MAHARAJA INSTITUTE OF TECHNOLOGY MYSORE

Belawadi, S.R Patna, MANDYA -571477.

C PROGRAMMING FOR PROBLEM SOLVING
Subject Code: 18CPS13/23

NOTES

Academic Year: 2019-20

By:
Mr. Honnaraju B
Asst. Prof.
Dept. of CSE
MIT Mysore

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MODULE 1

Introduction to computer Hardware and software

1. INTRODUCTION TO COMPUTER

What is Computer?

A Computer is an electronic machine that can solve different problems, process data, store & retrieve data and perform calculations faster and efficiently than humans.

"The term computer is derived from the Latin term '**computare**', this means to calculate or programmable machine. **Computer cannot do anything without a Program**. It represents the decimal numbers through a string of binary digits. The Word 'Computer' usually refers to the Center Processor Unit plus Internal memory."

Basic Functions of Computer

There are four basic functions of a Computer.

1. Input
2. Processing
3. Output
4. Storage

1. **INPUT**- You input data i.e. you provide data; set of instructions. You input data through input devices which are keyboard, mouse, scanner, etc.

2. **PROCESSING**- The computer processes it i.e. it manipulates the data which is done by the C.P.U

3. **OUTPUT**- After processing the data the computer displays the result, it gives an output. Output devices are the monitor, in the case of visual output. Speakers, in the case of audio output, printers, etc.

4. **STORAGE**- We can save our data for future use in the CPU itself which is stored in the computer's ROM. There are several other storage devices also like removable disks, CDs, etc.

OR

2. A Computer can process data, pictures, sound and graphics. They can solve highly complicated problems quickly and accurately. A block diagram of the computer as shown in Fig 1. Computer performs basically five major computer operations or functions irrespective of their size and make. These are

1) it accepts data or instructions by way of input,
2) it stores data,
3) it can process data as required by the user,
4) it gives results in the form of output, and
5) it controls all operations inside a computer.

1. Input: This is the process of entering data and programs in to the computer system. You should know that computer is an electronic machine like any other machine which takes as inputs raw data and performs some processing giving out processed data. Therefore, the input unit takes data from us to the computer in an organized manner for processing.

2. Storage: The process of saving data and instructions permanently is known as storage. Data has to be fed into the system before the actual processing starts. It is because the processing speed of Central Processing Unit (CPU) is so fast that the data has to be provided to CPU with the same speed. Therefore the data is first stored in the storage unit for faster access and processing. This storage unit or the primary storage of the computer system is designed to do the above functionality. It provides space for storing data and instructions.

The storage unit performs the following major functions:

- All data and instructions are stored here before and after processing.
- Intermediate results of processing are also stored here.

3. Processing: The task of performing operations like arithmetic and logical operations is called processing. The Central Processing Unit (CPU) takes data and instructions from the storage unit and makes all sorts of calculations based on the instructions given and the type of data provided. It is then sent back to the storage unit.

4. Output: This is the process of producing results from the data for getting useful information. Similarly the output produced by the computer after processing must also be kept somewhere inside the computer before being given to you in human readable form. Again the output is also stored inside the computer for further processing.

5. Control: The manner how instructions are executed and the above operations are performed. Controlling of all operations like input, processing and output are performed by control unit. It takes care of step by step processing of all operations inside the computer.

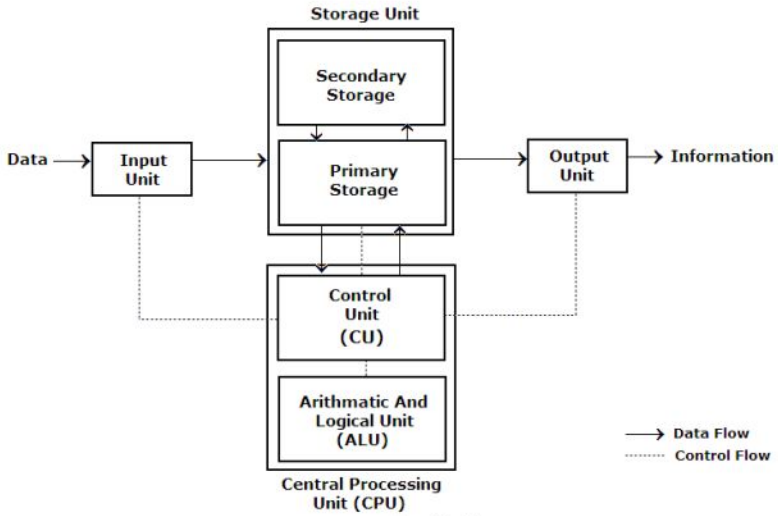


Figure 1: Block diagram of computer

2. GENERATION OF COMPUTER

The history of the computer goes back several decades however and there are five definable generations of computers.

Each generation is defined by a significant technological development that changes fundamentally how computers operate – leading to more compact, less expensive, but more powerful, efficient and robust machines.

- 1940 – 1956: **First Generation** – Vacuum Tubes. These early computers used vacuum tubes as circuitry and **magnetic drums** for **memory**.
- 1956 – 1963: **Second Generation** – Transistors.
- 1964 – 1971: **Third Generation** – Integrated Circuits.
- 1972 – 2010: **Fourth Generation** – Microprocessors.
- 2010 : **Fifth Generation** - Artificial Intelligence

(i) The First Generation:

The first generation computers made use of:

- Vacuum tube technology,
- Punched cards for data input,
- Punched cards and paper tape for output,
- Machine Language for writing programs,
- Magnetic tapes and drums for external storage.

The computers of the first generation were very bulky and emitted large amount of heat which required air conditioning. They were large in size and cumbersome to handle. They had to be manually assembled and had limited commercial use. The concept of operating systems was not known at that time. Each computer had a different binary coded program called a machine language that told it how to operate.

Punched cards *Paper tape* *Vacuum tube*

Figure 2: The first Generation Computer technology

Charles Babbage a British mathematician at Cambridge University invented the first **analytical engine or difference engine.** This machine could be programmed by instructions coded on punch cards and had mechanical memory to store the results. For his contributions in this field **Charles Babbage** is known as **'the father of modern digital computer.**

ENIAC (Electronic Numeric Integrator and Calculator) –

The first all-electronic computer was produced by a partnership between the US Government and the University of Pennsylvania. It was built using 18,000 vacuum tubes, 70,000 resistors and 1,500 relays and consumed 160 kilowatts of electrical power. The ENIAC computed at speed about thousand times faster than Mark I.

(ii) Second Generation

In the second generation computers:

- Vacuum tube technology was replaced by transistorized technology,
- Size of the computers started reducing,
- Assembly language started being used in place of machine language,
- Concept of **stored program** emerged,
- High level languages were invented.

This was the generation of **Transistorized Computers.** Vacuum tubes were replaced by transistors. As a result, the size of the machines started shrinking. These computers were smaller, faster, more reliable and more energy efficient. The first transistorized computer was TX-0. Typical computers of the second generation were the IBM 1400 and 7000 series, Honeywell 200 and General Electric.

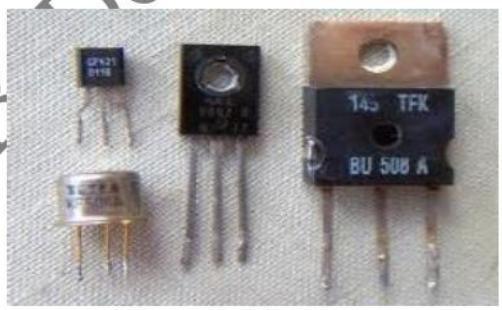


Figure 3: Transistors

The stored program concept and programming languages gave the computers flexibility to finally be cost effective and productive for business use. High level languages like COBOL, FORTRAN and AL- GOL were developed. Computers started finding vast and varied applications. The entire software industry began with the second generation computers.

(iii) The Third Generation:

The third generation computers were characterized by:

- Use of Integrated circuits,
- Phenomenal increase in computation speed,
- Substantial reduction in size and power consumption of the machines,
- Use of magnetic tapes and drums for external storage,
- Design-of Operating systems and new higher level languages,
- Commercial production of computers.

This generation was characterized by the invention of **Integrated Circuits (ICs).** The 1C combined electronic components onto a small chip which was made from quartz.

Figure 4: Integrated Circuits (ICs)

Later, even more components were fitted onto a single chip, called a **semiconductor.** Operating systems were designed which allowed the machine to run many different programs at once. A central program monitored and co-ordinate the computer s memory. Multiprogramming was made possible, whereby the machine could perform several jobs at the same time. Computers achieved speeds of executing millions of instructions per second. Commercial production became easier and cheaper. Higher level languages like Pascal and Report Program Generator (RPG) were introduced and applications oriented languages like FORTRAN, COBOL, and PL/1 were developed.

(iv)The Fourth Generation:

The general features of the fourth generation computers were:

- Use of Very Large Scale Integration,
- Invention of microcomputers,
- Introduction of Personal Computers,
- Networking,
- Fourth Generation Languages.

Figure 5: VLSI Chip

The third generation computers made use of 'Integrated Circuits that had 10-20 components on each chip, this was **Small Scale Integration (SSI).** The Fourth Generation realized **Large Scale Integration (LSI)** which could fit hundreds of components on one chip and **Very Large Scale integration (VLSI)** which squeezed thousands of components on one chip. The Intel 4004 chip, located all the components of a computer (central processing unit, memory, input and output controls) on a single chip and microcomputers were introduced. Higher capacity storage media like magnetic disks were developed. Fourth generation languages emerged and applications software's started becoming popular.

(v) The Fifth Generation:

Defining the fifth generation computers is somewhat difficult because the field is still in its infancy. The computers of tomorrow would be characterized by Artificial Intelligence (AI). An example of Al is Expert Systems. Computers could be developed which could think and reason in much the same way as humans. Computers would be able to accept spoken words as input (voice recognition).

Many advances in the science of computer design and technology are coming together to enable the creation of fifth generation computers. Two such advances are **parallel processing** where many CPUs work as one and advance in **superconductor technology** which allows the flow of electricity with little or no resistance, greatly improving the speed of information flow.

3. CLASSIFICATION OF COMPUTERS

Computers are broadly classified into two categories depending upon the logic used in their design as:

Digital Computers:

Class of devices capable of solving problems by processing information in discrete forms. It operates on data, including magnitudes, letters and symbols that are expressed in binary code, i.e., using only the two digits 0 or 1. By counting, comparing and manipulating these digits or their combinations according to a set

of instructions held in its memory, a digital computer can perform uh tasks as to control industrial processes and regulate the operations of machines; analyse and organize vat amount of business data; and simulate the behaviour of dynamic system.

A typical digital computer system has four basic functional elements: (1) input-output equipment, (2) main memory (3) control unit and (4) arithmetic unit.

Analog computers:

An analog is a computer which is used to process analog data. Analog computers store data in a continuous form of physical quantities and perform calculations with the help of measures. It is quite different from the digital computer, which makes use of symbolic numbers to represent results. Analog computers are excellent for a situation which requires data to be measured directly without converting into numerals or codes.

Analog computers were the earliest computer machines developed and were among the most complicated machines for analog computation and process control. Analog data is not discrete, but rather is of a continuous nature. Examples of such data are pressure, temperature, voltage, speed and weight.

There are certain advantages associated with analog computers. Real-time operation and simultaneous computation is possible with the help of analog computers. Analog computers can also provide the insight of the problems and errors in case of analog issues for user.

Hybrid Computers:

Hybrid Computers are a combination of Analog and Digital computers. It combines the best features of both types of computers, i.e., it has the speed of analog computer and the memory and accuracy of digital computer. Hybrid computers are used mainly in specialized applications where both kinds of data need to be processed. Therefore, they help the user to process both continuous and discrete data. For example, a petrol pump contains a processor that converts fuel flow measurement into quantity and price values. In hospital Intensive Care Unit(ICU), an analog device is used which measures patient's blood pressure and temperature, etc., which are then converted and displayed in the form of digits. Hybrid computers, for example, are used for scientific calculations, in defence and radar systems.

3.1 Different Types of Computer

Since the advent of the first computer different types and sizes of computers are offering different services. Computers can be as big as occupying a large building and as small as a laptop or a microcontroller in mobile & embedded systems.

Computers can be generally classified by size and power as follows, though there is considerable overlap:

1. **Personal computer:** A small, single-user computer based on a microprocessor.
2. **Workstation:** A powerful, single-user computer. A workstation is like a personal computer, but it has a more powerful microprocessor and, in general, a higher-quality monitor.
3. **Minicomputer:** A multi-user computer capable of supporting up to hundreds of users simultaneously.
4. **Mainframe:** A powerful multi-user computer capable of supporting many hundreds or thousands of users simultaneously.
5. **Supercomputer:** An extremely fast computer that can perform hundreds of millions of instructions per second.

Supercomputer and Mainframe

Supercomputer is a broad term for one of the fastest computers currently available.

- Supercomputers are very expensive and are employed for specialized applications that require immense amounts of mathematical calculations (number crunching). For example, weather forecasting requires a supercomputer.
- Other uses of supercomputers scientific simulations, (animated) graphics, fluid dynamic calculations, nuclear energy research, electronic design, and analysis of geological data (e.g. in petrochemical prospecting). Perhaps the best known supercomputer manufacturer is Cray Research. Ex: CRAY-2, NEC - 500, PARAM.

Mainframe was a term originally referring to the cabinet containing the central processor unit or "main frame" of a room-filling Stone Age batch machine. After the emergence of smaller "minicomputer" designs in the early 1970s, the traditional big iron machines were described as "mainframe computers" and eventually just as mainframes.

- Nowadays a Mainframe is a very large and expensive computer capable of supporting hundreds, or even thousands, of users simultaneously. The chief difference between a supercomputer and a mainframe is that a supercomputer channels all its power into executing a few programs as fast as possible, whereas a mainframe uses its power to execute many programs concurrently.
- In some ways, mainframes are more powerful than supercomputers because they support more simultaneous programs. But supercomputers can execute a single program faster than a mainframe. The distinction between small mainframes and minicomputers is vague, depending really on how the manufacturer wants to market its machines.

Minicomputer

It is a midsize computer. In the past decade, the distinction between large minicomputers and small mainframes has blurred, however, as has the distinction between small minicomputers and workstations. But in general, a minicomputer is a multiprocessing system capable of supporting from up to 200 users simultaneously.

