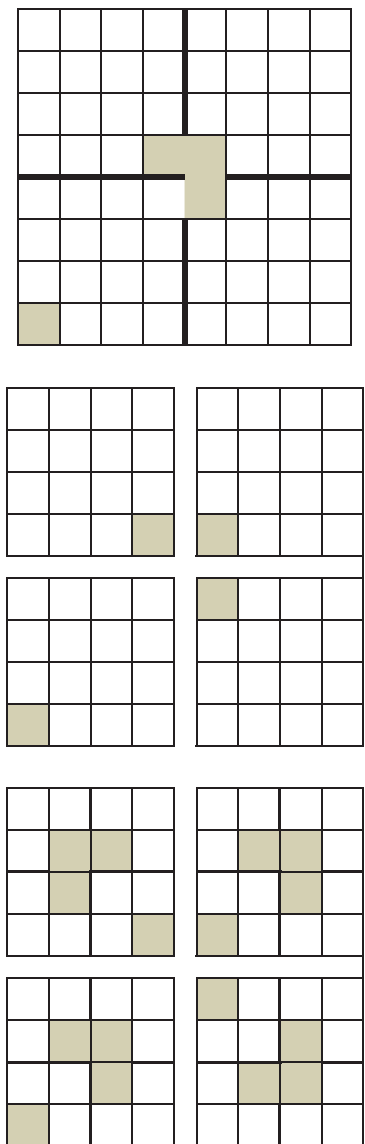
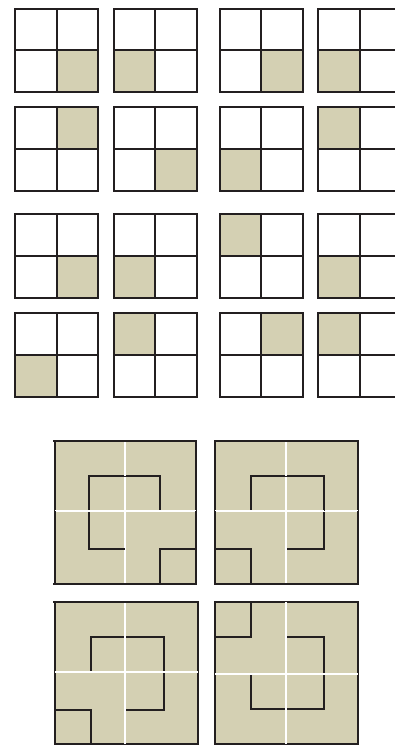


Figure 8.11: Recursive process of covering a corner-covered board of size $2^n \times 2^n$



We have 4 sub-problems whose size is strictly smaller than the size of the given problem. We can solve each of the sub-problems recursively.

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 $n-1$

The resulting recursive process for covering a $2^3 \times 2^3$ corner-covered board is illustrated in Figure 8.11.

Points to Remember

- Iteration repeats the two steps of evaluating a condition and executing a statement, as long as the condition is true.
- An expression involving variables, which remains unchanged by an assignment to one of these variables, is called an invariant of the assignment.
- An invariant for the loop body is known as a loop invariant.
- A loop invariant is true at
 - (a) at the start of the loop (just before the loop)
 - (b) at the start of each iteration (before loop body)
 - (c) at the end of each iteration (after loop body)
 - (d) at the end of the loop (just after the loop)
- When a loop ends, the loop invariant is true. In addition, the termination condition is also true.
- Recursion must have at least one base case.
- Recursion step breaks the problem into sub-problems of smaller size, assumes solutions for sub-problems are given by recursive calls, and constructs solution to the given problem.
- In recursion, the size of input to a sub-problem must be strictly smaller the size of the given input.

Evaluation



Part I

1. A loop invariant need not be true

(a) at the start of the loop.	(b) at the start of each iteration
(c) at the end of each iteration	(d) at the start of the algorithm
2. We wish to cover a chessboard with dominoes, the number of black squares and the number of white squares covered by dominoes, respectively, placing a domino can be modeled by

(a) $b := b + 2$	(b) $w := w + 2$	(c) $b, w := b+1, w+1$	(d) $b := w$
------------------	------------------	------------------------	--------------

3. If $m \times a + n \times b$ is an invariant for the assignment $a, b := a + 8, b + 7$, the values of m and n are

- (a) $m = 8, n = 7$ (b) $m = 7, n = -8$ (c) $m = 7, n = 8$ (d) $m = 8, n = -7$

4. Which of the following is not an invariant of the assignment?

$m, n := m+2, n+3$

- (a) $m \bmod 2$ (b) $n \bmod 3$ (c) $3 \times m - 2 \times n$ (d) $2 \times m - 3 \times n$

5. If Fibonacci number is defined recursively as

$$F(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F(n-1) + F(n-2) & \text{otherwise} \end{cases}$$

to evaluate $F(4)$, how many times $F()$ is applied?

- (a) 3 (b) 4 (c) 8 (d) 9

6. Using this recursive definition

$$a^n = \begin{cases} 1 & \text{if } n = 0 \\ a \times a^{n-1} & \text{otherwise} \end{cases}$$

how many multiplications are needed to calculate a^{10} ?

- (a) 11 (b) 10 (c) 9 (d) 8

Part II

1. What is an invariant?
2. Define a loop invariant.
3. Does testing the loop condition affect the loop invariant? Why?
4. What is the relationship between loop invariant, loop condition and the input-output recursively
5. What is recursive problem solving?
6. Define factorial of a natural number recursively.

