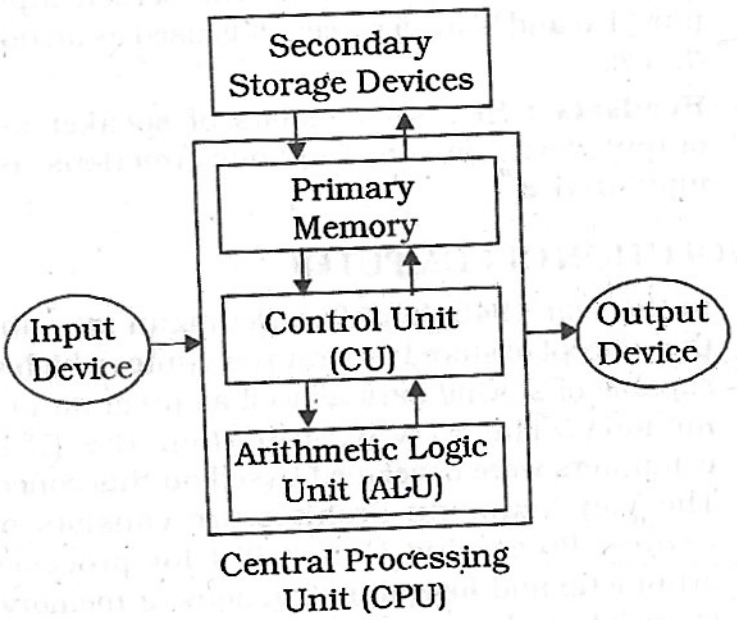
1 COMPUTER SYSTEM
(NCERT CLASS 11)

INTRODUCTION TO COMPUTER SYSTEM

- A computer is an electronic device, under the control of instructions stored in its memory that can accept data (input), process the data according to specified rules(Program) on processor & produces information (output), and store the information for future use.
- A computer along with additional hardware and software together is called a computer system.
- A computer system primarily comprises a central processing unit (CPU), memory, input/output devices and storage devices and all these components function together as a single unit to deliver the desired output.
- Following figure shows the block diagram of a computer system.The directed lines represent the flow of data and signal between the components.



- **Central processing unit (CPU) :** It is the electronic circuitry of a computer which carriesout the actual processing and usually referred as the brain of the computer. It is commonly called processor also. Physically, a CPU can be placed on one or more microchips called integrated circuits (IC). The Ics comprise semiconductor materials.CPU is also popularly known as microprocessor.
- The CPU receive instructions and data through programs and then the CPU fetches the program and data from the memory and performs arithmetic and logic operations as per the given instructions and stores the result back to memory.
- At the time of processing, the CPU stores the data as wellas instructions in its local memory which is known as registers. Registers are part of the CPU chip and they are limited in size and number. Different registers are used for storing data, instructions or intermediate results.
- The CPU has two main components other than register— Arithmetic Logic Unit (ALU) and Control Unit (CU).
- Arithmetic Logic Unit performs basic arithmetic operations such as addition and subtraction Performs logical operations such as AND, OR, and NOT. Most modern ALUs have a small amount of special storage units called registers that can be accessed faster than main memory.
- Control unit organizes the computer to work computer as single unit & generates control signals for various devices regarding read/write or execute operation. Or, we can say that Control Unit controls sequential instruction execution, interprets instructions and guides data flow through the computer's memory, ALU and input or output devices.
- **Input Devices :** In a simple word, it is a device through which data and programs from the outside world enter the computer system.Or we can say, it is the devices through which control signals are sent to a computer.

  Input devices convert the input data into a digital form which is acceptable by the computer system. Data entered through input device is temporarilystored in the main memory (also called RAM) of the computer system. For permanent storage and future use, the data as well as instructions are stored permanently in additional storage locations called secondary memory.

  Specially designed braille keyboards is used to help the visually impaired people for entering data into a computer. Besides that, we can also now enter data through voice, for example, we can use Google voice search to search the web where we can input the search string through our voice.

Examples of input devices are :

- **Keyboard :** It is an input device which sends data in to the computer. The data send depends on the key pressed by the user.
- **Mouse:** A mouse is a small handheld input device which controls a cursor in a graphical user interface. It can move and select text, files, folders etc. on our computer according to the user input.
- **Scanner:** Scanner optically reads and document, file or image and then changes it into digital signal and sends to the computer.

- **OMR** : Optical mark recognition/ reader, is used to read marks on a document and send them to computer.
- **OCR** : OCR stands for optical character Recognition, is an input device which reads printed text and sends that to computer.
- **MICR :** Magnetic Ink Character Reader is an input device which generally finds application is banks to process cheques.
- **Microphone :** It receives audio generated by some input source and sends it to a computer.
- **Webcam** : It sends the captured images to a computer.
- **Graphics Tablets** : This input device is used to draw using hand.
- **Trackballs :** An upside down mouse,encased within a socket. It is a cursor control device.
- **Barcode reader :** It is used to read the barcode of various items and feed the same to computer.
- **Gamepad** : Also known as joy pad is the input controller for video games.
- **Joystick** : these input devices are used to control video games.
- **Output Devices :** A device through which results stored in the computer memory are made available outside the computer system. Or we can say, it is the device that receives data from a computer system for display, physical production, etc.

  Output devices converts digital information into humanunderstandable form. A printer is the most commonly used device to get output in physical (hardcopy) form. Three types of commonly used printers are inkjet, laserjet and dot matrix. Now-a-days, there is a new type of printer called 3D-printer, which is used to build physical replica of a digital 3D design. These printers are being used in manufacturing industries to create prototypes of products. Their usage is also being explored in the medical field, particularly for developing body organs.

  A braille display monitor is useful for a visually challenged person to understand the textual output generated by computers.Example of output devices are:-
- **Monitor :** A monitor is an output device that is responsible for receiving data from a computer and displaying that information as text or images for users to see.
- **Speakers :** Receives sound signal from a computer and then plays that sound signal and thus we hear songs or music or any other audio.
- **Projector :** Gets data from a computer and displays or projects the same information onto a screen or a wall. Projector cannot directly accept data from a user and send that data to another device.
- **Both Input / Output Devices :** An input/output device is capable of receiving data from users or another devices and also sending data to another devices or computers. That means a devices which can be used as both input device and output device are called Input / Output (I/O) devices. Some examples of input/output devices are as:
- **USB drive :** Also known as pen drive or flash stick works as both input device to computer and as an output device. USB drives receive or save data from a computer as an input and it can also send data to a computer or another device.
- **Facsimile** : Facsimile or FAX machine has a scanner which is an input device and a small printer to provide output.
- **Modems :** It is used to transmit and receive data from one computer to another computer or other devices using telephone lines.
- CD-RW drives and DVD-RW drives: Receives data from a computer as input to copy onto and save into writable CD or DVD. We also use CDs or DVDs to transfer data to a computer.
- **Touch Screen :** Touch screen is both input and output device. By touching the screen input is provided and being a screen, it is used as an output device.
- **Headsets** : Headset consists of speaker as an output device and microphone functions as an input device