Workstation

It is a type of computer used for engineering applications (CAD/CAM), desktop publishing, software development, and other types of applications that require a moderate amount of computing power and relatively high quality graphics capabilities. Workstations generally come with a large, high-resolution graphics screen, at large amount of RAM, built-in network support, and a graphical user interface. Most workstations also have a mass storage device such as a disk drive, but a special type of workstation, called a diskless workstation, comes without a disk drive. The most common operating systems for workstations are UNIX and Windows NT. Like personal computers, most workstations are single-user computers. However, workstations are typically linked together to form a local-area network, although they can also be used as stand-alone systems.

N.B.: In networking, workstation refers to any computer connected to a local-area network. It could be a workstation or a personal computer.

Personal computer:

It can be defined as a small, relatively inexpensive computer designed for an individual user. In price, personal computers range anywhere from a few hundred pounds to over five thousand pounds. All are based on the microprocessor technology that enables manufacturers to put an entire CPU on one chip. Businesses use personal computers for word processing, accounting, desktop publishing, and for running spread sheet and database management applications. At home, the most popular use for personal computers is for playing games and recently for surfing the Internet.

Today, the world of personal computers is basically divided between Apple Macintoshes and PCs. The principal characteristics of personal computers are that they are single-user systems and are based on microprocessors.

Personal Computer Types

Actual personal computers can be generally classified by size and chassis / case. The chassis or case is the metal frame that serves as the structural support for electronic components. Every computer system requires at least one chassis to house the circuit boards and wiring. The chassis also contains slots for expansion boards. If you want to insert more boards than there are slots, you will need an expansion chassis, which provides additional slots.

Tower model

The term refers to a computer in which the power supply, motherboard, and mass storage devices are stacked on top of each other in a cabinet. This is in contrast to desktop models, in which these components are housed in a more compact box. The main advantage of tower models is that there are fewer space constraints, which makes installation of additional storage devices easier.

Desktop model

A computer designed to fit comfortably on top of a desk, typically with the monitor sitting on top of the computer. Desktop model computers are broad and low, whereas tower model computers are narrow and tall. Because of their shape, desktop model computers are generally limited to three internal mass storage devices. Desktop models designed to be very small are sometimes referred to as **slimline models**.

Notebook computer

An extremely lightweight personal computer. Notebook computers typically weigh less than 6 pounds and are small enough to fit easily in a briefcase. Aside from size, the principal difference between a notebook computer and a personal computer is the display screen. Notebook computers use a variety of techniques, known as flat-panel technologies, to produce a lightweight and non-bulky display screen. The quality of notebook display screens varies considerably. In terms of computing power, modern notebook computers are nearly equivalent to personal computers. They have the same CPUs, memory capacity, and disk drives. However, all this power in a small package is expensive. Notebook computers cost about twice as much as equivalent regular-sized computers. Notebook computers come with battery packs that enable you to run them without plugging them in. However, the batteries need to be recharged every few hours.

Laptop computer

A small, portable computer -- small enough that it can sit on your lap. Nowadays, laptop computers are more frequently called notebook computers.

Subnotebook computer

A portable computer that is slightly lighter and smaller than a full-sized notebook computer. Typically, subnotebook computers have a smaller keyboard and screen, but are otherwise equivalent to notebook computers.

Hand-held computer

A portable computer that is small enough to be held in one's hand. Although extremely convenient to carry, handheld computers have not replaced notebook computers because of their small keyboards and

screens. The most popular hand-held computers are those that are specifically designed to provide PIM (personal information manager) functions, such as a calendar and address book. Some manufacturers are trying to solve the small keyboard problem by replacing the keyboard with an electronic pen. However, these pen-based devices rely on handwriting recognition technologies, which are still in their infancy. Hand-held computers are also called PDAs, palmtops and pocket computers.

Palmtop

A small computer that literally fits in your palm. Compared to full-size computers, palmtops are severely limited, but they are practical for certain functions such as phone books and calendars. Palmtops that use a pen rather than a keyboard for input are often called hand-held computers or PDAs. Because of their small size, most palmtop computers do not include disk drives. However, many contain PCMCIA slots in which you can insert disk drives, modems, memory, and other devices. Palmtops are also called PDAs, hand-held computers and pocket computers.

PDA

Short for personal digital assistant, a handheld device that combines computing, telephone/fax, and networking features. A typical PDA can function as a cellular phone, fax sender, and personal organizer. Unlike portable computers, most PDAs are pen-based, using a stylus rather than a keyboard for input. This means that they also incorporate handwriting recognition features. Some PDAs can also react to voice input by using voice recognition technologies. The field of PDA was pioneered by Apple Computer, which introduced the Newton Message Pad in 1993. Shortly thereafter, several other manufacturers offered similar products. To date, PDAs have had only modest success in the marketplace, due to their high price tags and limited applications. However, many experts believe that PDAs will eventually become common gadgets.
PDAs are also called palmtops, hand-held computers and pocket computers.

4. BITS, BYTES and WORDS

Bits:

A bit (short for *binary digit*) is the smallest unit of data in a computer. A bit has a single binary value, either 0 or 1. Although computers usually provide instructions that can test and manipulate bits, they generally are designed to store data and execute instructions in bit multiples called bytes. In most of computer systems, there are eight bits in a byte.

Byte:

A byte is a unit of measurement used to measure data. One byte contains eight binary bit, or a series of eight zeros and ones. Therefore, each byte can be represents 256 different values. The byte was originally developed to store a single character, since 256 values is sufficient to represent all standard lowercase an uppercase letters, numbers, and symbols.

Half a byte (four bits) is called a nibble. In some systems, the term octet is used for an eight-bit unit instead of byte.

- 1 Bit = Binary Digit
- 8 Bits = 1 Byte
- 1024 Bytes = 1 Kilobyte
- 1024 Kilobytes = 1 Megabyte
- 1024 Megabytes = 1 Gigabyte
- 1024 Gigabytes = 1 Terabyte
- 1024 Terabytes = 1 Petabyte

Word:

A word is the natural size with which the processor is handling data, also referred to as the register size. The most common word sizes encountered today are 8, 16, 32 and 64 bits, but other sizes are possible. The word, by contrast, is biggest chunk of bits with which a processor can do processing at time.
The word size refers to the number of bits processed be a computer's CPU in one go. Data bus size, instruction size, address size are usually multiples of the word size.
The word size is what the majority of operations work with.

5. APPLICATIONS OF COMPUTERS

Today computers find widespread applications in all activities of the modern world. Some of the major application areas include:

Scientific, Engineering and Research:

This is the major area where computers find vast applications. They are used in areas which require lot of experiments, mathematical calculations, weather forecasting, and complex mathematical and engineering applications. Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) help in designing robotics, automobile manufacturing, automatic process control devices etc.

Business:

Record keeping, budgets, reports, inventory, payroll, invoicing, accounts are all the areas of business and industry where computers are used to a great extent. Database management is one of the major areas where computers are used on a large scale. The areas of application here include banking, airline reservations, etc. where large amounts of data need to be updated, edited, sorted, and searched from large databases.

Medicine:

Computerized systems are now in widespread use in monitoring patient data like, pulse rate, blood pressure etc. resulting in faster and accurate diagnosis. Modern day medical equipment's are highly computerized today. Computers are also widely used in medical research.

Information:

This is the age of information. Television, Satellite communication, Internet, networks are all based on computers.

Education:

The use of computers in education is increasing day by day. The students develop the habit of thinking more logically and are able to formulate problem solving techniques. CDs on a variety of subjects are available to impart education. On line training programs for students are also becoming popular day by day. All the major encyclopaedias, dictionaries and books are now available in the digital form and therefore are easily accessible to the student of today. Creativity in drawing, painting, designing, decoration, music etc. can be well developed with computers.

Games and Entertainment:

Computer games are popular with children and adults alike. Computers are nowadays also used in entertainment areas like movies, sports, advertising etc.

6. ADVANTAGES AND DISADVANTAGES OF COMPUTERS

Advantages of Computers:

Speed:

The speed of a computer is measured in terms of the number of instructions that it can perform or execute in a second. The speeds of computers are measured in milliseconds (10~3 sec), micro-seconds (10*6 sec), and nano-seconds (10~9sec).

Accuracy:

Computers are very accurate. They are capable of executing hundreds of instructions without any errors. They do not make mistakes in their computations. They perform each and every calculation with the same accuracy.

Efficiency

The efficiency of computers does not decrease with age. The computers can perform repeated tasks with the same efficiency any number of times without exhausting there selves.

Storage Capability

Computers are capable of storing large amounts of data in their storage devices. These devices occupy very less space and can store millions of characters in condensed forms.

Versatility

Computers are very versatile. They are capable not only of performing complex mathematical tasks of science and engineering, but also other non-numerical operations fielding air-line reservation, electricity bills, data base management etc.

Limitations of Computers:

Although the computers of today are highly intelligent and sophisticated they have their own limitations. The computer cannot think on its own, since it does not have its own brain. It can only do what is has been programmed to do. It can execute only those jobs that can be expressed as a finite set of instructions to achieve a specific goal. Each of the steps has to be clearly defined. The computers do not learn from previous experience nor can they arrive at a conclusion without going through all the intermediate steps. However the impact of computers on today's society in phenomenal and they are today an important part of the society.

7. COMPONENTS OF A COMPUTER SYSTEM

The basic parts of computer system are:

- Input Unit
- The Central Processing Unit
- Output Unit

The Input Unit:

Input devices are the devices which are used to feed programs and data to the computer. The input system connects the external environment with the computer system. The input devices are the means of communication between the user and the computer system. Typical input devices include the keyboard, floppy disks, mouse, microphone, light pen, joystick, magnetic tapes etc. The way in which the data is feed into the computer through each of these devices is different. However, a computer can accept data only in a specific form. Therefore these input devices transform the data fed to them, into a form which can be accepted by the computer. These devices are a means of communication and inter1 station between the user and the computer systems.

Thus the functions of the input unit are:

- accept information (data) and programs.
- convert the data in a form which the computer can accept.
- provide this converted data to the computer for further processing.

The Central Processing Unit:

This is the brain of any computer system. The central processing unit or CPU is made of three parts:

- The control unit.
- The arithmetic logic unit
- The primary storage unit

The Control Unit:

The Control Unit controls the operations of the entire computer system. The control unit gets the instructions from the programs stored in primary storage unit interprets these instruction are subsequently directs the other units to execute the instructions. Thus it manages and coordinates the entire computer system.

The Arithmetic Logic Unit:

The **Arithmetic Logic Unit** (ALU) actually executes the instructions and performs all the calculations and decisions. The data is held in the primary storage unit and transferred to the ALU whenever needed. Data can be moved from the primary storage to the arithmetic logic unit a number of times before the entire processing is complete. After the completion, the results are sent to the output storage section and the output devices.

The Primary Storage Unit:

This is also called as **Main Memory.** Before the actual processing starts the data and the instructions fed to the computer through the input units are stored in this primary storage unit. Similarly, the data which is to be output from the computer system is also temporarily stored in the primary memory. It is also the area where intermediate results of calculations are stored. The main memory has the storage section that holds the computer programs during execution. Thus the primary unit:

- Stores data and programs during actual processing
- Stores temporary results of intermediate processing
- Stores results of execution temporarily

Output Unit:

The output devices give the results of the process and computations to the outside world. The output units accept the results produced by the computer, convert them into a human readable form and supply them to the users. The more common output devices are printers, plotters, display screens, magnetic tape drives etc.

8. COMPUTER HARDWARE

Computer hardware is the collection of physical parts of a computer system. This includes the computer case, monitor, keyboard, and mouse. It also includes all the parts inside the computer case, such as the hard disk drive, motherboard, video card, and many others. Computer hardware is what you can physically touch.

- **Arithmetic/logic unit**

  Contains the electronic circuitry necessary to perform arithmetic and logical operations on data.

Communications Devices

Enable a computer to connect to other computers. Devices that enable a computer to connect to other components; includes modems and network interface cards.

Control Unit

The component in any computing system that works in coordinatin with the central processing unit to instruct, maintain and control the flow of information.

Central Processing Unit (CPU)

The component in any computing system that represents the circuitry necessary to interpret and execute program instructions, it consists of the Control Unit, arithmetic/logic unit and the controller.. It is the corollary to the brain in organic systems.

Hardware

Equipment that inputs, processes, outputs, and stores data. It consists of input devices, a system unit, output devices, storage devices, and communication devices.

Input Devices

Any computer peripheral used to enter data and/or control signals into a computer system. Some devices, such as modems, are capabile of both input as well as output.

Laptop or Notebook or Netbook

A portable, integrated device in which the central processing unit (CPU), storage (hard disk and/or memory), input system (keyboard, trackpad, etc.), display and various data input and output channels (USB, Firewire, Ethernet, WiFi, etc.) are combined in a self-contained unit. Laptop or Notebook or Netbook computers are generally smaller than desktop computers, lending to their portability.

Machine cycle

Machine Cycle or Processor Cycle or Instruction Cycle: the most basic logical mode of operation in a central processing unit (CPU). It consists of four steps that are "executed" continuously at a very high rate of speed: fetch, decode, execute and store. Only one execution cycle per machine cycle may be performed.

Memory

Integrated circuits that temporarily store program instructions and data that can be retrieved. Basic unit of memory is a byte.

RAM (Random Access Memory) – a volatile form of memory, RAM generally functions as a computers "desktop" - the space in which data that is actively under review and/or manipulation can be processed. As a result and as a general rule, the more RAM with which a computer is fitted, the more and faster data can be viewed and manipulated. RAM needs to be cyclically "refreshed" from an outside power source in order to maintain the information contained therein. When external power is removed, the data contents held in RAM disappears. For this reason, RAM is sometimes referred to as short term memory.

ROM (Read Only Memory) – a non-volatile form of memory, ROM stores data that does not commonly change, like startup instructions and data used when a computer if first turned on.

CMOS –used to store information about the computer system, such as the amount of memory, the type of keyboard and monitors, and the type and capacity of disk drives.

Output Devices

Devices that convert the results of processed data into a form that can be used and understood by the user. A computer display is an example of an output device, as is a printer. Whereas a computer display uses a screen to present visual information in virtual form, a printer produces hardcopy - a tangible form of the data or information. Audio speakers are another form of output device, converting electronic programming into human-audible sound. Some devices are capable of being output as well as input devices.