Part III

1. There are 7 tumblers on a table, all standing upside down. You are allowed to turn any 2 tumblers simultaneously in one move. Is it possible to reach a situation when all the tumblers are right side up? (Hint: The parity of the number of upside down tumblers is invariant.)
2. A knockout tournament is a series of games. Two players compete in each game; the loser is knocked out (i.e. does not play any more), the winner carries on. The winner of the tournament is the player that is left after all other players have been knocked out. Suppose there are 1234 players in a tournament. How many games are played before the tournament winner is decided?
3. King Vikramaditya has two magic swords. With one, he can cut off 19 heads of a dragon, but after that the dragon grows 13 heads. With the other sword, he can cut off 7 heads, but 22 new heads grow. If all heads are cut off, the dragon dies. If the dragon has originally 1000 heads, can it ever die? (Hint: The number of heads mod 3 is invariant.)

Part IV

1. Assume an 8×8 chessboard with the usual coloring. "Recoloring" operation changes the color of all squares of a row or a column. You can recolor repeatedly. The goal is to attain just one black square. Show that you cannot achieve the goal. (Hint: If a row or column has b black squares, it changes by $(|8 - b) - b|$).
2. Power can also be defined recursively as

$$a^n = \begin{cases} 1 & \text{if } n = 0 \\ a \times a^{n-1} & \text{if } n \text{ is odd} \\ a^{n/2} \times a^{n/2} & \text{if } n \text{ is even} \end{cases}$$

Construct a recursive algorithm using this definition. How many multiplications are needed to calculate a^{10} ?

3. A single-square-covered board is a board of $2^n \times 2^n$ squares in which one square is covered with a single square tile. Show that it is possible to cover the this board with triominoes without overlap.

COMPUTER SCIENCE – XI

List of Authors and Reviewers

Domain Experts

Dr. Chitra Babu
Professor and Head of the Department, Dept of Computer Science and Engineering, SSN College of Engineering, Chennai

Mrs. Bagyalakshmi P
Asst. Professor and Head of the Department, Dept of Computer Applications, Queen Mary's College, Chennai

Mrs. Sasikala k
Associate Professor, Dept of Computer Science, Queen Mary's College, Chennai

Dr. Radha P
Assistant Professor, Dept of Information Technology, Govt. Arts & Science College (A), Coimbatore

Dr. Nester Jeyakumar M
Associate Professor and Head Of the Department, Dept of Computer Science, Loyola College, Chennai

Dr. Srinivasan N
Professor, Dept of Computer Science and Engineering, Sathyabama Institute of Science & Technology, Chennai

Dr. Chandra Mohan B
Associate Professor, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore

Mr. Sethuraman R
Assistant Professor, Dept of Computer Science and Engineering, Sathyabama Institute of Science & Technology, Chennai

Mr. Sankar K
Assistant Professor, Dept of Computer Science, RKM Vivekananda College, Mylapore, Chennai

Experts Co-ordinator

Mr. Ravikumar Arumugam
Deputy Director, SCERT, Chennai

Art and Design Team

Chief Co-ordinator and Creative Head

Srinivasan Natarajan

Layout

THY designers and computers
Chennai

In-House

QC - Gopu Rasuvel
- Rajesh Thangapan
- Raghupathy, TamilKumaran. C
- Wrapper Design - Kathir Arumugam

Co-ordination

Ramesh Munisamy

Reviewers

Dr. Ranjani Parthasarathi
Professor, Dept of Info Sci and Tech, College of Engineering, Guindy, Anna University, Chennai

Mr. Munivel E
Scientist/Engineer 'C' IT Group (Information Security), NIELIT Calicut (MeitY, Govt. of India), NIT Campus, Calicut - KL (IN).

Authors

Mr. Kannan K
Post Graduate Teacher, Chennai Girls Hr Sec School, Rotler street, Chennai

Mr. Ramakrishnan V G
Post Graduate Teacher, Karnataka Sangha Hr Sec School, T Nagar, Chennai

Mrs. Bindhu Mohandas
Post Graduate Teacher, Vijayanta Model Hr Sec School, H.V.F Estate, Avadi, Chennai

Mr. Gowrisankar N.V
Post Graduate Teacher, Chennai Girls Hr Sec School, Nungambakkam, Chennai

Mr. Sreenivasan R
Post Graduate Teacher, Santhome Hr Sec School, Mylapore, Chennai

Mr. Lenin K
Post Graduate Teacher, Chennai Girls Hr Sec School, Saidapet, Chennai

Miss. Sangeetha A
Post Graduate Teacher, Govt. Hr Sec School, Rajanthangal, Thiruvannamalai Dt

Dr. Valarmathi K E
Post Graduate Teacher, Velammal Vidhyashram, Surapet, Chennai

Mrs. Gajalakshmi R
Post Graduate Teacher, Jaigopal Garodia Hindu Vidyalaya Hr Sec School, West Mambalam, Chennai

Academic Coordinators

Mrs. Nevedha Selvaraj
Assistant Professor, SCERT, Chennai

Mrs. Tamil Selvi R
B.T. Assistant,
Government High School, Poonampalayam, Trichy District

This book has been printed on 80 G.S.M.
Elegant Maplitho paper.

Printed by offset at: