

Based on National Curriculum of Pakistan 2022-23

Model Textbook of

# COMPUTER SCIENCE

Grade 9



Cantab Publisher  
Lahore, Pakistan



# Computer Science

Grade

9



**Cantab Publisher Lahore, Pakistan**

All rights reserved. This volume may not be reproduced in whole or in part in any form (abridged, photocopy, electronic etc.) without prior written permission from Cantab Publisher.

A Textbook of  
for Grade

**Authors**

Computer Science

**Editorial Board**

Prof. Dr. Shazia Naeem  
Prof. Dr. Naeem Khalid

**Supervision**

Dr. Mariam Chughtai  
Director, National Curriculum Council  
Ministry of Federal Education and Professional Training, Islamabad

**Reviewed by Internal Review Committee**

Nadeem Mahmood Shaikh  
Sana Talha  
Muhammad Nabeel  
Ms Lubna Zakia  
Ms Asma Haroon  
Sadiah Mujtaba  
Usman Adeel  
Ayaz Abdullah  
Khubab Hamza  
Sadaf Zehra Kazmi

**Reviewed by National Review Committee**

Mr. Muhammad Hashim.  
Sufain Matloob  
Mudassar Manzoor Qureshi  
Sadaf Zehra kazmi.  
Lubna Zakia  
Mr. Jahanzaib Khan, .  
Qurban Karim  
Mr. Mushtaq Ahmed

**Desk Officer (NCC)**

Zehra Khushal

**Management**

Prof. Dr. Shazia Naeem  
CEO Cantab Publisher

**First Edition - First Impression**

**Price:** PK 000/-

**Code:** , **ISBN:**

**Printer:**

**Note:** All the pictures, paintings and sketches used in this book are only for educational and promotional purpose in public interest.

For Information about other publications of Cantab Publisher,

Visit our Web site: [www.cantabpublisher.com](http://www.cantabpublisher.com)

or E-mail: [info@cantabpublisher.com](mailto:info@cantabpublisher.com)

to share feedback or correction, please send us an email to [info@cantabpublisher.com](mailto:info@cantabpublisher.com) and [textbooks@snc.gov.pk](mailto:textbooks@snc.gov.pk)

# SLO based Model Video lecture



## Salient Features

### Comprehensive Learning

Engage students with videos, simulations, and practical worksheets.

### Structured Lesson Plan

Well-organized with clear objectives, PPTs, and a question bank.

### Engaging Multimedia

Visual appeal through PPTs and interactive simulations.

### Assessment & Tracking

Diverse question bank and progress monitoring.

### Adaptable & Accessible

Scalable and accessible, suitable for all learners.



Simulation



Power Point Presentation

## SLO:CS-09-A-01

Students will define and describe types of systems (artificial, natural), computer hardware components such as computer architecture (cpu, microprocessors, etc.)

## Knowledge

### Knowledge 1.1 History of Computers

The first counting device was used by the primitive people. They used sticks, stones and bones as counting tools. As human mind and technology improved with time more computing devices were developed. Some of the popular computing devices starting with the first to recent ones are described below:

#### a. Abacus

The history of computers begins with the birth of abacus which is believed to be the first computer. It is said that Chinese invented Abacus around 8,000 years ago.

It was a wooden rack, which has metal rods with beads mounted on them. The beads were moved by the abacus operator according to some rules to perform arithmetic calculations. Abacus is still used in some economies like China, Russia and Japan. An image of this tool is shown below:



Fig. 1.1(a)

#### b. Napier's Bones

It was a manually-operated calculating device which was invented by John Napier (1550-1617) of Merchiston. In this calculating tool, he used 9 different ivory strips or bones marked with numbers to multiply and divide. So, this tool became known as "Napier's Bones". It was also the first machine to use the decimal point.



Fig. 1.1(b)

#### c. Pascaline

Pascaline is also known as Arithmetic Machine or Adding Machine. It was invented between 1642 and 1644 by a French mathematician-philosopher Blaise Pascal. It is believed that it was the first mechanical and automatic calculator.

Pascal invented this machine to help his father, a tax accountant. It could only perform addition and subtraction. It was a wooden box with a series of gears and wheels. When a wheel is rotated one revolution, it rotates the neighboring wheel. A series of windows is given on the top of the wheels to read the totals. An image of this tool is shown below:



Fig. 1.1(c)

#### d. Stepped Reckoner or Leibniz wheel

It was developed by a German mathematician-philosopher Gottfried Wilhelm Leibniz in 1673. He improved Pascal's invention to develop this machine. It was a digital mechanical calculator which was called the stepped reckoner as instead of gears it was made of fitted drums. See the following image:



Fig. 1.1(d)



Question Bank



Work sheet

55002

**Note:** Read the Notes carefully to Answer this worksheet questions student will not be able to answer the questions who will not carefully read the notes which is also provided this worksheet too.

Below template is created in Canva After creating the same template as this format take screen shot of your work and send it to your class teacher for the assessment of your skill in Canva.

Note Teacher will add marks of student's assignment as per their assessment in their exam.

The screenshot shows a digital form titled "Class SCHEDULE" designed for a teacher to fill out. The form is set against a light blue background with decorative orange and yellow circular elements. At the top, there's a yellow header bar. Below it, the title "Class SCHEDULE" is prominently displayed, with "Class" in a cursive font and "SCHEDULE" in a bold, orange, sans-serif font. Under the title, there are two input fields: "Name" and "Homeroom". The main part of the form is a table with five columns representing different times of the day: "MORNING", "MIDMORNING", "AFTERNOON", "EVENING", and "SIXTH". The rows are labeled with times from "8:00 AM" to "10:00 PM". The table cells are currently empty, ready for the teacher to input the schedule. At the bottom of the form, there is a yellow bar with the text "Teacher (Print)".

## Work sheet

### SLO-A01 (Exercise)

### Multiple-Choice Questions (MCQs):

1. Which generation of computers used vacuum tubes as the main electronic component?  
a) First Generation b) Second Generation c) Third Generation d) Fourth Generation
2. What is the primary function of a CPU (Central Processing Unit) in a computer?  
a) Storing data b) Processing instructions and calculations c) Providing power to the computer d) Controlling input and output devices.
3. What type of memory is volatile, meaning data is lost when the computer is powered off?  
a) Cache Memory b) Main Memory (RAM) c) Secondary Storage (HDD) d) Tertiary Storage (Tape Drives)
4. Which component is considered the "brain" of a computer?  
a) Motherboard b) CPU (Central Processing Unit) c) Hard Disk Drive (HDD) d) Monitor

## Question Bank





## Preface

Welcome to the exciting world of Computer Science! This book has been meticulously crafted to introduce you to the foundational principles and dynamic aspects of computing that have become integral to our modern lives. Whether you're just beginning your journey or seeking to deepen your understanding, this resource aims to spark curiosity, foster critical thinking, and ignite your passion for the incredible realm of technology.

We delve into the intricate architecture of computing systems, exploring hardware, software, and the interactions between them. Understanding the components that make up computers and their functionalities will lay the groundwork for comprehending the complexities behind digital devices we use every day.

Computational thinking forms the bedrock of problem-solving in computer science. Here, we'll unravel the essence of breaking down challenges into solvable steps, cultivating algorithmic thinking and the creation of efficient algorithms to tackle diverse problems.

Programming is the language through which we communicate with computers. In this chapter, we will embark on an exploration of programming fundamentals, learning the syntax, logic, and structure of various programming languages to create software and applications.

Data is the lifeblood of the digital age. This section will illuminate the significance of data, its types, collection, storage, and the methodologies used for analysis and interpretation. Uncover the power of data-driven decision-making and its transformative impact across industries.

From artificial intelligence to cybersecurity and beyond, this chapter showcases the diverse applications of computer science. Explore how technological innovations are shaping fields like healthcare, finance, entertainment, and transportation, revolutionizing our world.

Technology is a catalyst for change. This section examines the broader implications of computing on society, ethics, privacy, and the environment. Delve into the ethical considerations and societal impacts arising from the rapid advancement of technology.

In today's digital landscape, proficiency in navigating the online realm is crucial. This chapter equips you with the necessary skills to critically evaluate information, practice responsible digital citizenship, and safeguard against potential risks in the digital sphere.

Entrepreneurship and innovation thrive in the digital era. Learn how technological advancements have provided fertile ground for entrepreneurial endeavors, fostering creativity, and enabling the development of groundbreaking startups and businesses.

As you embark on this educational journey, engage with the content, experiment with hands-on activities, and embrace the challenges posed by each chapter. Computer science is not merely about mastering concepts but also about cultivating a mindset that thrives on curiosity, creativity, and problem-solving.

Let this book serve as a guiding light, empowering you to explore the depths of computer science and inspiring you to become active contributors to the ever-evolving technological landscape.

# Table of Content

## Domain

# A

## Computer Systems

# 01

### A-01 System Type and Architecture

Knowledge 1.1	1
Knowledge 1.2	3
Knowledge 1.3	5
Knowledge 1.4	7
Knowledge 1.5	8
Knowledge 1.6	10
Knowledge 1.7	11
Knowledge 1.8	13
Knowledge 1.9	15
Knowledge 1.10	16
Knowledge 1.11	17
Knowledge 1.12	17
Exercise	19
A-02 Software and Programming Languages	21
Knowledge 1.13	23
Knowledge 1.14	24
Knowledge 1.15	25
Knowledge 1.16	27
Exercise	33
A-03 Information Networks	35
Knowledge 1.17	38
Knowledge 1.18	39
Knowledge 1.19	40
Knowledge 1.20	41
Knowledge 1.21	42
Knowledge 1.22	43
Knowledge 1.23	43
Knowledge 1.24	44
Knowledge 1.25	47
Knowledge 1.26	49
Exercise	50
	53

## Domain

# B

## Computational Thinking & Algorithms

# 55

### B-01 Importance of Computational Thinking

Knowledge 2.1	55
Knowledge 2.2	57
Knowledge 2.3	59
Knowledge 2.4	61
Knowledge 2.5	64
	66

Knowledge 2.6

Knowledge 2.7

Exercise

### B-02 Role Of Computational Thinking

Knowledge 2.8	68
Knowledge 2.9	68
Knowledge 2.10	70
Knowledge 2.11	72
Knowledge 2.12	73
Knowledge 2.13	74
Exercise	76
	78
	79
	82
	84

## Domain

# C

## Programming Fundamentals

# 87

### C-01 Building of Website

Knowledge 3.1	87
Knowledge 3.2	89
Knowledge 3.3	90
Knowledge 3.4	91
Exercise	92
	94

### C-02 Introduction of HTML

Knowledge 3.5	96
Knowledge 3.6	97
Knowledge 3.7	98
Knowledge 3.8	99
Knowledge 3.9	99
Knowledge 3.10	99
Knowledge 3.11	100
Exercise	101
	102

### C-03 Introduction of Javascript

Knowledge 3.12	104
Knowledge 3.13	105
Knowledge 3.14	107
Exercise	108
	109

### C-04 Control Structures and Arrays in Javascript

Knowledge 3.15	111
Knowledge 3.16	112
Knowledge 3.17	112
Knowledge 3.18	113
Exercise	115
	117

### C-05 C-04 Dubugging in Javascript

Knowledge 3.19	119
Knowledge 3.20	120
Knowledge 3.21	122
Knowledge 3.22	123
Exercise	128
	131

## Domain

# D

## Data and Analysis

133

### D-01 Key Principles of Data science

Knowledge 4.1	133
Knowledge 4.2	135
Knowledge 4.3	136
Knowledge 4.4	137
Exercise	139

### D-02 Data and Data Set Representation

Knowledge 4.5	140
Knowledge 4.6	142
Knowledge 4.7	143
Knowledge 4.8	145
Knowledge 4.9	146
Knowledge 4.10	148
Exercise	149

### D-03 Big Data Representation

Knowledge 4.11	153
Knowledge 4.12	155
Knowledge 4.13	156
Knowledge 4.14	157
Knowledge 4.15	158
Knowledge 4.16	159
Knowledge 4.17	160
Knowledge 4.18	161
Exercise	165

Knowledge 6.2

Knowledge 6.3

Knowledge 6.4

Knowledge 6.5

Knowledge 6.6

Knowledge 6.7

Knowledge 6.8

Knowledge 6.9

Exercise

### F-02 Beneficial & Harmful Effects of Computing

Knowledge 6.10	189
Knowledge 6.11	190
Knowledge 6.12	191
Knowledge 6.13	193
Knowledge 6.14	194
Knowledge 6.15	194
Knowledge 6.16	195
Knowledge 6.17	196
Knowledge 6.18	197
Exercise	200

### F-02 Environment Impacts of Computing

Knowledge 6.19	201
Knowledge 6.20	202
Knowledge 6.21	203
Knowledge 6.22	203
Knowledge 6.23	205
Exercise	206

## Domain

# E

## Applications of Computer Science

171

### E-01 AI and Machine Learning Applications

Knowledge 5.1	171
Knowledge 5.2	173
Knowledge 5.3	174
Knowledge 5.4	175
Knowledge 5.5	175
Exercise	176

### E-02 Ethical Issues of AI

Knowledge 5.6	178
Knowledge 5.7	180
Knowledge 5.8	181
Knowledge 5.9	182
Exercise	183

## Domain

# F

## Impacts of Computing

187

### F-01 Data Ethics and Computing Laws

Knowledge 6.1	187
---------------	-----

## Domain

# H

## Entrepreneurship in Digital Age

221

### H-01 Business Development Skills

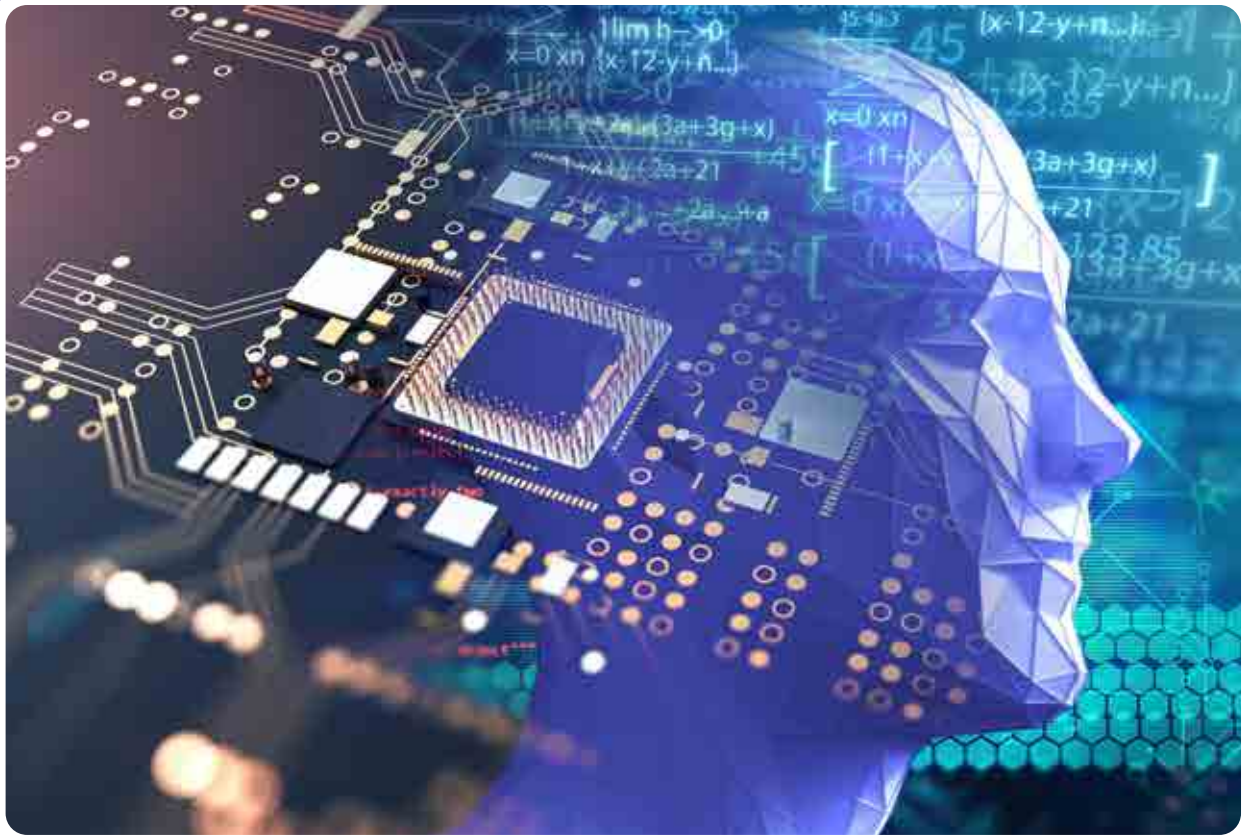
Knowledge 7.1	221
Knowledge 7.2	222
Knowledge 7.3	224
Knowledge 7.4	226
Knowledge 7.5	226
Exercise	228

### H-02 Use of Digital Tools for Business Development

Knowledge 7.6	229
Knowledge 7.7	231
Knowledge 7.8	232
Knowledge 7.9	232
Knowledge 7.10	233
Knowledge 7.11	234
Knowledge 7.12	234
Exercise	235



# Computer Systems

**SLO: A01****System Types and Architecture****Standard**

Student will understand and explain the various components of a computer system and the different levels of interactions between these. They will have a basic understanding of digital logic, the different stages of the software life cycle and concepts of scalability, reliability and security for computer systems.

## ● Students' Learning Outcomes-1 ●



students will define and describe types of systems (artificial, natural), computer hardware components such as computer architecture (CPU, microprocessors.)



### Knowledge

#### Student will understand that

- ✔ Brief history of computer systems and generation of computers.
- ✔ Basic concept of a system, types of systems.
- ✔ Name and identify the core components of a computer (input/output devices, system unit (motherboard, memory, CPU, power supply, etc.), and storage devices.
- ✔ Understand and identify Von Neumann Architecture.
- ✔ Computer architecture how data is transmitted within a computer system.
- ✔ What are some of the requirements for a functioning computer system and what are some key concepts?

#### Students will know

- ✔ The core parts of a computer system and how they all work together, including definitions and key functions of computer architecture such as central processing unit (CPU), arithmetic logic unit (ALU), control unit (CU), memory, operating system and application software, and data representation in computers (bit, byte, binary, denary/decimal, hexadecimal)
- ✔ Difference between hardware engineering and software engineering
- ✔ Difference between natural and artificial systems
- ✔ Types and hierarchy of memory with respect to their volatility/retention, speed, storage capacity, cost
- ✔ Difference between necessary and auxiliary components of a computer system



### Skills

#### Students will be able to:

- ✔ Recognize and describe key components of a computer system, such as:
- ✔ Differentiate between natural and artificial systems  
Outline the architecture of the central processing unit (CPU) and the functions of the arithmetic logic unit (ALU) and the control unit (CU) and the registers within the CPU.
- ✔ Differentiate between primary memory, cache and secondary memory
- ✔ Describe the main functions of operating systems and application systems.
- ✔ Explain how the various components interact together to transmit data and instructions
- ✔ Illustrate the hierarchy of memory and storage devices with respect to their volatility/retention, speed, storage capacity, cost
- ✔ Differentiate between necessary and auxiliary components of a computer system

## Diagnostic Assessment

As a class activity for grade 9 level students to introduce the chapter on computer Systems, some possible prerequisite questions could include:

- a. What is a computer system?
- b. What is a computer network?
- c. Define input and output devices. Give examples of each.

In the previous grade, we embarked on an extraordinary voyage through the realm of technology, where we immersed ourselves in the captivating domains of innovation. Our exploration included exciting encounters with cutting-edge concepts such as robotics, Augmented Reality (AR), Virtual Reality (VR), Artificial Intelligence (AI), 3D, and holographic imaging. Engaging in hands-on activities, we delved into the boundless possibilities these technologies offer.

Our journey involved constructing and programming robots, marveling at their precision and efficiency in task execution. We donned VR headsets, transporting ourselves to mesmerizing virtual worlds and redefining our understanding of storytelling and education. Witnessing the power of AI, we trained machines to learn and make intelligent decisions. Exploring the wonders of 3D, we designed and brought our creations to life through innovative printing techniques. The allure of holographic images captivated us, appearing to materialize right before our eyes.

These immersive experiences cultivated a profound appreciation for the profound impact of technology on our lives. They equipped us with the confidence to navigate the ever-evolving technological landscape. As we eagerly anticipate the journey ahead in the next grade, we are poised to continue our exploration, venturing even deeper into the world of technology. We are prepared to unveil its transformative potential across various fields, further enriching our understanding and skills in this dynamic and ever-expanding domain.

## Knowledge 1.1 | History of Computers

The first counting device was used by the primitive people. They used sticks, stones and bones as counting tools. As human mind and technology improved with time more computing devices were developed. Some of the popular computing devices starting with the first to recent ones are described below;

### a. Abacus

The story of computers starts with the creation of the abacus, considered the earliest form of a computer. Around 4,000 years ago, it's believed that the Chinese invented this device. The abacus was made of wood and had metal rods holding beads. By following specific rules, people using the abacus could perform math calculations by moving these beads. Even today, countries like China, Russia, and Japan continue to use the abacus for calculations. Below is an image of this tool.



Fig. 1.1 (a) Abacus

### b. Napier's Bones

John Napier, who lived from 1550 to 1617 in Murchison, invented a





Fig. 1.1 (b) Napier's Bones



Fig. 1.1 (c) Pascaline



Fig. 1.1 (d) Stepped Reckoner or Leibniz wheel

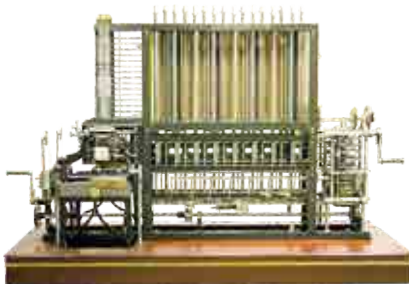


Fig. 1.1 (e) Difference Engine

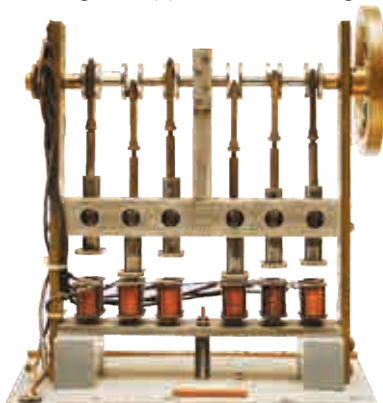


Fig. 1.1 (f) Analytical Engine

calculating device that needed to be operated by hand. This tool, called "Napier's Bones," used nine ivory strips or bones with numbers on them to help with multiplication and division. It was also the first machine to include the use of the decimal point.

### c. Pascaline

The Pascaline, also known as the Arithmetic Machine or Adding Machine, was created by a French mathematician named Blaise Pascal between 1642 and 1644. This invention is believed to be the first calculator that worked automatically using machinery.

Blaise Pascal made this machine to assist his father, who worked as a tax accountant. It was capable of performing addition and subtraction only. The Pascaline appeared as a wooden box containing gears and wheels. When one wheel turned utterly, it made the next wheel turn. There were windows on top of the wheels to show the calculated totals. Here's an image of this tool.

### d. Stepped Reckoner or Leibniz wheel

In 1673, a German mathematician-philosopher named Gottfried Wilhelm Leibniz created an improved version of Pascal's invention. He developed a digital mechanical calculator known as the stepped reckoner. Unlike Pascal's machine, which used gears, this one was made of fluted drums. Please take a look at the image below to see what it looks like.

### e. Difference Engine

Around the early 1820s, Charles Babbage, often referred to as the "Father of Modern Computer," created a mechanical computer. This machine could do basic calculations and was powered by steam. Its main purpose was to solve sets of numbers, such as logarithm tables.

### f. Analytical Engine

Charles Babbage also created a calculating machine around 1830. This mechanical computer used punch cards for input. It could solve various types of mathematical problems and store information as permanent memory.

### g. Tabulating Machine

Around 1890, Herman Hollerith, an American statistician, invented a mechanical tabulator. This machine used punch cards and was capable of tabulating statistics, recording, and sorting data or

information. It was notably utilized in the 1890 U.S. Census. Hollerith established the Hollerith's Tabulating Machine Company, which eventually became International Business Machines (IBM) in 1924.

#### h. Differential Analyzer

In 1930, the United States saw the introduction of the first electronic computer. It was an analogue device created by Vannevar Bush. This machine used vacuum tubes to switch electrical signals for doing calculations. It could complete 25 calculations within a few minutes.

#### i. Mark I

Significant developments in computer history started in 1937 when Howard Aiken aimed to create a machine capable of handling complex calculations with big numbers. By 1944, the Mark I computer was constructed through a collaboration between IBM and Harvard. This was the initial programmable digital computer.

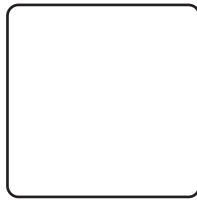


#### — Learning Activity

Share questions with students on how computers work and then show the given video in class to explain in their own words

**How computers work and what they do:**

<https://www.youtube.com/watch?v=92TaQRBwPSs>



### Knowledge 1.2 | Generations of Computer

A computer generation means the different ways computer technology gets better over time. In 1946, they made electronic pathways called circuits to do counting tasks. These circuits took the place of gears and other mechanical parts that were used for counting in older computers.

As each new generation came along, the circuits got tinier and better than the ones before. Making them more minor helped computers become faster, have more memory, and be more powerful. There are five generations of computers, which we'll talk about below.

#### a. First Generation (1940s - 1950s)

The first electronic digital computers came about during World War II. Famous ones were ENIAC (Electronic Numerical Integrator and Computer) and UNIVAC. They used vacuum tubes, which were extensive, costly, and not very dependable. Here are the main features of the first generation of computers (1940s-1950s):

- Essential electronic part: vacuum tube



Fig. 1.1 (g) Tabulating Machine

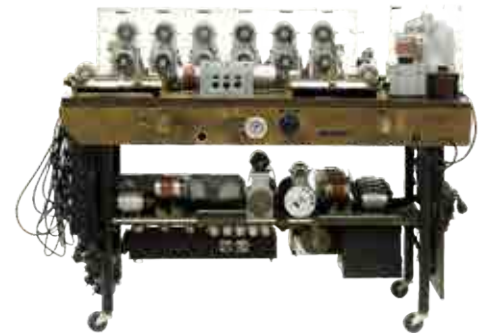


Fig. 1.1 (h) Differential Analyzer



Fig. 1.1 (i) Mark I



Fig. 1.2 (a) Vacuum tube

- Primary memory: magnetic drums and magnetic tapes
  - Language used for programming: machine language
  - Energy usage: used a lot of electricity and made a lot of heat
  - Speed and size: very slow and huge (often filling entire rooms)
- Examples include ENIAC, UNIVAC1, IBM 650, IBM 701, and more.

#### b. Second Generation (1950s - 1960s)

The next generation of computers, called the second generation, used transistors instead of vacuum tubes. Transistors were smaller, more reliable, and used less energy. These computers became smaller and faster. Here are the main points about the second generation of computers:

- Essential electronic part: transistor
- Memory: magnetic core and magnetic tape/disk
- Programming language: assembly language
- Power and size: used less power, produced less heat and were smaller compared to first-generation computers
- Examples include IBM 1401, IBM 7090 and 7094, UNIVAC 1107, and more.

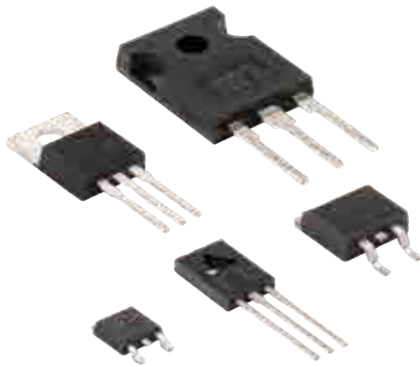


Fig. 1.2 (b) Transistors

#### c. Third Generation (1960s - 1970s)

In the third generation of computers, which was in the 1960s-1970s, there came integrated circuits (ICs). These ICs let more parts fit onto a single chip, making computers smaller, faster, and more dependable. Here are the main features of the third generation of computers:

- Essential electronic part: integrated circuits (ICs)
- Memory: large magnetic core, magnetic tape/disk
- Programming language: high-level languages like FORTRAN, BASIC, Pascal, COBOL, C, and more
- Examples include IBM 360, IBM 370, PDP-11, UNIVAC 1108, and others.



Fig. 1.2 (c) Transistors

#### d. Fourth Generation (1970s - 1990s)

This era saw the development of microprocessors, which made personal computers possible. IBM introduced the IBM PC in the early 1980s, and Apple released the Macintosh. Software, like the operating system MS-DOS, also became essential. The main characteristics of fourth generation of computers (1970s-present)

- Main electronic component – very large-scale integration (VLSI) and microprocessor.
- VLSI– thousands of transistors on a single microchip.
- Memory – semiconductor memory (such as RAM, ROM, etc.)



Fig. 1.2 (d) Transistors



- Programming language – high level language (Python, C#, Java, JavaScript, Rust, Kotlin, etc.).
- Examples–IBM PC, STAR 1000, APPLE II, Apple Macintosh, etc.
- **e. Fifth Generation (1990s - Present)**

From the 1990s until now, fifth-generation computers use advanced ULSI technology with microprocessor chips having millions of components. They rely on parallel processing and AI software, making computers think like humans. Programming languages like C, C++, Java, and .Net are used to create powerful software in this generation.



Fig. 1.2 (e) Fifth generation computer

### Knowledge 1.3 | Types of Computers

#### a. Digital

Personal computers belong to a type of computer called digital computers. They take in information using 0s and 1s. When you give them information, they process it and give you an answer. These computers can do the math and make logical decisions. Before the computer does anything with the information you give it, it changes the input into the 0s and 1s language. Examples of these computers are laptops, PCs, mobile phones, and desktops.

#### b. Analog

Analog computers handle information that doesn't have fixed values. This kind of data keeps changing without specific steps. These computers understand continuous changes in what they're given, figure things out, and give an answer. They work accurately but are slower and less exact than digital computers. Analogue computers are used for things like measuring speed, temperature, frequency, voltage signals and checking the resistance of a capacitor.

#### c. Hybrid

Hybrid computers are a blend of analogue and digital



#### Skill:1.1 (a)

#### Identify Key Components of the Computer System.

#### Unraveling Natural and Artificial Systems

**Objective:** The aim of this activity is to help grade 9 students differentiate between natural and artificial systems through observation and analysis. Scan the QR code for further details.



Fig. 1.3 (a)



Fig. 1.3 (b)



Fig. 1.3 (c)



Fig. 1.4 (a) Super Computer

computers. They're really good at doing lots of calculations. These computers work fast and get things done well. They take in information that's analogue, change it into digital, and then figure things out to answer. Scientists use hybrid computers for laborious calculations. For instance, in hospitals, they use them to measure patients' heartbeats, and in research, centers use them to measure things like earthquakes and other natural disasters.

## Knowledge 1.4 | Classification of Computers

Computers can be categorized in different ways, considering their size and how they manage data. Let's discuss each type of computer in detail. First, let's take a look at the different types:

- Super Computer
- Mainframe Computer
- Mini Computer
- Micro Computer

### a. Supercomputer

When we consider speed in computers, the first type that stands out is the supercomputer. These machines are the largest and quickest when it comes to processing data. Supercomputers are built to handle massive amounts of data, like handling trillions of instructions or data in just one second. They achieve this due to having thousands of connected processors. They're mainly used for scientific and engineering tasks such as predicting weather, conducting scientific simulations, and researching nuclear energy. The first one was created by Roger Cray in 1976.

Characteristics of Supercomputers:

- Supercomputers are the fastest but are also very costly.
- They can perform up to ten trillion individual calculations per second, which makes them incredibly fast.
- They're used in stock markets and large organizations for managing online currencies like Bitcoin.
- In scientific research areas, they're used to analyze data collected from exploring the solar system, satellites, and more.

### b. Mainframe computer

Mainframe computers are specially designed to help many people at once and handle lots of tasks at the same time. They're perfect for big companies like banks and telecom companies because they deal with lots of information.

Here are some things that make mainframe computers unique:



- Mainframe computers can be expensive, but they're mighty and can do a lot of things at once.
- They can store a vast amount of information and work really fast. This is important, especially in banking, where there's a ton of data to handle.
- Mainframes are great at handling large amounts of data, especially in the banking sector, where there's so much information to process.
- These computers can keep working well for a long time. They're reliable and have a long life.

### c. Minicomputer

A minicomputer is a mid-sized computer that can do many things at the same time. It has two or more processors and can be used by 4 to 200 people together. It's similar to a microcomputer but a bit bigger.

Here are some things that make minicomputers unique:

- Minicomputers are not very heavy, which makes them easy to move around.
- Because they're not heavy, you can carry them from place to place without much trouble.
- Minicomputers are cheaper than vast mainframe computers.
- They work fast and get things done quickly.

### d. Microcomputer

A microcomputer is a small-sized computer made for one person to use at a time. It's also known as a personal computer (PC) or a device that uses a single-chip microprocessor. Laptops and desktops are examples of common microcomputers.



Fig. 1.4 (d) Microcomputer



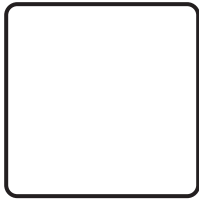
Fig. 1.4 (b) Mainframe Computer



Fig. 1.4 (c) Minicomputer

### Do you know?

that the average smartphone today has more computing power than the computers used for the Apollo 11 moon landing in 1969? It's true! The Apollo Guidance Computer (AGC), which helped navigate the spacecraft and land on the moon, had a processing speed of about 0.043 MHz and 64 KB of memory. In contrast, the latest smartphones boast processing speeds measured in gigahertz (GHz) and storage capacities of hundreds of gigabytes or even terabytes. This means that the device you carry in your pocket or purse is exponentially more powerful than the technology that helped humanity achieve one of its greatest accomplishments—landing on the moon! It's a testament to the incredible advancements in computing technology over the past few decades.

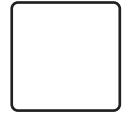


### — Learning Activity

Watch the following video to understand how the different components of a computer process and explain in your own words.

**Various components of a computer explained:**

<https://www.youtube.com/watch?v=ExxFxD4OSZ0>



## Knowledge 1.5

### Input/output Devices

In previous grades, we have acquired foundational knowledge about input and output devices, essential components of computer systems that facilitate interaction between users and computers. Through explorations in earlier coursework, we have developed an understanding of how input devices enable users to input data and commands into a computer system, while output devices present processed information in a usable format. We have learned about a variety of input devices, including keyboards, mice, touchscreens, and microphones, each serving distinct purposes in capturing and transmitting user input. Additionally, we have explored various output devices such as monitors, printers, speakers, and projectors, which translate processed data into human-readable forms or convey information through sound or visual displays.

For a quick review discuss the following input and output devices shown in the figure.



Keyboard



Mouse



Microphone



Joystick



microphone



Light pen



Projector



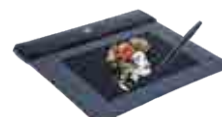
Printer



Track ball



Scanner



Scanner



Speakers



### Skill:1.1 (b)

**Identify Key Components of the Computer System.**

#### Inside the Brain: Unraveling CPU Architecture

**Objective:** The goal of this activity is to help grade 9 students understand the architecture of the CPU (Central Processing Unit), focusing on the functions of the ALU (Arithmetic Logic Unit), CU (Control Unit), and registers within the CPU. Scan the QR code for further details.



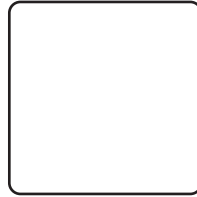


### — Learning Activity

Share questions with students on how computers work and then show the given video in class to explain in their own words.

**How computers work and what they do:**

<https://www.youtube.com/watch?v=92TaQRBwPSs>



## Knowledge 1.6 | CPU (Central Processing Unit)

In the video, we saw all the parts that make up a computer. Now, let's talk about how these parts actually work together. Think of a computer like a big team, where each part has a special job to do. The video showed us who's on the team: the motherboard, CPU, RAM, storage, and other pieces. Now, we're going to learn about what each team member does. Get ready to explore how the computer thinks, remembers things, and talks to us. It's like solving a fun puzzle, and by the end, you'll know a lot about how our tech buddies, the computers, work their magic!

### CPU - Central Processing Unit

The CPU, also known as the "Central Processing Unit" or the "Brain" of the system, is a crucial part of a computer. It's placed in a special socket on the motherboard. While working, the CPU gets hot, so it's connected to a heat sink to stay calm.

### Importance of CPU

The CPU is super essential for a computer to work well. It helps all the devices in the computer run smoothly. The CPU manages instructions it gets from the computer's hardware and software, works on them, and finally shows the result on devices like the monitor or printer. That's why the CPU is an essential part of the computer system.

### Parts of the CPU

Let's talk about the different parts inside the CPU and what they do.

#### 1. Memory Unit

The memory unit is fundamental because it stores instructions and sends them to other parts of the CPU. Computer memory is split into two main types: primary and secondary memory. The performance of the computer depends on the type of memory used, like SRAM or DRAM. After the CPU finishes working on instructions, the memory unit helps store the result and sends it to output devices.

#### 2. Control Unit

The control unit's main job is to control all the operations in the CPU. It moves data and instructions between different parts of the system. The memory unit receives instructions and data from the control unit,

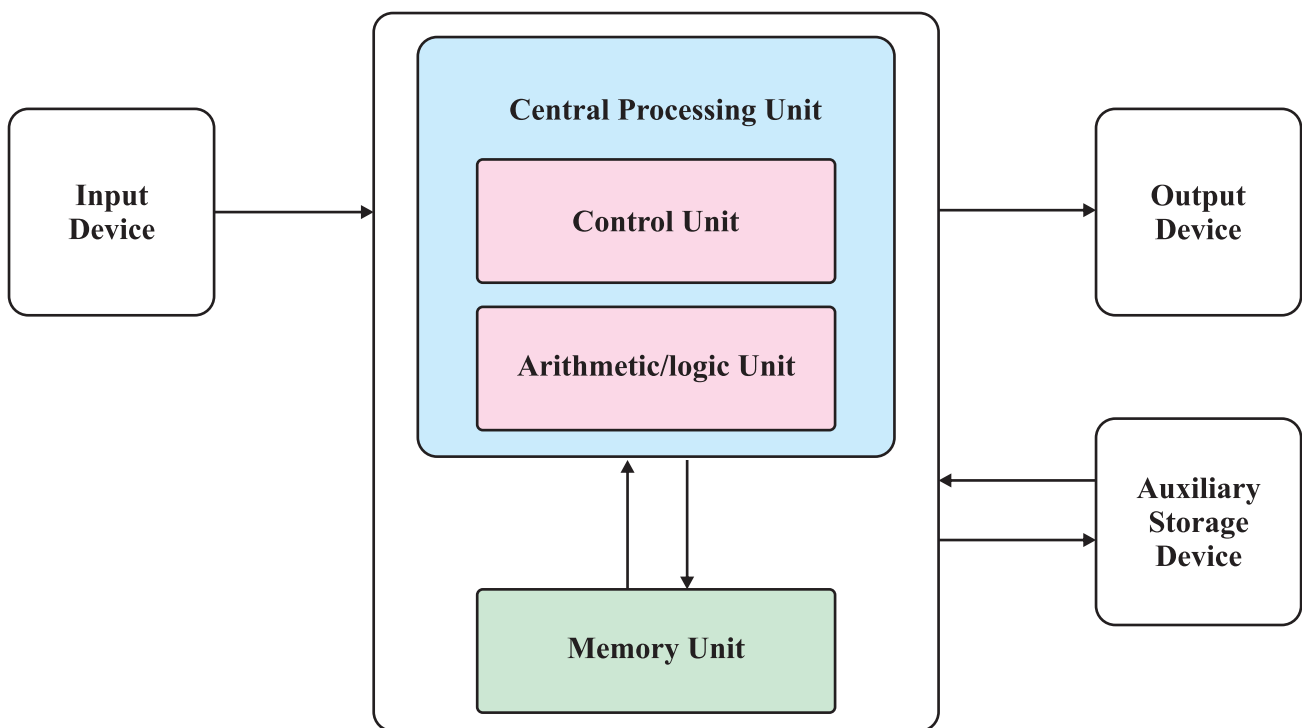
understands them, and then sends the instructions to different parts of the system. It's like a messenger between input and output devices.

### 3. Arithmetic and Logic Unit (ALU)

The ALU has two parts: arithmetic and logic. It's made of digital circuits called registers that help solve math problems and logical operations. The arithmetic unit does things like addition, subtraction, multiplication, and division. The logic unit's main job is to compare, select, match, and merge different data.

### 4. Input/output unit

The input unit includes devices like the keyboard, mouse, and touchpad – these help users put instructions into the computer. Each input device has its hardware controller linked to the CPU, giving instructions on how to use them. The output unit includes devices like the monitor, printer, and speaker – they show the results in text or images after the computer works on the data. The CPU takes 0s and 1s (binary code) and changes them to the correct format needed by the output devices.



#### a. Motherboard

A motherboard is like a computer's main circuit board. It's essential because it connects everything inside a computer. All the parts and things you plug into your computer connect to the motherboard.

You can find motherboards on almost all computers, especially on desktops and laptops. Different parts like chipsets, the CPU (the computer's brain), and memory connect to the motherboard. Also, things like Wi-Fi, Ethernet, and graphics cards with the GPU connect

through it. Company like Acer, ASRock, Asus, Gigabyte Technology, Intel, and Micro-Star International make motherboards. They create these boards that help all the parts in your computer talk to each other.

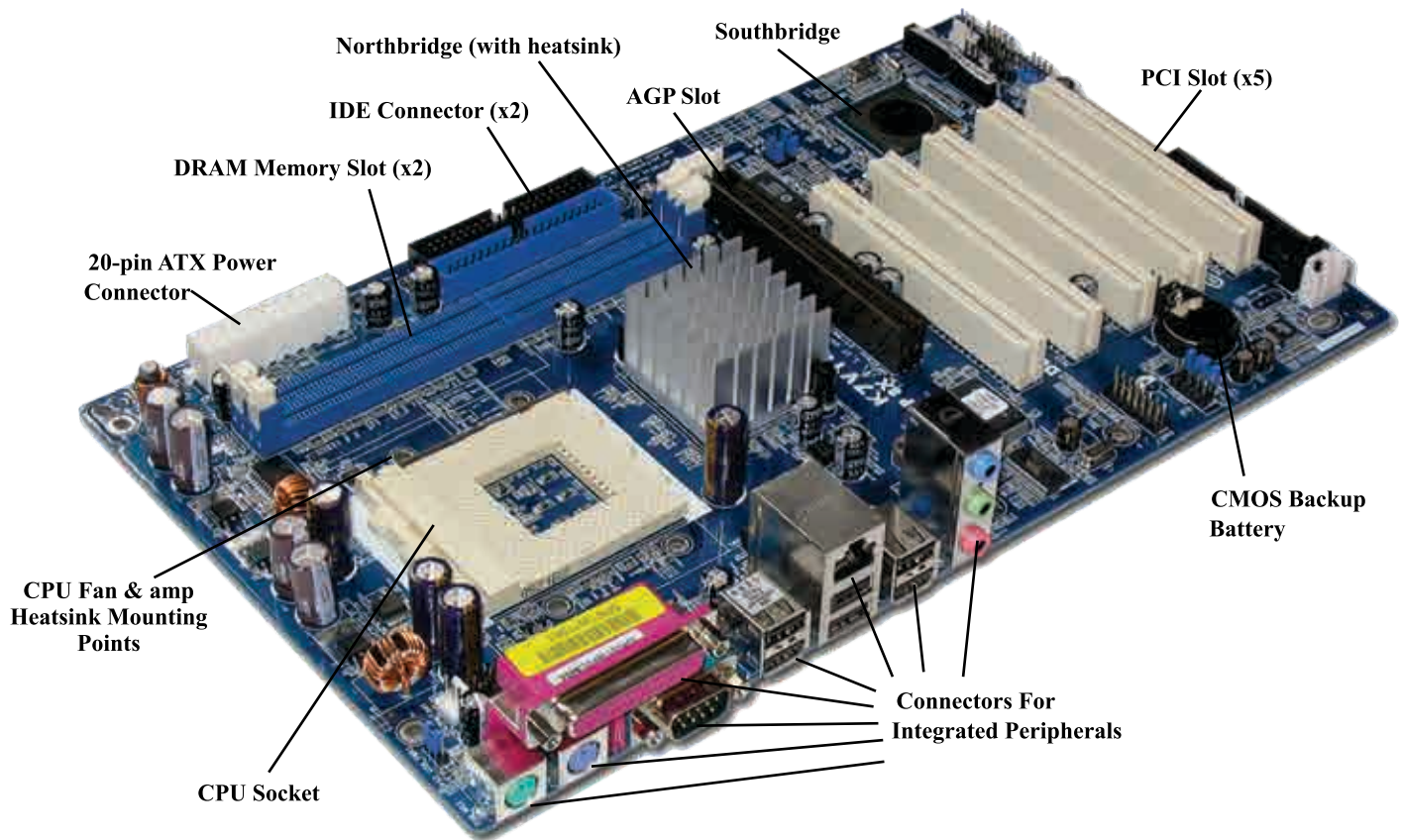


Fig. 1.6 (a) Motherboard

## b. Power Supply Unit

The power supply, also known as PSU (power supply unit) or PS, is a part inside a computer. Its job is to give power to all the other parts of the computer. It changes the regular electric power from your home into a lower power that the computer can use. This power is measured in watts. For example, the picture is of an Antec True 330, which is a power supply that makes 330 watts of power.



Fig. 1.6 (b) Power Supply Unit

## Knowledge 1.7 | Types of Memory

### Primary Storage

#### 1. Memory (RAM - Random Access Memory):

RAM, which stands for "Random Access Memory," is a vital part of a computer. It helps the computer access important data and programs quickly from its cache memory. RAM is also known as "Primary Memory" or "Main Memory," and it's a hardware component found on the computer's motherboard.



Fig. 1.7 (1) RAM

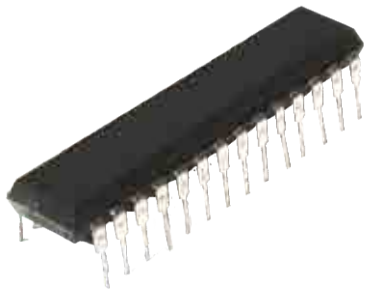


Fig. 1.7 (1a) Static RAM (SRAM)



Fig. 1.7 (1b) Dynamic RAM (DRAM)



Fig. 1.7 (2a) Hard disk



Fig. 1.7 (2b) Permanent storage

RAM is a type of memory that can store data and instructions when the computer is turned on. When you start the computer, RAM collects necessary data and instructions from the hard disk and stores them. The CPU uses this stored data from RAM to do specific tasks.

### Types of RAM (Random Access Memory):

#### a. Static RAM (SRAM)

SRAM, or "Static Random-Access Memory," retains information in a static form until the power supply is turned off. This type of memory is volatile, meaning it loses data when the power is off. SRAM is mainly used to create Cache Memory, and it's faster than dynamic RAM (DRAM). It's more expensive and consumes more power but offers higher speed. SRAM stores data in flip-flops that use 4-6 transistors for each memory cell.

#### b. Dynamic RAM (DRAM)

DRAM, or "Dynamic Random-Access Memory," is another type of semiconductor memory. It's designed to store data or program files required by computer processors to function. DRAM stores data in capacitors. It can store more data than SRAM but needs frequent refreshing of its circuit for charging. As a result, it consumes more power than SRAM.

## 2. Secondary Storage:

Secondary storage, also called external memory, refers to various storage media where a computer can store data and programs. This storage can be fixed or removable. Fixed storage media is internal, like a hard disk.

#### a. Hard Disk:

A hard disk is a magnetic storage medium for computers. It's a flat, round plate made of aluminum or glass coated with magnetic material. Hard drives can store terabytes of information. There are two types: Internal hard disks and External hard disks.

#### b. Permanent Storage

Tertiary storage includes high-capacity data storage that uses removable media like optical tapes or discs. These are especially useful for massive data archives that need human operator access. Examples include magnetic tapes and optical discs like Blu-ray, CDs, and DVDs, tape Drive.

#### c. Offline Storage:

Offline storage needs physical connection to a computer every time



it's used. Examples include portable hard drives, floppy disks, zip diskettes, USB flash drives, and SD cards. These devices must be plugged in or attached to the computer system for use.

## Knowledge 1.8 | Von Neuman Architecture

The Von Neumann architecture is a way of designing computers that was created by a mathematician named John Von Neumann in the 1940s. This architecture divides the computer's memory into two parts: one part stores the instructions (like rules) for the computer to follow, and the other part holds the data that the computer works on.

In this design, both the instructions and data are kept in the same memory unit. This helps the computer to read and process information faster. It's like having a system where the computer can both read instructions and work on data at the same time. This architecture made computers more efficient and faster. It also made them less expensive because the same parts could be used for different tasks. The Von Neumann architecture is used in most modern computers, phones, and tablets.

A "bus" is like an internal highway inside the computer. It's used to send messages and information between the processor (the brain of the computer) and other parts of the computer. This helps everything in the computer work together smoothly.

### The Basic Process of a computer built in Von Neumann Architecture

- The input comes in and is stored in the memory (RAM).
- The input becomes the command and ready to be processed.
- The CU in the CPU fetches the command from the memory and checks which operations to be performed.

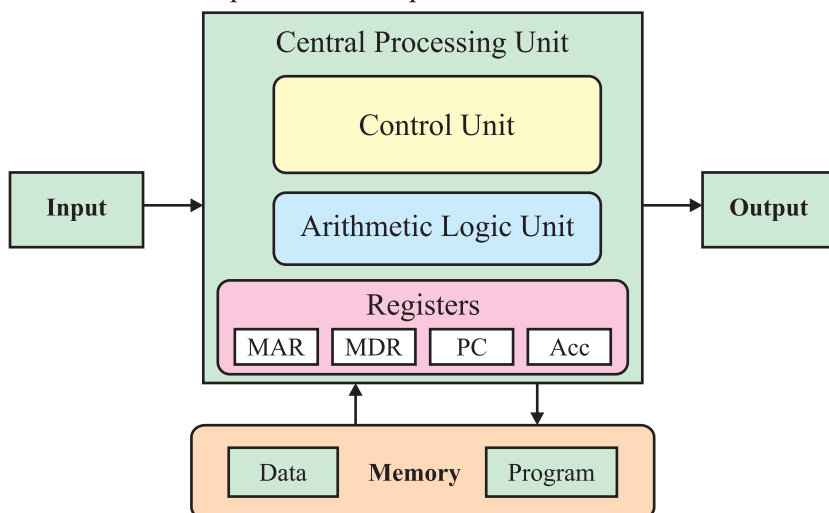


Fig. 1.8 Von Neumann Architecture diagram



Fig. 1.7 (2c) Offline storage



#### Skill:1.1 (c)

#### Identify Key Components of the Computer System.

##### Navigating Primary, Cache, and Secondary Memory

**Objective:** The goal of this activity is to help grade 9 students differentiate between primary memory (RAM), cache memory, and secondary memory (e.g., hard drives, SSDs) through interactive exploration.

Scan the QR code for further details.



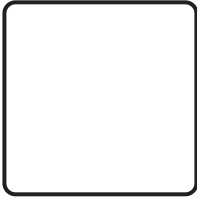
#### Skill:1.1 (d)

#### Identify Key Components of the Computer System.

##### Unveiling the Functions of Operating Systems and Applications

**Objective:** The objective of this activity is to enable grade 9 students to describe the main functions of operating systems and application systems through a collaborative exploration.

Scan the QR code for further details.



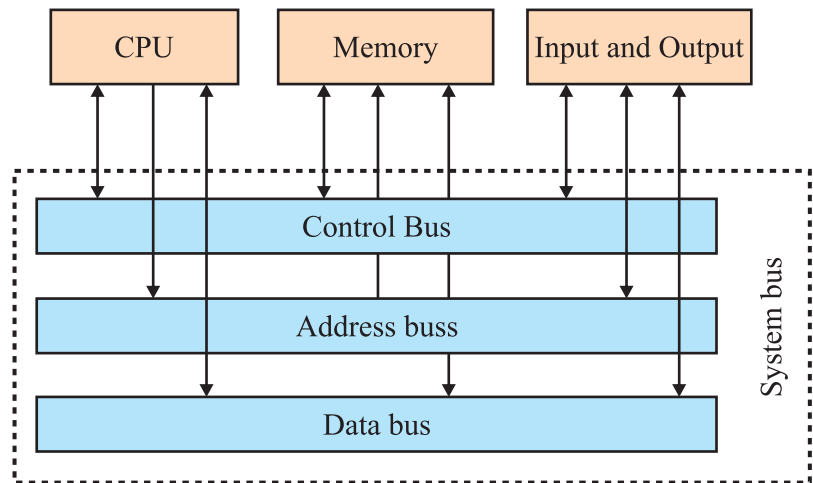
- The fetched command is processed by CU and ALU in the CPU.
- After processing completes, the control unit stores the result in memory again.
- Finally, the output is displayed via an output device.

## Knowledge 1.9 | Bus Interconnection

### System Bus

In computers, think of the architecture as a system of roads that helps different parts of the computer communicate with each other or even with other computers. It's like a network of roads connecting various places in a city.

Now, this system, known as the bus, has three main types:



#### Skill:1.2

**Interaction of various components for data transmission**

#### Interactive Exploration of Component Communication

**Objective:** The goal of this activity is to engage grade 9 students in a hands-on experience that simulates the transmission of data and instructions between various computer components. Scan the QR code for further details.

#### a. Control Bus

This is like the traffic police of the computer. It carries signals that control and coordinate activities across the computer. It's generated by the control unit inside the computer's brain (CPU). These signals help the CPU talk to different parts and devices by identifying them.

#### b. Address Bus

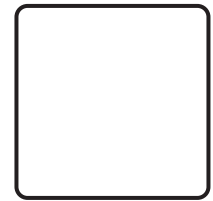
Imagine this as the system of street addresses in the city. It carries memory addresses within the computer, allowing the CPU to find specific locations in the computer's memory. The address bus connects the CPU to other parts and carries signals about where in the computer's memory it needs to read from or write to. For example, if it can carry 8 bits at a time, the CPU could manage up to 256 memory locations.

#### c. Data Bus

This is like a highway that moves data between different places in the computer. When a device needs to send important information, it places that data onto these lines, and the CPU uses this "highway" to

move the data around. The data bus connects the CPU, memory, and other parts of the computer. It allows data to flow back and forth between the processor and memory, as well as peripheral devices like printers or hard drives.

So, these three types of buses—control, address, and data buses—help in the smooth and fast communication between different parts of the computer, ensuring everything works together properly.



## Knowledge 1.10 | Data Representation

Data representation in computers involves several common numerical systems, including binary, decimal, and hexadecimal, as well as units of measurement like bits and bytes. Here's an overview of each of these elements:

### a. Bit (Binary Digit)

A bit is like a tiny switch in computers. It's the smallest unit of information and can only have two values: 0 or 1. Everything in computers is made from bits.

### b. Byte

Eight bits together make up a byte. It's like a small container that can hold information. Bytes are often used to represent things like letters, numbers, or symbols.

### c. Binary (Base-2) Representation

Computers use the binary system. It's all about using just two numbers, 0 and 1. Each 0 or 1 in binary represents a different power of 2. For example, 1101 in binary means  $(1 * 2^3) + (1 * 2^2) + (0 * 2^1) + (1 * 2^0)$ , which is 13 in regular numbers.

### d. Decimal (Base-10) Representation

This is the kind of numbers we use daily with digits from 0 to 9. Each digit in a decimal number represents a power of 10. For instance, 427 means  $(4 * 10^2) + (2 * 10^1) + (7 * 10^0)$ , which is simply 427.

### e. Hexadecimal (Base-16) Representation

Hexadecimal is a mix of numbers and letters (from 0 to 9 and A to F). It's useful in computers because it represents big binary numbers in a shorter way. Each digit in hexadecimal stands for 4 bits (half a byte). For example, 1A3 in hexadecimal means  $(1 * 16^2) + (10 * 16^1) + (3 * 16^0)$ , which equals 419 in regular numbers.

## Knowledge 1.11 | Hardware Engineering vs Software Engineering

Both hardware and software engineers play different roles in making computers and programs work well.



### Skill:1.3

#### Hierarchy of Memory and Storage Devices

##### Building the Hierarchy of Memory and Storage

**Objective:** The goal of this activity is to visually illustrate the hierarchy of memory and storage devices for grade 9 students, emphasizing their respective roles and characteristics.

Scan the QR code for further details.



### Skill:1.4

#### Necessary and Auxiliary Components of a Computer System

##### Identifying Necessary and Auxiliary Components

**Objective:** The goal of this activity is to engage grade 9 students in a hands-on exploration to differentiate between necessary and auxiliary components of a computer system.

Scan the QR code for further details.

### a. Hardware Engineers

Hardware engineers are professionals who specialize in designing, building, and maintaining the physical components of electronic devices. They use knowledge of mathematics, physics, and computer science to create efficient and reliable systems. Hardware engineers work on tasks such as making models and samples of new systems, assembling and testing computer parts like chips and boards, and troubleshooting problems with hardware components.

Software engineers, on the other hand, are professionals who focus on designing, creating, and testing computer programs and applications. They primarily work with coding languages and problem-solving skills to develop software that performs specific tasks. Software engineers are involved in activities such as checking the functionality of programs, updating existing software, making programs user-friendly, and collaborating with other software developers to bring new programs to market. They also play a role in planning and improving software based on user feedback and data analysis.

### b. Software Engineers

Software engineers, on the other hand, are professionals who focus on designing, creating, and testing computer programs and applications. They primarily work with coding languages and problem-solving skills to develop software that performs specific tasks. Software engineers are involved in activities such as checking the functionality of programs, updating existing software, making programs user-friendly, and collaborating with other software developers to bring new programs to market. They also play a role in planning and improving software based on user feedback and data analysis.

Hardware Engineering	Software Engineering
Involves designing and building physical components of electronic devices	Involves designing, creating, and testing computer programs and applications
Uses math, physics, and computer science	Uses coding languages and problem-solving skills
Crafts tangible parts of technology we use daily	Crafts digital instructions that make computers and devices work as intended
Makes models and samples of new systems	Checks how well a program works



Helps fix and upgrade computer systems	Updates software that's already there
Puts together and checks computer parts like chips and boards	Looks into making programs easier for people to use
Explains new models to bosses and others	Helps put new programs into use
Does tests in labs and studies the results	Teaches new users how to use the software
Listens to what users need and suggests changes	Works together with other software creators
Fixes problems with computer parts	Draws plans for software and gives them to developers
Keeps an eye on making computer parts	Gathers and analyzes data to improve software and plan for the future them to developers

## Knowledge 1.12 | Natural and Artificial Systems

Natural systems, inherent to the natural world, arise spontaneously through natural processes and encompass interconnected components such as ecosystems and biological organisms, exhibiting adaptability and complexity. In contrast, artificial systems, human-made and designed with specific purposes, comprise man-made components like machines and computer programs, typically exhibiting predictability and serving defined human needs. While natural systems evolve through processes like natural selection, artificial systems are designed to fulfill specific objectives, reflecting the diverse interplay between nature's spontaneity and human intentionality in shaping the world around us. Artificial systems can be adaptable to some extent through programming or design modifications. However, they usually require human intervention or updates to adapt to significant changes and new situations.



### Example

Natural systems: Examples include ecosystems, biological organisms, weather patterns, celestial bodies, etc.  
Artificial systems: Examples encompass technology-based systems like computers, artificial intelligence, engineered structures (such as buildings, bridges), and social systems (like governments, economies).

### Difference between Natural and Artificial systems

Storage Type	Volatility	Speed	Storage Capacity	Cost
<b>Registers</b>	Non-volatile	Extremely fast	Very limited, typically a few bytes	Very expensive
<b>Cache Memory (L1, L2, L3)</b>	Volatile	Extremely fast, slower than registers	L1 (smallest, fastest), L2, L3 (increasing)	Relatively expensive compared to RAM
<b>Main Memory (RAM)</b>	Volatile	Slower than cache memory	Typically, gigabytes to several terabytes	Less expensive per unit compared to cache
<b>Secondary Storage (HDDs, SSDs)</b>	Non-volatile	Slower than RAM	HDDs (larger capacity, slower), SSDs (faster, smaller)	Less expensive per unit compared to RAM
<b>Tertiary Storage (Optical Discs, Tape Drives)</b>	Non-volatile	Slower than secondary storage	Very high storage capacity	Relatively cost-effective for archival
<b>Cloud Storage</b>	Non-volatile	Variable depending on internet connection	Virtually unlimited, depends on cloud provider	Variable, depends on data and plan

**A Multiple Choice Questions**

1. Which generation of computers used vacuum tubes as the main electronic component?
  - a. First Generation
  - b. Second Generation
  - c. Third Generation
  - d. Fourth Generation
2. What is the primary function of a CPU (Central Processing Unit) in a computer?
  - a. Storing data
  - b. Processing instructions and calculations
  - c. Providing power to the computer
  - d. Controlling input and output devices
3. What type of memory is volatile, meaning data is lost when the computer is powered off?
  - a. Cache Memory
  - b. Main Memory (RAM)
  - c. Secondary Storage (HDD)
  - d. Tertiary Storage (Tape Drives)
4. Which component is considered the "brain" of a computer?
  - a. Motherboard
  - b. CPU (Central Processing Unit)
  - c. Hard Disk Drive (HDD)
  - d. Monitor
5. What type of computer is known for processing data at extremely high speeds and is used for scientific and engineering applications?
  - a. Mainframe Computer
  - b. Supercomputer
  - c. Mini Computer
  - d. Personal Computer (PC)
6. Which generation of computers introduced integrated circuits (ICs) and high-level programming languages?
  - a. First Generation
  - b. Second Generation
  - c. Third Generation
  - d. Fourth Generation
7. What is the primary function of the Control Unit (CU) in a CPU?
  - a. Execution of mathematical calculations
  - b. Interpretation of instructions and coordination of operations
  - c. Storage of temporary data
  - d. Generation of electrical power
8. Which type of storage device is known for its non-volatile and long-term data retention characteristics?
  - a. RAM (Random Access Memory)
  - b. HDD (Hard Disk Drive)
  - c. SSD (Solid-State Drive)
  - d. Tertiary Storage (Tape Drives)
9. What is the purpose of the Von Neumann architecture in computer design?
  - a. To increase the speed of data processing
  - b. To allow for parallel processing
  - c. To separate program code and data for efficient execution
  - d. To reduce the need for peripheral components
10. Which type of computer is designed for technical or scientific applications and typically includes a high-speed graphic adapter?
  - a. Mainframe Computer
  - b. Mini Computer
  - c. Workstation Computer
  - d. Supercomputer

## **B** Short Questions

1. Explain the concept of the Von Neumann architecture and its significance in computer design.
2. Describe the role of cache memory (L1, L2, L3) in a computer system and how it improves performance.
3. Differentiate between primary (necessary) components and auxiliary (peripheral) components of a computer system. Provide examples of each.
4. Briefly explain the significance of data representation in computing, including binary, decimal, and hexadecimal systems.
5. What are the key functions of the Control Unit (CU) in a CPU, and how does it contribute to the execution of instructions?
6. Discuss the role of the ALU (Arithmetic Logic Unit) in a CPU and its significance in data processing.
7. Explain the concept of data representation in binary and how binary digits (bits) are used to represent information in computer systems.
8. Compare and contrast volatile and non-volatile memory in a computer system. Provide examples of each.
9. What are the key responsibilities of the motherboard in a computer system, and how does it facilitate communication between various components?
10. Describe the functions and significance of power supply units (PSUs) in computer systems, including their role in converting electrical power.

## **C** Long questions.

1. Discuss the evolution of computer generations from the first generation to the fifth generation, highlighting key developments and technological advancements in each generation.
2. Compare and contrast hardware engineering and software engineering, including their roles, responsibilities, and typical tasks performed by professionals in each field.
3. Explain the various types of memory in a computer system, including registers, cache memory, main memory (RAM), and secondary storage. Discuss their characteristics, functions, and roles in data processing.
4. Describe the essential components of a computer system, such as the CPU, motherboard, storage devices, and input/output devices. Explain the function and significance of each component in the overall operation of a computer.
5. Explore the concept of data storage in a computer system, including primary storage (RAM), secondary storage (HDDs/SSDs), tertiary storage (optical discs and tape drives), and cloud storage. Discuss their differences, advantages, and use cases.

**Hands on lab:** Install, configure, and maintain the major hardware components.

**Hands on lab:** Analyze problems and resolve system faults using basic troubleshooting tools.

**In-class activity:** Oral presentations that involve demonstration of skills such as


1. Independent research using the Internet
2. Research on product specifications
3. Downloading drivers or other information from vendor sites
4. Gathering information on new hardware devices
5. New software (third party) diagnosis tools



## Standard




Students will understand and explain the various components of a computer system and the different levels of interactions between these. They will be able to define & explain digital logic, the different stages of the software life cycle and the concepts of scalability, reliability and security for computer systems.

### Students' Learning Outcomes-1

 Students will be able to identify and explain system software, application software, low-level and high-level programming languages, and their uses


#### Knowledge

##### Student will understand that

-  The main functions of systems software with some examples
-  The main functions of operating systems and application software
-  Outline the uses of various application software.

#### Skills

##### Students will be able to:

-  Select an appropriate medium to create artifacts (Planning the document / information flow, editing and alignment of page, paragraphs, text, tables, and graphics) to communicate ideas in various digital tools such as:
  - Image processing tools (like Photoshop, Canva.com, GIMP etc)
  - Word processors (like MS Word, Google Docs etc)
  - Presentations (like MS PowerPoint or Google Slides etc)
  - Spreadsheets (like MS Excel or Google Sheets)

System software is a special kind of computer program that does essential jobs to make a computer work. It helps control and watch over everything that happens on a computer and makes it easier for people to use. There are different types of system software:

General-purpose software is versatile and applicable to a wide range of tasks, serving a broad audience with diverse needs. Examples include operating systems, productivity suites, web browsers, multimedia players, and graphic design software.

Special-purpose software is tailored to specific industries, professions, or applications, meeting the unique requirements of particular user groups or niche markets. Examples include accounting software, CAD/CAM software, medical software, educational software, and scientific software.

**a. Operating System (OS):**

An operating system is like a computer's boss. It tells the computer what to do and ensures everything runs smoothly. It also helps you use programs like games or word processors and manages how files are stored and organized on the computer. The operating system is essential for making your computer work well and do what you want.

**b. Device Drivers:**

These are like translators between the computer and hardware devices, like printers or graphics cards. They help the computer understand what to do with these devices.

**c. Utilities:**

Utilities in system software are like helpful tools for your computer. They do tasks like cleaning up files, organizing data, and keeping your system safe from viruses. They're like little helpers that make sure your computer runs smoothly and stays healthy.

**d. Language Translators:**

These tools change the code that people write into a language that the computer can understand and follow. There are three types:

- Compilers, which turn whole programs into computer language.
- Interpreters, which read and run code step by step.
- Assemblers change a special kind of code into computer language.

**e. Firmware:**

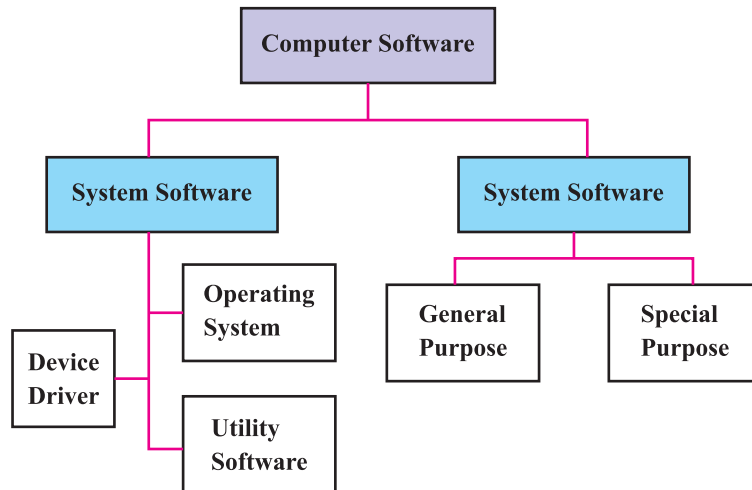
This is a special kind of system software that lives inside hardware parts. It helps control these parts, making sure they work right. For example, it's like the boss of the computer's brain.

#### f. Virtual Machine Managers and Emulators:

These are tools that let you run different operating systems on one computer. It's like having several computers inside one!

#### g. System Libraries:

These are ready-made pieces of code that make it easier for people to create new programs. They're like building blocks that programmers can use to make their software.



### Knowledge 1.14 | Functions of operating system software

An **operating system** is like a computer's manager. It handles all the essential tasks that make the computer work, and it helps programs and people communicate with the computer's hardware. The main job of an operating system is to share resources like memory, devices, and processors among different programs. It also comes with tools to manage these resources, such as a traffic controller, a scheduler, memory management, programs for input/output, and a file system. Basically, the operating system provides a safe environment for other programs to do their work.

#### Need of an Operating Systems:

Think of the operating system as the middle person between the computer and the user. It plays several roles:

- It controls everything on the computer.
- It offers essential services to programs.
- It makes sure programs run smoothly together.
- It gives resources to programs.
- It gives users a way to interact with the computer.
- It can do different things at different times.
- It watches over programs to prevent mistakes.



## Functions of an Operating Systems:

### a. Memory Management:

The operating system takes care of the computer's memory. Memory is like the computer's fast storage, and it needs to be managed well. The operating system keeps track of which part of memory belongs to which program. It decides when one program can use memory and when it can't. It also loads programs into memory when they need to run and clears memory when they're done.

### b. Processor Management:

In a busy computer, many programs want to use the processor (the computer's brain) at the same time. The operating system decides which program gets to use the processor and for how long. This is called process scheduling.

### c. Device Management:

The computer has various devices like keyboards, printers, and screens. The operating system looks after these devices and makes sure they work correctly. It also decides which program can use a device and when.

### d. File Management:

Your computer stores files, like documents and pictures. The operating system keeps track of where these files are, who can use them, and how they are organized. It makes sure files are safe and protected.

### e. User Interface:

The operating system provides a way for you to talk to the computer. This can be through commands or a graphical interface that you can click and touch. It's how you use software and control the hardware.

### f. Booting:

When you turn on your computer, it goes through a startup process called booting. The operating system helps the computer get ready for you to use.

### g. Security:

The operating system keeps your computer safe. It uses passwords and other tricks to protect your data and make sure only authorized users can access programs and files.



## Knowledge 1.15

### Application software uses and their function

#### Application Software:

Application software is like the helpful tools on your computer or phone that do specific jobs for you. When you use software directly to do something, like writing a document or playing a game, that's application software. It's all about helping you with specific tasks. Examples of application software include Microsoft Word and Excel for work and web browsers like Firefox and Google Chrome for surfing the Internet. You also have mobile apps. Apps for checking the weather or finding transportation are part of this group, too, as are apps for connecting with businesses. Other examples are GPS apps, graphic design programs, multimedia software, and programs for making presentations.

**Installed applications** are programs that you can use on your computer even when you're not connected to the internet. These are like software that lives inside your computer and doesn't need the internet to work. Installed applications are stored on your computer's hard drive.

Let's talk about the good and not-so-good things about using installed software:

**Spreadsheets** are software applications used to organize, analyze, and manipulate data in a tabular format. Here's what spreadsheets can do:

#### a. Data Organization:

Spreadsheets allow users to input data into rows and columns, making it easy to organize information in a structured manner.

#### b. Calculations:

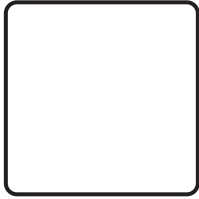
Spreadsheets can perform various calculations using formulas and functions. Users can apply mathematical operations, such as addition, subtraction, multiplication, and division, to manipulate data within the spreadsheet.

#### c. Data Analysis:

Spreadsheets can analyze data by generating summaries, calculating averages, identifying trends, and performing statistical analyses. Users can create charts and graphs to visualize data and gain insights.

#### d. Automated Tasks:

Spreadsheets can automate repetitive tasks by using macros and scripts. This can save time and reduce errors by automating routine processes, such as data entry or formatting.



### — Learning Activity

Create a chart from a table in a spreadsheet using an application like MS Excel.



#### e. Data Visualization:

Spreadsheets allow users to create visual representations of data, such as charts, graphs, and pivot tables. These visualizations make it easier to interpret complex data and communicate findings effectively.

#### f. Data Validation:

Spreadsheets can enforce data validation rules to ensure that data entered into cells meets specific criteria. This helps maintain data integrity and accuracy within the spreadsheet.

#### g. Collaboration:

Spreadsheets support collaboration by allowing multiple users to work on the same document simultaneously. Users can track changes, leave comments, and share spreadsheets with others in real-time.

#### h. Data Filtering and Sorting:

Spreadsheets provide tools for filtering and sorting data based on specific criteria. Users can quickly find and analyze subsets of data by applying filters and sorting options.

Overall, spreadsheets are versatile tools that offer a wide range of capabilities for organizing, analyzing, and manipulating data effectively. They are commonly used in various industries and professions for tasks such as financial analysis, budgeting, project management, inventory tracking, and data reporting.

### Word processing software

Word processing software is a type of program designed to create, edit, format, and print text documents. It provides users with a range of tools and features to manipulate text, graphics, and other elements within a document. Here are some key points about word processing software:

- **Document Creation:**

Word processors allow users to create new documents from scratch or start with pre-existing templates. Users can type text, insert images, tables, and other multimedia elements to create visually appealing documents.

- **Text Editing and Formatting:**

Word processors offer a variety of text editing and formatting options, including font styles, sizes, colors, alignment, indentation, and bullet points. Users can easily modify the appearance of text to suit their preferences or the requirements of the document.

- **Document Organization:**

Users can organize content within a document using features like headers, footers, page numbering, sections, and bookmarks. These



tools help users structure their documents logically and make them easier to navigate.

- **Spell Checking and Grammar Checking:**

Word processors often include built-in spell-checking and grammar-checking tools to help users identify and correct errors in their documents. These tools can improve the overall quality and professionalism of the text.

- **Collaboration and Sharing:**

Many word processing software programs offer collaboration features that allow multiple users to work on the same document simultaneously. Users can also share documents electronically via email, cloud storage, or file-sharing platforms.

- **Document Review and Revision:**

Word processors facilitate document review and revision by providing tools for tracking changes, adding comments, and comparing different versions of a document. These features are particularly useful for collaborative projects or document approval processes.

- **Integration with Other Software:**

Word processing software often integrates with other software programs, such as spreadsheet software, presentation software, and email clients. This integration enables users to easily incorporate data, charts, and other content from various sources into their documents.

Examples of word processing software include Microsoft Word, Google Docs, Apple Pages, and LibreOffice Writer. These programs offer similar functionality but may vary in terms of user interface, compatibility with different file formats, and collaboration capabilities. Overall, word processing software provides users with a versatile tool for creating, editing, and sharing text-based documents for a variety of purposes.

## Presentation software

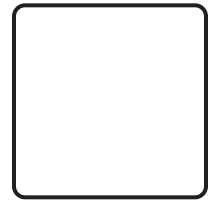
Presentation software is designed to create slideshows or multimedia presentations that visually communicate information to an audience. Here's what presentation software can do:

- **Slide Creation:**

Presentation software allows users to create slides containing text, images, videos, charts, graphs, and other multimedia elements. Users can design slides to convey information in a visually appealing and engaging manner.

- **Slide Formatting:**

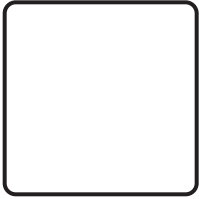
Presentation software offers tools for formatting slides, including



### — Learning Activity

Creating a written report on a research topic, the submission should include cover page, title, table of contents, headers, subheadings, paragraphs in a word processor like MS Word.





options for changing fonts, colors, backgrounds, and layouts. Users can customize the appearance of slides to match the theme or style of their presentation.

- **Slide Transitions:**

Presentation software allows users to add transitions between slides, such as fades, wipes, or animations. These transitions enhance the visual appeal of the presentation and help maintain the audience's attention.

- **Slide Animation:**

Presentation software enables users to animate individual elements within slides, such as text, images, or objects. Animations can be used to highlight key points, demonstrate processes, or add visual interest to the presentation.

- **Presenter Notes:**

Presentation software often includes features for adding presenter notes to slides. These notes can help speakers remember important points, provide additional context or information, and guide the flow of the presentation.

- **Slide Show Playback:**

Presentation software allows users to play their slideshows in full-screen mode, either manually or automatically. Users can control the pace of the presentation, navigate between slides, and interact with multimedia elements during playback.

- **Audience Interaction:**

Some presentation software offers features for audience interaction, such as polling, quizzes, or live chat. These features engage the audience and encourage participation during the presentation.

- **Export and Sharing:**

Presentation software enables users to export their slideshows in various formats, such as PowerPoint files, PDFs, or videos. Users can share their presentations electronically via email, cloud storage, or online platforms.

Overall, presentation software is a versatile tool for creating visually compelling presentations that effectively communicate information to an audience. It is commonly used in business meetings, academic lectures, training sessions, sales pitches, and conferences to deliver engaging and impactful presentations.

## Image and diagram software

Image and diagram software are specialized tools used to create, edit, and manipulate images, diagrams, and visual representations of information. Here's what image and diagram software can do:



### — Learning Activity

Create a presentation on a topic using an application like MS PowerPoint.





**Image Editing:** Image software allows users to edit and enhance digital images by adjusting parameters such as brightness, contrast, saturation, and sharpness. Users can also crop, resize, and rotate images, as well as remove imperfections or unwanted elements.

- **Drawing Tools:**

Image software provides drawing tools such as brushes, pencils, shapes, and text, allowing users to create custom graphics and illustrations from scratch. Users can draw freehand or use predefined shapes and templates to design graphics.

- **Layers and Transparency:**

Image software supports layers, allowing users to work on different elements of an image separately and combine them to create complex compositions. Users can adjust the opacity and transparency of layers to create interesting visual effects.

- **Filters and Effects:**

Image software offers a variety of filters and effects that can be applied to images to alter their appearance. Users can add artistic effects, blur backgrounds, apply color adjustments, and simulate various photographic techniques.

- **Diagram Creation:**

Diagram software enables users to create diagrams, flowcharts, mind maps, organizational charts, and other visual representations of information. Users can choose from a library of shapes, symbols, and connectors to create professional-looking diagrams quickly.

- **Templates and Themes:**

Diagram software provides templates and themes that users can use as a starting point for their diagrams. Templates offer pre-designed layouts and styles, making it easy to create diagrams for specific purposes or industries.

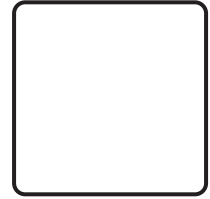
- **Collaboration and Sharing:**

Image and diagram software often support collaboration features that allow multiple users to work on the same project simultaneously. Users can share images and diagrams with others via email, cloud storage, or collaboration platforms.

- **Export and Integration:**

Image and diagram software enable users to export their creations in various formats, such as JPEG, PNG, PDF, or SVG. Users can also integrate images and diagrams into other documents, presentations, or websites seamlessly.

Popular examples of image and diagram software include Adobe Photoshop, Adobe Illustrator, CorelDRAW, Microsoft Visio, Lucidchart, and draw.io. These tools are widely used in graphic



#### — Learning Activity

Creating a diagram in an image processing tool like MS Paint.



design, web development, engineering, education, and many other fields to create compelling visual content and communicate ideas effectively.

### Web-based applications

Web-based applications are like software that is kept on big computers on the internet or inside a private network. To use them, you have to be connected to the internet because these programs are not on your computer, but on these big computers far away. When you're deciding whether to use installed or web-based programs, think about what you need and your situation. For instance, if you have internet access, web-based applications will work for you.

#### Web Browsers:

A web browser is a program that helps you look at things on the internet, like web pages. There are different ones you can use, like Microsoft Edge, Google Chrome, and Firefox.

some examples of document management software are Microsoft SharePoint, Google Workspace, Dropbox Business.

**Google Chrome:** This is a free web browser developed by Google.

#### Features of Google Chrome:

- 1. Speed and Performance:** Chrome is known for its fast browsing speed and performance. It uses a powerful JavaScript engine that loads web pages quickly.
- 2. Simplicity and User-Friendly Interface:** The browser has a clean and intuitive interface that allows easy navigation and use, even for beginners.
- 3. Customization:** Users can personalize their browsing experience with various themes, extensions, and settings available in the Chrome Web Store.

**Firefox:** This is a free open-source web browser developed by Mozilla.

#### Features of Firefox:

- 1. Privacy and Security:** Firefox emphasizes user privacy and security. It offers enhanced tracking protection, blocking third-party trackers by default and providing options for stricter privacy settings. It also has a Private Browsing mode (called "Firefox Private Browsing" or "Private Window") that doesn't save browsing history, cookies, or site data after the session ends.
- 2. Customization:** Firefox allows extensive customization through themes, extensions, and add-ons available in the Firefox Add-ons marketplace. Users can personalize their browsing experience according to their preferences.
- 3. Enhanced Tracking Protection:** By default, Firefox blocks

known trackers and offers options for users to control the level of tracking protection they want. This helps in preventing advertisers and websites from tracking users across the web.

## Knowledge 1.16 | Programming Languages and Their Types

This article is here to help you understand the different kinds of computer programming languages. It will provide definitions and examples to make things clear.

There are mainly two types of computer programming languages:

1. Low-level language
2. High-level language

### Low-Level Languages

Low-level programming languages are very close to the computer's language, which is all about 0s and 1s. In these languages, program instructions are written in binary form.

Types of low-level languages:

There are two types of low-level computer languages:

- Machine language
- Assembly language

#### Machine Language

Machine language, or machine code, is all about those binary instructions that the computer's central brain (CPU) can understand without any translation. It's like the computer's native language, and it's super basic, using only 0s and 1s. Writing in machine language is challenging because of this, and it could be more practical.

Address		Machine Language				Assembly Language	
0000	0000	0000	0000	0000	0000	TOTAL	• BLOCK 1
0000	0001	0000	0000	0000	0010	ABC	• WORD 2
0000	0010	0000	0000	0000	0011	XYZ	• WORD 3
0000	0011	0001	1101	0000	0001		LOAD REGD, ABC
0000	00100	0001	1110	0000	0010		LOAD REGE, XYZ
0000	00101	0101	1111	1101	1110		ADD REGE, REGD, REGE
0000	00110	0010	1111	0000	0000		STORE REGF, REGD, REGE
0000	00111	1111	0000	0000	0000		STORE REGF, TOTAL

#### Assembly Language

Assembly language is another low-level language, and it's like a step up from machine language. It's also known as a second-generation language. In assembly language, you write instructions using symbols instead of binary codes. These symbols are like short names for operations, making them easier to understand.

Here's the information presented in a table format:

Advantages of Assembly Language	Disadvantages of Assembly Language
Easier to understand and use compared to machine language	Tied to a specific computer, limiting portability
Finding and fixing errors is more accessible	Requires extensive knowledge of hardware
Ease of making changes to instructions	Writing programs is challenging and time-consuming

This table provides a concise overview of the advantages and disadvantages of assembly

### High-Level Languages

High-level languages are the ones that are more like human languages, such as English. They're way easier to learn and use than low-level languages.

Some examples of high-level languages are Python, C, C++, and Java. These languages use words like "print" and "input" that look more like English. But each high-level language has its own rules and grammar for writing instructions, called syntax. High-level programs need to be translated into machine code to run, and each language has its translator.

Advantages of High-Level Languages	Disadvantages of High-Level Languages
Easier to learn due to English-like instructions	Programs might not be as efficient as those in low-level languages
Programs are easy for other programmers to understand	Limitations in controlling CPU, memory, and registers
Quick to write new programs	
Simple error finding and fixing	
Built-in functions save time	
Portability across different types of computers	

- **Procedural Languages**

In procedural languages, programs are built using procedures, which are like sets of instructions with their names. These instructions need to be followed in a specific order to solve a problem.

Examples of procedural languages include FORTRAN, BASIC, Pascal, ADA, and C.

- **Non-Procedural Languages**

Non-procedural languages, also called fourth-generation languages, don't care about the order of instructions; they focus on what needs to be done rather than how to do it. These languages are user-friendly and make program development easier.



A famous non-procedural language is SQL (Structured Query Language), which is used for working with databases.

- **Object-Oriented Programming Languages**

Object-oriented programming (OOP) languages use objects, which are like building blocks of a program. Each object has data and modules (called methods) to work with that data. OOP is a modern approach to programming and makes it easy to reuse code in different software.

C++ and Java are popular object-oriented programming languages

## Exercise



### A Multiple Choice Questions

- 1) What is machine language also known as?
  - a) First-generation language
  - b) Third-generation language
  - c) Second-generation language
  - d) Fourth-generation language
- 2) What does assembly language use instead of binary codes for instructions?
  - a) Octal numbers
  - b) Symbols
  - c) Hexadecimal numbers
  - d) Characters
- 3) What are the advantages of high-level languages over low-level languages?
  - a) More efficient use of memory
  - b) Tied to specific computers
  - c) Easier to learn and use
  - d) Limited syntax and grammar
- 4) Which category of high-level languages focuses on procedures to solve a problem?
  - a) Procedural languages
  - b) Non-procedural languages
  - c) Object-oriented programming languages
  - d) Structured Query Language (SQL)
- 5) What makes Object-Oriented Programming (OOP) languages different?
  - a) They don't use objects in programming
  - b) Objects have data and methods to work with that data
  - c) They focus solely on procedural approaches
  - d) OOP languages don't allow code reusability.
- 6) What is the main function of an operating system?
  - a) To design web pages
  - b) To create documents and spreadsheets
  - c) To develop mobile applications
  - d) To manage hardware resources and enable communication between software and hardware
- 7) Which system software manages the computer's memory, ensuring efficient usage?
  - a) Device Drivers
  - b) Utilities
  - c) Language Translators
  - d) Operating System
- 8) What does the operating system do in terms of device management?
  - a) Manages printers only
  - b) Takes care of keyboards and mice
  - c) Ensures devices work correctly and assigns them to programs
  - d) Controls only the display screens
- 9) What role does the operating system play in terms of file management?
  - a) Keeps track of where files are stored
  - b) Only organizes files alphabetically
  - c) Protects files from viruses
  - d) Manages printer files
- 10) What distinguishes application software from system software?
  - a) Application software manages hardware resources.
  - b) Application software controls the operating system.

- c) Application software performs specific tasks for the user.
- d) Application software maintains memory usage.
- 11) Which of the following is an example of application software used for creating documents?
  - a) Microsoft Excel
  - b) Adobe Photoshop
  - c) Microsoft Word
  - d) Google Chrome
- 12) Which type of language is close to the computer's language and written in binary form?
  - a) High-level language
  - b) Assembly language
  - c) Procedural language
  - d) Machine language
- 13) What are the advantages of high-level languages over low-level languages?
  - a) They offer better control over hardware.
  - b) They are less efficient in using memory.
  - c) They are tied to specific computers.
  - d) They are easier to learn and use.
- 14) Which language among the following is considered a high-level language?
  - a) Assembly language
  - b) COBOL
  - c) Machine language
  - d) SQL

## **B** Short Questions

- 1) What is the central role of an operating system in a computer?
- 2) Name one function of device drivers in a computer system.
- 3) What are utilities in the context of system software?
- 4) Explain memory management and its significance in an operating system.
- 5) How does the operating system manage devices in a computer system?
- 6) Define file management in the context of an operating system.
- 7) Distinguish between system software and application software.
- 8) Please provide examples of application software and briefly describe their functions.
- 9) What is the fundamental difference between low-level and high-level programming languages?
- 10) Describe one advantage and one disadvantage of using high-level languages compared to low-level languages.

## **C** Long questions.


- 1) Explain in detail the functions of an operating system. Discuss at least three core functions, such as memory management, device management, and file management, and elaborate on their significance in maintaining the computer system's efficiency and functionality.
- 2) Define application software and provide examples of various types of application software along with their specific functions. Describe how these applications serve different purposes and contribute to enhancing user experience and productivity.
- 3) Compare and contrast low-level and high-level programming languages. Highlight their fundamental differences, advantages, and disadvantages. Discuss how the choice of programming language impacts software development and execution.
- 4) Detail the different components of system software, including operating systems, device drivers, utilities, language translators, firmware, and virtual machine managers. Explain the roles and functions of each component in ensuring efficient computer operations.
- 5) Explain why operating systems are crucial for computer functionality and user interaction. Discuss the significance of various operating system functions such as memory management, processor management, and user interface in providing a seamless computing experience.



## Standard








Students will understand and explain the various components of a computer system and the different levels of interactions between these. They will have a basic understanding of digital logic, the different stages of the software life cycle and the concepts of scalability, reliability and security for computer systems.

### Students' Learning Outcomes-1

 Students will be able to identify and analyze data communication, computer networks, networking devices, basic networking systems and understand how data is transmitted and key concepts such as protocols, speeds, etc.


### Knowledge

#### Student will understand that

-  How data is transmitted across a computer network for example circuit switching, packet switching, layering, encapsulation, and protocols
-  Common network topologies and transmission modes
-  Outline the advantages and disadvantages of wireless networks
-  Outline the advantages and evolution of the Internet
-  Outline common applications of the Internet
-  The internet is the largest computer network ever built and learn how it works
-  Explain these key terms to students: 7-layer OSI networking model, LAN, WAN, packet switching, circuits, circuit switching, router, TCP/IP, IP, UDP, DNS, DHCP, host, browsers, layering, encapsulation and protocols

### Skills

#### Students will be able to:

-  Identify common network problems and provide possible solutions



## Knowledge 1.17 | History of Internet

**Internet** was first brought into use by the U.S. Department of Defense's Advanced Research Project Agency (ARPA) in 1969 for the sole purpose of communication amongst them in case of any emergency war situation. Eventually, the scientist fraternity gradually used the ARPANet for their research work. But still, then, the internet was not in everyday use among the general public as a standard way of communication was not there. In the 1970s, Bob Kahn published his research at ARPA, which later evolved into Transmission Control Protocol (TCP) or Internet Protocol (I.P.), the two main protocols of the International Internet Protocol Suite. In 1983, the scientists officially established TCP/IP, and the birthday of the internet was on 1st January 1983. In 1986, the National Science Federation (NSF) of the U.S. set up interconnectivity among several universities of the United States. Thus, the United States made the emergence of computer architectures such as Domain Name System (DNS) and TCP/IP official. At the same time, along with the USA, Australia also started to adopt the new Internet system. In 1990 the first internet, ARPANET, was officially decommissioned. Several private connections and commercial entities came up across the United States, thus removing the restrictions on the use of the internet among the general public and for commercial purposes.

### Computer networks and the internet

The internet is actually a network of networks that connects billions of digital devices worldwide. Standard protocols allow communication between these devices. Those protocols include hypertext transfer protocol (the 'http' in front of all website addresses). Internet protocol (or IP addresses) are the unique identifying numbers required of every device that accesses the internet. IP addresses are comparable to your mailing address, providing unique location information so that information can be delivered correctly.

Internet Service Providers (ISPs) and Network Service Providers (NSPs) provide the infrastructure that allows the transmission of packets of data or information over the internet. Every bit of information sent over the internet doesn't go to every device connected to the internet. It's the combination of protocols and infrastructure that tells information exactly where to go.

### How do they work?

Computer networks connect nodes like computers, routers, and switches using cables, fiber optics, or wireless signals. These connections allow devices in a network to communicate and share

information and resources.

Networks follow protocols, which define how communications are sent and received. These protocols allow devices to communicate. Each device on a network uses an Internet Protocol or IP address, a string of numbers that uniquely identifies a device and allows other devices to recognize it.

Routers are virtual or physical devices that facilitate communications between different networks. Routers analyze information to determine the best way for data to reach its ultimate destination. Switches connect devices and manage node-to-node communication inside a network, ensuring that bundles of information traveling across the network reach their ultimate destination.

### Knowledge 1.18 | Application of the Internet

This table presents a concise overview of various applications and their descriptions, highlighting key functionalities and purposes for each application.

Application	Description
Communication	Refers to exchanging ideas and thoughts between or among people to create understanding. In organizations, both formal and informal communications simultaneously take place. Formal communications refer to official communications in orders, notes, circulars, agenda, minutes, etc. Informal communications are usually in the form of rumors, whispers, etc.
Web Browsing	One of the applications of the internet, allowing users to interact with all the data in the World Wide Web (WWW) through web browsers such as Google Chrome, Firefox, Safari, Internet Explorer, Opera, Microsoft Edge, and Netscape.
Online Shopping	The internet enables virtual shops to be available 24x7, providing all necessary product details on websites for users to choose as per their needs.
Real-Time Update	Allows quick access to real-time updates on various topics such as sports, politics, business, finance, etc., facilitating informed decisions.
Social Media	Platforms where users can communicate with others, promote businesses, share thoughts, pictures, and videos.
Job Search	Users can search, apply, and get jobs easily through online platforms such as LinkedIn, Monster.com, Naukari.com, Indeed, Glassdoor, and Upwork.
Education	The internet plays a vital role in teaching and learning, allowing teachers to upload notes or learning videos, enhancing the learning process.
Travel	Users can search and plan trips, book holiday packages, cabs, hotels, flight tickets, and more through websites like goibibo.com, makemytrip.com, and olacabs.com.

Stock Market Update	Provides latest information and news related to financial markets, particularly the stock market, including current stock prices, trading volumes, market capitalization, and price movements.
Video Conferencing	Using computers to provide a video link between two or more people, enabling face-to-face meetings among individuals in different locations.

## Knowledge 1.19 | Networking Devices

Network devices, also known as networking hardware, are physical devices that allow hardware on a computer network to communicate and interact with one another. For example, Repeater, Hub, Bridge, Switch, Routers, Gateway, Brouwer, and NIC, etc.

### 1. Repeater:

A repeater operates at the physical layer. Its job is to regenerate the signal over the same network before the signal becomes too weak or corrupted to extend the length to which the signal can be transmitted over the same network. An important point to be noted about repeaters is that they not only amplify the signal but also regenerate it. When the signal becomes weak, they copy it bit by bit and regenerate it at its star topology connectors connecting following the original strength. It is a 2-port device.

### 2. Hub:

A hub is a basically multi-port repeater. A hub connects multiple wires coming from different branches, for example, the connector in star topology which connects different stations. Hubs cannot filter data, so data packets are sent to all connected devices. In other words, the collision domain of all hosts connected through Hub remains one. Also, they do not have the intelligence to find out the best path for data packets which leads to inefficiencies and wastage.

### 3. Bridge:

A bridge operates at the data link layer. A bridge is a repeater, with add on the functionality of filtering content by reading the MAC addresses of the source and destination.

### 4. Switch:

A switch is a multiport bridge with a buffer and a design that can boost its efficiency (a large number of ports imply less traffic) and performance. A switch is a data link layer device. The switch can perform error checking before forwarding data, which makes it very efficient as it does not forward packets that have errors and forward good packets selectively to the correct port only. In other words, the switch divides the collision domain of hosts, but fig 1.19(5) the broadcast domain remains the same.





Fig 1.19(5)



Fig 1.19(6)

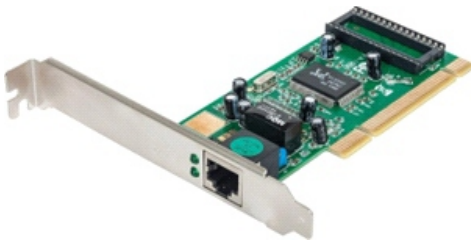


Fig 1.19(7)

## 5. Routers:

A router is a device like a switch that routes data packets based on their IP addresses. The router is mainly a Network Layer device. Routers normally connect LANs and WANs and have a dynamically updating routing table based on which they make decisions on routing the data packets. The router divides the broadcast domains of hosts connected through it.

## 6. Gateway:

A gateway, as the name suggests, is a passage to connect two networks that may work upon different networking models. They work as messenger agents that take data from one system, interpret it, and transfer it to another system. Gateways are also called protocol converters and can operate at any network layer. Gateways are generally more complex than switches or routers. A gateway is also called a protocol converter.

## 7. Network Interface Card:

NIC or network interface card is a network adapter that is used to connect the computer to the network. It is installed in the computer to establish a LAN. It has a unique id that is written on the chip, and it has a connector to connect the cable to it. The cable acts as an interface between the computer and the router or modem. NIC card is a layer 2 device which means that it works on both the physical and data link layers of the network model.

## Knowledge 1.20 | Key Terms related to Networks

### Web Browser

A web browser is a software that enables users to access and view content on the World Wide Web. Its primary function is to locate and retrieve web pages, images, videos, documents, and other files from servers and display them on the user's device.

### Packet switching:

Packet switching is the transfer of small pieces of data across various networks. These data chunks or “packets” allow faster, more efficient data transfer.

### Circuit switching:

Circuit switching is a type of network configuration in which a physical path is obtained and dedicated to a single connection between two endpoints in the network for the duration of a dedicated connection.

### Encapsulation:

Encapsulation adds information to a packet as it travels to its

destination. Decapsulation reverses the process by removing the info, so a destination device can read the original data.

A Network Topology is the arrangement with which computer systems or network devices are connected to each other. Topologies may define both physical and logical aspect of the network. Both logical and physical topologies could be same or different in a same network.

## Knowledge 1.21 | Network Protocols

**Protocol**, in computer science, are set of rules or procedures for transmitting data between electronic devices, such as computers. In order for computers to exchange information, there must be a preexisting agreement as to how the information will be structured and how each side will send and receive it.

### a. TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol) is the fundamental communication protocol used by the Internet. It handles the breakdown of data into packets, the routing of these packets across the network, and then reassembly at the destination.

### b. UDP

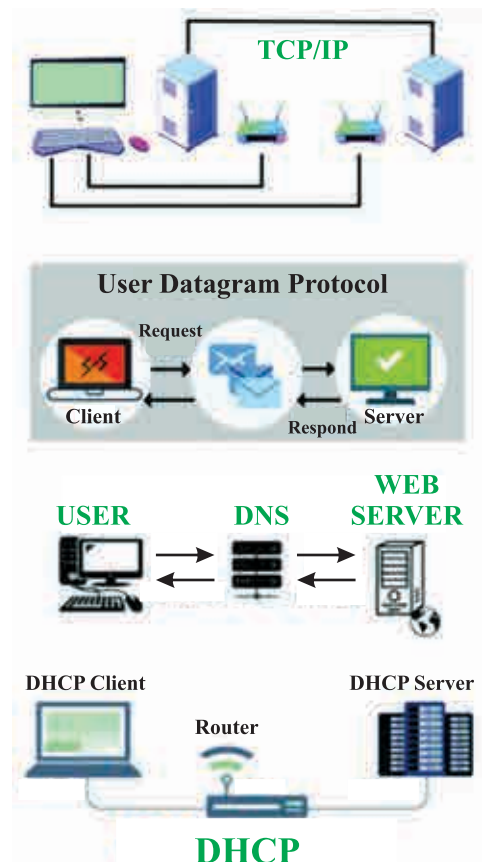
UDP (User Datagram Protocol) is frequently used when communications are time-sensitive. For users, it is better to have the overall transmission arrive on time than wait for it to get there in a near-perfect state. For this reason, UDP is commonly used in Voice over Internet Protocol (VoIP) applications as well.

### c. DNS (Domain name system)

The domain name system (i.e., “DNS”) is responsible for translating domain names into a specific IP address so that the initiating client can load the requested Internet resources.

### d. DHCP (Dynamic Host Configuration Protocol)

DHCP (Dynamic Host Configuration Protocol) is a network management protocol used to dynamically assign an IP address to any device, or node, on a network so it can communicate using IP.



## Knowledge 1.22 | Wireless Networks

The connection between the two devices is wireless, makes information to be accessible anywhere at any time and up to date. Regardless of the size of the equipment or a device, the device is embedded in a wireless connection. So, the point to be noticed is no cable or a wire can restrict the user. This table provides a clear comparison of the advantages and disadvantages of wireless



networks, highlighting key points for consideration when choosing between wireless and wired network options.

Advantages of Wireless Network	Disadvantages of Wireless Network
<b>Accessibility:</b> Users can communicate while moving without being tethered by wires, leading to increased productivity.	<b>Security:</b> Incorrectly configured or maintained wireless networks can be vulnerable to security threats, including hacking.
<b>Easy Installation:</b> Faster and simpler to set up compared to wired networks, reducing safety risks associated with cables.	<b>Limited Bandwidth:</b> Cannot support high-bandwidth applications like video conferencing, and bandwidth can be limited or stolen by neighboring networks.
<b>Wider Reach:</b> Can extend to areas where wired networks are not feasible, providing flexibility in deployment.	<b>Speed:</b> Slower than wired networks, especially at greater distances from the access point, affecting file transfer speeds.
<b>Flexibility:</b> Facilitates remote work and improves accessibility to data, enhancing productivity.	<b>Cost:</b> Initial setup costs can be high, including equipment and installation expenses, though they may be cheaper in the long run.
<b>Efficiency:</b> Allows faster data communication between users, enhancing overall efficiency.	<b>Prone to Interference:</b> Susceptible to interference from external factors like weather or neighboring networks, leading to potential disruptions.
<b>Cost-Effective:</b> Cheaper and easier to install than wired networks, leading to lower overall expenses over time.	<b>Coverage:</b> Limited coverage area compared to wired networks, typically ranging from 150 to 300 feet for a typical wireless router.
	<b>Requires Basic Computer Knowledge:</b> Setting up a wireless network may require computer expertise, posing challenges for inexperienced users.

## Knowledge 1.23 | Network Topologies

A network topology is the physical and logical arrangement of nodes and connections in a network. Nodes usually include devices such as switches, routers and software with switch and router features. Here are some types of network topologies.

### a. Bus Topology

In a Bus network, all devices share one communication line. If many devices try to talk simultaneously, there might be a problem. To solve this, Bus networks use special rules or choose one device to manage the communication. It's simple. If one device has trouble, others usually keep working, but if the shared line has a problem, everyone might have to stop.

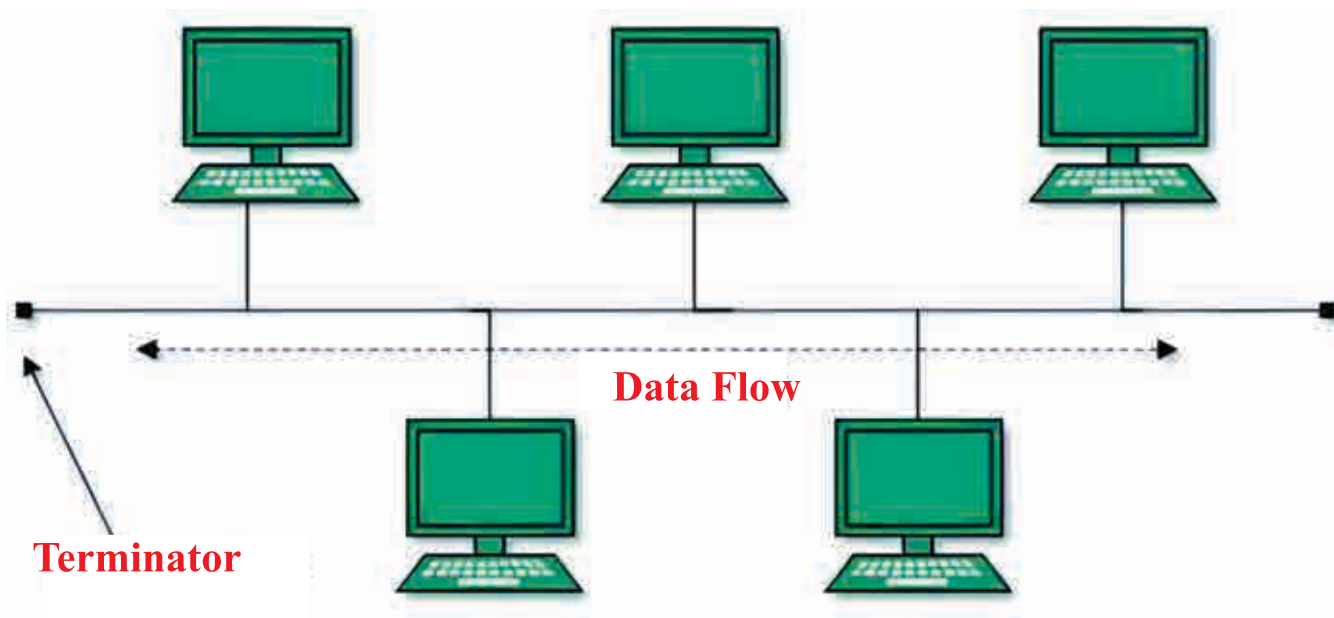


Fig 1.23 (a)

### b. Star Topology

In a star topology, devices connect to a central hub, forming a star-shaped network. Communication between devices relies on the central hub, a switch or a hub. If the central hub is faulty, it can affect the entire network's communication.

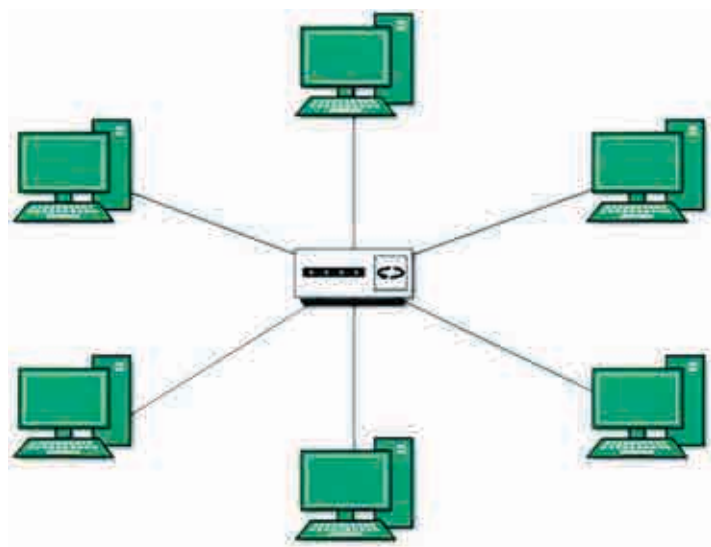


Fig 1.23 (b)

### c. Ring Topology

In a ring topology, computers connect in a circle. Messages travel through all computers to reach distant ones. Adding a computer needs just one extra cable. If any computer fails, the whole ring may stop, some use a backup ring for safety.

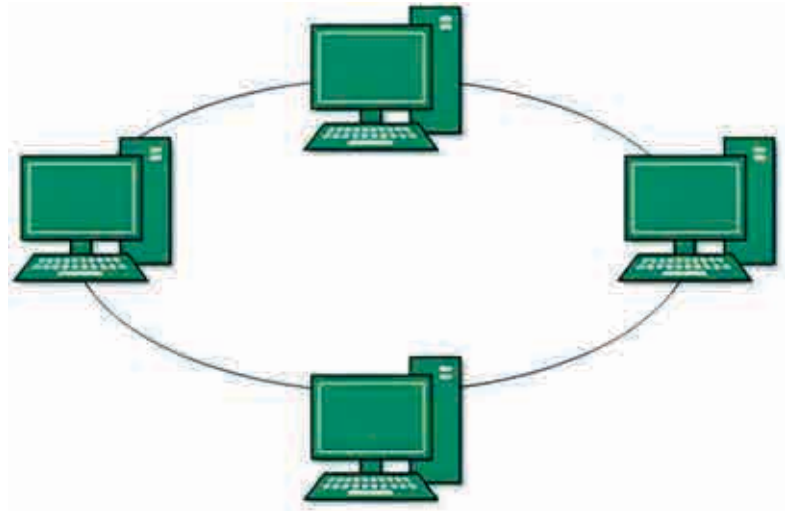


Fig 1.23 (c)

#### d. Mesh Topology

In Mesh topology, computers connect directly, forming a network. They can have direct links with all computers (Full Mesh) or only some (Partial Mesh). Computers in Mesh help relay messages to those without direct connections. Full Mesh is highly reliable, while Partial Mesh is used when only specific computers need extra reliability.

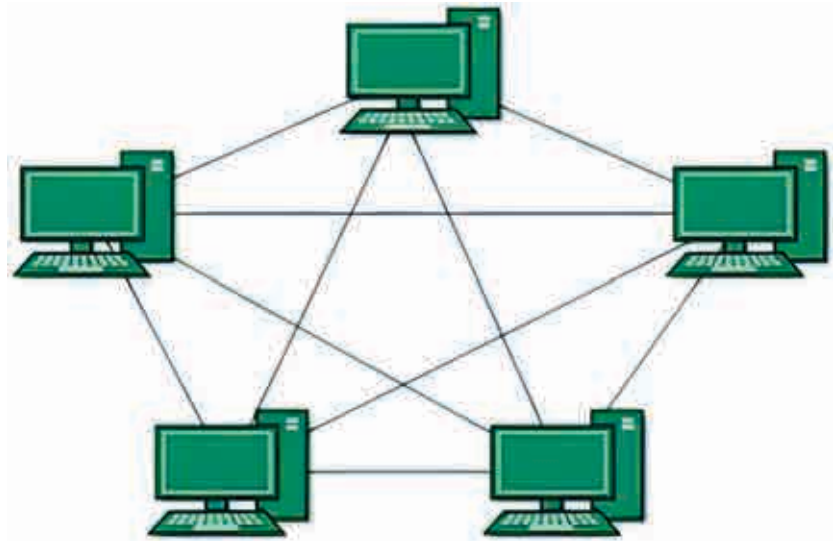


Fig 1.23 (d)

#### e. Tree Topology

Hierarchical Topology, or Extended Star, is widely used in networks. It divides the network into layers, access, distribution, and core. Each layer connects devices point-to-point, similar to a Bus. The whole network can be affected if the central core has an issue, as every connection point is a potential problem.

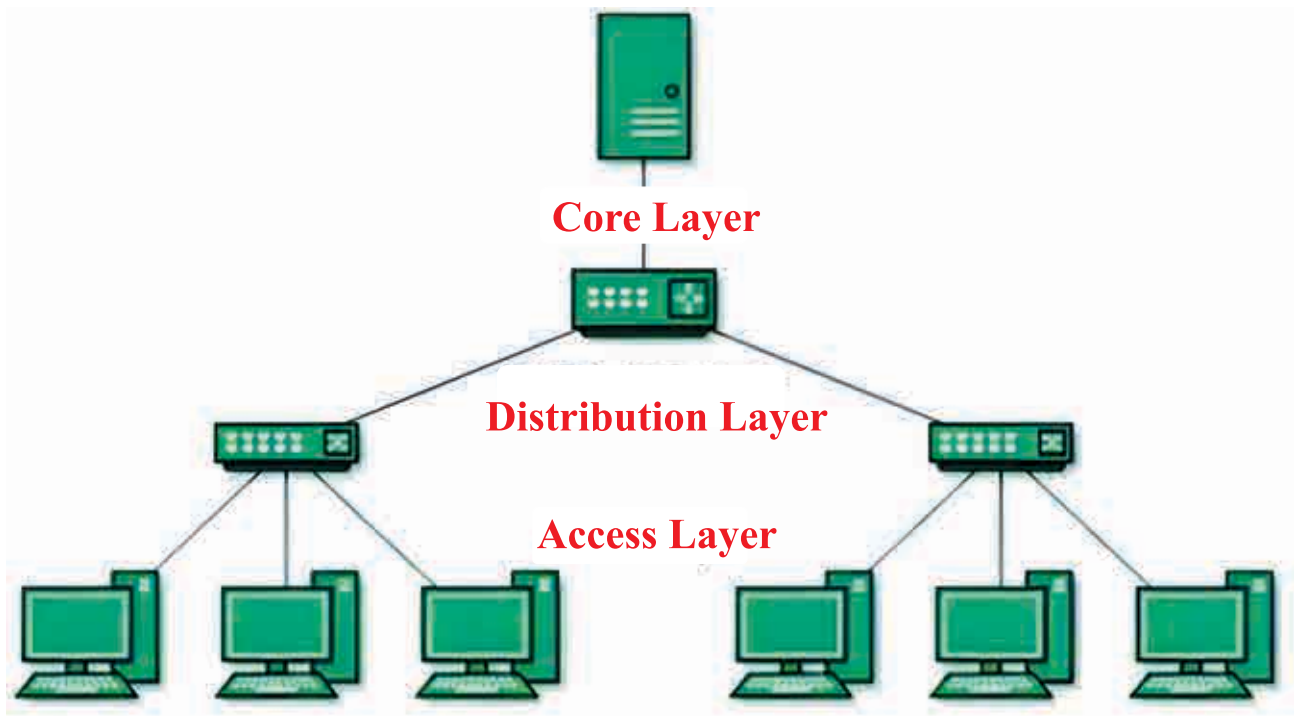


Fig 1.23 (e)

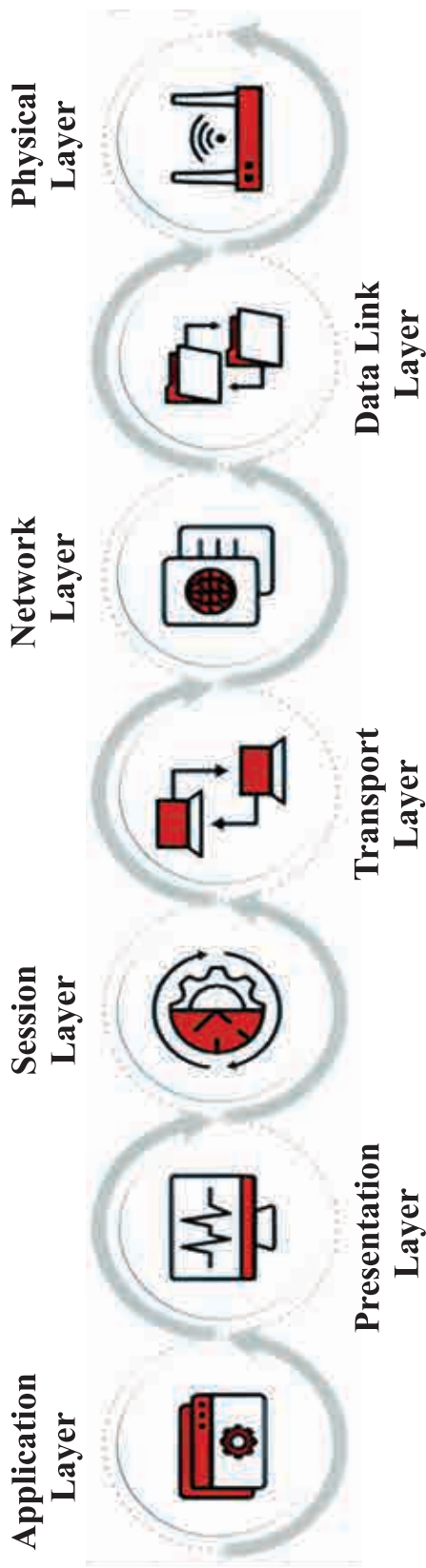
## Knowledge 1.24 | 7 Layers of an OSI Model

### Layer 7 – Application

The Application Layer in the OSI model is the layer that is the “closest to the end user”. It receives information directly from users and displays incoming data to the user, applications themselves do not reside at the application layer. Instead the layer facilitates communication through lower layers in order to establish connections with applications at the other end. Web browsers (Google Chrome, Firefox, Safari, etc.) Telnet, and FTP, are examples of communications that rely on Layer 7.

### Layer 6 - Presentation

The Presentation Layer represents the area that is independent of data representation at the application layer. In general, it represents the preparation or translation of application format to network format, or from network formatting to application format. In other words, the layer “presents” data for the application or the network. A good example of this is encryption and decryption of data for secure transmission; this happens at Layer 6.



## Layer 5 - Session

When two computers or other networked devices need to speak with one another, a session needs to be created, and this is done at the Session Layer. Functions at this layer involve setup, coordination (how long should a system wait for a response, for example) and termination between the applications at each end of the session.

## Layer 4 – Transport

The Transport Layer deals with the coordination of the data transfer between end systems and hosts. How much data to send, at what rate, where it goes, etc. The best-known example of the Transport Layer is the Transmission Control Protocol (TCP), which is built on top of the Internet Protocol (IP), commonly known as TCP/IP. TCP and UDP port numbers work at Layer 4, while IP addresses work at Layer 3, the Network Layer.

## Layer 3 - Network

Here at the Network Layer is where you'll find most of the router functionality that most networking professionals care about and love. In its most basic sense, this layer is responsible for packet forwarding, including routing through different routers. You might know that your Boston computer wants to connect to a server in California, but there are millions of different paths to take. Routers at this layer help do this efficiently.

## Layer 2 – Data Link

The Data Link Layer provides node-to-node data transfer (between two directly connected nodes), and also handles error correction from the physical layer. Two sublayers exist here as well--the Media Access Control (MAC) layer and the Logical Link Control (LLC) layer. In the networking world, most switches operate at Layer 2. But it's not that simple. Some switches also operate at Layer 3 in order to support virtual LANs that may span more than one switch subnet, which requires routing capabilities.

## Layer 1 - Physical

At the bottom of our OSI model we have the Physical Layer, which represents the electrical and physical representation of the system. This can include everything from the cable type, radio frequency link (as in a Wi-Fi network), as well as the layout of pins, voltages, and other physical requirements. When a networking problem occurs, many networking pros go right to the physical layer to check that all of the cables are properly connected and that the power plug hasn't been pulled from the router, switch or computer.



## Knowledge 1.25 | Data Communication

Data transmission is the transfer of data from one digital device to another. This transfer occurs via point-to-point data streams or channels. These channels may previously have been in the form of copper wires but are now much more likely to be part of a wireless network.

### Data Communication System Components:

There are mainly five components of a data communication system:

- Message
- Sender
- Receiver
- Transmission Medium
- Protocol

#### a. Message:

This is most useful asset of a data communication system. The message simply refers to data or piece of information which is to be communicated. A message could be in any form, it may be in form of a text file, an audio file, a video file, etc.

#### b. Sender:

To transfer message from source to destination, someone must be there who will play role of a source. Sender plays part of a source in data communication system. It is simple a device that sends data message. The device could be in form of a computer, mobile, telephone, laptop, video camera, or a workstation, etc.

#### c. Receiver:

It is destination where finally message sent by source has arrived. It is a device that receives message. Same as sender, receiver can also be in form of a computer, telephone mobile, workstation, etc.

#### d. Transmission Medium:

In entire process of data communication, there must be something which could act as a bridge between sender and receiver, Transmission medium plays that part. It is physical path by which data or message travels from sender to receiver. Transmission medium could be guided (with wires) or unguided (without wires), for example, twisted pair cable, fiber optic cable, radio waves, microwaves, etc.

#### d. Protocol:

To govern data communications, various sets of rules had been already designed by the designers of the communication systems, which represent a kind of agreement between communicating

devices. These are defined as protocol. In simple terms, the protocol is a set of rules that govern data communication. If two different devices are connected but there is no protocol among them, there would not be any kind of communication between those two devices. Thus, the protocol is necessary for data communication to take place.

## Knowledge 1.26 | Data Transmission Modes

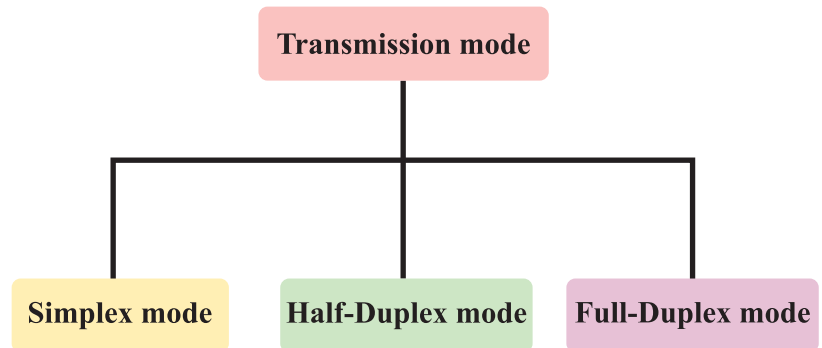


Fig 1.26

These are explained as following below.

### a. Simplex Mode

In Simplex mode, the communication is unidirectional, as on a one-way street. Only one of the two devices on a link can transmit, the other can only receive. The simplex mode can use the entire capacity of the channel to send data in one direction.

**Example:** Keyboard and traditional monitors. The keyboard can only introduce input, the monitor can only give the output.

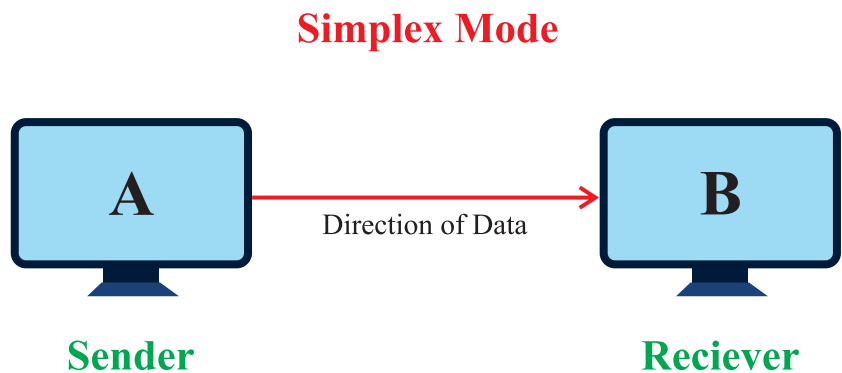


Fig 1.26 (a)

This table provides a clear comparison of the advantages and disadvantages of simplex mode communication, highlighting key points for consideration when choosing this mode for communication purposes.

Advantages of Wireless Network	Disadvantages of Wireless Network
Easiest and most reliable mode of communication.	Only one-way communication is possible.
Most cost-effective, requiring only one communication channel.	No way to verify if transmitted data has been received correctly.
No need for coordination between transmitting and receiving devices.	Not suitable for applications requiring bidirectional communication.
Particularly useful in situations where feedback or response is not required.	

### b. Half-Duplex Mode

In half-duplex mode, each station can both transmit and receive, but not at the same time. When one device is sending, the other can only receive, and vice versa. The half-duplex mode is used in cases where there is no need for communication in both directions at the same time. The entire capacity of the channel can be utilized for each direction.

**Example:** Walkie-talkie in which message is sent one at a time and messages are sent in both directions.

**Channel capacity=Bandwidth \* Propagation Delay**

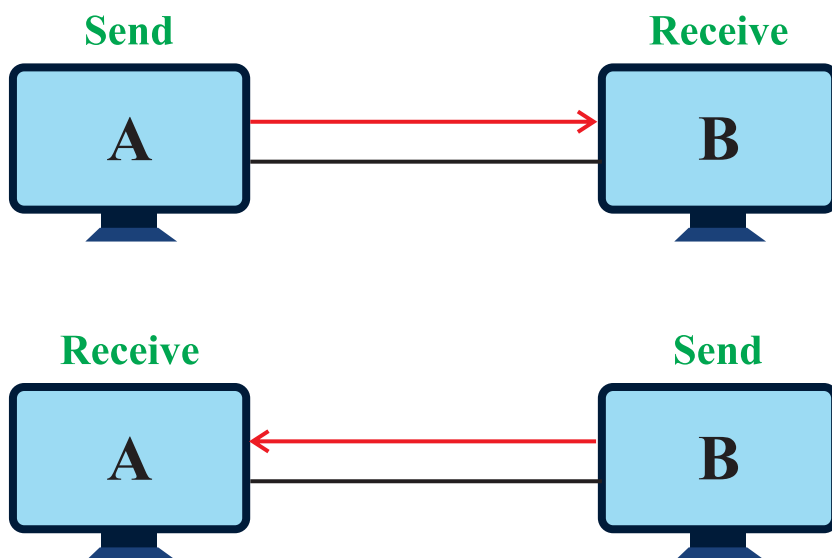
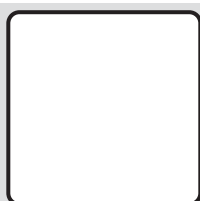


Fig 1.26 (b)

Advantages of Half-Duplex Mode	Disadvantages of Half-Duplex Mode
Allows bidirectional communication.	Less reliable than Full-Duplex mode.
More efficient than simplex mode.	Delay between transmission and reception.
Less expensive than full-duplex mode	Coordination needed between transmitting and receiving devices.



#### Skill:1.6

#### Identify common network problems and provide possible solutions

#### Network Troubleshooting Challenge

**Objective:** The aim of this activity is to empower grade 9 students to identify common network problems and propose effective solutions through hands-on troubleshooting. Scan the QR code for further details

### c. Full-Duplex-Mode

In full-duplex mode, both stations can transmit and receive simultaneously. In full duplex mode, signals going in one direction share the capacity of the link with signals going in another direction, this sharing can occur in two ways:

Either the link must contain two physically separate transmission paths, one for sending and the other for receiving.

Or the capacity is divided between signals traveling in both directions.

Full-duplex mode is used when communication in both directions is required all the time. The capacity of the channel, however, must be divided between the two directions.

**Example:** Telephone Network in which there is communication between two persons by a telephone line, through which both can talk and listen at the same time.

$$\text{Channel Capacity} = 2 * \text{Bandwidth} * \text{propagation Delay}$$

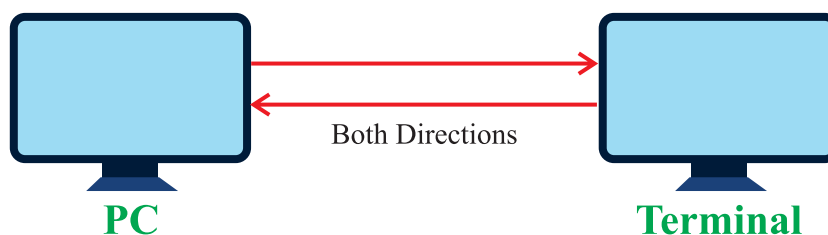


Fig 1.23 (c)

Advantages of Half-Duplex Mode	Disadvantages of Half-Duplex Mode
Allows simultaneous bidirectional communication.	Most expensive mode, requiring two communication channels.
Most efficient mode, enabling simultaneous data transmission and reception.	More complex than simplex and half-duplex modes.

Provides high reliability and accuracy, minimizing the need for error correction mechanisms.

May not be suitable for all applications due to high bandwidth requirements.

## Exercise



### A Multiple Choice Questions

1. What is the main purpose of a router in a computer network?
  - a) It connects devices within the same network.
  - b) It facilitates communication between different networks.
  - c) It assigns IP addresses to devices.
  - d) It encrypts data for secure transmission.
2. What is the primary function of an Internet Service Provider (ISP)?
  - a) Creating websites
  - b) Hosting email services
  - c) Providing network infrastructure for internet access
  - d) Developing software applications
3. Which transmission mode allows for bidirectional communication but not at the same time?
  - a) Simplex
  - b) Half-Duplex
  - c) Full-Duplex
  - d) Multiplex
4. What type of communication occurs via rumors and whispers and is unofficial and unrecorded?
  - a) Formal communication
  - b) Informal communication
  - c) One-way communication
  - d) Secure communication
5. What is the primary role of a web browser in the context of the internet?
  - a) To host websites
  - b) To provide internet access
  - c) To facilitate video conferencing
  - d) To interact with web content
6. What does TCP/IP stand for in the context of computer networking?
  - a) Total Control Protocol/Internet Protocol
  - b) Transmission Control Protocol/Internet Protocol
  - c) Textual Communication Protocol/Internet Protocol
  - d) Transfer Control Protocol/Internet Protocol
7. Which network topology connects all devices to a central hub or switch?
  - a) Bus Topology
  - b) Star Topology
  - c) Ring Topology
  - d) Mesh Topology
8. What is the primary function of the Session Layer in the OSI model?
  - a) Data transfer between end systems
  - b) Packet forwarding and routing
  - c) Presentation of data for the application layer
  - d) Establishing, coordinating, and terminating sessions between applications
9. Which type of transmission mode allows devices to transmit and receive data simultaneously?
  - a) Simplex
  - b) Half-Duplex
  - c) Full-Duplex
  - d) Multiplex
10. What is the role of Domain Name System (DNS) in computer networks?
  - a) Assigning IP addresses to devices
  - b) Facilitating real-time video conferencing
  - c) Translating domain names into IP addresses
  - d) Managing network security



## **B** Short Questions

1. Explain the differences between simplex, half-duplex, and full-duplex transmission modes.
2. Describe the role of Internet Service Providers (ISPs) in enabling access to the internet.
3. Discuss the key functions of a router in computer networks, including how it determines the path for data transmission.
4. How has the internet transformed the field of education? Provide examples of how it's used in teaching and learning.
5. Explain the seven layers of the OSI model and briefly describe the function of each layer.
6. Compare and contrast the advantages and disadvantages of bus, star, and mesh network topologies.
7. Define packet switching and circuit switching. Provide examples of situations where each is used.
8. Describe the concept of encapsulation in data transmission and its significance in network communication.

## **C** Long questions.


1. Provide a comprehensive history of the internet, highlighting its origins, major milestones, and its evolution into a global network.
2. Explore the advantages and disadvantages of wireless networks. How have wireless networks revolutionized communication and connectivity in the modern world?
3. Discuss the impact of social media on society and communication. Analyze both its positive and negative effects on individuals and communities.
4. Describe the components of a typical computer network, including devices like routers, switches, and servers. Explain their roles and how they contribute to network functionality.

# Computational Thinking & Algorithms

**SLO: B01****Importance of Computational thinking****Standard**








Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.

## ● Students' Learning Outcomes-1 ●

 Understand and apply techniques to decompose problems.






### Knowledge

#### Student will understand that

-  The importance of computational thinking and problem-solving in computer science.
  - Principles of computational thinking:
  - Logical thinking
-  Algorithmic thinking
-  How to identify steps in identifying a computing problem
-  How to identify the inputs, processes, and outputs of a problem
-  Different methods to design and construct a solution to a simple problem, such as flow charts, and/or concept maps.
-  Steps to produce simple diagrams to show:
  - The structure of a problem
-  Subsections and their links to other subsections.

### Skills

#### Students will be able to:

-  Explain the role of computational thinking in computer science.
-  Read and interpret simple computational problems
-  Apply computational thinking principles to define and refine problems
-  Identify the procedure appropriate to solving a problem.
-  Evaluate whether the order in which activities are undertaken will result in the required outcome.  
Identify the inputs and outputs required in a solution.

## Diagnostic Assessment

As a class activity for grade 9 level students to introduce the chapter on computational thinking and problem-solving, some possible prerequisite questions could include:

What is computational thinking, and why is it essential in problem-solving using computers?

Describe a situation where abstraction could be used to simplify a problem or task in everyday life.

### How do algorithms differ from computational thinking concepts like patterns or decomposition?

Welcome back to exploring computational thinking and algorithms! This chapter aims to strengthen your problem-solving skills and deepen your logical reasoning abilities, building upon what you learned in earlier grades. Computational thinking is a powerful way of dealing with problems by breaking them into smaller parts and creating organized solutions. When you think like a computer scientist, you learn how to examine problems closely, find similarities between them, and create step-by-step plans to solve them efficiently. In previous grades, you discovered key concepts like decomposition, which means breaking big problems into smaller, easier-to-handle pieces. Also, pattern recognition helped you find common features or similarities in problems and their solutions.

## Knowledge 2.1

### Understand the Importance of Computational thinking and Problem solving in Computer Science .

Computational thinking means making ideas clear and organized enough so a computer can understand and do them. This skill is essential because it helps us solve problems in a clever and organized way.

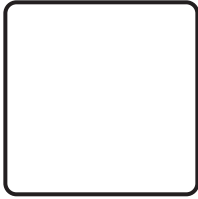
- Computational thinking is a skill about making ideas clear for computers.
- It helps us think wisely and solve problems.
- Nowadays, it's super important in our digital world.
- Many jobs need people with these skills.
- Knowing computational thinking can help us get better jobs in the digital world. China, Russia, and Japan continue to use the abacus for calculations. Below is an image of this tool.

## The 4 Cornerstones of Computational Thinking

Computational thinking is like a problem-solving toolbox. It helps us smartly solve challenging problems. It's not just for computers; we can use it in many parts of our life.

Four main parts of computational thinking make it strong: breaking things down, spotting patterns, simplifying, and making step-by-step plans.

These four parts help us understand problems better and solve them smarter. Let's learn more about each part and how they help us



become better problem-solvers.

## Decomposition

Decomposition is like a big puzzle-solving trick in computational thinking. It means breaking complex problems into smaller, more manageable pieces. It helps us solve challenging tasks by dividing them into more straightforward jobs we can handle.

## Pattern Recognition

Pattern recognition is an integral part of thinking in computers. It helps in solving problems by finding similar things and noticing what's different. When we find similarities in problems, it makes them easier to solve because we can use one solution for many parts. And when we see differences, it helps us understand each part better and find the right solution for each one.

## Abstraction

Abstraction is an important idea in thinking like a computer. It means picking out the most essential parts from broken-down problems and using those to solve the whole problem. It helps us focus on what matters most and ignore things that don't help us find the solution.

## Algorithmic Thinking

Algorithmic Thinking is an essential part of Computational Thinking. It's like making a recipe for solving problems step by step. You break a big problem into smaller steps and put them in order. Algorithms are like clear instructions that help solve a problem. They're like a recipe that everyone, including computers, can follow strictly.



### Skill:2.1

#### Role of Computational thinking in Computer Science

##### Exploring Computational Thinking

**Objective:** The goal of this activity is to introduce grade 9 students to fundamental aspects of computational thinking through a hands-on coding-based activity.

Scan the QR code for further details.



### Learning Activity

Let students play simple games on paper, for example, tic tac toe, connecting dots game or traditional such as connecting dots within a group.

<https://kidpillar.com/games-kids-think-criticallycritical-thinking>.

Apply simple steps to solve problems such as tic tac toe.

Ask students to make an algorithm for playing Tic Tac Toe.





## Knowledge 2.2

### Principles of computational thinking

Computational thinking is a problem-solving methodology that draws upon principles and techniques used in computer science. It involves breaking down complex problems into smaller, more manageable parts, and then systematically analyzing and solving them using computational tools and techniques. Computational thinking emphasizes abstraction, algorithmic thinking, pattern recognition, and the use of logic and reasoning to tackle challenges across various domains, not just within computer science.

At its core, computational thinking involves approaching problems in a structured and logical manner, often leveraging concepts from computer science such as algorithms, data structures, and programming languages. It emphasizes the ability to formulate problems in a way that a computer, or human, can understand and solve them efficiently. Computational thinking is not limited to experts in computer science but is increasingly recognized as a valuable skill for individuals in all fields, as it helps in understanding and addressing complex issues using systematic and analytical approaches.

### Algorithmic Thinking

Algorithmic thinking is a problem-solving approach that involves breaking down complex problems into smaller, more manageable steps. It is a process of logically analyzing and organizing procedures to create a set of instructions that can be executed by a computer or human.

**Definition:** Algorithmic thinking involves using logic and critical thinking skills to develop algorithms, which are sets of instructions that can be used to solve problems.

**Purpose:** The purpose of algorithmic thinking is to efficiently and effectively solve problems, either by humans or computers, through the creation of step-by-step instructions.

**Components:** Algorithmic thinking includes elements such as exploring the problem space, decomposing complex problems into smaller, more manageable parts, recognizing patterns, and testing potential solutions.

**Importance:** It is an essential skill in computer science, programming, and STEM fields, providing the foundation for problem-solving and logical reasoning in technology-driven environments.

**Skills Development:** Algorithmic thinking helps to develop critical



thinking skills, logic, and problem-solving abilities, which are valuable in various professional and academic settings.

**Applications:** Algorithmic thinking has diverse applications across fields such as education, data analysis, machine learning, robotics, and operating systems.

**Education:** In education, algorithmic thinking is used to develop data-driven instruction, instructional planning, and problem-solving strategies for students.

**Data Analysis:** It is employed in data analysis to create algorithms for sorting, analyzing, and interpreting large datasets efficiently.

**Machine Learning:** Algorithmic thinking is crucial in machine learning for developing algorithms that recognize patterns, make predictions, and learn from data.

**Robotics:** In robotics, it is utilized to devise algorithms for controlling robot movements, making decisions, and interacting with the environment.

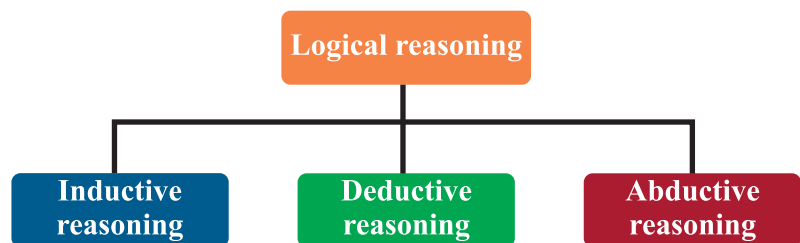
**Operating Systems:** Algorithmic thinking is integral to developing algorithms for managing resources, scheduling tasks, and optimizing system performance in operating systems.

Overall, algorithmic thinking is a foundational skill that enables individuals to approach problems systematically, develop efficient solutions, and harness the power of technology to address complex challenges in various domains.

## Logical thinking

Logical thinking is the process of reasoning, analyzing, and problem-solving based on principles of logic and rationality. It involves systematically evaluating information, identifying patterns, making deductions, and drawing conclusions using logical reasoning.

Key components of logical thinking include:



1. **Analysis:** Logical thinking involves breaking down complex information or problems into smaller, more manageable components. This process requires examining the details and understanding the relationships between different elements.
2. **Pattern Recognition:** Logical thinkers are adept at recognizing patterns, trends, and regularities within data or information. They

can identify similarities, differences, and underlying structures that help them make sense of the world.

3. **Inference:** Logical thinking involves drawing logical conclusions based on evidence, observations, and known facts. It requires making reasoned judgments and predictions about outcomes or solutions.
4. **Deductive Reasoning:** Deductive reasoning involves drawing specific conclusions from general principles or premises. It follows a top-down approach, starting with general principles and applying them to specific cases to derive logical conclusions.
5. **Inductive Reasoning:** Inductive reasoning involves drawing general conclusions from specific observations or evidence. It follows a bottom-up approach, starting with specific observations and patterns and generalizing them to form broader principles or theories.
6. **Critical Thinking:** Logical thinking is closely related to critical thinking, which involves evaluating arguments, evidence, and reasoning to assess their validity and credibility. Critical thinkers question assumptions, evaluate evidence, and consider alternative perspectives before forming conclusions.
7. **Problem-Solving:** Logical thinking is essential for problem-solving, as it helps individuals analyze problems, devise strategies, and evaluate potential solutions based on logic and rationality. It enables individuals to approach problems systematically and methodically, leading to more effective problem-solving outcomes.
8. Overall, logical thinking is a fundamental cognitive skill that underpins various aspects of human cognition, including decision-making, problem-solving, and analytical reasoning. It enables individuals to navigate complex situations, make informed judgments, and arrive at logical conclusions based on evidence and reasoning. Developing and honing logical thinking skills can enhance critical thinking abilities and contribute to more effective decision-making and problem-solving in various domains of life.

### Knowledge 2.3

#### Identify steps involved in Computing Problems

The following are six steps that must be followed to solve a problem using computer:

- Problem Analysis
- Program Design - Algorithm, Flowchart and Pseudocode
- Coding

- Compilation and Execution
- Debugging and Testing
- Program Documentation

## Problem Analysis

Problem-solving in computer science refers to the process of identifying, analyzing, and resolving computational problems using algorithmic techniques and logical reasoning. It involves breaking down complex problems into smaller, more manageable sub-problems, devising efficient algorithms to solve these sub-problems, and then combining the solutions to address the overall problem.

Key aspects of problem-solving in computer science include:

1. **Understanding the Problem:** This involves carefully examining the problem statement, identifying the inputs and outputs, defining constraints and requirements, and gaining a clear understanding of the problem's objectives.
2. **Algorithm Design:** Once the problem is understood, the next step is to devise an algorithmic solution. This involves determining the steps or procedures required to solve the problem, considering factors such as efficiency, correctness, and scalability.
3. **Data Structures:** Problem-solving often involves working with various data structures, such as arrays, lists, stacks, queues, trees, graphs, and hash tables. Choosing the appropriate data structure can significantly impact the efficiency and effectiveness of the solution.
4. **Coding:** After designing the algorithm, the solution is implemented in a programming language using appropriate syntax and coding practices. This step involves translating the algorithm into executable code that can be run on a computer.
5. **Testing and Debugging:** Once the code is written, it is tested with different inputs to ensure that it produces the correct output and behaves as expected. Debugging involves identifying and fixing any errors or bugs in the code that prevent it from functioning correctly.
6. **Optimization:** After the initial solution is developed and tested, it may be necessary to optimize the code for improved performance, reduced memory usage, or better scalability. Optimization techniques may include algorithmic improvements, data structure optimizations, or code refactoring.
7. **Documentation:** Finally, it is essential to document the problem-solving process, including the problem statement, algorithm design, implementation details, testing procedures, and any optimizations made. Documentation helps in understanding

the solution and facilitates collaboration and maintenance in the future.

Overall, problem-solving in computer science requires critical thinking, analytical skills, creativity, and proficiency in algorithm design and programming. It is a fundamental skill for computer scientists, software engineers, and programmers, enabling them to tackle a wide range of computational challenges effectively

## Algorithm

An algorithm is a step-by-step procedure or set of rules designed to solve a specific problem or perform a particular task. In computer science, algorithms are fundamental to solving computational problems efficiently and accurately. They serve as the blueprint or roadmap for solving problems by outlining the sequence of steps needed to achieve the desired outcome.

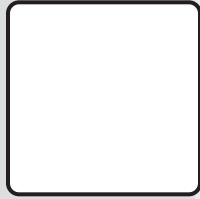
Key characteristics of algorithms include:

1. **Input:** Algorithms take input, which could be data, variables, or conditions, on which they operate to produce output.
2. **Output:** Algorithms produce output, which could be a result, solution, or action, based on the input provided.
3. **Finite:** Algorithms must be finite, meaning they must eventually terminate after a finite number of steps. They cannot loop indefinitely or run forever.
4. **Well-defined:** Algorithms must be well-defined, with each step precisely defined and unambiguous. Each step must be executable and clearly understandable.
5. **Effective:** Algorithms must be effective, meaning they should solve the problem correctly and efficiently within a reasonable amount of time and resources.

Algorithms can be expressed in various forms, including:

1. **Natural Language:** Algorithms can be described using everyday language, such as English, to outline the sequence of steps needed to solve a problem.
2. **Pseudocode:** Pseudocode is a high-level description of an algorithm that combines elements of natural language and programming language syntax. It provides a more formalized representation of an algorithm's logic without adhering to the specific syntax of any programming language.
3. **Flowcharts:** Flowcharts are graphical representations of algorithms using symbols and arrows to illustrate the sequence of steps and **Programming Languages:** Algorithms can be implemented in programming languages such as Python, Java, C++, etc., using the syntax and constructs of the chosen language to translate the algorithm into executable code.





- Algorithms are essential in various fields, including computer science, mathematics, engineering, and beyond, where they are used to solve problems ranging from simple arithmetic calculations to complex data analysis and optimization tasks. They form the foundation of computational thinking and play a crucial role in the development of software, systems, and technologies.

## Knowledge 2.4

### Identifying Inputs, Processes and Outputs

#### Identify Inputs:

First, figure out what you need to start solving a problem - like information, data, or tools. These things you need are called inputs. They can be things you can touch, like numbers or things you can see, or things you can't touch, like instructions or conditions.

#### Define Processes

Next, think about what you'll do with those inputs to solve the problem. These are the processes. It's like the steps you take or the actions you do to change the inputs into what you want.

#### Determine Outputs:

Then, see what happens because of those processes - these are the outputs. It's what you get at the end. Outputs can be numbers, descriptions, or even changes that solve the problem. When designing and constructing a solution to a simple problem, various methods and visual tools can be employed to help with the process.

Two common methods are flowcharts and concept maps:

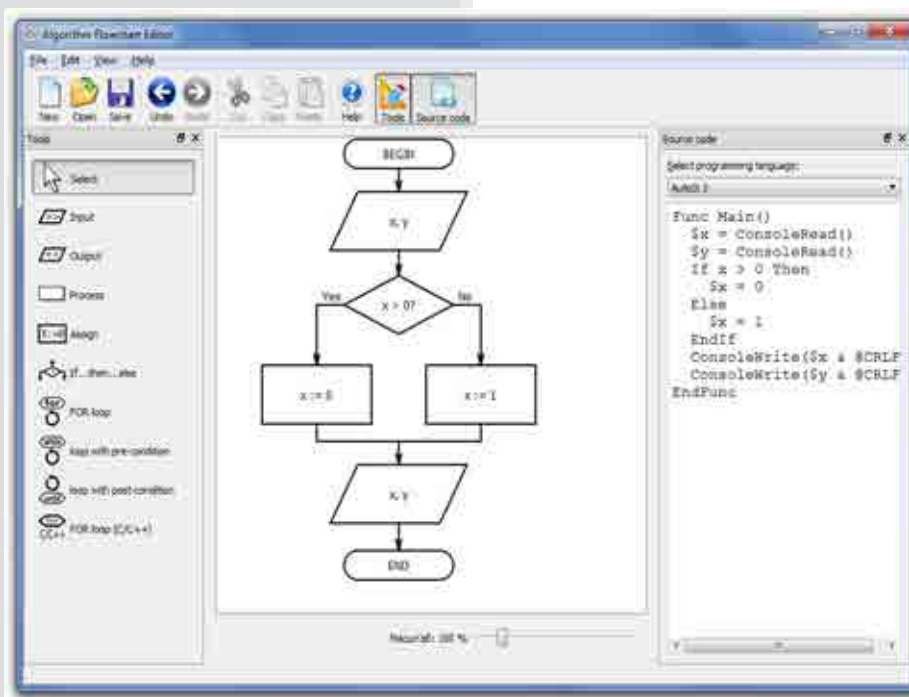
By breaking down a problem into its inputs, processes, and outputs, you get a better idea of how it works and what you need to fix or change to solve it. This is really important in solving problems in computer science, engineering, or business.

When you're working on a solution, there are ways to make it easier. Two common ways are using flowcharts or concept maps. These tools help to plan out the steps or show how things are connected in a problem.



#### Learning Activity

Draw flowchart for calculator program using software like LARP.



## Flowchart

A flowchart is a visual representation of a process or algorithm, using symbols and arrows to illustrate the sequence of steps, decisions, and actions involved. Here's how you can create a flowchart:

**Start/End:** Use an oval-shaped symbol to represent the start and end points of the process. Label the symbol with "Start" at the beginning and "End" at the end.

**Process:** Use rectangular symbols to represent individual steps or actions within the process. Each step describes a specific action or operation to be performed. Label each process symbol with a brief description of the action.

**Decision:** Use diamond-shaped symbols to represent decision points in the process where a choice must be made based on a condition or criteria. Label the symbol with a question that represents the decision to be made.

**Input/Output:** Use parallelogram-shaped symbols to represent points where data enters or exits the process. Label input symbols with a description of the data or information being provided, and label output symbols with a description of the result or outcome produced.

**Flow Arrows:** Use arrows to connect the symbols and indicate the sequence of steps and the direction of flow within the process. Arrows show the order in which actions are performed and how the process progresses from one step to the next.

**Annotations:** Use text annotations or comments to provide additional explanation or details about specific steps or decision points in the flowchart.

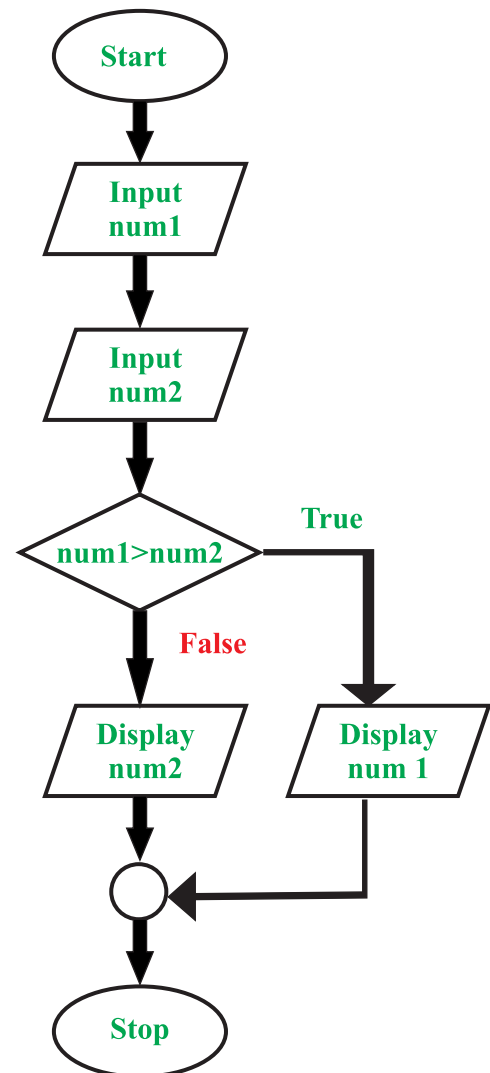
**Testing and Validation:** Once the flowchart is created, review it to ensure it accurately represents the process and logic of the algorithm. Test the flowchart with sample inputs to verify that it produces the expected outputs.

Overall, flowcharts are valuable tools for visualizing, analyzing, and communicating processes and algorithms, enabling individuals and organizations to streamline operations, solve problems, and make informed decisions. They provide a clear and concise overview of complex processes, making them easier to understand and analyze.

## Concept Map

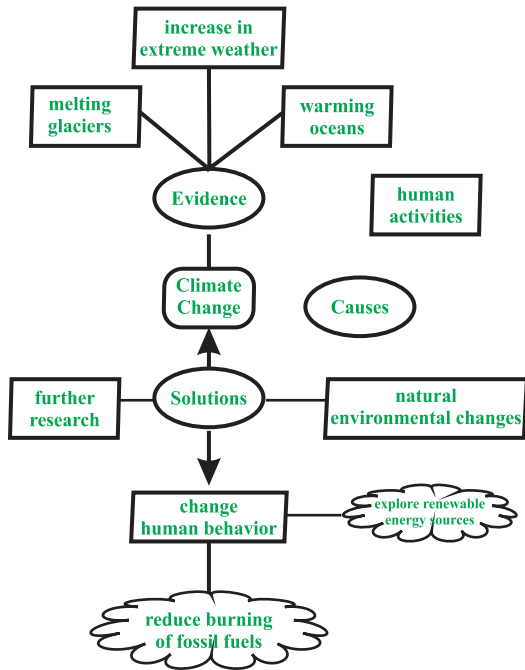
A concept map is a visual representation of knowledge or information that illustrates the relationships between different concepts or ideas. Here's how you can create a concept map:

1. **Identify Key Concepts:** Start by identifying the main concepts or ideas that you want to include in the concept map. These could



be topics, themes, theories, or any other units of knowledge relevant to your subject.

2. **Organize Concepts:** Arrange the concepts in a hierarchical or relational manner, with the most general or overarching concepts at the top and more specific or detailed concepts below them. Use shapes or nodes to represent each concept.
3. **Establish Relationships:** Connect the concepts with lines or arrows to indicate the relationships between them. Relationships could include hierarchical relationships (e.g., parent-child), causal relationships (e.g., cause-effect), associative relationships (e.g., similarity or association), or temporal relationships (e.g., sequence or order).
4. **Label Connections:** Label the lines or arrows with the nature of the relationship between the connected concepts. This helps clarify the meaning of the connections and how the concepts are related to each other.
5. **Add Descriptions:** Include brief descriptions or keywords next to each concept or connection to provide additional context or information. This helps clarify the meaning of each concept and how it relates to other concepts in the map.
6. **Visual Elements:** Use colors, shapes, icons, or formatting to differentiate between different types of concepts or relationships, and to make the concept map visually appealing and easy to understand.
7. **Review and Refine:** Review the concept map to ensure that it accurately represents the relationships between concepts and effectively communicates the intended information. Make any necessary revisions or refinements to improve clarity and coherence.



Concept maps are effective tools for organizing and synthesizing information, identifying connections between ideas, and promoting deeper understanding of complex topics. They can be used for brainstorming, note-taking, studying, teaching, and presenting information in a visually engaging format.

## Knowledge 2.5

### Different Methods for Designing and Constructing Solutions of a Problem.



#### Example 1

##### Calculate the Average of Two Numbers.

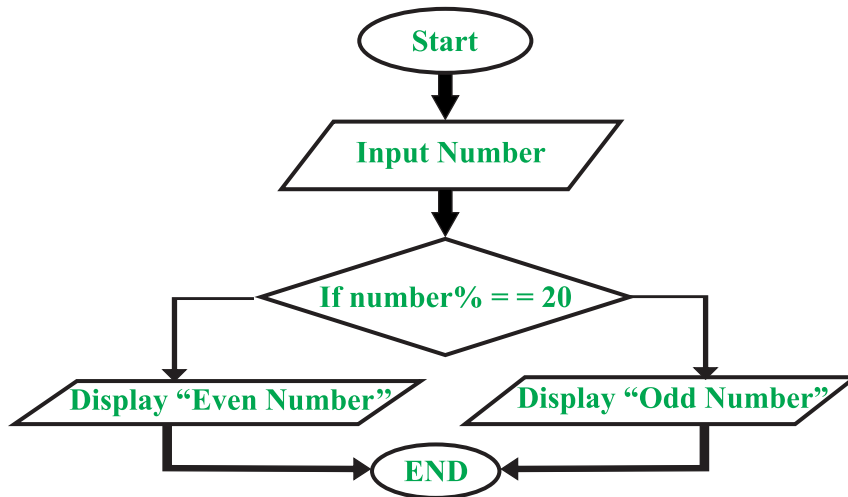
To calculate the average of two numbers, you can sum up those two numbers and divide the result by 2. In other words, the formula for

calculating the average of two numbers is:  $(\text{number 1} + \text{number 2})/2$ .  
So, the basic shapes of the chart should be like this flowchart.

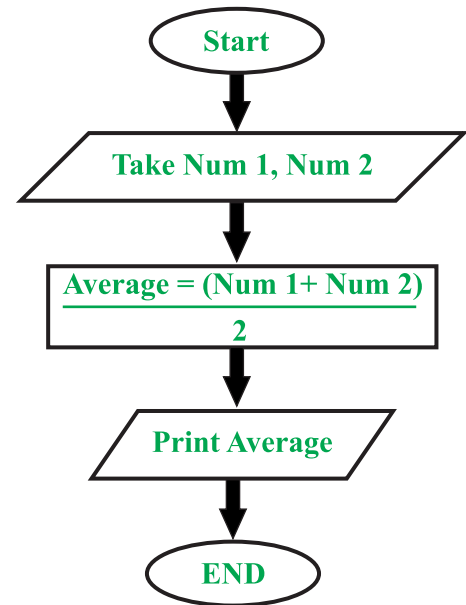
## Example 2

### Input Number and Check If They Are Odd or Even:

Another relatively simple program is checking odd/even numbers. It is a basic conditional operation that involves: inputting a number, determining whether it is odd or even, printing the result on-screen. The chart should be something like this:



Input Number and Check If They Are Odd or Even Algorithm Flowchart.



## Concept Maps

A concept map is like a picture that helps organize information and show how ideas are connected. It's great for understanding complex things by drawing lines and linking different thoughts or parts of a problem. People use concept maps a lot in schools, research, and solving problems because they help us see how different ideas fit together. They're fantastic for brainstorming and showing how one idea leads to another. Flowcharts are similar but better for showing steps in a process, like making a recipe or writing code for a computer. They're suitable for following a sequence or steps in order. Deciding which one to use depends on the problem you're working on. Sometimes, it's helpful to use both because they're good at different things. Mixing them up can help solve problems better because each one has its strengths.

The concept map example depicted above gives an insight into how living things function. To simplify the flow of information - dogs are animals and all animals are living things. Sheep eat herbs, which are plants and therefore all come under one umbrella living things.

## Knowledge 2.6

### Steps to produce simple diagrams to show the structure of a problem

#### a. Define the Problem:

Start by clearly defining the problem you want to represent. This involves understanding the problem's scope, context, and specific challenges. Ensure that you have a thorough grasp of what the problem entails before proceeding with diagram creation. The more precise your problem definition, the more effective your diagram will be.

#### b. Select a Diagram Type:

Choose an appropriate type of diagram that best aligns with the problem and your objectives. Different diagram types are suitable for various scenarios, such as flowcharts for process-oriented problems, mind maps for brainstorming, or tree diagrams for hierarchical structures.

#### c. Identify Key Components:

List the essential components, variables, or factors related to the problem. These are the building blocks of your diagram and should represent the core elements that need to be considered.

#### d. Establish Relationships:

Determine how these components are interconnected and how they influence one another. Understanding the relationships between the components is vital in effectively representing the problem's structure.

#### e. Begin Diagram Creation:

Start creating the diagram, beginning with a central element or concept that represents the main problem or issue. This element serves as the starting point for building the diagram's structure.

#### f. Use Symbols and Shapes:

Depending on the chosen diagram type, use appropriate symbols, shapes, lines, and connectors to represent the components and their relationships. These visual elements help convey meaning and clarify the structure.

## Knowledge 2.7

### Techniques to decompose the Problems

Decomposing problems involves breaking down complex issues into smaller, more manageable parts to better understand, analyze, and



solve them. Here are several techniques to effectively decompose problems:

**a. Divide and Conquer:**

Break the problem into smaller sub-problems, solve them individually, and then combine their solutions to solve the larger issue. This technique is common in algorithms and problem-solving strategies.

**b. Mind Mapping:**

Create a visual representation of the problem by starting with a central idea and branching out into smaller components or related concepts. This helps in understanding relationships and dependencies between different parts of the problem.

**c. Top-Down Approach:**

Start by understanding the big picture and then break it down into smaller components. Begin with an overview and gradually dive deeper into each sub-component.

**d. Bottom-Up Approach:**

Begin with the smallest elements or details of the problem and then build up to understand the larger context. This can help in understanding the fundamental building blocks before tackling the entire problem.

**e. Functional Decomposition:**

Divide the problem based on its functionalities or operations. Identify the various functions required to solve the problem and break them into smaller, manageable tasks.

**f. Dependency Analysis:**

Identify dependencies between different components of the problem. Understand which parts rely on others and how changes in one area can affect others.

**g. Abstraction:**

Simplify complex elements or processes into more manageable and understandable representations. This involves hiding unnecessary details to focus on the essential aspects of the problem.

**h. Use Case Analysis:**

Identify different scenarios or use cases related to the problem. Break down the problem by examining each specific situation separately to find tailored solutions.

**i. Storyboarding or Flowcharting:**

Create a visual representation of the problem's workflow or steps. This helps in understanding the sequence of actions required to solve the problem and identifies potential bottlenecks or issues.

**j. Modularization:**

Divide the problem into modules or components based on their functions or responsibilities. This helps in managing complexity by organizing related tasks together.

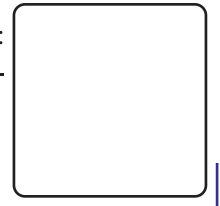
**k. Refinement:**

Continuously refine and iterate upon the decomposition process. As you gain more insights or information, adjust the decomposition to better suit the problem-solving approach.



**— Learning Activity**

Problem definition and Steps to solve an algorithm:  
<https://ramahanishagunda.medium.com/asearch-algorithm-8233683c5d60>



**Exercise**



**A**

**Multiple Choice Questions**

1. What does computational thinking involve?
  - a) Programming computers
  - b) Applying logical reasoning to solve complex problems
  - c) Studying computer hardware
  - d) Engaging in computer simulations
2. Why is computational thinking crucial in computer science?
  - a) Supporting hardware design
  - b) Assisting with real-time data analysis
  - c) Facilitating efficient problem-solving and algorithm creation
  - d) Concentrating on optimizing computer speed
3. Which demonstrates logical thinking?
  - a) Solving a Rubik's cube
  - b) Writing a computer program
  - c) Drawing a concept map
  - d) Designing a flowchart
4. After gathering information to identify a computing problem, what should be done next?
  - a) Seek technical support immediately
  - b) Isolate the problem
  - c) Define the problem
  - d) Check for software updates
5. What are the three main steps for recognizing inputs, processes, and outputs of a problem?
  - a) Gather, Isolate, Run
  - b) Define, Break, Recognize
  - c) Identify, Define, Determine
  - d) Inputs, Processes, Outputs
6. What is computational thinking best described as?
  - a) Focusing on hardware design and engineering

- b) Studying programming languages
  - c) Problem-solving through decomposition and abstraction
  - d) A method for optimizing computer performance
7. In computer science, what is the primary role of problem-solving?
    - a) Debugging computer hardware
    - b) Writing complex code
    - c) Designing efficient algorithms and making informed decisions
    - d) Analyzing network protocols
  8. What is the importance of algorithmic thinking in the field of robotics?
    - a) Creating visually appealing robot designs
    - b) Developing algorithms for recognizing patterns and making predictions
    - c) Focusing on hardware development
    - d) Mainly related to 3D modeling for robots
  9. When gathering information to identify a computing problem, what should be collected?
    - a) The solution to the problem
    - b) Error messages, system changes, and specific problem triggers
    - c) Information on the history of computer science
    - d) Data on the latest hardware components
  10. How does logical thinking contribute to successful Rubik's cube solving?
    - a) Speeding up the movement of cube parts
    - b) Visualizing cube colors
    - c) Enabling efficient problem-solving strategies
    - d) Having no impact on solving the cube

## **B** Short Questions

1. Explain the importance of logical thinking in problem-solving.
2. Describe the steps involved in identifying computing problems.
3. Discuss the significance of algorithmic thinking in computer science and STEM fields.
4. Explain the role of algorithmic thinking in data analysis and how it can improve decision-making.
5. Describe the primary steps in identifying computing problems, focusing on the "Check for Updates" step.
6. How can concept maps be utilized to organize information effectively in educational settings?

## **C** Long questions.

1. Elaborate on the principles of computational thinking and provide examples of how it can be applied to real-world problem-solving.
2. Compare and contrast flowcharts and concept maps as tools for representing complex problems. Explain when each is most useful and provide examples.
3. In the context of computer science, how do algorithmic thinking and logical thinking complement each other in the problem-solving process? Provide a detailed example.
4. How can the systematic breakdown of a problem into its inputs, processes, and outputs aid in understanding and solving complex challenges? Use a practical scenario to illustrate.

## Standard









Students will identify and decompose simple and complex problems, create & evaluate appropriate solutions using computational approaches, and understand and apply common algorithms used in solving computational problems.

### Students' Learning Outcomes-1

 Solve simple and complex problems computationally.







#### Knowledge

##### Student will understand that

-  The importance of computational thinking and problem-solving in computer science.
-  Principles of computational thinking:
  - Logical thinking
  - Algorithmic thinking
-  How to identify steps in identifying a computing problem
-  How to identify the inputs, processes, and outputs of a problem
-  Different methods to design and construct a solution to a simple problem, such as flow charts, and/or concept maps.
-  Steps to produce simple diagrams to show:
-  The structure of a problem
-  Subsections and their links to other subsections

#### Skills

##### Students will be able to:

-  Explain the role of computational thinking in computer science.
-  Read and interpret simple computational problems.
-  Apply computational thinking principles to define and refine problems.
-  Identify the procedure appropriate to solving a problem.
-  Evaluate whether the order in which activities are undertaken will result in the required outcome.
-  Identify the inputs and outputs required in a solution.

## Knowledge 2.8

### Role of computational thinking in computer science

Computational thinking plays a crucial role in computer science as it serves as the fundamental framework for problem-solving and designing algorithms. Here's a breakdown of its roles within the realm of computer science:

#### Problem Solving:

Algorithmic thinking means breaking big problems into smaller, easier parts. It's like breaking a complicated task into simple steps to solve it more easily. Abstraction is another idea where you pay attention only to the important stuff and ignore things that don't matter. This helps computer experts simplify tricky real-world problems, making them easier to work with and solve.

#### Algorithm Design and Analysis:

Algorithm development is about using computational thinking to make step-by-step plans for doing things. It helps figure out the right order of steps to solve a problem well. Efficiency and optimization mean using computational thinking to make sure these plans work well. Computer scientists use this way of thinking to check and improve plans to make them faster and use as little as possible, like time or memory, while still getting the right answers.

#### Data Representation and Manipulation:

Computational thinking helps pick the right ways to organize data, like using arrays, linked lists, or trees, so it's easy to manage. It also helps computer experts work with data in really smart ways. They use this way of thinking to change and handle data well, using special methods and plans for studying it, changing it, and understanding what it means.

#### Automation and Modeling:

Computational thinking means making things that can-do tasks by themselves, like creating systems and software that can-do jobs without people having to do them all the time. It also includes making pretend versions of real-life situations on computers. This helps us look at these situations closely, predict what might happen, and make smart choices based on these pretend situations.

#### Logical and Critical Thinking:

Computational thinking helps us think logically and carefully to create strong and mistake-free systems. It's about checking and fixing





plans and programs step by step. It also teaches us to break big problems into smaller parts so we can understand and solve them better.

### Integration Across Disciplines:

Computational thinking isn't only for computer science—it's used in many subjects like biology, physics, and economics too! It's like a helpful way of thinking that gives us a system to solve problems neatly. Think of it as the main idea behind computer science, teaching us how to understand problems, make models, and solve them using special plans and tech tools. It helps us learn how to tackle big challenges step by step in an organized way.

## Knowledge 2.9

### Interpret simple computational problems

Students can effectively read and interpret simple computational problems by following a structured approach and practicing specific strategies. Here are steps and tips to help students comprehend and solve such problems:

#### a. Understand the Problem Statement:

Make sure to read the problem carefully so you understand what you need to do.

Look for the information given and what you're supposed to find or do as a result. That way, you know what you're starting with and what you're aiming to get at the end.

#### b. Define the Problem:

Be sure you know what the problem wants you to do and what you need to get. Explain the problem using your own words to make sure you really understand it. That way, you're clear on what you're supposed to do and what the goal is.

#### c. Break Down the Problem:

If the problem seems hard, try to split it into smaller pieces to make it easier. Find the important parts or smaller jobs you need to do to solve it. This way, you can manage each part step by step and make it simpler to figure out.

#### d. Identify Patterns or Strategies:

Try to remember if you've solved a similar problem before or look for problems that seem alike. Doing this can help you notice if there are any familiar tricks or ways to solve it. It's like finding connections between problems you've already figured out and the one you're working on now.

**e. Visualize or Draw Diagrams:**

Making pictures or diagrams can make it easier to understand the problem, especially in tasks like geometry or things that involve shapes and space. Drawing these visual representations helps you see the problem in a clearer way, making it simpler to figure out what to do.

**f. Develop a Plan:**

Help students pick the right way to solve the problem. This could mean choosing things like doing math, thinking logically, or using step-by-step plans. It's about figuring out which method or approach will work best to find the answer.

**g. Start Solving:**

Tell students to solve the problem by doing one thing at a time in order. This means breaking it into smaller steps and working through them one after the other. Also, remind them to stop and look at how they're doing from time to time to make sure they're going in the right direction.

**h. Verify and Reflect:**

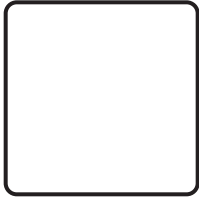
Once you find an answer, double-check to make sure it's what the problem asked for. See if your solution matches what was needed. Also, think back on how you solved the problem. Figure out what things went right and what things you could do better next time. This helps you learn and improve how you solve problems in the future.

**i. Practice and Persistence:**

Doing lots of different problems often can help you get used to different kinds of problems and make you better at solving them. Keep trying, even if it's hard at first. Remember, making mistakes or finding things tough at the beginning is normal when you're learning. Just keep going, and you'll get better!

**j. Seek Help and Collaboration:**

If you're having trouble understanding or solving a problem, don't



hesitate to ask for help from your teacher, friends, or online sources. Working together with classmates can be really helpful too because everyone might have different ideas that can make solving the problem easier.

## Knowledge 2.10

### Computational Thinking Principals



#### Skill:2.2

**Ability to understand, read, and interpret simple computational problems**

#### Computational Problem Solving

**Objective:** The objective of this activity is to help grade 9 students develop the ability to understand, read, and interpret simple computational problems through a series of engaging challenges. Scan the QR code for further details.

Computational thinking principles can significantly aid in defining and refining problems effectively. Here's how these principles can be applied in this context:

#### 1. Abstraction:

To figure out what's really important in the problem, try focusing on the main parts. Forget about extra details and think about the main question or what you're supposed to do. This way, you can see the most important stuff clearly without getting confused by things that don't matter as much.

#### Examples of Abstraction

- A world map is an abstraction of the earth in terms of longitude and latitude, helping us describe the location and geography of a place
- When we write a book report, we summarize and discuss only the theme or critical aspects of the book; it is an abstraction
- When we tell a story or describe a movie to our friends, why don't we describe every single detail of the story or movie?

#### 2. Decomposition:

Split the big problem into smaller parts that are easier to work with. Find the important things or smaller jobs that make up the whole problem. Make a list of these tasks or questions and work on them one by one. This helps to handle each part separately and solve the problem step by step.

#### Examples of Decomposition

- When we taste an unfamiliar dish and identify several ingredients based on the flavor, we are decomposing that dish into its individual ingredients
- When we give someone directions to our house, we are decomposing the process of getting from one place to another

(e.g., city, interstate, etc.)

- When we break a course project into several steps, we are decomposing the task into smaller, more manageable subtasks.

### 3. Algorithmic Thinking:

Make a plan with clear steps to solve each smaller part of the problem. Think about different ways you could solve it and pick the best way that works for you. Also, try to find things in the problem that are alike or follow a similar pattern. Finding these patterns can help you solve the problem more generally, which means it might work for similar problems too.

#### Examples of Algorithm Design

- When a cook writes a recipe for a dish, he or she is creating an algorithm that others can follow to replicate the dish
- When your friend writes down the instruction to get to her house, he or she is specifying a sequence of steps— that is, an algorithm—for you to follow (See Google maps!)
- When you follow an installation manual to assemble a bookshelf, you are executing an algorithm.

### 4. Pattern Recognition:

See if the problem looks like any other problems you've already solved before. Remember what you did in the past when you faced similar problems. Look for ways or tricks that worked before, as they might help with this problem too.

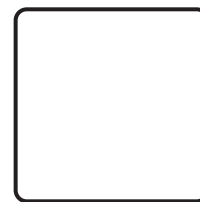
#### Examples of Pattern Recognition

- We look for patterns when choosing a registrar when we checkout
- Drivers look for patterns in traffic to decide whether and when to switch lanes.
- People look for patterns in stock prices to decide when to buy and sell
- Scientists look for patterns in data to derive theories and models.

### 5. Logical Reasoning:

Figure out how different parts of the problem are connected to each other. Think about how one thing might depend on or affect another thing in the problem. Use your logic and reasoning to understand how changing one part might make changes in other parts too.

#### Examples of Logical Reasoning



**Skill:2.3**

**Apply computational thinking principles .**

#### Applying Computational Thinking

**Objective:** The goal of this activity is to enable grade 9 students to apply computational thinking principles in defining and refining problems through a collaborative and hands-on exploration. Scan the QR code for further details.

- When troubleshooting, systematically identifying and eliminating possible causes of a problem.

## Knowledge 2.11

Identify the procedure appropriate to solving a problem

### 1. Understand the Problem:

First, read the problem carefully to understand what it's asking for and any rules or goals. Next, figure out the main parts of the problem, like what you need to start with, what you're supposed to get, and the main thing you're trying to do.

### 2. Plan and Strategize:

Break the problem into smaller pieces that are easier to work on. Then, pick a good way to solve each part of the problem based on what type of problem it is. It could mean using special plans, smart guesses, or trying different things until you find the answer.



### 3. Develop a Solution:

Make a clear plan with steps for each smaller part of the problem using algorithmic thinking. Then, follow this plan step by step to solve each part one after another. Stick to your plan and work through it carefully to find the solution.

### 4. Test and Evaluate:

Use the plan you made to solve the problem, making sure to do each step correctly. After finding the answer, check if it matches what the problem asked for. Make sure it's right and that you haven't missed anything important.

### 5. Refine and Iterate:

Look at how well your solution worked. If it doesn't match what the problem needed, figure out what might have gone off track. Then, change your plan, the steps you took, or the way you solved the problem based on what you found out. Keep doing this until you get closer to the right solution, trying different ways if needed.

### 6. Reflect and Improve:



Think about how you solved the problem and what you did well or could do better. It's important to understand what you did right and what didn't work so well. When you make mistakes, remember their chances to learn something new. Use these lessons to get better at solving problems in the future.

## 7. Documentation:

Write down how you solved the problem, including the way you did it, the steps you took, and the answer you found. Doing this helps you learn from what you did and share how you solved the problem with others. It's like making a note so you remember how you did it and can help someone else too.

## 8. Seek Feedback:

If it makes sense, ask friends, teachers, or people who know a lot about the topic for advice or thoughts. Getting help from others can give you new ideas and ways to solve the problems that you might not have thought about.

## 9. Iterate as Necessary:

If your first solution doesn't work well, don't worry! Try again and again using different ways until you find a solution that works best. It's like trying different ideas or plans until you get the right answer. Keep going until you find a solution that makes sense and solves the problem well.

## 10. Closure:

When you find a solution that works well, take a moment to look back at how you solved the problem. Make sure your solution matches what the problem was asking for at the start. Think about how you got there and make sure it all fits together with what the problem needed.

**Knowledge 2.12** | Activities Undertakes will result in the required outcome.

Evaluating whether the order of activities undertaken will result in the required outcome involves assessing the sequence and arrangement of steps taken to achieve a specific goal or solve a problem. Here's a structured approach to evaluating the order of activities:

### 1. Understanding the Requirements:

Make sure you have a clear understanding of what you want to



**Skill:2.4**

**Identify and match appropriate procedures to solve specific problems**

**Matching Problems with Solutions**

**Objective:** The objective of this activity is to help grade 9 students develop the ability to identify and match appropriate procedures to solve specific problems through a hands-on and interactive puzzle. Scan the QR code for further details.

achieve. Know the goal or outcome you're aiming for.

**Example:**

Before starting a school project, make sure you understand the assignment requirements. Knowing the goal, such as creating a presentation, helps in planning the steps to achieve it.

## **2. Sequencing Activities:**

Make sure you know exactly what you want to achieve. After that, write down and organize the steps you need to take to reach your goal. Check if some steps depend on others, meaning you have to finish one before starting the next. Understanding these connections will help you plan and achieve your goal more effectively.

**Example:**

If the goal is to bake a cake, sequencing activities involves organizing steps like gathering ingredients, mixing the batter, and baking. Understanding that you need to mix before baking helps in efficient planning.

## **3. Assessing Feasibility:**

Make sure to check if you have all the things you need, like time, people, tools, and information, for each step in your plan. Additionally, think about any rules or limitations that could affect the order in which you do things. Considering these factors will help you organize your activities more effectively and work towards your goal.

**Example:**

In a construction project, identifying the critical path involves determining tasks like foundation, framing, and roofing. These tasks must be completed on time for the project to finish as scheduled.

## **4. Critical Path Analysis:**

Find the most important path: Figure out the order of tasks that must be done on time for the project to finish as planned.

**Example:**

In a construction project, identifying the critical path involves determining tasks like foundation, framing, and roofing. These tasks must be completed on time for the project to finish as scheduled.

## **5. Analyzing Impact and Risk:**

Consider the impact of changing the order of tasks by thinking about

how it might affect what you want to achieve, the resources you have, and when you plan to finish. Also, look for possible challenges or issues that could come up with the way you've planned things. Identify any risks, like delays or difficulties, to be better prepared and ensure a smoother process.

**Example:**

When organizing a work schedule, consider alternative arrangements to improve efficiency. For instance, rearranging tasks might reduce commuting time, making the schedule more practical.

## **6. Considering Alternatives:**

If the way things are organized doesn't seem to be the best, explore different ways to arrange tasks. Look for options that might work better. Consider if changing the order of tasks could make things run more smoothly, reduce risks, or increase the chances of reaching your goal. It's like trying to find the best path to success by considering different possibilities and choosing the one that makes things work better.

**Example:**

When organizing a work schedule, consider alternative arrangements to improve efficiency. For instance, rearranging tasks might reduce commuting time, making the schedule more practical.

## **7 .Simulation or Modeling:**

Try Simulation Tools: If possible, use special computer programs or ways of creating models to see and understand how different ways of doing things might affect the outcome.

**Example:**

Using project management software to simulate a software development process helps identify potential bottlenecks and improve the overall workflow before implementation.

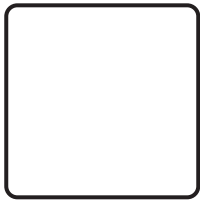
## **8. Decision Making:**

Review and Decide: After looking at everything, decide if the way tasks are currently organized is good or if changes are necessary.

**Example:**

After analyzing various ways to organize a product launch, a marketing team decides on a specific order of tasks based on factors like timing, resource availability, and impact on the target audience.

## **9. Iteration and Continuous Improvement:**



### Skill:2.5

**Evaluate whether the order in which activities are undertaken will result in the required outcome.**

#### Order for Success

**Objective:** The objective of this activity is to develop the ability of grade 9 students to evaluate and determine the optimal order of activities to achieve the desired outcome in various scenarios. Scan the QR code for further details.

Know that figuring out the best way to organize tasks takes time and may need to be done more than once. Keep looking at it, getting feedback, and making it better based on what you learn.

#### Example:

In software development, the iterative process involves regular testing and refining of code. Continuous improvement is essential for adapting to changing requirements and enhancing the final product.

## 10. Documentation and Monitoring:

Once you've decided on the best way to organize tasks, write it down and explain why you picked that order. This is like making a plan. After that, keep checking how things are going regularly. Make sure the plan you made is working and bringing you closer to what you want. It's important to keep an eye on progress to stay on track and make adjustments if needed.

#### Example:

After deciding on the best way to organize a research project, document the plan with task sequences. Regularly monitor progress by checking completed tasks against the plan to ensure the project stays on track. Adjustments can be made if any deviations are identified.

### Knowledge 2.13

Identify the inputs and outputs required in a solution.

Identifying inputs and outputs is essential when designing a solution to a problem. Inputs are the data, information, or resources required to initiate or carry out a process, while outputs are the results or outcomes produced by that process. Here's how to identify inputs and outputs in a solution:

### 1. Understand the Problem:

First, make sure you understand what the problem is asking you to do. Figure out what you need to solve or achieve. It's like knowing the main question or task that needs an answer or solution.

### 2. Identify Inputs:

Find out what information you need to start solving the problem. This might be things like numbers you already know, details you measure, or things people tell you. Also, figure out what tools or things you might need to solve the problem, like equipment or other helpful

resources.

### **3. Determine Processing or Action:**

Figure out the things you need to do to handle the information and get the solution. This means knowing the steps or actions you should take with the information you have to get the answer. Also, understand the math or changes you'll need to make to the information to get the right answer in the end.

### **4. Identify Outputs:**

Figure out what results you need to get to solve the problem. These could be things like the numbers or answers you calculate, pictures or reports you make, or even changes that happen because of solving the problem. It's like knowing what you're trying to find or create at the end to solve the problem.

### **5. Dependency Analysis:**

Think about how the things you start with (inputs) are connected to the things you want to get (outputs). Figure out how they depend on each other and how different pieces of information or steps are linked together. It's about seeing how changing one thing might affect another to get the right results.

### **6. Validation and Verification:**

Decide how to check if the information you start with (inputs) is correct, complete, and right for the problem. Then, figure out how to make sure that the answers or results you get (outputs) are what the problem needs. It's like having rules or tests to make sure everything is accurate and fits the problem well.

### **7. Iteration and Refinement:**

Realize that figuring out what you need at the start and what you're aiming for might change as you work on solving the problem. Keep improving and making them better based on what you learn and how things are going. It's like changing or making them clearer as you go along and get more information.

### **8. Documentation and Communication:**

Write down the things you're starting with and what you want to get. This helps everyone know what's needed to solve the problem and



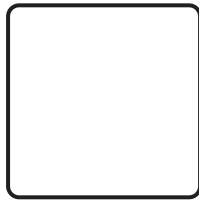


### Skill:2.6

**Identify and understand the inputs and outputs required in a solution**

#### Input-Output Explorer: Unraveling Solutions

**Objective:** This activity aims to help grade 9 students develop the ability to identify and understand the inputs and outputs required in a solution through hands-on exploration. Scan the QR code for further details.



makes sure everyone understands what's supposed to happen. It's like making a note so everyone knows what to do and what's expected.

### 9. Validation and Testing:

Try using different starting things and see if you get the right answers before saying you're finished. This means checking if different information gives the correct results. It's like trying out different scenarios to make sure everything works the way it should.

### 10. Adaptation to Changes:

Be ready to change the things you start with and what you want to get if the problem's needs or rules change while you're solving it. It's like being flexible and making adjustments if things are different from what you first thought.

## Exercise



### A

## Multiple Choice Questions

- Which principle of computational thinking involves focusing on the essential details while ignoring irrelevant information?
  - Algorithmic Thinking
  - Abstraction
  - Data Representation
  - Logical Reasoning
- What role does decomposition play in problem-solving?
  - Breaking down complex problems
  - Selecting data structures
  - Developing algorithms
  - Automating tasks
- Which step is crucial in the problem-solving process to ensure understanding and comprehension?
  - Break down the problem
  - Define the problem
  - Develop a plan
  - Verify and reflect
- Which aspect of computational thinking involves selecting appropriate data structures to organize and manage data efficiently?
  - Abstraction
  - Data Representation
  - Algorithmic Thinking
  - Logical Reasoning
- What is the primary purpose of automation in computational thinking?
  - Fostering critical thinking skills
  - Designing efficient algorithms
  - Reducing human effort and errors
  - Analyzing data structures
- Which computational thinking principle emphasizes creating computational models for analysis and prediction?
  - Decomposition
  - Modeling and Simulation
  - Logical Reasoning
  - Automation

## **B** Short Questions

1. Explain how computational thinking aids in the creation of algorithms.
2. Describe the significance of abstraction in problem-solving and give an example.
3. Why is it important to break down complex problems into smaller components during problem-solving?
4. Explain how logical reasoning is employed in computational thinking when designing algorithms.
5. Why is it important for computer scientists to understand and select appropriate data structures for problem-solving?
6. Describe how automation contributes to the efficiency of problem-solving processes in computer science.

## **C** Long questions.

1. Discuss the key principles of computational thinking and elaborate on their roles in problem-solving with examples.
2. Explain the step-by-step approach a student should follow when solving a computational problem. Provide examples to illustrate each step.
3. Detail the process involved in evaluating the order of activities in problem-solving. How does this assessment impact the outcome?
4. Discuss the interplay between abstraction and problem-solving, providing real-world examples of abstraction simplifying complex problems.
5. Explain the role of critical thinking in the context of computational thinking principles. How does it aid in algorithm development and problem-solving strategies?

“One Curriculum, One Nation”

## قومی ترانہ

پاک سرزمین شاد باد      کشورِ حسین شاد باد  
تو نشانِ غمِ عالی شان      ارضِ پاکستان  
مرکزِ یقین شاد باد  
پاک سرزمین کا نظام      قوتِ اخوتِ عوام  
قوم، ملک، سلطنت      پائندہ تابندہ باد  
شاد باد منزلِ مُراد  
پرچمِ ستارہ و ہلال      رہبرِ ترقی و کمال  
ترجمانِ ماضی شانِ حال      جانِ استقبال  
سایہٴ خدائے ذوالجلال

(حفیظ جالندھری)