Input Devices:

An input device is any hardware device that sends data to a computer, allowing you to interact with and control it. The most commonly used or primary input devices on a computer are the keyboard and the mouse. However, there are dozens of other devices that can also be used to input data into the computer.

- **Examples of Input Devices**
- Biofeedback Input Devices—special equipment like gloves, body suits, and eyeglasses to translate movements, temperature, or skin-based electrical signals
- Digital Camera—record photographs in the form of digital data that can be stored on a computer.
- Digitizer—converts points, lines, and curves from a sketch, drawing, or photograph to digital impulses and transmits them to a computer
- Electronic Whiteboard—captures anything drawn on special whiteboard
- Joystick—uses the movement of a vertical stem to direct the pointer. These are often used with computer games and have buttons you can press to activate events, depending upon the software.
- Graphics Tablet—similar to a digitizer, but it also contains unique characters and commands that can be generated automatically by the person using the tablet
- Image Scanner (page scanner)– an input device that can electronically capture an entire page of text or images such as photographs or art work.
- Keyboard—most commonly used input device. You enter data by pressing keys on the keyboard
- Light Pen—used by touching it on the display to create or modify graphics
- Microphone—used to record sound.

- Modem—a device that converts data into a form suitable for both receipt and transmission by wire or radio such that it can be reconstructed at the destination point.
- Mouse—a small palm-sized input device that you move across a flat surface to control the movement of the pointer on the screen
- Pen Input—used to (1) input data using hand-written characters and shapes that the computer can recognize (2) as a pointing device like a mouse to select items on the screen, and (3) to gesture (special symbols made with the pen that issue a command)
- Optical Recognition Devices—use a light source to read codes, marks, and characters and convert them into digital data that can be processed by a computer
- Pointing Stick (trackpoint or isometric pointing device)—a small device shaped like a pencil eraser that moves the insertion point as pressure is applied to the device
- Terminals—consist of a keyboard and a screen - commonly used for special purpose input such as POS (Point-of Sale) information entry.
- Touchpad (trackpad)—a flat rectangular surface that senses the movement of a finger on its surface to control the movement of the insertion point
- Touch Screen—allows you to touch areas of the screen with your fingers to enter data.
- Trackball—A pointing device like a mouse only with the ball on the top of the device instead of the bottom

Output Devices:

An output device is any device used to send data from a computer to another device or user. Most computer data output that is meant for humans is the form of audio of video. Thus, most output devices used by humans are in these categories. Examples include monitors, printers, etc.

- **Examples of Output Devices**
  - Computer display or monitor
  - Plotter- Designed for line drawing; often used for computer-aided design; some units can handle large paper sizes
  - Modem—a device that converts data into a form suitable for both receipt and transmission by wire or radio such that it can be reconstructed at the destination point.
  - Multifunction Devices—Combines printer, fax, scanner, and copier
  - Printers: A printer is an external output device that takes data from a computer and generates output in the form of graphics / text on a paper.

  Different types of printers:

  1. Impact printers
     - Dot-Matrix Printers
     - Daisy-wheel printers
     - Line printers
     - Drum printer
     - Chain printers
     - Band printers
  2. Non-impact printers
     - Ink-jet printers
     - Laser printers
  - Projectors

9. MEMORY

Memory is defined as the place or locations where the data and/or the instructions are stored. Memory is required for the following purpose:

- Memory is required to store both the data and instructions given to the computer to achieve a specific task.
- Memory is also required to store the partial or complete results during processing.

The memory can be broadly classified as shown below:

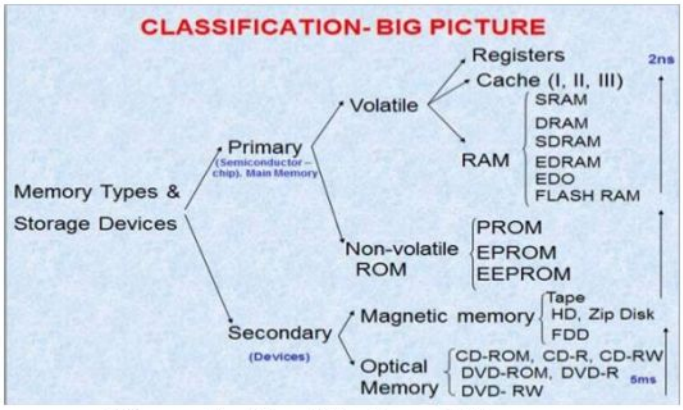


Figure 6: Classification of Memory

9.1 PRIMARY MEMORY

Primary memory is computer memory that a processor or computer accesses first or directly. It allows a processor to access running execution applications and services that are temporarily stored in a specific memory location.

Primary memory is also known as **primary storage or main memory or internal memory**. This primary memory allows the CPU to store and retrieve data quickly.

Primary memory is a computer system's volatile storage mechanism. It may be random access memory (RAM), cache memory or data buses, but is primarily associated with RAM.

As soon as a computer starts, primary memory loads all running applications, including the base operating system (OS), user interface and any user-installed and running software utility. A program/application that is opened in primary memory interacts with the system processor to perform all application-specific tasks. Primary memory is considered faster than secondary memory.

Ex: RAM, ROM, EPROM, Flash Memory

Advantages:

- The data or instructions can be accessed at very high speed
- To execute a program, at first step, all the instructions or data used by CPU have to be loaded into main memory. Otherwise, we cannot execute the program.

Disadvantage:

- The primary memory is volatile i.e, the data or instructions stored in the primary memory will be lost as soon as the computer is turned off and the data cannot be retrieved back.
- Very expensive
- Huge amount of data cannot be stored.

9.2 SECONDARY MEMORY

The memory that can store all the data and instructions even when the computer is turned off is called secondary memory or auxiliary memory. This type of memory is normally called secondary storage. The secondary storage is used to store the data and instructions permanently.

Secondary memory (or secondary storage) is the slowest and cheapest form of memory. It cannot be processed directly by the CPU. It must first be copied into primary storage (also known as RAM).

Secondary memory devices include magnetic disks like hard drive sand floppy disks ; optical disks such as CDs and CDROMs ; and magnetic tapes, which were the first forms of secondary memory.

Advantages:

- The data stored in the secondary memory are not lost even when the computer is turned off
- Huge amount of data can be stored.
- Very cheap
- They are portable i.e, they can be easily moved from one place to another place.

Disadvantages:

- Accessing of data is very slow.

Difference between Primary memory and Secondary Memory

BASIS FOR COMPARISON	PRIMARY MEMORY	SECONDARY MEMORY
Basic	Primary memory is directly accessible by Processor/CPU.	Secondary memory is not directly accessible by CPU.
Altered Name	Main memory.	Auxiliary memory.
Data	Instructions or data to be currently executed are copied to main memory.	Data to be permanently stored is kept in secondary memory.
Volatility	Primary memory is usually volatile.	Secondary memory is non-volatile.
Formation	Primary memories are made of semiconductors.	Secondary memories are made of magnetic and optical material.
Access Speed	Accessing data from primary memory is faster.	Accessing data from secondary memory is slower.
Access	Primary memory is accessed by the data bus.	Secondary memory is accessed by input-output channels.
Size	The computer has a small primary memory.	The computer has a larger secondary memory.
Expense	Primary memory is costlier than secondary memory.	Secondary memory is cheaper than primary memory
Memory	Primary memory is an internal memory.	Secondary memory is an external memory.

Volatile Memory:

Volatile memory, in contrast to non-volatile memory, is computer memory that requires power to maintain the stored information; it retains its contents while powered on but when the power is interrupted, the stored data is quickly lost.

Volatile memory has several uses including as main memory. In addition to usually being faster than forms of mass storage such as a hard disk drive, volatility can protect sensitive information as it becomes unavailable on power-down. Most of the general-purpose random-access memory (RAM) is volatile.

RAM (Random Access Memory):

RAM is a semiconductor memory. RAM (*pronounced ramm*) is an acronym for ***r**andom **a**ccess **m**emory*, a type of computer memory that can be accessed randomly; that is, any byte of memory can be accessed without touching the preceding bytes.

The data and the programs that are entered from the keyboard are stored temporarily in RAM and so on, RAM is a temporary storage medium. RAM is found in servers, PCs, tablets, smartphones and other devices, such as printers.

The data can be stored in the RAM as long as the computer is on.

Main Types of RAM

There are two main types of RAM:

1. DRAM (Dynamic Random Access Memory)
2. SRAM (Static Random Access Memory)

DRAM (Dynamic Random Access Memory) – The term dynamic indicates that the memory must be constantly refreshed or it will lose its contents. DRAM is typically used for the main memory in computing devices. If a PC or smartphone is advertised as having 4-GB RAM or 16-GB RAM, those numbers refer to the DRAM, or main memory, in the device.

SRAM (Static Random Access Memory) – While DRAM is typically used for main memory, today SRAM is more often used for system cache. SRAM is said to be static because it doesn't need to be refreshed, unlike dynamic RAM, which needs to be refreshed thousands of times per second. As a result, SRAM is faster than DRAM. However, both types of RAM are volatile, meaning that they lose their contents when the power is turned off

ROM (Read Only Memory)

Read-only memory (ROM) is a type of storage medium that permanently stores data on personal computers (PCs) and other electronic devices. It contains the programming needed to start a PC, which is essential for boot-up; it performs major input/output tasks and holds programs or software instructions.

Because ROM is read-only, it cannot be changed; it is permanent and non-volatile, meaning it also holds its memory even when power is removed. By contrast, **random access memory (RAM)** is volatile; it is lost when power is removed.

There are numerous ROM chips located on the motherboard and a few on expansion boards. The chips are essential for the basic input/output system (BIOS), boot up, reading and writing to peripheral devices, basic data management and the software for basic processes for certain utilities.

Other types of non-volatile memory include:

- Programmable Read-Only Memory (PROM)
- Electrically Programmable Read-Only Memory (EPROM)
- Electrically Erasable Programmable Read-Only Memory (EEPROM; also called Flash ROM)
- Electrically Alterable Read-Only Memory (EAROM)

Some ROM is non-volatile but can be reprogrammed, this includes:

- **Programmable read-only memory (PROM)**: This is read-only memory (ROM) that can be modified once by a user. PROM is a way of allowing a user to tailor a microcode program using a special machine called a *PROM programmer*.
- **Erasable Programmable Read-Only Memory (EPROM):** This is programmed with the use of very high voltages and exposure to approximately 20 minutes of intense ultraviolet (UV) light.
- **Electrically-Erasable Programmable Read-Only Memory (EEPROM):** This is used in many older computer BIOS chips, is non-volatile storage that can be erased and programmed several times and allows only one location at a time to be written or erased. An updated version of EEPROM is flash memory; this allows numerous memory locations to be altered simultaneously.
- **Electrically Alterable Read-Only Memory**. A **read-only memory** that can be reprogrammed **electrically** in the field a limited number of times, after the entire **memory** is erased by applying an appropriate **electric** field.

Cache Memory:

This is a very special type of high speed memory. This memory cannot be accessed by the user. The main function of this cache memory is to make the programs and data available to the CPU very fast.

Access time of memory is generally very high as compared to the execution time of the GPU. Therefore a cache, which is a very small but fast memory, is used between the CPU and the main memory. This memory also called a high speed buffer. A cache stores those segments of programs and data which are frequently needed. It makes available this data to the CPU at a very fast rate thus increasing the efficiency**.**

Difference between RAM and ROM

RAM	ROM
1. 1. The data can be read and written into RAM	1. The data can be read (read only), but the data cannot be written
2. RAM is volatile i.e. its contents are lost when the device is powered off.	2. It is non-volatile i.e. its contents are retained even when the device is powered off.
3. The programs are brought into RAM just before execution.	3.The programs to be executed are already available in ROM
4. Temporary storage medium.	4. Permanent Storage medium
5. User can write the data into RAM	5. User cannot write the data into ROM

SECONDARY STORAGE

Alternatively referred to as **external memory**, **secondary memory**, and **auxiliary storage**, a **secondary storage device** is a non-volatile device that holds data until it is deleted or overwritten. Secondary storage is about two orders of magnitude cheaper than primary storage. Consequently, hard drives (a prime example of secondary storage) are the go-to solution for nearly all data kept on today's computers.

Information stored on secondary storage devices can be accessed in two ways:

- **Sequential Access and**
- **Direct Access:**

i) Sequential Access Devices:

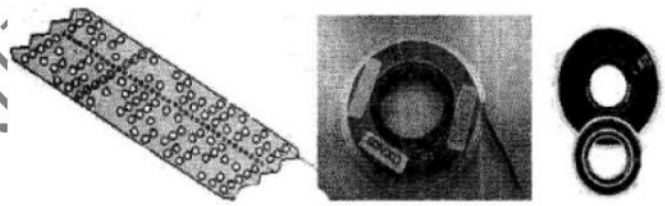
Sequential or serial access

In **sequential access** data can be accessed only in the sequence in which it has been stored. Typical sequential access storage device is the magnetic tape. These types of devices are useful in applications like pay slip printing where the data is to be accessed one after the other.

Types of Sequential Access Devices:

a) Punch Paper Tape:

Punched paper tapes were the early devices of data storage. Data is coded onto a paper tape as a combination of punched holes across the width of the tape. Each row on the tape represents one character. The data has to be coded on the tapes using special coding systems. The punched paper tapes are a low cost storage medium and their storage capacity is unlimited. However, the paper is susceptible to wear and tear and mishandling. Nowadays, punched paper tapes are rarely being used wear and tear and mishandling. Nowadays, punched paper tapes are rarely being used.



Paper tape and magnetic tape

Figure 7: Sequential Access Devices

b) Magnetic Tape:

A magnetic tape is a ribbon of Mylar which is coated with a thin layer of iron oxide material on one side.

A magnetic tape drive is used to read data from the tape or write information to the tape. The tape drive has a read/write head to access or store information respectively.

Magnetic tape is a read write device where the data can be written as well as erased and new data recorded on the same area. The tape is divided into vertical columns and horizontal rows. The columns are called **frames** and the rows are called **tracks**.

The actual number of characters that can be stored on an inch of a tape is known as the **density** of the tape.

The storage capacity of magnetic tapes is very large. This capacity is measured in terms of bytes per inch. Most common tape densities are 800 bpi, 1600 bpi. Nowadays tapes with much higher densities of the order of 6000 bpi are also available.

The records in a tape can be of any size.

Advantages of Magnetic Tapes:
- High data density and virtually unlimited storage
- Low in cost
- Easy to handle and portable from one computer to another.

Limitations are:
- Support only Sequential access
- Tapes are sensitive to dust; humidity and temperature, hence require proper storage facilities.

(ii) Direct Access Storage Devices

Random or direct access

In **random access** the data at any location on the storage unit can be accessed directly without having to follow the sequence in which it has been stored. Typical devices that support direct access are the magnetic disk and magnetic drum.

a) Magnetic Disk:

A magnetic disk is a thin metallic/Mylar platter circular in shape. It is coated on both sides with magnetic material. A number of these disks are mounted on a disk pack, on a central shaft. Thus all the disks in the disk pack move at the same speed, simultaneously in the same direction. These disks are also called as hard disks or fixed disks. Hard disk can be permanently installed in the drive or can also be in the form of a removable cartridge. Each disk is divided into a number of concentric circles called **tracks.** All the corresponding tracks in all the surfaces are together called a **cylinder.** Information is not stored on the outer surface of the upper plate and the lower surface of the bottom plate.

b) Floppy Disks:

Floppy disks are made up of flexible Mylar coated with iron oxide. This disk is enclosed in a square plastic jacket to protect the surface of the disk from dust. A floppy disk is to be inserted in the floppy disk drive of the computer system to read or write information. The read/write head of the drive makes a direct contact with the floppy disk.

While accessing or storing data, Floppy disks come in various sizes

- 8 inch floppy disks
- 51/4 inch floppy disks
- 31/2 inch floppy disks

Floppy disks are a very popular storage medium since they are small in size, relatively cheap and can store data on line. Floppy disks are also very portable. They can be carried form one place to another very easily.

c) Winchester Disk:

In a Winchester, the disks are permanently enclosed in a sealed container. The disks are coated with a special lubricant to reduce friction with the read/write head. This technology allows for an increase in the number of tracks on the disk, and higher storage density. Winchester disks are fast and highly reliable. They are used extensively in micro-computers.

d) Magnetic Drum:

This is a cylinder whose outer surface is coated with a thin layer of magnetic material. A motor rotates on the cylinder at a constant speed. Data is recorded on the tracks of the drum as magnetized spots. A set of stationary read/write heads are positioned slightly away from the surface of the drum. Data is read from and written onto this drum with the help of these heads. The drum rotates are relatively fast speeds of the order of a several thousand rotations per minute. Magnetic drums have faster data transfer rates as compared to disks. However their storage capacity is limited. Magnetic drums are rarely used today.

e) Optical Disk:

An optical disk is made up of a rotating disk which is coated with a thin reflective metal. To record data on the optical disk, a laser beam is focused on the surface of the spinning disk. The laser beam is turned on and off at varying rates! Due to this, tiny holes (pits) are burnt into the metal coating along the tracks. When data

stored on the optical disk is to be read, a less powerful laser beam is focused on the disk surface. The storage capacity of these devices is tremendous.

Optical disk access time is relatively fast. The biggest drawback of the optical disk is that it is a permanent storage device. Data once written cannot be erased. Therefore it is a read only storage medium. A typical example of the optical disk is the CD-ROM.

10. Ports and Connection:

A connection is a term that describes the link between a plug or connector into a port or jack. For example, your monitor, mouse, and keyboard all must connect to the computer before they work.

In computer hardware, a **port** serves as an interface between the computer and other computers or peripheral devices. In computer terms, a port generally refers to the part of connection available for connection between one computer to peripherals like input and output ones. Computer ports have many uses, to connect a monitor, webcam, speakers, or other peripheral devices. On the physical layer, a computer port is a specialized outlet on a piece of equipment to which a plug or cable connects. Electronically, the several conductors where the port and cable contacts connect provide a method to transfer signals between devices.

A Computer Port is an interface or a point of connection between the computer and its peripheral devices. Some of the common peripherals are mouse, keyboard, monitor or display unit, printer, speaker, flash drive etc.

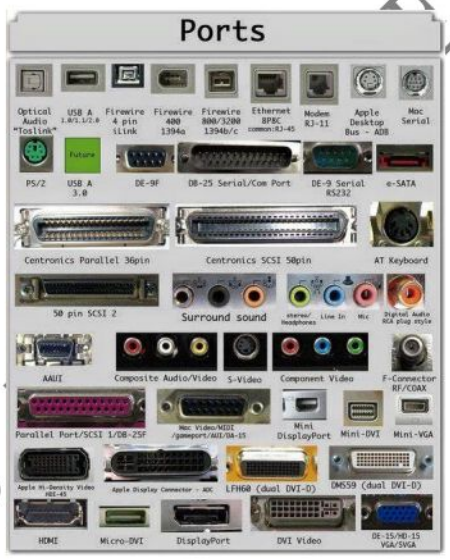


Figure 8: 16 different types of ports

Serial Port:

A *serial port* is an *interface* on a computer system with which information is transferred in or out one *bit* at a time.

An interface is a connection between two subsystems. In this case, it is an electrical connector built into a computer into which another, external connector can be easily inserted (and removed) by a user to allow data to be transferred to or from peripheral devices (e.g., display units, keyboards, mice, printers and scanners).

A bit is the most basic unit of information in computing and communications. Each bit has a value of either zero or one. This value is stored as an electrical charge in a single capacitor within a memory device (i.e., a RAM) or a CPU (central processing unit) or as the magnetization of a tiny area of magnetic material on a hard disk drive (HDD) or floppy disk.

In contrast to serial ports, a *parallel port* consists of multiple wires over which bits are transferred *in parallel*, i.e., simultaneously. There are usually several extra wires on the port that are used for control signals to indicate when data is ready to be sent or received.

Throughout most of the history of personal computers, serial connections have been made using the RS-232 standard.

Parallel Port:

A parallel port is a type of interface found on computers (personal and otherwise) for connecting peripherals. The name refers to the way the data is sent; parallel ports send multiple bits of data at once, in parallel communication, as opposed to serial interfaces that send bits one at a time. To do this, parallel ports require multiple data lines in their cables and port connectors, and tend to be larger than contemporary serial ports which only require one data line.

PS/2 Port

- Used for old computer keyboard and mouse
- Also called mouse port
- Most of the old computers provide two pS/2 port, each for the mouse and keyboard.
- IEEE 1284 – compliant Centronics port.

Universal Serial Bus (or USB) port

- It can connect all kinds of external USB devices such as external hard disk, printer, scanner, mouse, keyboard, etc.
- It was introduced in 1997.
- Most of the computers provide two USB ports as minimum
- Data travels at 12 megabits per seconds.
- USB compliant devices can get power from a USB port.

VGA Port

- Connects monitor to a computer's video card.
- It has 15 holes
- Similar to the serial port connector. However, serial port connector has pins, VGA port has holes.

Power Connector

- Three-pronged plug.
- Connects to the computer's power cable that plugs into a power bar or wall socket.

Firewire Port

- Transfers large amount of data at very fast speed.
- Connects camcorders and video equipment to the computer.
- Data travels at 400 to 800 megabits per second.
- Invented by Apple.
- It has three variants: 40-pin firewire 400 connector, 6-pin firewire 400 connector, 9-pin firewire 800 connector.

Modem Port

- Connects a PC's modem to the telephone network.

Ethernet Port

- Connects to a network and high speed Internet network.
- Connects the network cable to a computer
- This port reside on an Ethernet Card
- Data travels at 10 megabits to 1000 megabits per seconds depending upon the network bandwidth.

Game Port

- Connect a joystick to a PC
- Now replaced by USB

Digital Video Interface (DVD) Port

- Connects flat panel LCD monitor to the computer's high-end-video graphic cards.
- Very popular among video card manufactures.

Sockets:

- Sockets connect the microphone and speakers to the sound card of the computer.

11. COMPUTER NETWORK

A computer network is a set of computers connected together for the purpose of sharing resources such as:

- Hardware
- Software

using communication media. The most common resource shared today is connection to the Internet. Other shared resources can include a printer or a file server. The Internet itself can be considered a computer network.

Computer Network Defined

A **computer network** is a set of connected computers. Computers on a network are called **nodes**. The connection between computers can be done via cabling, most commonly the Ethernet cable, or wirelessly through radio waves. Connected computers can share resources, like access to the Internet, printers, file servers, and others. A network is a multipurpose connection, which allows a single computer to do more.

The uses of Computer Network:

The various benefits of using a network are shown below:

a). **Sharing the data:** Exchanging of information between the organization that are geographically far apart is easier.

b). **Sharing of peripheral Devices**: Various peripheral devices such as printer, scanner, hard disk etc. can be shared by many users.

c). **Personal Communication**: We can communicate with people in any part of the world using:

E-mail: We can exchange the message or text including graphic, audio and video.

Using videoconferencing or audio conferencing two or more people share their information in real time.

d) **Easier data backup**: Since data in extremely important in the business, the users have to back-up their data. This is done by simply moving the files to the severs in the common shared area.

e) **Load sharing:** If one computer is loaded with many jobs, some of the jobs can be transferred to another machine and can be executed there.

f) **Working from different Place**: The computer network allows the flexible working environment. Many people sit at home or hotel and work from there connecting to a network.

g) **E-commerce:** It is gaining popularity using which people can purchase any items from anywhere in the world without leaving their place.

COMMON TYPES OF NETWORKS

One way to categorize the different types of computer network designs is by their scope or scale. For historical reasons, the networking industry refers to nearly every type of design as some kind of *area network*. Common types of area networks are:

- LAN - Local Area Network
- WAN - Wide Area Network
- WLAN - Wireless Local Area Network
- MAN - Metropolitan Area Network
- SAN - Storage Area Network, System Area Network, Server Area Network, or sometimes Small Area Network
- CAN - Campus Area Network, Controller Area Network, or sometimes Cluster Area Network
- PAN - Personal Area Network

LAN and WAN are the two primary and best-known categories of area networks, while the others have emerged with technology advances.

LAN: Local Area Network

A LAN connects network devices over a relatively short distance. A networked office building, school, or home usually contains a single LAN, though sometimes one building will contain a few small LANs (perhaps one per room), and occasionally a LAN will span a group of nearby buildings. In TCP/IP networking, a LAN is often but not always implemented as a single IP subnet.

In addition to operating in a limited space, LANs are also typically owned, controlled, and managed by a single person or organization. They also tend to use certain connectivity technologies, primarily Ethernet and Token Ring.

Advantages of LAN or Local Area Network

1. Resource Sharing

Computer hardware resources like printers, modems, DVD-ROM drives and hard disks can be shared with the help of local area networks. This will reduce cost of hardware purchases.

2. Software Applications Sharing

It is cheaper to use same software over network instead of purchasing separate licensed software for each client in a network.

3. Easy and Cheap Communication

Data and messages can easily be transferred over networked computers. It saves a lot of time and money.

4. Centralized Data

The data of all network users can be saved on hard disk of the server computer. This will help users to use any workstation in a network to access their data. Because data is not stored on workstations locally. But it is stored on a server computer.

5. Data Security

Since, data is stored on server computer centrally, it will be easy to manage data at only one place and the data will be more secure too, because of more security for the server computer.

6. Internet Sharing

Local Area Network provides the facility to share a single internet connection among all the LAN users. In Net Cafes, single internet connection sharing system keeps the internet expenses cheaper.

Disadvantages of LAN or Local Area Network

1. High Setup Cost

Although the LAN will save cost over time due to shared computer resources but the initial setup costs of installing Local Area Networks is high. This is because any organization that will setup a network will have to purchase necessary hardware equipment for networking. It may require a sophisticated server computer - a Mini Computer, Network LAN cards, Network Routers, HUBS / Switches, Networking Cables (for wired networks only) and connectors etc.

2. Privacy Violations

The LAN administrator has the rights to check personal data files of each and every LAN user. Moreover he can check the internet history and computer use history of the LAN users.

3. Data Security Threat

Unauthorized users can access important data of an organization if centralized data repository is not secured properly by the LAN administrator. LAN Administer is responsible for the security of the whole data resource in an organization. LAN administrator will implement a fully functional security policy for database safety.

4. LAN Maintenance Job

Local Area Network requires a LAN Administrator. Because, there are problems of software installations or hardware failures or cable disturbances in Local Area Network. A LAN Administrator is needed at this full time job.
A LAN Administrator may be with an M.C.S. or B.S.C.S. degree holder person optionally with a course / diploma in networking field or with relative experience in the field.

5. Covers Limited Area

Local Area Network covers a small area like one office, one building or a group of nearby buildings.

WAN: Wide Area Network

As the term implies, a WAN spans a large physical distance. The Internet is the largest WAN, spanning the Earth.

A WAN is a geographically-dispersed collection of LANs. A network device called a router connects LANs to a WAN. In IP networking, the router maintains both a LAN address and a WAN address.

A WAN differs from a LAN in several important ways. Most WANs (like the Internet) are not owned by any one organization but rather exist under collective or distributed ownership and management. WANs tend to use technology like ATM, Frame Relay and X.25 for connectivity over the longer distances.

Advantages of WAN are:

- It covers large geographical area therefore networks can be made between long distances
- Everyone on that network can access the same data. This avoids problems where some users may have older information than others.
- Different peripherals can be shared with all the computers in the network.
- Shares software and resources with connecting workstations.

Disadvantage of WAN are:

- Setting up the network could be expensive.
- Maintaining a network is a full-time job which requires network supervisors and technicians to be employed.
- Security is a real issue when many different people have the ability to use information from other computers.
- Protection against hackers and viruses adds more complexity and expense.
- Need a good firewall to restrict outsiders from entering and disrupting the network.

LAN, WAN, and Home Networking

Residences typically employ one LAN and connect to the Internet WAN via an Internet Service Provider (ISP) using a broadband modem. The ISP provides a WAN IP address to the modem, and all of the computers on the home network use LAN (so-called *private*) IP addresses. All computers on the home LAN can communicate directly with each other but must go through a central network gateway, typically a broadband router, to reach the ISP.

Other Types of Area Networks

While LAN and WAN are by far the most popular network types mentioned, you may also commonly see references to these others:

- **Wireless Local Area Network** - A LAN based on Wi-Fi wireless network technology
- **Metropolitan Area Network** - A network spanning a physical area larger than a LAN but smaller than a WAN, such as a city. A MAN is typically owned and operated by a single entity such as a government body or large corporation.

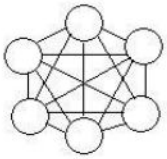
- **Campus Area Network** - A network spanning multiple LANs but smaller than a MAN, such as on a university or local business campus.
- **Storage Area Network** - Connects servers to data storage devices through a technology like Fibre Channel.
- **System Area Network** (also known as Cluster Area Network) - Links high-performance computers with high-speed connections in a cluster configuration.

Types of Network Connections

Computer networks can be broken down historically into **topologies**, which is a technique of connecting computers. The most common topology today is a **collapsed ring**. This is due to the success of a network protocol called the Ethernet. This protocol, or network language, supports the Internet, Local Area Networks, and Wide Area Networks.

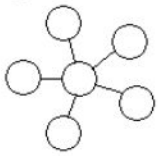
Mesh Topology

Mesh Topology: In a mesh network, devices are connected with many redundant interconnections between network nodes. In a true mesh topology every node has a connection to every other node in the network.



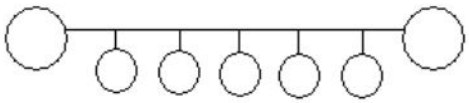
Star Topology

A **star topology** is a design of a network where a central node extends a cable to each computer on the network. On a star network, computers are connected independently to the center of the network. If a cable is broken, the other computers can operate without problems. A star topology requires a lot of cabling.



Bus Topology

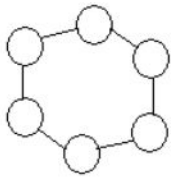
A **bus topology** is another type of design where a single cable connects all computers and the information intended for the last node on the network must run through each connected computer. If a cable is broken, all computers connected down the line cannot reach the network. The benefit of a bus topology is a minimal use of cabling.



Ring Topology

A similar topology is called a **ring**. In this design, computers are connected via a single cable, but the end nodes also are connected to each other. In this design, the signal circulates through the network until it finds the intended recipient. If a network node is not configured properly, or it is down temporarily for another reason, the signal will make a number of attempts to find its destination.

A **collapsed ring** is a topology where the central node is a network device called a hub, a router, or a switch. This device runs a ring topology internally and features plugins for cables. Next, each computer has an independent cable, which plugs into the device. Most modern offices have a **cabling closet**, or a space containing a switch device that connects the network. All computers in the office connect to the cabling closet and the switch. Even if a network plug is near a desk, the plug is connected via a cable to the cabling closet.



NETWORK DEVICES

The data that is transmitted between PCs should be channelled properly so that data reaches destination. The various hardware used in computer network during data transfers are:

a) Network Interface Cards (NICs)
b) Network linking devices

Network Interface Cards:

A Network Interface Card (NIC) is a computer hardware component that allows a computer to connect to a network. NICs may be used for both wired and wireless connections.

A NIC is also known as a network interface controller (NIC), network interface controller card, expansion card, computer circuit board, network card, LAN card, network adapter or network adapter card (NAC).

Network Linking Devices:

Network Hub:

Network Hub is a networking device which is used to connect multiple network hosts. A network hub is also used to do data transfer. The data is transferred in terms of packets on a computer network. So when a host sends a data packet to a network hub, the hub copies the data packet to all of its ports connected to. Like this, all the ports know about the data and the port for whom the packet is intended, claims the packet.

Network Switch:

Like a hub, a switch also works at the layer of LAN (Local Area Network) but you can say that a switch is more intelligent than a hub. While hub just does the work of data forwarding, a switch does 'filter and forwarding' which is a more intelligent way of dealing with the data packets.

Modem:

A Modem is somewhat a more interesting network device in our daily life. So if you have noticed around, you get an internet connection through a wire (there are different types of wires) to your house. This wire is used to carry our internet data outside to the internet world.

A modem stands for (**Mo**dulator + **De**modulator). That means it modulates and demodulates the signal between the digital data of a computer and the analog signal of a telephone line.

Network Router:

A router is a network device which is responsible for routing traffic from one to another network. These two networks could be a private company network to a public network. You can think of a router as a traffic police who directs different network traffic to different directions.

Bridge:

If a router connects two different types of networks, then a bridge connects two sub-networks as a part of the same network. You can think of two different labs or two different floors connected by a bridge.

Repeater:

A repeater is an electronic device that amplifies the signal it receives. In other terms, you can think of repeater as a device which receives a signal and retransmits it at a higher level or higher power so that the signal can cover longer distances.

Gateway:

A gateway, as the name suggests, is a passage to connect two networks together that may work upon different networking models. They basically works as the 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 switch or router.

12. SOFTWARE BASICS

Sometimes abbreviated as **SW** and **S/W**, **software** is a collection of instructions or set of programs that enable the user to interact with a computer, its hardware, or perform specific tasks. Without software, most computers would be useless. For example, without your Internet browser software, you could not surf the Internet or read this page. Without an operating system, the browser could not run on your computer. The picture to the right shows a Microsoft Excel box, an example of a spread sheet software program.

Software is often divided into three categories:

- **System software** serves as a base for application software. System software includes device drivers, operating systems (OSs), compilers, disk formatters, text editors and utilities helping the computer to operate more efficiently. It is also responsible for managing hardware components and providing basic non-task-specific functions. The system software is usually written in C programming language.
- **Programming software** is a set of tools to aid developers in writing programs. The various tools available are compilers, linkers, debuggers, interpreters and text editors.
- **Application software** is intended to perform certain tasks. Examples of application software include office suites, gaming applications, database systems and educational software. Application software can be a single program or a collection of small programs. This type of software is what consumers most typically think of as "software."

13. COMPUTER LANGUAGES

A language is the main medium of communicating between the Computer systems and the most common are the programming languages. As we know a Computer only understands binary numbers that is 0 and 1 to perform various operations but the languages are developed for different types of work on a Computer. A language consists of all the instructions to make a request to the system for processing a task.

Low Level Language:

Low level languages are the machine codes in which the instructions are given in machine language in the form of 0 and 1 to a Computer system. It is mainly designed to operate and handle all the hardware and instructions set architecture of a Computer. The main function of the Low level language is to operate, manage and manipulate the hardware and system components.

The main advantage of using Machine language is that there is no need of a translator or interpreter to translate the code, as the Computer directly can understand. But there are some disadvantages also like you have to remember the operation codes, memory address every time you write a program and also hard to find errors in a written program. It is a machine dependent and can be used by a single type of Computer.

- **Assembly Language** is the second generation programming language that has almost similar structure and set of commands as Machine language. Instead of using numbers like in Machine languages here we use words or names in English forms and also symbols. The programs that have been written using words, names and symbols in assembly language are converted to machine language using an Assembler. Because a Computer only understands machine code languages that's why we need an Assembler that can convert the Assembly level language to Machine language so the Computer gets the instruction and responds quickly.

The main disadvantage of this language is that it is written only for a single type of CPU and does not run on any other CPU. But its speed makes it the most used low level language till today which is used by many programmers.

High Level Language:

The high level languages are the most used and also more considered programming languages that helps a programmer to read, write and maintain. It is also the third generation language that is used and also running till now by many programmers. They are less independent to a particular type of Computer and also require a translator that can convert the high level language to machine language. The translator may be an interpreter and Compiler that helps to convert into binary code for a Computer to understand. There is various high level programming languages like C, FORTRAN or Pascal that are less independent and also enables the programmer to write a program.

14. LANGUAGE PROCESSORS

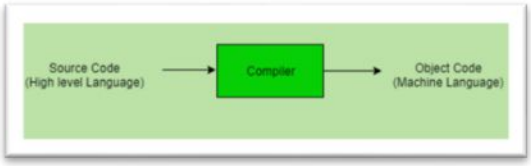
A system software is used to translate the program written in high-level language into machine code is called **Language Processor** and the program after translated into machine code (object program / object code).

The language processors can be any of the following three types:

1. **Compiler** –

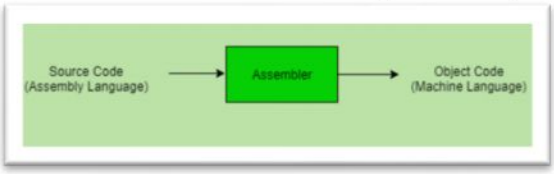
The language processor that reads the complete source program written in high level language as a whole in one go and translates it into an equivalent program in machine language is called as a Compiler.
Example: C, C++, C#, Java

In a compiler, the source code is translated to object code successfully if it is free of errors. The compiler specifies the errors at the end of compilation with line numbers when there are any errors in the source code. The errors must be removed before the compiler can successfully recompile the source code again.



2. **Assembler**

The Assembler is used to translate the program written in Assembly language into machine code. The source program is a input of assembler that contains assembly language instructions. The output generated by assembler is the object code or machine code understandable by the computer.



3. **Interpreter**

The translation of single statement of source program into machine code is done by language processor and executes it immediately before moving on to the next line is called an interpreter. If there is an error in the statement, the interpreter terminates its translating process at that statement and displays an error message. The interpreter moves on to the next line for execution only after removal of the error. An Interpreter directly executes instructions written in a programming or scripting language without previously converting them to an object code or machine code.
Example: Perl, Python and Matlab.

Difference between Compiler and Interpreter

COMPILER	INTERPRETER
A compiler is a program which coverts the entire source code of a programming language into executable machine code for a CPU.	Interpreter takes a source program and runs it line by line, translating each line as it comes to it.
Compiler takes large amount of time to analyze the entire source code but the overall execution time of the program is comparatively faster.	Interpreter takes less amount of time to analyze the source code but the overall execution time of the program is slower.
Compiler generates the error message only after scanning the whole program, so debugging is comparatively hard as the error can be present any- where in the program.	Its Debugging is easier as it continues translating the program until the error is met
Generates intermediate object code.	No intermediate object code is generated.
Examples: C, C++, Java	Examples: Python, Perl

OVERVIEW OF C

C is a structured programming language developed by Dennis Ritchie in 1973 at Bell Laboratories. It is one of the most popular computer languages today because of its structure, high-level abstraction, machine independent feature. C language was developed with UNIX operating system, so it is strongly associated with UNIX, which is one of the most popular network operating system in use today and heart of internet data superhighway.

History of C Language

C language has evolved from three different structured language ALGOL, BCPL and B Language. It uses many concepts from these languages and introduced many new concepts such as data types, struct, pointer. In 1988, the language was formalised by **American National Standard Institute** (ANSI). In 1990, a version of C language was approved by the **International Standard Organisation** (ISO) and that version of C is also referred to as C89.

IMPORTANCE

The increasing popularity of C probably due it its many desirable qualities. It is a robust language whose rich set of built-in functions and operators can be used to write any complex programs.
The C compiler combines the capabilities of an assembly language with the features of a high-level language and therefore it is well suited for writing both system software and business package.
Programs written in C are efficient and fast. This is due to its variety of data types and powerful operator.
There are only 32 keywords in ANSI C and its strength lies in its built-in functions.
C is highly portable. This means that C programs written for one computer can be run on another with little or no modification. Portability is important if we plan to use a new computer with a different operating system.

C language is well suited for structured programming, thus requiring the user to think of a program in terms of function modules or blocks.

Another important feature of C is its ability to extend itself. A C program is basically a collection of functions that are supported by the C library. We can continuously add our own functions to C library. With the availability of a large number of functions, the programming task become simple.

BASIC STRUCTURES OF C PROGRAMS

Basic Structure of C Programs
Documentation Section
Link Section
Definition Section
Global Declaration Section

main() Function Section
{
Declaration Part
Executable Part
}

Subprogram Section Function 1 Function 2 Function 3 - - - Function n

Figure 9: Basic Structure of C Program

The basic structure of C program is as follows:

1. Documentation
2. Link Section
3. Definition Section
4. Global Declaration Section
5. Main Function Section
6. Sub Program Section

1. Documentation:

The documentation section consists of a set of comment lines giving the name of the program, the author and other details, which the programmer would like to use later.

2. Link section:

The link section provides instructions to the compiler to link functions from the system library such as using the #include directive.

3. Definition section:

The definition section defines all symbolic constants such using the #define directive.

4. Global declaration section:

There are some variables that are used in more than one function. Such variables are called global variables and are declared in the global declaration section that is outside of all the functions. This section also declares all the user-defined functions.

5. main () function section:

Every C program must have one main function section. This section contains two parts; declaration part and executable part

1. Declaration part: The declaration part declares all the variables used in the executable part.

2. Executable part: There is at least one statement in the executable part. These two parts must appear between the opening and closing braces. The program execution begins at the opening brace and ends at the closing brace. The closing brace of the main function is the logical end of the program. All statements in the declaration and executable part end with a semicolon.

6. Subprogram section:

If the program is a multi-function program then the subprogram section contains all the user-defined functions that are called in the main () function. User-defined functions are generally placed immediately after the main () function, although they may appear in any order.

Ex:

```
/* Program to find the sum of two numbers */    /* Comment Section */
#include<stdio.h>                                 /* Link Section */
void main()                                       /* Function Section */
{
      int a=10, b= 30, sum=0;                     /* Declaration Part */
      sum = a + b;                                /* Exactable Part */
      printf(" Sum = %d\n", sum);
}
```

Executing a C Program:

Programming Environment

Local Environment Setup

If you want to set up your environment for C programming language, you need the following two software tools available on your computer, (a) Text Editor and (b) The C Compiler.

Text Editor

This will be used to type your program. Examples of few a editors include Windows Notepad, OS Edit command, Brief, Epsilon, EMACS, and vim or vi.

The name and version of text editors can vary on different operating systems. For example, Notepad will be used on Windows, and vim or vi can be used on windows as well as on Linux or UNIX.

The files you create with your editor are called the source files and they contain the program source codes. The source files for C programs are typically named with the extension "**.c**".

Before starting your programming, make sure you have one text editor in place and you have enough experience to write a computer program, save it in a file, compile it and finally execute it.

The C Compiler

The source code written in source file is the human readable source for your program. It needs to be "compiled", into machine language so that your CPU can actually execute the program as per the instructions given.

The compiler compiles the source codes into final executable programs. The most frequently used and free available compiler is the GNU C/C++ compiler, otherwise you can have compilers either from HP or Solaris if you have the respective operating systems.

The following section explains how to install GNU C/C++ compiler on various OS. We keep mentioning C/C++ together because GNU gcc compiler works for both C and C++ **programming languages.**

Installation on UNIX/Linux

If you are using **Linux or UNIX**, then check whether GCC is installed on your system by entering the following command from the command line −

```
$ gcc -v
```

If you have GNU compiler installed on your machine, then it should print a message as follows −

```
Using built-in specs.
Target: i386-redhat-linux
Configured with: ../configure --prefix=/usr .....
Thread model: posix
```

gcc version 4.1.2 20080704 (Red Hat 4.1.2-46)

If GCC is not installed, then you will have to install it yourself using the detailed instructions available at https://gcc.gnu.org/install/

This tutorial has been written based on Linux and all the given examples have been compiled on the Cent OS flavor of the Linux system.

Installation on Mac OS

If you use Mac OS X, the easiest way to obtain GCC is to download the Xcode development environment from Apple's web site and follow the simple installation instructions. Once you have Xcode setup, you will be able to use GNU compiler for C/C++.

Xcode is currently available at developer.apple.com/technologies/tools/.

Installation on Windows

To install GCC on Windows, you need to install MinGW. To install MinGW, go to the MinGW homepage, www.mingw.org, and follow the link to the MinGW download page. Download the latest version of the MinGW installation program, which should be named MinGW-<version>.exe.

While installing Min GW, at a minimum, you must install gcc-core, gcc-g++, binutils, and the MinGW runtime, but you may wish to install more.

Add the bin subdirectory of your MinGW installation to your **PATH**environment variable, so that you can specify these tools on the command line by their simple names.

After the installation is complete, you will be able to run gcc, g++, ar, ranlib, dlltool, and several other GNU tools from the Windows command line.

Compile and Execute C Program

Let us see how to save the source code in a file, and how to compile and run it. Following are the simple steps −

- Open a text editor and add the above-mentioned code.
- Save the file as *hello.c*
- Open a command prompt and go to the directory where you have saved the file.
- Type *gcc hello.c* and press enter to compile your code.
- If there are no errors in your code, the command prompt will take you to the next line and would generate *a.out* executable file.
- Now, type *a.out* to execute your program.
- You will see the output *"Hello World"* printed on the screen.

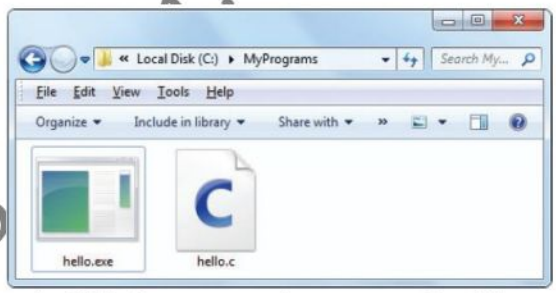
```
$ gcc hello.c
$ ./a.out
Hello, World!
```

Make sure the gcc compiler is in your path and that you are running it in the directory containing the source file hello.c.

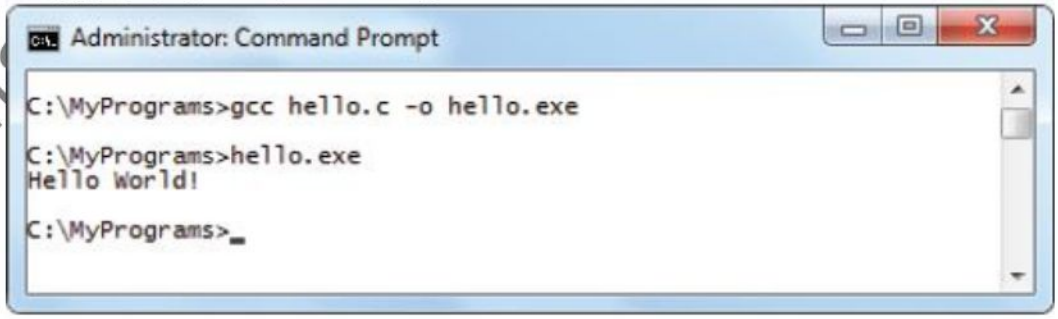
Compiling and Executing C program:

The C source code files for the examples in this book are stored in a directory created expressly for that purpose. The directory is named "MyPrograms" and its absolute address on Windows is C:\MyPrograms, whereas on Linux it's at /home/MyPrograms. The hello.c source code file, created by following the steps on the previous page, is saved in this directory awaiting compilation to produce a version in executable byte code format.

1. At a command prompt issue a cd command with the path to the MyPrograms directory to navigate there l
2. At a command prompt in the MyPrograms directory type gcc hello.c then hit Return to compile the program When the compilation succeeds the compiler creates an executable file alongside the original source code file. By default this file will be named a.out on Linux systems and a.exe on Windows systems. Compiling a different C source code file in the MyPrograms directory would now overwrite the first executable file without warning. This is obviously unsatisfactory so a custom name for the executable file must be specified when compiling hello.c. This can be achieved by including a -o option followed by a custom name in the compiler command.
3. At a command prompt in the MyPrograms directory type gcc hello.c -o hello.exe then hit Return to compile the program once more On both Linux and Windows systems an executable file named hello.exe is now created alongside the C source code file:



4. At a command prompt in Windows type the executable filename then hit Return to run the program – the text string is output and the print head moves to the next line



Because Linux does not by default look in the current directory for executable files, unless it is specifically directed to do so, it is necessary to prefix the filename with ./ to execute the program.

5. At a command prompt in Linux type ./hello.exe then hit Return to run the program – the text string is output and the print head moves to the next line

We can also use ./a.out for program execution.

CHARACTER SET

The character that can be used to form words, numbers and expressions depend upon the computer on which the program is run. However, a subset of characters is available that can be used on most personal, micro, mini and mainframe computers. The characters in C are grouped into the following categories:

1. Letters
2. Digits
3. Special Characters
4. White spaces

Most ANSI-compatible C compilers accept the following ASCII characters for both the source and execution character sets. Each ASCII character corresponds to a numeric value. The ASCII characters and their numeric values.

- The 26 lowercase Roman characters:

a b c d e f g h i j k l m n o p q r s t u v w x y z

- The 26 uppercase Roman characters:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

- The 10 decimal digits:

0 1 2 3 4 5 6 7 8 9

- The 30 graphic characters:

! # % ^ & * ( ) - _ = + ~ ' " : ; ? / | \ { } [ ] , . < > $

Five white space characters:

Space	( )
Horizontal tab	(\t)
Form feed	(\f)
Vertical tab	(\v)
New-line character	(\n)

The ASCII execution character set also includes the following control characters:

- New-line character (represented by \n in the source file),
- Alert (bell) tone (\a)
- Backspace (\b)
- Carriage return (\r)
- Null character (\0)

Trigraph Sequences

To write C programs using character sets that do not contain all of C's punctuation characters, ANSI C allows the use of nine ***trigraph sequences*** in the source file. These three- character sequences are replaced by a single character in the first phase of compilation.

Trigraph Sequence	Character Equivalent
??=	#
??(	[
?? /	\
??)	]
??'	^
??<	{

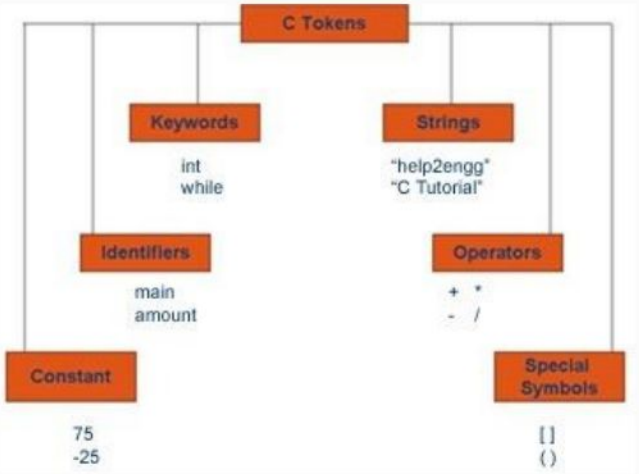
??!	|
??>	}
??-	~

C TOKENS:

- In a passage of text, individual words and punctuation marks are called Tokens. Similarly, in a C program the smallest individual units are known as C tokens.
- C tokens are the basic buildings blocks in C language which are constructed together to write a C program.
- Each and every smallest individual units in a C program are known as C tokens.

C tokens are of six types. They are,

1. Keywords (eg: int, while),
2. Identifiers (eg: main, total),
3. Constants (eg: 10, 20),
4. Strings (eg: "total", "hello"),
5. Special symbols (eg: (), {}),
6. Operators (eg: +, /,-,*)



KEYWORDS IN C LANGUAGE

Every C word is classified as either a keyword or an identifier. All keywords have fixed meanings and these meanings cannot be changed.

- Keywords are pre-defined words in a C compiler.
- Each keyword is meant to perform a specific function in a C program.
- Since keywords are referred names for compiler, they can't be used as variable name.

C language supports 32 keywords which are given below.

auto	double	case	enum
int	struct	register	typedef
const	float	default	goto
short	unsigned	sizeof	volatile
break	else	char	extern
long	switch	return	union
continue	for	do	if
signed	void	static	while

IDENTIFIERS:

Identifiers refer to the names of variables, functions and arrays. These are user-defined names and consist of a sequence of letters and digits, with a letter as a first character. Both uppercase and lowercase letters are permitted, although lowercase letters are commonly used. The underscore character is also permitted in identifiers. It is usually used as a link between two words in long identifiers.

Rules for Identifiers:

1. First character must be an alphabet (or Underscore)
2. Must consist of only letters, digits or underscore.
3. Only first 31 characters are significant.
4. Cannot use a keyword.
5. Must not contain white space.

CONSTANT

Constants refer to fixed values that the program may not alter during its execution. These fixed values are also called literals. **Constants** can be of any of the basic data types like an integer **constant**, a floating **constant**, a character **constant**, or a string literal. There are enumeration **constants** as well.

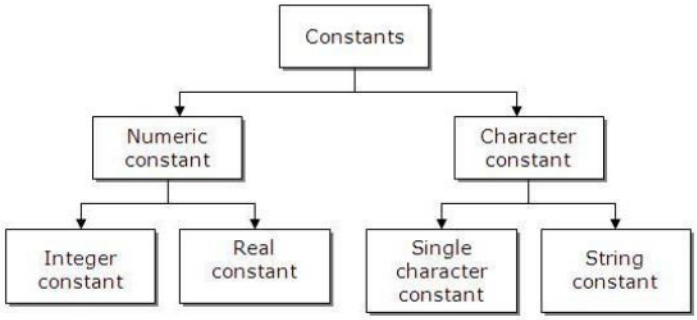


Figure: Basic Types of Constants

Integer Constants:

An integer constant is a sequence of digits from 0 to 9 without decimal points or fractional part or any other symbols. There are 3 types of integers namely decimal integer, octal integers and hexadecimal integer.

Decimal Integers consists of a set of digits 0 to 9 preceded by an optional + or - sign. Embedded spaces, commas and non-digit characters are not permitted between digits.

Example for valid decimal integer constants are:

int y=123; //here 123 is a decimal integer constant
int n = -321; // Is valid constant
int m = 15 750; // Illegal numbers. Because space is not allowed.

Octal Integers constant consists of any combination of digits from 0 through 7 with a O at the beginning. Some examples of octal integers are

int X=O123; // here 0123 is a octal integer constant .

Hexadecimal integer constant is preceded by OX or Ox, they may contain alphabets from A to F or a to f. The alphabets A to F refers to 10 to 15 in decimal digits. Example of valid hexadecimal integers are

int x=Ox12 // here Ox12 is a Hexa-Decimal integer constant

Real Constants:

Real Constants consists of a fractional part in their representation. Integer constants are inadequate to represent quantities that vary continuously. These quantities are represented by numbers containing fractional parts like 26.082. Example of real constants are

float x = 6.3; //here 6.3 is a double constant.
float y = 6.3f; //here 6.3f is a float constant.
float z = 6.3 e + 2; //here 6.3 e + 2 is a exponential constant.
float s = 6.3L ; //here 6.3L is a long double constant

Real Numbers can also be represented by exponential notation. The general form for exponential notation is mantissa exponent. The mantissa is either a real number expressed in decimal notation or an integer. The exponent is an integer number with an optional plus or minus sign.

Single Character Constants:

A Single Character constant represent a single character which is enclosed in a pair of quotation symbols.

Example for character constants are

```
char p ='ok' ;  // p will hold the value 'O' and k will be omitted
char y ='u';    // y will hold the value 'u'
char k ='34' ; // k will hold the value '3, and '4' will be omitted
char e =' ';    // e will hold the value ' ' , a blank space
chars ='\45'; // swill hold the value ' ' , a blank space
```

All character constants have an equivalent integer value which are called ASCII Values.

String Constants:

A string constant is a set of characters enclosed in double quotation marks. The characters in a string constant sequence may be a alphabet, number, special character and blank space. Example of string constants are

"VISHAL" "1234" "God Bless" "!.....?"

Backslash Character Constants [Escape Sequences]:

Backslash character constants are special characters used in output functions. Although they contain two characters they represent only one character. Given below is the table of escape sequence and their meanings.

Constant	Meaning
'\a'	.Audible Alert (Bell)
'\b'	.Backspace
'\f'	.Form feed
'\n'	.New Line
'\r'	.Carriage Return
'\t'	.Horizontal tab
'\v'	.Vertical Tab
'\''	.Single Quote
'\"'	.Double Quote
'\?'	.Question Mark
'\\'	.Back Slash
'\0'	.Null

Table: Backslash Character Constant

Note that each one of them represents one character, although they consist of two characters. These characters combinations are known as escape sequences.

VARIABLES

A variable is a data name that may be used to store a data value. Unlike constants that remain unchanged during the execution of a program, a variable may take different values at different times during execution.

A variable name can be chosen by the programmer in a meaningful way so as to reflect its function or a nature in the program. Some examples of such names are:

Average
height
Total
Counter_1

class_strength

Variable names may consist of letters, digits, and underscore(_) character, subject to the following conditions:

Rules for defining a variable

1. The first character in the variable should be a letter or an underscore. The first character can be followed by letters or digits or underscore.
2. ANSI standard recognizes a length of 31 characters.
3. Keywords should not be used as variable-names.
4. Uppercase and lowercase are significant. That is, the variable Total is not same as total or Total.
5. White space is not allowed.

Some examples of valid variable names are:

Manu	value	T_value
M	N	distance

Invalid examples include

123	(area)
%	25th

DATA TYPES

The data type defines the type of data stored in a memory location. The data type determines how much memory should be allocated for a variable. Various types of data such as integer constant, floating point constant, character constant etc. can be stored in memory during execution of a program. The data types in C language are classified mainly into three groups:

- Primitive data types or Primary Data types or Fundamental Data Types
- Derived data types
- User-defined data types

Primitive data types or Primary Data types or Fundamental Data Types

The data types that can be manipulated by machine instructions are called primitive data types. They are also called basic data types or simple data types or fundamental data types.

All C compilers supports five fundamental data types, namely integer (**int**), character (**char**), floating point (**float**), double-precision floating point (**double**) and void. Many of them also extended data types such as **long int** and **long double**.

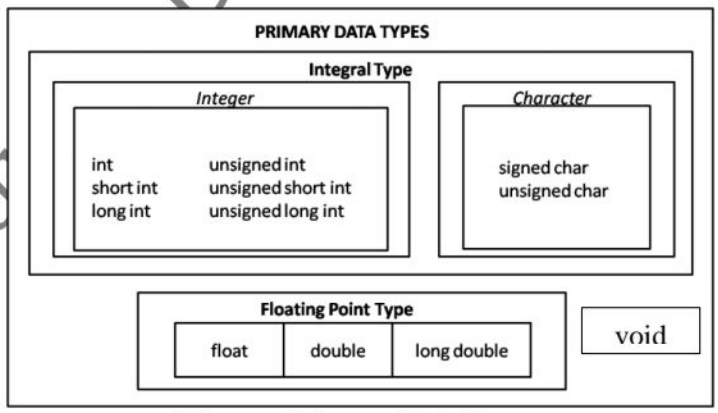


Figure: Primary Data Types

Integer Types:

Integers are whole numbers with a range of values supported by particular machine. Generally, integers occupy one word of storage, and since the word sizes of machines vary (typically, 16 or 32 bits) the size of an integer that can stored depends on the computer.

If we use a 16 bit word length, the size of the integer value is limited to the range -32768 to +32767 (that is, -2^{15} to $+2^{15}$ - 1). A signed integer uses one bit for sign and 15 bits for the magnitude of the number.

In order to provide some control over the range of numbers and storage space, C has three classes of integer storage, namely short int, int, and long int, in both signed and unsigned forms. Table below shows all the allowed combinations of basic types and qualifiers and their size and range on a 16-bit machine.

Size and Range of Data Types on a 16-bit Machine

Type	Size (bits)	Range
char or signed char	8	–128 to 127
unsigned char	8	0 to 255
int or signed int	16	–32,768 to 32,767
unsigned int	16	0 to 65535
short int or signed short int	8	–128 to 127
unsigned short int	8	0 to 255
long int or signed long int	32	–2,147,483,648 to 2,147,483,647
unsigned long int	32	0 to 4,294,967,295
float	32	3.4E – 38 to 3.4E + 38
double	64	1.7E – 308 to 1.7E + 308
long double	80	3.4E – 4932 to 1.1E + 4932

Floating Point Types:

Floating point (or real) numbers are stored in 32-bits (on all 16bit and 32 bit machines), with 6 digits of precision. Floating points numbers are defined in C by the keyword **float**. When the accuracy provided by a **float** number are not sufficient, the type **double** can be used to define the number. A **double** data type number uses 64 bits giving a precision of 14 digits. These are known as *double precision* numbers,

Floating Point Data Types

Type	Size(in bytes)	Range	Digits of Precision
float	4	$3.4 \cdot 10^{-38}$ to $3.4 \cdot 10^{38}$	7
double	8	$1.7 \cdot 10^{-308}$ to $1.7 \cdot 10^{308}$	15
long double	10	$3.4 \cdot 10^{-4932}$ to $3.4 \cdot 10^{4932}$	18

Void Types:

The **void** type has no values. This is usually used to specify the type of functions. The type of a function is said to be **void** when it does not return any value to the calling function. It can also play the role of a generic type, meaning that it can represent any of the other standard types.

Character Types:

A single character can be defined as a character (char) type data. Characters are usually stored in 8 bits (one byte) of internal storage. The qualifier signed or unsigned may be explicitly applied to char. While unsigned chars have values between 0 and 255, signed chars have values from -128 to 127.

DECLARATION OF VARIABLES

After designing suitable variable names, we must declare them to the compiler. Declaration does two things:

1. It tells the compiler what the variable name is.
2. It specifies what type of data the variable will hold.

The declaration of variables must be done before they are used in the program.

Primary Type Declaration:

A variable can be used to store a value of any data type. That is, the name has nothing to do with its type. The syntax for declaring a variable is as follows:

Data-type v1,v2,v3,…vn;

v1, v2, v3, …… vn are the names of variables. Variables are separated by commas. A declaration statement must end with a semicolon. For example, valid declarations are:

```
int count;
int number, total;
double ratio;
```

int and double are the keywords to represent integer type and real type data values respectively.

Below shows the various keywords and their equivalents data types:

- int - an integer; reflects size of integers on host machine

- float - single-precision floating point
- double - double-precision floating point
- char - character, a single byte
- unsigned char - Unsigned character
- signed short int - Signed short integer
- signed long int - Signed Long integer
- unsigned int - Unsigned integer
- unsigned short int - Unsigned Short Integer
- unsigned long int - Unsigned Long Integer
- long double - Extended double-precision floating point

Below program segment illustrates declaration of variables.

```
void main()
{
        / ****************** Declaration ********************/
        float x, y;
        int code;
        short int count;
        long int amount;
        double n;
        unsigned num;
        char c;
        /****************** End Declaration *****************/
}
```

USER-DEFINED TYPE DECLARATION

In C programming, a feature known as "type definition" is available which allows a programmer to define an identifier that represents an existing data type. The user defined identifier can be used later in the program to declare variables.

The general syntax of declaring a variable by user-defined type declaration is:

typedef type identifier;

Here, *type* is an existing data type and identifier is the "new name" given to the data type. Here, the new type is 'new' only in name but not the data type.

Consider the example:

```
typedef int age;
typedef float weight;
```

Here, *age* represents *int* and *weight* represent *float* which can be used later in the program to declare variables as follows:

```
age boy1,boy2;
weight b1,b2;
```

Here, *boy1* and *boy2* are declared as as integer data type and *b1* & *b2* are declared as floating integer data type.

The main advantage of using user-defined type declaration is that we can create meaningful data type names for increasing the readability of a program.

Another user-defined data type is enumerated data type. The general syntax of enumerated data type is:

enum identifier {value 1,value 2,...value n};

Here, *identifier* is a user-defined enumerated data type which can be used to declare variables that can have one of the values enclosed within the braces. The values inside the braces are known as enumeration constants. After this declaration, we can declare variables to be of this 'new' type as:

enum identifier v1, v2, ... vn;

The enumerated variables v1, v2, ... vn can only have one of the values value1, value2, ... valuen. The following kinds of declarations are valid:

```
v1=value5;
v3=value1;
```

User-defined Type Declaration Example

```
enum month {January, February, ..., December};
enum month day_st, day_end;
day_st = January;
day_end = December;
if (day_st == February)
day_end = November;
```

The compiler automatically assigns integer digits begriming with 0 to all the enumeration constants. That is, the enumeration constant *value1* is assigned 0, *value2* is assigned 1, and so on. However, the automatic assignments can be overridden by assigning values explicitly to thee enumeration constants.

For example:

```
enum mnth {January = 1, February, ..., December};
```

Here, the constant *January* is assigned value 1. The remaining values are assigned values that increase successively by 1.

The definition and declaration of enumerated variables can be combined in one statement. For example;

```
enum mnth {January, ... December} day_st, day_end;
```

DECLARATION OF STORAGE CLASS

Variables in C can have not only data type but also storage class that provides information about their location and visibility. The storage class decides the portion of the program within which the variables are recognized. Consider the following example:

```
/*  Example of storage Classes */
int m;
main()
{
        int i;
        float balance;
        ……..
        Function1();
}
Function1()
{
        int i;
        float sum;
        ……
        ……
}
```

The variable **m** which has been declared before the main is called ***global variable***. It can be used in all the functions in the program. It need not be declared in other functions. A global variable is also known as an ***external*** variable.

The variables i, balance and sum are called local variables because they are declared inside a function. Local variables are visible and meaningful only inside the functions in which they are declared. They are not known to other function.

Storage class determines the scope and lifetime of a variable.

There are 4 types of storage class:

1. automatic
2. external

3. static
4. register

Local Variable

The variables declared inside the function are automatic or local variables.

The local variables exist only inside the function in which it is declared. When the function exits, the local variables are destroyed.

```
int main() {
   int n; // n is a local variable to main() function
   ... .. ...
}

void func() {
  int n1; // n1 is local to func() function
}
```

Register Variable

The register keyword is used to declare register variables. Register variables were supposed to be faster than local variables.

However, modern compilers are very good at code optimization and there is a rare chance that using register variables will make your program faster.

Unless you are working on embedded system where you know how to optimize code for the given application, there is no use of register variables.

ASSIGNING VALUES TO VARIABLES

Variables are created for use in program statements such as,

Value = amount + inrate * amount;

Here variable Value is called the target variable.

Assignment Statement:

Values can be assigned to variables using the assignment operator = as follows:

variable_name = constant;

Examples are:

```
initial_value = 0;
final_value = 100;
balance     = 75.84;
yes = 'x';
```

C permits multiple assignments in one line. For example

Initial_value = 0; final_value=100;

are valid statements.

An assignment statement implies that the value of the variable on the left of the 'equal sign' is set equal to the value of the quantity (or the expression) on the right.

Year = Year + 1;

means that the 'new value' of year is equal to the 'old value' of Year plus 1.

It is also possible to assign a value to a variable at the time the variable is declared. This takes the following form:

Data_type variable_name = constant;

Some examples are:

int final_value = 100;
char yes = 'x';

The process of giving initial values to variables is called "initialization".

C permits the initialization of more than one statement using multiple assignment operators. For example the statements

p = q = s = 0;
x = y = z = MAX;

```
void  main ()
{
  /* variable definition: */
        int a, b;
        int c;
       float f;
   /* actual initialization */
        a = 10;
        b = 20;
       c = a + b;
       printf("value of c : %d \n", c);
       f = 70.0/3.0;
       printf("value of f : %f \n", f);
}
```

OPERATORS

The symbols which are used to perform logical and mathematical operations in a C program are called C operators.

These C operators join individual constants and variables to form expressions.

Operators, functions, constants and variables are combined together to form expressions.

Consider the expression A + B * 5. where, +, * are operators, A, B are variables, 5 is constant and A + B * 5 is an expression.

TYPES OF C OPERATORS:

C language offers many types of operators. They are,

1. Arithmetic operators
2. Assignment operators
3. Relational operators
4. Logical operators
5. Bit wise operators
6. Conditional operators (ternary operators)
7. Increment/decrement operators
8. Special operators

ARITHMETIC OPERATORS:

C provides all the basic arithmetic operators. They are listed below:

Operator	Description	Example
+	Adds two operands.	A + B = 30
−	Subtracts second operand from the first.	A − B = -10
*	Multiplies both operands.	A * B = 200
/	Divides numerator by de-numerator.	B / A = 2
%	Modulus Operator and remainder of after an integer division.	B % A = 0

The operators +, -, %, *, and / all work the same way as they do in other languages. The unary minus operator, in effect, multiplies its single operand by -1. Therefore, a number preceded by a minus sign changes its sign.

Integer division truncates any fractional part. The modulo division operation produces the remainder of an integer division.

Integer Arithmetic:

When both the operands in a single arithmetic expression such as a + b are integers, the expression is called as integer expression, and the operation is called integer arithmetic. Integer arithmetic always yields an integer value.

A = 20
B = 10
A – B = 10
A + B = 30
A * B = 200

During integer division, if both the operands are of the same sign, the result is truncated towards zero. If one of them is negative, the direction of truncation is implementation dependent. That is,

6/7 = 0 and -6/-7 = 0

Similarly, during modulo division, the sign of the result is always the sign of the first operand (the divided). That is

- 14 % 3 = -2
- 14 % -3 = -2
14 % -3 = 2

Real Arithmetic:

An arithmetic operation involving only real operands is called real arithmetic. A real operand may assume values either in decimal or exponential notation. Since floating point values are rounded to the number of significant digits permissible, the final value is an approximation of the correct result.

Ex:

X = 6.0/7.0
Y = -2.0 / 3.0 = -0.666667

The operator % cannot be used with real operands.

Mixed-mode Arithmetic:

When one of the operands is real and the other is integer, the expression is called a mixed-mode arithmetic expression. If either operand is of the real type, then only the real operation is performed and the result is always a real number. Thus

15 / 10.0 = 1.5

Where as

15/ 10 = 1

RELATIONAL OPERATORS

We often compare two quantities and depending on their relation take certain decisions. For example, we my compare the age of two persons, or the price of two items, and so on. These comparisons can be done with the help of relational operators.

• The output of relational expression is either true (1) or false (0).
• For example

a>b //If a is greater than b, then a>b returns 1 else a>b returns 0.

• The expression containing a relational operator is returned as a relational expression.
  • The 2 operands may be constants, variables or expressions.
• There are 6 relational operators:

Operator	Meaning of Operator	Example
>	Greater than	5 > 3 returns true (1)
<	Less than	5 < 3 returns false (0)
>=	Greater than or equal to	5 >= 3 returns true (1)
<=	Less than or equal to	5 <= 3 return false (0)
==	Equal to	5 == 3 returns false (0)
!=	Not equal to	5 != 3 returns true (1)

• For ex:

Condition	Return values
2>1	1 (or true)
2>3	0 (or false)
3+2<6	1 (or true)

• Example: Program to illustrate the use of all relational operators.

```
#include<stdio.h>
void main()
{
printf("4>5 : %d \n", 4>5);
printf("4>=5 : %d \n", 4>=5);
printf("4<5 : %d \n", 4<5);
printf("4<=5 : %d \n", 4<=5);
printf("4==5 : %d \n", 4==5);
printf("4!=5 : %d ", 4!=5);
}
```

Output:

```
4>5 : 0
4>=5 : 0
4<5 : 1
4<=5 :1
4==5 : 0
4!=5 : 1
```

LOGICAL OPERATORS

• These operators are used to perform logical operations like negation, conjunction and disjunction.
• The output of logical expression is either true(1) or false(0).
  • An expression
• There are 3 logical operators:

Operator	Meaning	Example
&&	Logical AND	If c=5 and d=2 then ((c==5) && (d>5)) returns false.
||	Logical OR	If c=5 and d=2 then ((c==5) || (d>5)) returns true.
!	Logical NOT	If c=5 then ! (c==5) returns false.

• All non -zero values(i.e. 1, -1, 2, -2) will be treated as true.
While zero value(i.e. 0) will be treated as false.

Example:

a > b && x == 10

An expression of this kind, which combines two or more relational expressions, is termed as a logical expression or a compound relational expression.

Truth table

Truth table

A	B	A&&B	A||B	!A
0	0	0	0	1
0	1	0	1	
1	0	0	1	0
1	1	1	1	

• Example: Program to illustrate the use of all logical operators.

```
#include<stdio.h>
void main()
{
    clrscr ( );
    printf("7 && 0 : %d \n", 7 && 0 );
    printf("7 || 0 : %d \n", 7 || 0 );
    printf(" !0 : %d", !0 );
}
```

Output:

```
7 && 0 : 1
7 || 0 : 1
!0 : 1
```

• Example: Program to illustrate the use of both relational & Logical operators.

```
#include<stdio.h>
void main()
{
        printf("5>3 && 5<10 : %d \n", 5>3 && 5<10);
        printf(" 8<5 || 5= =5 : % d \n", 8<5 || 5==5);
                printf("!(8 = =8) : %d ", !(8==8) ;
}
```

Output:

```
5>3 && 5<10 : 1
8<5 || 5= =5 : 1
!(8 = =8) : 0
```

ASSIGNMENT OPERATOR

Assignment operators are used to assign the result of an expression to a variable. We have seen the usual assignment operator, '='.

In addition, C has a set of 'shorthand' assignment operators of the form

v op = exp;

Where v is a variable, exp is an expression and op is a C binary arithmetic operator. The operator op= is known as the shorthand assignment operator.

The assignment statement

v op = exp;

is equivalent to

v = v op (exp);

with v evaluated only once. Consider as example

x += y + 1;

This is same as the statement

x = x + (y + 1);

The shorthand operator += means 'add y+1 to x' 'increment x by y+1'.

- This operator assigns the value in right side to the left side.
- The syntax is shown below:

  variable = expression;
- For ex:

  c=5; //5 is assigned to c

  b=c; //value of c is assigned to b

  5=c; // Error! 5 is a constant.
- The operators such as +=,*= are called shorthand assignment operators.
- For ex,

  a=a+10: can be written as a+=10;
- In the same way, we have:

Operator	Example	Same as
-=	a-=b	a=a-b
*=	a*=b	a=a*b
/=	a/=b	a=a/b
%=	a%=b	a=a%b

The use of the shorthand assignment operators has three advantages:

1. What appears on the left-hand side need not be repeated and therefore it becomes easier to write.
2. The statement is ore concise and easier to read.
3. The statement is more efficient.

These advantage may be appreciated if we consider a slightly more involved statement like

value (5 * j – 2) = value (5 * j – 2) + delta;

with the help of the += operator, this can be written as follows:

value (5 * j – 2) += delta;

It is a easier to read and understand and is more efficient because the expression 5 * j – 2 is evaluated only once.

INCREMENT OPERATOR & DECREMENT OPERATOR

C allows two very useful operators not generally found in other languages. These are the increment and decrement operators:

++ and –

The operator ++ adds 1 to the operand, while – subtracts 1.

Increment Operator

• ++ is an increment operator.

• As the name indicates, increment means increase, i.e. this operator is used to increase the value of a variable by 1.

• For example:

If b=5

then b++ or ++b; // b becomes 6

• The increment operator is classified into 2 categories:

1) Post increment Ex: b++

2) Pre increment Ex: ++b

• As the name indicates, post-increment means first use the value of variable and then increase the value of variable by 1.

• As the name indicates, pre-increment means first increase the value of variable by 1 and then use the updated value of variable.

• For ex:

If x is 10,

then z= x++; sets z to 10

but z = ++x; sets z to 11

Example: Program to illustrate the use of increment operators.

```
#include<stdio.h>
void main()
{
    int x=10,y = 10, z ;
    z= x++;
    printf(" z=%d x= %d\n", z, x);
    z = ++y;
    printf(" z=%d y= %d", z, y);
}
```

Output:

z=10 x=11

z=11 y=11

DECREMENT OPERATOR

• -- is a decrement operator.

• As the name indicates, decrement means decrease, i.e. this operator is used to decrease the value of a variable by 1.

• For example:

If b=5

then b-- or --b; // b becomes 4

• Similar to increment operator, the decrement operator is classified into 2 categories:

1) Post decrement Ex: b--

2) Pre decrement Ex: --b

• For ex:

If x is 10,

then z= x--; sets z to 10,

but z = --x; sets z to 9.

Example: Program to illustrate the use of decrement operators.

```
void main()
{
int x=10,y = 10, z ;
z= x--;
printf(" z=%d x= %d\n", z, x);
z = --y;
printf(" z=%d y= %d", z, y);
}
```

Output:

```
z=10 x=9
z=9 y=9
```

Rules for ++ and – operators

- Increment and decrement operators are unary operators and they require variable ad their operands.
- When postfix ++ (or --) is used with a variable in an expression, the expression is evaluated first using the original value of the variable and then the variable is incremented (or decremented) by one.
- When prefix ++ (or --) is used in an, the variable is incremented (or decremented) first and then the expression is evaluated using the new value of the variable.
- The precedence and associativity of ++ and – operator are the same as those of unary + and unary -.

CONDITIONAL OPERATOR

• The conditional operator is also called ternary operator it takes three operands.
• Conditional operators are used for decision making in C.
• The syntax is shown below:

(exp1)? exp2: exp3;

where exp1 is an expression evaluated to true or false;

If exp1 is evaluated to true, exp2 is executed;
If exp1 is evaluated to false, exp3 is executed.

Example: Program to find largest of 2 numbers using conditional operator.

```
#include<stdio.h>
void main()
{
    int a,b, max ;
    printf(" enter 2 distinct numbers \n");
    scanf("%d %d", &a, &b);
    max=(a>b)? a : b;
    printf(" largest number =%d ", max);
}
```

Output:

```
enter 2 distinct numbers
3 4
largest number = 4
```

BITWISE OPERATORS

C has a distinction of supporting special operators known as bitwise operators for manipulation of data at bit level. These operators are used for testing the bits, or shifting them right or left.

Bitwise operators may not be applied to **float** or **double**.

- These operators are used to perform logical operation (and, or, not) on individual bits of a binary number.

• There are 6 bitwise operators:

Operators	Meaning of operators
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
~	Bitwise complement
<<	Shift left
>>	Shift right

Truth Table

Truth Table

A	B	A&B	A|B	A^B	~A
0	0	0	0	0	1
0	1	0	1	1	
1	0	0	1	1	0
1	1	1	1	0	

- Ex for ~ (bitwise complement)
  a = 13 0000 1101
  ~a= 11110010
- Ex for || (bitwise OR)
  a = 13 0000 1101
  b = 6 0000 0110
  a | b 0000 1111

Ex for & (bitwise AND)
a = 13 0000 1101
b = 6 0000 0110
a & b 0000 0100

Ex for ^ (bitwise xor)
a = 13 0000 1101
b = 6 0000 0110
a ^ b 0000 1011

• The operator that is used to shift the data by a specified number of bit positions towards left or right is called shift operator.
• The syntax is shown below for << The syntax is shown below for >>
b=a << num; b=a >> num;
where a is value to be shifted
num is number of bits to be shifted
• Ex for <<(left shift): Ex for >>(right shift):
a = 13 0000 1101 a = 13 0000 1101
b=a<<1 0001 1010 b=a<<1 0000 0110

SPECIAL OPERATORS

C supports some special operators of interest such as comma operator, **sizeof** operator, pointer operators (& and *) and member selection operators (. And ->). T

The Comma Operator:

The comma operator can be used to link the related expressions together. A comma-linked list of expressions are evaluated left to right and the value of right-most expression is that value of the combined expression. For example, the statement

value = (x = 10, y= 5, x + y)

first assigns the value 10 to x, then assigns 5 to y, and finally assigns 15 to value. Since comma operator has the lowest precedence of all operators, the parentheses are necessary.

The sizeof Operator:

The sizeof is a compile time operator and, when used with an operand, it returns the number of bytes the operand occupies. The operand may be a variable, a constant or a data type qualifiers.

Example:

M = sizeof(sum);
N = sizeof(long int);

ARITHMETIC EXPRESSION

An arithmetic expression is a combination of variables, constants, and operators arranged as per the syntax of the language.

Some examples are:

Algebraic Expression	C expression
a x b – c	a * b – c
(m+n)(x+y)	(m + n) * (x +y)
(ab/c)	a * b/c
$3x^2 + 2x + 1$	3 * x * x + 2 *x + 1

EVALUAION OF EXPRESSION

Expressions are evaluated using an assignment statement of the form:

Variable = expression

Variable is any valid C variable name. When the statement is encountered, the expression is evaluated first and the result then replaces the previous value of the variable on the left hand side.

All variables used in the expression must be assigned values before evaluation is attempted. Examples of evaluation statements are:

```
x = a * b – c;
y = b / c * a;
z = a – b / c + d;
```

The blank space around an operator is optional and adds only to improve readability. When these statements are used in a program, the variables a, b, c, and d must be defined before they are used in the expression.

PRECEDENCE OF ARITHMETIC OPERATORS

An arithmetic expression without parentheses will be evaluated from left to right using the rules of precedence of operators. There are two distinct priority levels of arithmetic operators in C:

High priority * / %

Low priority + -

The basic evaluation procedure includes 'two' left-to-right passes through the expression. During the first pass, the high priority operators (if any) are applied as they are encountered. During the second pass, the low priority operators (if any) are applied as they are encountered.

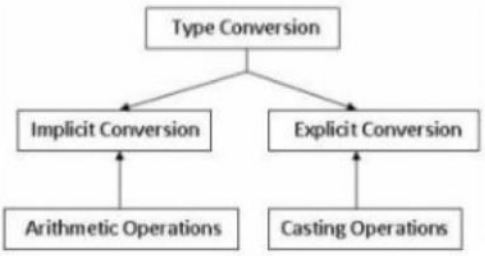
Rules for Evaluation of Expression

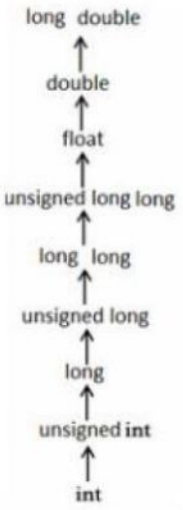
1. First, parenthesized sub expressions from left to right are evaluated.
2. If parentheses are nested, the evaluation begins with the innermost sub-expression.
3. The precedence rule is applied in determining the order of application of operators in evaluating sub-expressions.
4. The associativity rule is applied when two or more operators of the same precedence level appear in a sub-expression.
5. Arithmetic expressions are evaluated from left to right using the rules of precedence.
6. When parentheses are used, the expressions within parentheses assume highest priority.

TYPE CONVERSION

• Type conversion is used to convert data of one type to data of another type.

• Type conversion is of 2 types as shown in below figure:





datatype hierarchy

IMPLICIT TYPE CONVERSION

C permits mixing of constants and variables of different types in an expression. C automatically converts any intermediate values to the proper type so that the expression can be evaluated without losing any significance. This automatic conversion is known as *implicit type Conversion*.

During evaluation it adheres to very strict rules of type conversion. If the operands are of different types, the 'lower' type is automatically converted to the 'higher' type before the operation proceeds. The result is of the higher type.

The sequence of rules that are applied while evaluating expressions.

All short and char are automatically converted to int; then

1. If one of the operands is **long double**, the other will be converted to **long double** and the result will be **long double**;
2. Else, if one of the operands is **double**, the other will be converted to **double** and the result will be double;
3. Else, if one of the operands is **float**, the other will be converted to **float** and result will be **float**;
4. Else, if one of the operands is **unsigned long int**, the other will be converted to **unsigned long int** and result will be **unsigned long int**;
5. Else, if one of the operands is **long int**, the other will be converted to **unsigned int,** then
   (i) If **unsigned int** can be converted to **long int**, the **unsigned int** operand will be converted as such and the result will be **long int**;
   (ii) Else, both operands will be converted to **unsigned long int** operand will be **unsigned long int**;
6. Else, if one of the operands is **long int**, the other will be converted to **long int** and result will be **long int**;
7. Else, if one of the operands is **unsigned int**, the other will be converted to **unsigned int** and result will be **unsigned int**;

Note that some versions of C automatically convert all floating-point operands to double precision

The final result of an expression is converted to the type of the variable on the left of the assignment sign before assigning the value to it. However, the following changes are introduced during the final assignment.

1. **float** to **int** causes truncation of the fractional part.
2. **double** to **float** causes rounding of digits.
3. **long int** to **int** causes dropping of the excess higher order bits.

• Example: Program to illustrate implicit conversion.

```
#include<stdio.h>
void main()
{
int a = 22, b=11;
float d ;
d=b/c;
printf("d Value is : %f ", d );
}
```

Output:

```
d Value is : 0.500000
```

EXPLICIT CONVERSION

• When the data of one type is converted explicitly to another type with the help of some pre-defined functions, it is called as explicit conversion.

• There may be data loss in this process because the conversion is forceful.

• The syntax is shown below:

```
data_type1 v1;
data_type2 v2= (data_type2) v1;
```

where v1 can be expression or variable

• For ex:
float b=11.000000;
int c = 22;
float d = b/(float)c =11.000000/22.000000 = 0.500000
• Example: Program to illustrate explicit conversion.

```
#include<stdio.h>
void main()
{
        float b=11.000000;
int c = 22;
float d;
d=b/(float)c;
printf("d Value is : %f ", d );
}
```

Output:
d Value is : 0.500000