EVOLUTION OF COMPUTER

- In the year 1945 John Von Neumann introduced the concept of stored program computer which was capable of storing data as well as program in the memory. The EDVAC and then the ENIAC computers were developed based on this concept. The Von Neumann architecture consists of a Central Processing Unit (CPU) for processing arithmetic and logical instructions, a memory to store data and programs, input and output devices and communication channels to send or receive the output data. Electronic Numerical Integrator and Computer (ENIAC) is the first binary programmable computer based on Von Neumann architecture.
- During the 1970s, Large Scale Integration (LSI) ofelectronic circuits allowed integration of complete CPU on a single chip, called microprocessor. Moore's Law predicted exponential growth in the number of transistors that could be assembled in a single microchip.
- In 1980s, the processing power of computers increased exponentially by integrating around 3 million components on a small-sized chip termed as Very Large Scale Integration (VLSI). Further advancement in technology has made it feasible

to fabricate high density of transistors and other components (approx 106 components) on a single IC called Super Large Scale Integration (SLSI).

- IBM introduced its first personal computer (PC) for the home user in 1981 and Apple introduced Macintosh machines in 1984.
- By the introduction of Graphical User Interface (GUI) based operating systems by Microsoft and others in place of computers with only command line interface, like UNIX or DOS increases the popularity of the personal computer.
- Around 1990s, the growth of WorldWide Web (WWW) further accelerated mass usage of computers.Further, with the introduction of laptops, personal computing was made portable to a great extent. This was followed by smartphones, tablets and other personal digital assistants. These devices have leveraged the technological advancements in processor miniaturisation, faster memory, high speed data and connectivity mechanisms.
- The next wave of computing devices includes the wearable gadgets, such as smart watch, lenses, headbands, headphones, etc. Further, smart appliances are becoming a part of the Internet of Things (IoT), by leveraging the power of Artificial Intelligence (AI).

COMPUTER MEMORY

- Memory is used to store the data and instructions for processing in a computer system.
- **Units of Memory :** Bytes are grouped together to make bigger chunks or units of memory.Computer system uses binary numbers to store and process data. The binary digits 0 and 1, which are the basic units of memory, are called bits. These bits are grouped together to form words. A 4-bit word is called a Nibble. Examples of nibble are 1001, 1010, 0010, etc. A two nibble word, i.e., 8-bit word is called a byte, for example, 01000110, 01111100, 10000001, etc.
- **Types of Memory :** Computers have two types of memory — primary and secondary.

  Primary memory : It is an essential component of a computer system as program and data are loaded into the primary memory before processing. The CPU interacts directly with the primary memory to perform read or write operation. It is of two types viz. (I) Random Access Memory (RAM) and (II) Read Only Memory (ROM).

(I) **Random Access Memory (RAM)** – It is a type of volatile memory that is stores information on an integrated circuit which hold the data mainly when the program is being executed by the CPU. As it is volatile in nature so it can't store data permanently. RAM is usually referred to as main memory and it is faster than the secondary memory or storage devices.

(II) **Read Only Memory (ROM)** – It is a non-volatile memory chip in which data are stored permanently, and can not be altered by the programmerwhichmeans its contents are not lost even when the power is turned off. For example, the startup program (boot loader) that loads the operating system into primary memory, is stored in ROM.

Cache Memory: It is a small high speed memory, which is used to increase the speed of processing by making current programs and data available to the CPU at a rapid rate.It is the volatile computer memory which is very nearest to the CPU,so also called CPU memory, and is between CPU and RAM all the Recent Instructions are Stored into the Cache Memory. It is the fastest memory that provides high-speed data access to a computer microprocessor.

Secondary Memory : It is a storage, which suppliments the main memory of a computer. Often refferred to as secondary storage, this section of computer's memory is nonvolatile and has low cost per bit stored, but it generally has an operating speed far slower than that of the primary storage.Examples of secondarymemory devices include Hard Disk Drive (HDD), CD/DVD, Memory Card, etc.

DATA TRANSFER BETWEEN MEMORY AND CPU

- Data are transferred between different components of a computer system using physical wires called bus. Bus is of three types :-

  (I) **Data bus :** Used to transfer data between different components.

  (II) **Address bus :** Used to transfer addresses between CPU and main memory.The address of the memory location that the CPU wants to read or write from is specified in the address bus.

  (iii) **Control bus :** Used to communicate control signals between different components of a computer.

  All these three buses collectively make the system bus.
- CPU interacts directly with main memory therefore any data entered from input device or the data to be accessed from hard disk needs to be placed in the main memory for further processing. The data is then transferred between CPU and main memory using bus.
- CPU may require to read data from main memory or write data to main memory, a data bus is bidirectional. But the control bus and address bus are unidirectional. To write data into memory, the CPU places the data on the data bus, which is then written to the specific address provided through the address bus.

- In case of read operation, the CPU specifies the address, and the data is placed on the data bus by a dedicated hardware, called memory controller. The memory controller manages the flow of data into and out of the computer's main memory.

MICROPROCESSORS

- A processor (CPU) which is implemented on a single microchip is called microprocessor. Microprocessor is a small-sized electronic component inside a computer that carries out various tasks involved in data processing as well as arithmetic and logical operations. These days, a microprocessor is built over an integrated circuit comprising millions of small components like resistors, transistors and diodes.
- **Microprocessor Specifications :**
  (I) **Word Size :** Word size is the maximum number of bits that a microprocessor can process at a time. At present, the minimum word size is 16 bits and maximum word size is 64 bits.
  (II) **Memory Size :** Depending upon the word size, the size of RAM varies.Initially, As word size increased to 64 bits, it has become feasible to use RAM of size upto 16 Exabytes (EB).
  (III) **Clock Speed :** Computers have an internal clock that generates pulses (signals) at regular intervals of time. Clock speed simply means the number of pulses generated per second by the clock inside a computer. The clock speed indicates the speed at which the computer can execute instructions. Earlier, it was measured in Hertz (Hz) and Kilohertz (kHz). But with advancement in technology and chip density, it is now measured in Gigahertz (GHz), i.e., billions of pulses per second.
  (IV) **Cores :** Core is a basic computation unit of the CPU. CPU with two, four, and eight cores is called dual-core, quad-core and octa-core processor, respectively.
- **Microcontrollers :** It is a small computing device which has a CPU, a fixed amount of RAM, ROM and other peripherals all embedded on a single chip as compared to microprocessor that has only a CPU on the chip. Examples of microcontrollers are Keyboard, mouse, washing machine, digital camera, pendrive, remote controller, microwave etc. The simple use of microcontroller has permitted repetitive execution of tedious tasks automatically without any human intervention, thereby saving precious time.

DATA AND INFORMATION

- Data are raw numbers or other findings which, by themselves, are of limited value. Information is data that has been converted into a meaningful and useful context.
- **Data and Its Types :** A computer system has many input devices, which provide it with raw data in the form of facts, concepts, instructions, etc., Internally everything is stored in binary form (0 and 1), but externally, data can be input to a computer in the text form consisting of English alphabets A–Z, a–z, numerals 0 – 9, and special symbols like @, #, etc.Primarily, there are three types of data.
  (I) **Structured Data :** Data which follows a strict record structure and is easyto comprehend is called structured data. Such data with pre-specified tabular format may be stored in a data file to access in the future.
  (II) **Unstructured Data :** Data which are not organised in a pre-defined record format is called unstructured data. Examples include audio and video files, graphics, text documents, social media posts, satellite images, etc. Such data are unstructured as they consist of textual contents as well as graphics, which do not follow a specific format.
  (III) **Semi-structured Data :** Data which have no well-defined structure but maintains internal tags or markings to separate data elements are called semi-structured data. Examples include email document, HTML page, comma separated values (csv file), etc.
- **Data Capturing :** It involves the process of gathering data from different sources in the digital form. This capturing may vary from simple instruments like keyboard, barcode readers used at shopping outlets, comments or posts over social media, remote sensors on an earth orbiting satellite, etc.
- **Data Storage :** It is the process of storing the captured data for processing later. Now-a-days data is being produced at a very high rate, and therefore data storage has become a challenging task.
- **Data Retrieval :** It involves fetching data from the storage devices, for its processing as per the user requirement. As databases grow, the challenges involved in search and retrieval of the data in acceptable time, also increase. Minimising data access time is crucial for faster data processing.
- **Data Deletion and Recovery :** Deleting digitally stored data means changing the details of data at bit level, which can be very timeconsuming. Therefore, when any data is simply deleted, its address entry is marked as free, and that much space is shown as empty to the user, without actually deleting the data.

  In case data gets deleted accidentally or corrupted, there arises a need to recover the data. Recovery of the data is possible only if the contents or memory space marked as deleted have not been overwritten by some other data. Data recovery is a process of retrieving deleted, corrupted and lost data from secondary storage devices.

There are usually two security concerns associated with data:-

(I) Its deletion by some unauthorised person or software.

(II) unwanted recovery of data by unauthorised user or software.

SOFTWARE

- Software is a generic term for organized collections of computer data and instructions,often broken into two major categories: system software thatprovides the basic nontaskspecific functions of the computer, and application software which is used by users to accomplish specific tasks.

  Hardware : Computer hardware is the collection of physical elements/parts that constitutes a computer system,such as the monitor, mouse, keyboard, computer data storage, hard drive disk (HDD), system unit (graphic cards, sound cards, memory, motherboard and chips), etc. all of which are physical objects & can be touched.

- **Need of Software :** Purpose of a software is to make the computer hardware useful and operational. A software knows how to make different hardware components of a computer work and communicate with each other as well as with the end-user. Software acts as an interface between human users and the hardware.

  Depending on the mode of interaction with hardware and functions to be performed, the software can be broadly classified into three categories viz:-

(I) **System software :** The software that provides the basic functionality to operate a computer by interacting directly with its constituent hardware is termed as system software.A system software knows how to operate and use different hardware components of a computer. It provides services directly to the end user, or to some other software. Examples of system software include operating systems, system utilities, device drivers, etc.

(a) Operating system is a system software that operates the computer. An operating system is the most basic system software, without which other software cannot work. The operating system manages other application programs and provides access and security to the users of the system. Some of the popular operating systems are Windows, Linux, Macintosh, Ubuntu, Fedora, Android, iOS, etc.

(b) **System Utilities :** Software used for maintenance and configuration of the computer system is called system utility. Some system utilities are shipped with the operating system for example disk defragmentation tool, formatting utility, system restore utility, etc. Another set of utilities are those which are not shipped with the operating system but are required to improve the performance of the system, for example, anti-virus software, disk cleaner tool, disk compression software, etc.

(c) **Device Drivers :** As the name signifies, the purpose of a device driver is to ensure proper functioning of a particular device. When it comes to the overall working of a computer system, the operating system does the work. The device driver acts as an interface between the device and the operating system. It provides required services by hiding the details of operations performed at the hardware level of the device. Just like a language translator, a device driver acts as a mediator between the operating system and the attached device.

(II) **Programming tools :** A programming language is a vocabulary and set of grammatical rules for instructing a computer or computing device to perform specific tasks. In order to get some work done by the computer, we need to give instructions which are applied on the input data to get the desired outcome.

(a) **Classification of Programming Languages :** Two major categories of computer programming languages are low-level languages and high-level languages.

Low-level languages are machine dependent languages and include machine language and assembly language. Machine language uses 1s and 0s to write instructions which are directly understood and executed by the computer.

High level languages are machine independent and are simpler to write code into. Instructions are using English like sentences and each high level language follows a set of rules, similar to natural languages. However, these languages are not directly understood by the computer. Hence, translators are needed to translate high-level language codes into machine language. Examples of high level language include C++, Java, Python, etc.

(b) **Language Translators :** The language translators are used to convert high-level language programs into low-level language programs. They are classified into three catgories:

Assembler: Assembler is used to convert assembly language programs to machine language. The program written in the assembly or high-level language is known as source code. The assembles convert this code into machine language that can be read by the machine. The assembler can understand the microprocessor commands written in the program.

Compiler: Compiler converts high-level language program into machine language. It translates the source code into machine code and creates an object file. Once the code is translated, it brings the information from the object file. It compiles the whole program once and then reports the errors.

Interpreter: The interpreter executes the source code line by line. It stops the program if any error is reported then and there only. It accesses the source code every time when you run the program.

(c) Programming development tools are used to write programs, design interfaces, and deploy a program using high-level programming languages. As we are aware that the computer doesn't know or understand our language. So these programming language tools help to convert the program into machine language through language translators.

The IDE (Integrated Development Environment) tools provide the facility to write the source code, design the interface, and debugging. It is an integrated package of code editor, drag and drop tools to design interface and debugger. So these types of software allow to write code, design the software and debug the program for errors.

(III) **Application software :** The application software provides specific requirements for the end-users. This software runs on top of the system software. There are two broad categories of application software:

(a) **General Purpose Application Software :** General purpose application software is used to perform general-purpose tasks like documentation, spreadsheet-based tasks, presentation, graphics, or photo editing. Examples are MS word, MS Excel, MS Powerpoint, Adobe Photoshop, etc.

(b) **Customized Application Software :** Customized application software is tailormade application software designed to meet the specific requirements of the organization or an individual. This software is made as per the need of the organization and individual. For example ERP (Enterprise Resource Planning) software, accounting software, school management system software etc.

- Proprietary or Free and Open Source Software:- The developers of some application software provide their source code as well as the software freely to the public, with an aim to develop and improve further with each other's help. Such software is known as Free and Open Source Software (FOSS). For example, the source code of operating system Ubuntu is freely accessible for anyone with the required knowledge to improve or add 'new functionality. More examples of FOSS include Python, Libreoffice, Openoffice, Mozilla Firefox, etc. Sometimes, software are freely available for use but source code may not be available. Such software are called freeware. Examples of freeware are Skype, Adobe Reader, etc.

OPERATING SYSTEM

Operating System mainly operated the entire computer system. Without an operating system, the computer system cannot work. It manages and controls other application programs, provides access and security to the users of computers. Some examples are Windows, Linux, Macintosh, Ubuntu, Fedora, Android, iOS etc.

In other words, an operating system can be considered as a resource manager that manages all the resources of a computer like CPU, RAM, Disk, Network, and other input-output devices. Along with this, it is also responsible to control various application software and device drivers manage system security and users as well.

- **OS User Interface :** There are various user interface of operating system each of which provides a different functionality.

(I) **Command Based User Interface or Character User Interface (CUI) :** In this type of OS, the user enters a command to do a task and no graphics or images are supported. The only keyboard can be used as an input device. The user must remember the command to do the task. For example, UNIX, DOC etc.

(II) **Graphical User Interface (GUI) :** This type of OS supports a graphical interface. Users need not remember any command to perform any task. The keyboard and mouse can be used as an input device. For Example, Windows.

(III) **Touch-based user interface :** This type of OS mostly operated on mobiles, tables, or touch screen laptops. For Example iOS, Android etc.

(IV) **Voice-based user interface :** It accepts inputs by voice and performs the tasks. This type of OS can be used by only those users who cannot use the keyboard, mouse, etc. Some iOS devices supported Siri is an example of this type of OS.

(V) **Gesture-based interface :** Some operating supports gestures like waiving, tilting, eye motion, and shaking for operating mobile.

- Functions of Operating System:-The functions of the operating system are as follows:

(I) **Process Management :** process management concerns the management of multiple processes, allocation of required resources, and exchange of information among processes.

(II) **Memory Management :** memory management concerns with management of main memory so that maximum memoryis occupied or utilised by large number of processes while keeping track of each and every location within the memory as free or occupied.

(III) **File Management :** File management system manages secondary memory, while memory management system handles the main memory of a computer system.

(IV) **Device Management :** Just like files, devices also need security measures and their access to different devices must be restricted by the operating system to the authorised users, software and other hardware only. ❐❐❐

2 ENCODING SCHEMES AND NUMBER SYSTEM (NCERT CLASS 11)

INTRODUCTION

- The mechanism of converting data into an equivalent cipher using specific code is called encoding.
- Cipher means something converted to a coded form to hide/conceal it from others. It is also called encryption (converted to cipher) and sent to the receiver who in turn can decrypt it to get back the actual content.

A computer can handle numeric and non numeric data like letters, punctuation marks and other special characters. Some pre defined codes are used to represent numeric and non numeric characters. There are various standard encoding schemes each part of data is assigned a unique code. Some of popular encoding schemes are mentioned below :

(I) **American Standard Code for Information Interchange (ASCII) :** Total number of different characters on theEnglish keyboard that can be encoded by 7-bit ASCII code is $2^7 = 128$.Out of 7 bits, 3 are zone bits and 4 are numeric bits. ASCII-8 can represent 256 characters. It is an extended form of ASCII-7.

(II) **Indian Script Code for Information Interchange (ISCII) :** A lot of efforts have gone into facilitating the use of Indian languages on computers. In 1991, the Bureau of Indian Standards adopted the ISCII. It is an 8-bit code representation for Indian languages which means it can represent 2^8=256 characters. It retains all 128 ASCII codes and uses rest of the codes (128) for additional Indian language character set. Additional codes have been assigned in the upper region (160– 255) for the 'aksharas' of the language.

(III) **UNICODE :** Unicode is a new universal coding standard adopted by all new platforms. It ispromoted by Unicode Consortium which is a non profit organization. Unicode provides a unique number for every character irrespective of the platform, program and the language. It is a character coding system designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages. Commonly used UNICOD Eencodings areUTF-8, UTF-16 and UTF-32. It is a superset of ASCII, and the values 0–128 have the same character as in ASCII.

NUMBER SYSTEM

- Each number system has a base also called a Radix. A decimal number system is a system of base 10; binary is a system of base 2; octal is a system of base 8; and hexadecimal is a system of base 16. What are these varying bases? The answer lies in what happens when we count up to the maximum number that the numbering system allows. In base 10, we can count from 0 to 9, that is,10 digits.

Number System	Base	Symbols used
Binary	2	0,1
Octal	8	0,1,2,3,4,5,6,7
Decimal	10	0,1,2,3,4,5,6,7,8,9
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F where A = 10; B = 11; C = 12; D = 13; E = 14; F = 15

(I) **Decimal Number System :** Decimal number system is known as base-10 system since 10 digits (0 to 9) are used.

(II) **Binary Number System :** Binary number system is formed by two digit 0 and 1.This system is also referred as base-2 system as it has two digits only. Some examples of binary numbers are 1001011, 1011.101, 111111.01. A binary number can be mapped to an equivalent decimal number that can be easily understood by the human.

Following table represent Binary value for (0–9) digits of decimal number system :

Decimal	Binary
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

(III) **Octal Number System :** Octal number system was devised for compact representation of the binary numbers. Octal number system is called base-8 system as it has total eight digits (0-7), and positional value is expressed in powers of 8. Three

binary digits ($8=2^3$) are sufficient to represent any octal digit.

Following Table represent Decimal and binary equivalent of octal numbers 0 – 7:

Octal digit	Decimal value	3-bit Binary Number
0	0	000
1	1	001
2	2	010
3	3	011
4	4	100
5	5	101
6	6	110
7	7	111

(IV) **Hexadecimal Number System :** Hexadecimal numbers are also used for compact representation of binary numbers. It consists of 16 unique symbols (0 – 9, A–F), and is called base-16 system. In hexadecimal system, each alphanumeric digit is represented as a group of 4 binary digits because 4 bits ($2^4=16$) are sufficient to represent 16 alphanumeric symbols.

Following table represent decimal and binary equivalent of hexadecimal numbers 0–9, A–F:-

HEXADECIMAL SYMBOL	DECIMAL VALUE	4-BIT BINARY NUMBER
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

- Applications of Hexadecimal Number System

(I) Usually, size of a memory address is 16-bit or 32-bit. To access 16-bit memory address, a programmer has to use 16 binary bits, which is difficult to deal with. To simplify the address representation, hexadecimal and octal numbers are used.

(II) Hexadecimal numbers are also used for describing the colours on the webpage. Each colour is made up of three primary colours red, green and blue, popularly called RGB (in short). In most colour maps, each colour is usually chosen from a palette of 16 million colours. Therefore, 24 bits are required for representing each colour having three components (8 bits for Red, 8 bits for Green, 8 bits for Blue component).

CONVERSION BETWEEN NUMBER SYSTEM

Converting a number from one Base to another:-

(I) **Binary to Decimal :** Method to convert Binary to Decimal:

1. Start at the rightmost bit.
2. Take that bit and multiply by 2n where n is the current position beginning at 0 and increasing by 1 each time. This represents a power of two.
3. Sum each terms of product until all bits have been used.

Example

Convert the Binary number 101011 to its Decimal equivalent.

$1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0$

$32 + 0 + 8 + 0 + 2 + 1 = (43)_{10}$

Example

Convert the Binary number 1001 to its Decimal equivalent.

$1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0$

$8 + 0 + 0 + 1 = (9)_{10}$

Example

Convert $(11011.101)_2$ to decimal

$2^4\ 2^3\ 2^2\ 2^1\ .\ 2^0\ 2^{-1}\ 2^{-2}\ 2^{-3}$

1 1 0 1 1 1 0 1

$= (1\times2^4)+(1\times2^3)+(0\times2^2)+(1\times2^1)+(1\times2^0)+(1\times2^{-1})+(0\times2^{-2})+(1\times2^{-3})$

$= 16+8+0+2+1+0.5+0+0.125$

$= (27.625)_{10}$

(II) **Decimal to Binary :** Method to convert a Decimal number into its Binary equivalent:

1. Divide the decimal number by 2.
2. Take the remainder and record it on the side.
3. Divide the quotient by 2.
4. REPEAT UNTIL the decimal number cannot be divided further.
5. Record the remainders in reverse order and you get the resultant binary number.

Example

Convert the Decimal number 125 into its Binary equivalent.

125 / 2 = 62	1
62 / 2 = 31	0
31 / 2 = 15	1
15 / 2 = 7	1
7 / 2 = 3	1
3 / 2 = 1	1
1 / 2 = 0	1

Answer: $(1111101)_2$

Example

Convert $(105.15)_{10}$ to binary

Let us convert 105 first.

$(105)_{10} = (1101001)_2$

Let us convert $(0.15)_{10}$

Multiply 0.15 by 2	0.30
Multiply 0.30 by 2	0.60
Multiply 0.60 by 2	1.20
Multiply 0.20 by 2	0.40
Multiply 0.40 by 2	0.80
Multiply 0.80 by 2	1.60

Reading the integers from top to bottom $(0.15)_{10} = (0.001001)_2$

Final result $(105.15)_{10} = (1101001.001001)_2$

(III) **Decimal to Octal :** The method to convert a decimal number into its octal equivalent:

1. Divide the decimal number by 8.
2. Take the remainder and record it on the side.
3. Divide the quotient by 8.
4. REPEAT UNTIL the decimal number cannot be divided further.
5. Record the remainders in reverse order and you get the resultant binary

Example

Convert the Decimal number 125 into its Octal equivalent.

125 / 8 = 15	5
15/ 8 = 1	7
1/8 =0	1

Answer: $(175)_8$

Example

Convert $(0.75)_{10}$ to Octal

Multiply the given fraction by 8. Keep the integer in the product as it is and multiplythe new fraction in the product by 8. Continue the process and read the integers in the products from top to bottom.

Given fraction: 0.75

Multiply 0.75 by 8: 6.00

Reading the integers from top to bottom 0.75 in decimal number system is 0.6 in octal number system.

(IV) **Octal to Decimal :** Method to convert Octal to Decimal:

1. Start at the rightmost bit.
2. Take that bit and multiply by 8n where n is the current position beginning at 0 and increasing by 1 each time. This represents the power of 8.
3. Sum each of the product terms until all bits have been used.

Example

Convert the Octal number 321 to its Decimal equivalent.

$3 * 8^2 + 2 * 8^1 + 1 * 8^0$

$192+16+1 = (209)_{10}$

Example

Convert $(23.25)_8$ to decimal

$8^1 \quad 8^0 . \quad 8^{-1} \quad 8^{-2}$

$2 \quad 3 \quad 2 \quad 5$

$= (2 \times 8^1)+ (3 \times 8^0)+ (2 \times 8^{-1})+ (5 \times 8^{-2})$

$= 16+3+0.25+0.07812$

$= (19.32812)_{10}$

(V) **Decimal to Hexadecimal :** Method to convert a Decimal number into its Hexadecimal equivalent:

1. Divide the decimal number by 16.
2. Take the remainder and record it on the side.
3. REPEAT UNTIL the decimal number cannot be divided further.
4. Record the remainders in reverse order and you get the equivalent hexadecimal number.

Example

Convert the Decimal number 300 into its hexadecimal equivalent.

300 / 16 = 18	12-(C)
18 / 16 = 1	2
1 / 16 = 0	1

Answer: $(12C)_{16}$

Example

Convert $(0.75)_{10}$ to hexadecimal

Multiply the given fraction by 16. Keep the integer in the product as it is and multiply the new fraction in the product by 16. Continue the process and read the integers in the products from top to bottom.

Given fraction 0.75

Multiply 0.75 by 16 12.00 - C

Reading the integers from top to bottom 0.75 in decimal number system is 0C in Hexadecimal number system.

(VI) **Hexadecimal to Decimal :** Method to convert Hexadecimal to Decimal:

1. Start at the rightmost bit.
2. Take that bit and multiply by 16n where n is the current position beginning at 0 and increasing by 1 each time. This represents a power of 16.
3. Sum each terms of product until all bits have been used.

Example

Convert the Hexadecimal number AB to its Decimal equivalent.

$=A * 16^1 + B * 16^0$

$=10 * 16^1 + 11 * 16^0$

$=160+11 = (171)_{16}$

Example

Convert $(1E.8C)_{16}$ to decimal

$16^1\ 16^0\ .\ 16^{-1}\ 16^{-2}$

1E 8 C

$= (1 \times 16^1)+ (14 \times 16^0)+ (8 \times 16^{-1})+ (12 \times 16^{-2})$

$= 16+14+0.5+0.04688$

$= (30.54688)_{10}$

(VII) **Binary to Hexadecimal :** The hexadecimal number system uses the digits 0 to 9 and A, B, C, D, E, F. Method to convert a Binary number to its Hexadecimal equivalent is:

We take a binary number in groups of 4 and use the appropriate hexadecimal digit init's place. We begin at the rightmost 4 bits. If we are not able to form a group of four,insert 0s to the left until we get all groups of 4 bits each. Write the hexadecimal equivalent of each group. Repeat the steps until all groups have been converted.

Example

Convert the binary number 1000101 to its Hexadecimal equivalent.

0100 0101

4 5

Note that we needed to insert a 0 to the left of 100.

Answer: $(45)_{16}$

In case of a fractional binary number form groups of four bits on each side of decimal point. Then replace each group by its corresponding hexadecimal number.

Example

Convert $(11100.1010)_2$ to hexadecimal equivalent.

0001 1100 . 1010

1 C . A

Answer : $(1C.A)_{16}$

(VIII) **Hexadecimal to Binary :**Method to convert a Hexadecimal number to its Binary equivalent is: Convert each digit of Hexadecimal Number to it's binary equivalent and write them in 4 bits. Then, combine each 4 bit binary number and that is the resulting answer.

Example

Convert the Hexadecimal number $(10AF)_{16}$ to its Binary equivalent.

1 0 A F

0001 | 0000 | 1010 | 1111

Answer: $(0001000010101111)_2$

Example

Convert the Hexadecimal number $(A2F)_{16}$ to its Binary equivalent.

A 2 F

1010 | 0010 | 1111

Answer: $(1010\ 0010\ 1111)_2$

(IX) **Binary to Octal and Octal to Binary :** To convert Binary to Octal, as the octal system is a power of two (2^3), we can take the bits into groups of 3 and represent each group as an octal digit. The steps are the same for the binary to hexadecimal conversions except we are dealing with the octal base now. To convert from octal to binary, we simply represent each octal digit in it's three bit binary form.

Example

Convert the Octal number $(742)_8$ to its Binary equivalent.

7 | 4 | 2

111 | 100 | 010

Answer: $(111100010)_2$

(X) **Hexadecimal to Octal and Octal to Hexadecimal :** To convert Hexadecimal to Octal, Convert each digit of Hexadecimal Number to it's binary equivalent and write them in 4 bits. Then, combine each 3 bit binary number and that is converted into octal.

Example

Convert the Hexadecimal number $(A42)_{16}$ to its Octal equivalent.

A | 4 | 2

1010 | 0100 | 0010

101 | 001 | 000 | 010

Answer: $(5102)_8$

To convert Octal to hexadecimal, convert each digit of Octal Number to it's binary equivalent and write them in 3 bits. Then, combine each 4 bit binary number and that is converted into hexadecimal.

Example

Convert the Octal number $(762)_8$ to its hexadecimal equivalent.

7 | 6 | 2
101 | 110 | 010
0001 | 0111 | 0010

Answer: $(172)_{16}$

BINARY REPRESENTATION OF INTEGERS

Binary number can be represented only by using 0's and 1's, but can not use the sign (–) to denote the negative number or sign (+) to denote the positive number. So it must be either 0 or 1. There are three methods to represent binary number. They are:-

(I) **Sign and magnitude method :** In this method, first bit is considered as a sign bit. Here positive number starts with 0 and negative number starts with 1.

Example:- Take a number 25.

25/2 = 12	1
12/2 = 6	0
6/2 = 3	0
3/2 = 1	1
1/2 = 0	1

So the binary number is $(11001)_2$. If we take the size of the word is 1 byte, then the number 25 will be represented as 00011001.

Suppose, if the number is -25, and then it will be represented as 10011001.

(II) **One's complement method :** In this method, the positive number is represented as same as the binary number. If the number is negative, then we need to find one's complement of a binary number. The one's complement of a binary number will replace every 0 with 1 and vice- versa.

Example

(a) Represent 86 in one's complement method (I byte representation)

86/2 =43	0
43/2 =21	1
21/2 =10	1
10/2 =5	0
5/2 =2	1
2/2 =1	0
1/2 =0	1

The binary number is 1010110.

1 byte representation of number 86 is 01010110.

(b) Represent -55 in one's complement method (I byte representation)

55/2 =27	1
27/2 =13	1
13/2 =6	1
6/2 =3	0
3/2 =1	1
1/2 =0	1

The binary number is 110111.

1 byte representation is 00110111

The given number is negative; hence we need to calculate one's complement

One's complement of 00110111 is 11001000 (convert 1 into 0 and 0 into 1)

Thus, the 1 byte representation of number -55 is 11001000.

(III) **Two's complement method:** In this method, the positive number is represented as the binary number. If the number is negative, then we need to calculate two's complement of a binary number. The two's complement of a binary number is calculated by adding 1 to its one's complement.

Example

(a) Represent 87 in two's complement method (I byte representation)

87/2 =43	1
43/2 =21	1
21/2 =10	1
10/2 =5	0
5/2 =2	1
2/2 =1	0
1/2 =0	1

The binary number is 1010111

Hence, the 1 byte representation of number 86 is 01010111.

(b) Represent -54 two's complement method (I byte representation)

54/2 =27	0
27/2 =13	1
13/2 =6	1
6/2 =3	0
3/2 =1	1
1/2 =0	1

The binary number is 110110

Hence, the 1 byte representation is 00110110

The given number is negative; hence we need to calculate two's complement.

One's complement of 00110110 is 11001001 (convert 1 into 0 and 0 into 1).

Add 1 to one's complement

```
      1
11001001 (1+1=2, binary equivalent =11)
    + 1
-----------
11001010
-----------
```

Thus, 1 byte representation of number -54 is 11001010 □□□

3 EMERGING TRENDS
(NCERT CLASS 11)

INTRODUCTION

Many emerging trends in computer science involve various technologies that have grown at a faster pace. Emerging trends could be identified by simply being updated with journals and newspapers, and social gatherings. Technology has been increasing tremendously over the years. The infrastructure provided by various companies for their growth has been an excellent resource for innovating new technologies. The emerging trends in the information technology industry like Artificial Intelligence (AI), Robotics, Data Analytics, Cyber Security, Bioinformatics, Education involving technological assistance, etc., have seen a positive trend in recent years. These trends also have the potential of growing exponentially to the foreseeable future.

ARTIFICIAL INTELLIGENCE (AI)

- Artificial Intelligence is comprised of two words Artificial and Intelligence, where Artificial means "man-made," and intelligence means "thinking power", hence AI means "a man-made thinking power.
- "Artificial Intelligence exists there ,where a machine can have human based skills such as learning, reasoning, and solving problems.
- According to the father of Artificial Intelligence, John McCarthy, it is "The science and engineering of making intelligent machines.
- **Goals of AI :**
  - (I) To Create Expert Systems ? The systems which holds intelligent behavior, learn, demonstrate, explain, and advice its users.
  - (II) To Implement Human Intelligence in Machines? Creating systems that can understand, think, learn, and behave like humans.

Applications of AI :- AI has been dominant in various fields such as –

(I) Gaming
(II) Natural Language Processing
(III) Expert Systems
(IV) Vision Systems
(V) Speech Recognition
(VI) Handwriting Recognition
(VII) Intelligent Robots

Artificial intelligence is a science and technology based on disciplines such as Computer Science, Biology, Psychology, Linguistics, Mathematics, and Engineering.

Difference between Normal Programming and AI Programming :

	NORMAL/REGULAR PROGRAMMING	AI PROGRAMMING
INPUT	Input is a sequence of alphanumeric symbols presented and stored as per some given set of previously stipulated rules and that uses a limited set of communication media such as keyboard, mouse, disc, etc.	Input may be a sight, sound, touch, smell or taste. Sight means one dimensional symbols such as typed text, two dimensional objects or three dimensional scenes.
PROCESSIN	Processing means manipulation of the stored symbols by a set of previously defined algorithms.	Processing includes knowledge representation and pattern matching, search, logic, problem solving and learning.
OUTPUT	output is a sequence of alphanumeric symbols, may be in a given set of colors that is placed on such a medium as a CRT screen, paper, or magnetic disk.	Output can be in the form of printed language and synthesized speech, manipulation of physical objects or locomotion i.e., movement in space.

- **Advantages of Artificial Intelligence :**
  - (I) **High Accuracy with less errors :** it takes decisions as per preexperience or information.
  - (II) **High-Speed.**

(III) **High reliability :** can perform the same action multiple times with high accuracy.

(IV) **Useful for risky areas :** helpful in situations such as defusing a bomb, exploring the ocean floor, where to employ a human can be risky.

(V) **Digital Assistant :** Such as used by various E-commerce websites to show the products as per customer requirement.

(VI) **Useful as a public utility :** such as a self-driving car which can make our journey safer and hassle-free, facial recognition for security purpose, Natural language processing to communicate with the human in human-language, etc.

Artificial intelligence can be divided into three subfields:

(I) Artificial intelligence

(II) Machine learning

(III) Deep learning

MACHINE LEARNING

Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. As its name, it gives the computer that makes it more similar to humans: The ability to learn.

- Applications of Machine Learning:-
  - (I) Image Recognition.
  - (II) Speech Recognition.
  - (III) Traffic prediction.
  - (IV) Product recommendations.
  - (V) Self-driving cars.
  - (VI) Email Spam and Malware Filtering.
  - (VII) Virtual Personal Assistant.
  - (VIII) Online Fraud Detection.
  - (IX) Stock Market trading.
  - (X) Medical Diagnosis.
  - (XI) Automatic Language Translation.
- **WORKING OF MACHINE LEARNING :**
  - (I) Clustering is the most common unsupervised learning technique. It is used for exploratory data analysis to find hidden patterns or groupings in data. Applications for cluster analysis include gene sequence analysis, market research, and object recognition.
  - (II) Classification techniques predict discrete responses—for example, whethera n email is genuine or spam, or whether a tumor is cancerous or benign.
  - (III) Regression techniques predict continuous responses—for example, changes in temperature or fluctuations in power demand.

NATURAL LANGUAGE PROCESSING (NLP)

NLP is a way of computers to analyze, understand and derive meaning from a human languages such as English, Spanish, Hindi, etc. It is the technology that is used by machines to understand, analyse, manipulate, and interpret human's languages.

- **Components of NLP :** There are the following two components of NLP:-
  - (I) Natural Language Understanding (NLU).
  - (II) Natural Language Generation (NLG).
- Applications of NLP:-
  - (I) Question Answering
  - (II) Spam Detection
  - (III) Sentiment Analysis
  - (IV) Machine Translation
  - (V) Spelling correction
  - (VI) Speech Recognition
  - (VII) Chatbot
  - (VIII) Information extraction
- An "immersive experience" pulls a person into a new or augmented reality, enhancing everyday life via technology. It often use one or more technologies linked together. The three pillars of immersive experiences are visual quality, sound quality, and intuitive interactions. Full immersion can only be achieved by simultaneously applied all these three.
- **How does Augmented Reality work :** It involves technologies like S.L.A.M. (simultaneous localization and mapping), depth tracking (briefly, a sensor data calculating the distance to the objects), and the following components:
  - (I) **Cameras and sensors :** Collecting data about user's interactions and sending it for processing.
  - (II) **Processing :** AR devices eventually should act like little computersto be able to measure speed, angle, direction, orientation in space, and so on.
  - (III) **Projection :** This refers to a miniature projector on AR headsets, which takes data from sensors and projects digital content.
  - (IV) **Reflection :** Some AR devices have mirrors to assist human eyes to view virtual images.
- **Applications of AR :**
  - (I) Most popular applications of AR is gaming. New AR games provide much better experiences to players, some even promote a more active outgoing way of life (PokemonGo, Ingress).

(II) AR in retail may act to bring better customer engagement and retention, as well as brand awareness and more sales. Some features may also help customers make wiser purchases – providing product data with 3D models of any size or color.

- Virtual Reality (VR) is use of computer technology to create a simulated environment. Unlike traditional user interfaces, VR places the user inside an experience. Instead of viewing a screen in front of them, users are immersed and able to interact with 3D worlds/objects.
- **The Basics of how VR Works :** Every headset is used to perfect their approach to creating an immersive 3D environment. Each VR headset puts up a screen in front of eyes thus, eliminating any interaction with the real world. Two autofocus lenses are generally placed between the screen and the eyes that adjust based on individual eye movement and positioning.

  The visuals on the screen are rendered either by using a mobile phone or HDMI cable connected to a PC. A frame rate of minimum 60fps, an equally competent refresh rate and minimum 100-degree field of view (FOV) is required for true VR.
- **Applications of VR :**
  (I) Automotive industry.
  (II) Healthcare.
  (III) Retail
  (IV) Tourism
  (V) Real estate
  (VI) Architecture
  (VII) Gambling
  (VIII) Learning and Development.

BIG DATA

Big Data is also data but with a huge size/volume and yet growing exponentially with time. In short such data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently.

- **Benefits of Big Data Processing :**
  (I) Businesses can utilize outside intelligence while taking decisions.
  (II) Improved customer service.
  (III) Early identification of risk to the product/ services, if any.
  (IV) Better operational efficiency.
- **Characteristics Of Big Data :**
  (I) **Volume :** The name Big Data itself is related to a size which is enormous.
  (II) **Variety :** Variety refers to heterogeneous sources and the nature of data, both structured and unstructured. During earlier days, spread sheets and databases were the only sources of data considered by most of the applications as structured big data type. Nowadays, data in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. are also being considered in the analysis applications. This variety of unstructured data poses certain issues for storage, mining and analyzing data.
  (III) **Velocity :** It means speed of generation of data. How fast the data is generated and processed to meet the demands, determines real potential in the data.
  (IV) **Variability :** This refers to the inconsistency which can be shown by the data at times.

INTERNET OF THINGS (IOT)

- The IOT concept was initially proposed by a member of the Radio Frequency Identification (RFID) development community in 1999, and now it has become more relevant to the practical world as the use of mobile devices, embedded devices, communication, cloud computing and data analytics has increased. Internet connects all people means "Internet of People" IoT connects all things means "Internet of Things".
- Interconnection of Things/Objects/Machines, e.g., sensors, mobilephones, electronic devices, home appliances, any existing items and interact with each other via Internet.
- Internet of Things technology can include any sensor, electronic devices or software which are connected to the internet and can be utilized remotely and can exchange data. Here devices works themselves without human intervention for the welfare of humans.
- **MAJOR CHARACTERISTICS OF IOT :**
  (I) Very Large Scale.
  (II) Heterogeneity.
  (III) Pervasivity : Computing and Communication technologies embedded in our environments.
- **How Does the Internet of Things Work?**

  The Internet of Things is an aggregation of internet enabled sensors, smart devices and software that can be manipulated by scripts, applications and user interfaces across long distances.

- **Applications of IOT :**

(I) **Smart house :** Suppose we are not at home and doubts starts in our mind. Did I turn the coffee maker off? Did I set the security alarm? etc. With a smart home,we can quiet all of these worries with a quick glance at smartphone/tablet. we can connect the devices and appliances in our home so they can communicate with each other and with us and can work with the commands given over smartphone remotely.

(II) **Smart car :** the driverless car (now a prototype) where taxis work based on AI and take the passengers safely and accurately to the desired destination.

(III) **Elderly care :** Patient surveillance can be life-saving; automatically detecting when someone falls down or when they begin to experience a heart attack so that emergency care can be sent immediately.

(IV) **Disaster warning :** Sensors can collect critical information about the environment, allowing for early detection of environmental disasters like earthquakes, tsunamis, etc., thus saving lives.

(V) **Delivery Drones :** drones being used to deliver item with the help of smart grid/geospatial data.

(VI) A smart city is a framework, predominantly composed of Information and Communication Technologies (ICT), to develop, deploy, and promote sustainable development practices to address growing urbanization challenges. A big part of this ICT framework is essentially an intelligent network of connected objects and machines that transmit data using wireless technology and the cloud.

- **IoT Platform :** It is an integrated service which offers the things to bring physical objects online. It easily allow to configure devices for machine-to-machine communication through millions of devices connects simultaneously .

Sensors are useful and very important for the devices in order to fetch the data. The data can be real-time, which includes the current temperature, pressure or humidity.

List of Sensors most commonly used in the IoT devices:-

(I) Temperature Sensor

(II) Pressure Sensor

(III) Proximity Sensor

(IV) Accelerometer and Gyroscope Sensor

(V) IR Sensor

(VI) Optical Sensor

(VII) Gas Sensor

(VIII) Smoke Sensor

- **IoT Platform Types :**

(I) **End-to-end IoT Platforms :** provide the hardware, software, connectivity, security, and device management tools to handle connection of millions of concurrent device.

(II) **Connectivity Management Platforms :** It offer low power and low cost connectivity management solutions through Wi-Fi and cellular technologies.

(III) **IoT Cloud Platforms :** It's aim to get rid of the complexity of building our own complex network .

(IV) **Data Platform –** It deals with data in some way with the tools we need to route device data and manage / visualize data analytics.

CLOUD COMPUTING

Cloud refers to a Internet or Network or present at remote location. Cloud Computing refers to remote access of hardware/software resources for access, configuration, manipulation. Cloud computing offers online data storage, infrastructure, and application. Applications such as customer relationship management (CRM) ,e-mail, web conferencing, execute on cloud. It can work on public and private networks, i.e., WAN, LAN or VPN.

Cloud computing offers platform independency, because software is not required to be installed locally on the PC.Thus applications are being mobile and collaborative.

- **Uses of cloud computing :**

(I) Create new apps and services.

(II) Store, back up and recover data.

(III) Host websites and blogs.

(IV) Stream audio and video.

(V) Deliver software on demand.

(VI) Analyze data for patterns and make predictions.

DIFFERENCE BETWEEN PUBLIC AND PRIVATE CLOUD	
PUBLIC CLOUD	**PRIVATE CLOUD**
Hosted at service provider site.	Hosted at Enterprise or service provider server.
Cheaper than private cloud.	Costlier than public cloud.
Utilizes shared infrastructure.	Utilizes own infrastructure.
Supports connectivity over internet.	Supports connectivity over internet/Private WAN.
Require higher level of security.	Require medium level of security.
Supports multiple customers.	Supports one customer.
Shared server.	Dedicated server.
Fixed cost. Multitenant architecture.	Variable cost. Dedicated customer architecture.
Example - ESDS's eNlight Cloud, Amazon Elastic Compute Cloud (EC2), IBM's Blue Cloud, Sun Cloud, Google AppEngine and Windows Azure Services Platform.	Hewlett Packard Enterprise (HPE) — offers the Helion Cloud Suite software, Helion CloudSystem hardware, Helion Managed Private Cloud and Managed Virtual Private Cloud services

SOFTWARE AS A SERVICE (SaaS)

SaaS is a fully-developed software solution ready for purchase and use over the internet on a subscription basis. The SaaS provider manages the infrastructure, operating systems, middleware, and data necessary to deliver the program, ensuring that the software is available whenever and wherever customers need it. Many SaaS applications run directly through web browsers, eliminating the need for downloads or installations. This greatly reduces software management issues for internal IT teams. Examples of SaaS: Microsoft Office 365, Salesforce, Cisco WebEx, Google Apps.

PLATFORM AS A SERVICE (PaaS)

PaaS is extremely helpful for any company that develops software and web-based applications. Many of the tools needed to develop for multiple platforms (computers, mobile devices, browsers, etc) can be quite expensive. By using PaaS, customers can access the development tools they need, when they need them, without having to purchase them outright. Since the platform is accessible over the internet, remote development teams can all access the same assets to speed up product development. Examples of PaaS: AWS Elastic Beanstalk, Apache Stratos, Google App Engine, Microsoft Azure.

INFRASTRUCTURE AS A SERVICE (IaaS)

It provides a completely virtualized computing infrastructure that is provisioned and managed over the internet. An IaaS provider manages the physical end of the infrastructure (servers, data storage space, etc) in a data center, but allows customers to fully customize those virtualized resources to suit their specific needs. Examples of IaaS: Microsoft Azure, Amazon Web Services (AWS), Cisco Metacloud, Google Compute Engine (GCE).

BLOCKCHAIN TECHNOLOGY

It typically refers to the transparent, trustless, publicly accessible ledger that allows us to securely transfer the ownership of units of value using public key encryption and proof of work methods. The technology uses decentralized consensus to maintain the network, means not centrally controlled by a bank, corporation, or government. In fact, the larger the network grows and becomes increasingly decentralized, the more secure it becomes. The potential for blockchain technology is not limited to bitcoin. As such, it has gained a lot of attention in a variety of industries including: financial services, charities and nonprofits, the arts, and e-commerce.

Grid computing

- It is a computer network in which each computer's resources are shared with every other computer in the system. Processing power, memory and data storage are all community resources that authorized users can tap into and work/use for specific tasks.
- **Grid can be of two types**
  (i) Data grid, used tomanage large and distributed data having required multi-user access, and
  (ii) CPU or Processor grid, where processing is moved from one PC to another as needed or a large task is divided into subtasks and divided to various nodes for parallel processing.
- To set up a grid, by connecting numerous nodes in terms of data as well as CPU, a middleware is required to implement the distributed processor architecture. The Globus toolkit (http://toolkit.globus.org/toolkit) is one such software toolkit used for building grids, and it is open source. It includes software for security, resource management, data management, communication, fault detection, etc. ❐❐❐

4 INTRODUCTION TO PROBLEM SOLVING (NCERT CLASS 11)

INTRODUCTION

The success of a computer in solving a problem depends on how correctly and precisely we define the problem, design a solution (algorithm) and implement the solution (program) using a programming language. Thus, problem solving is the process of identifying a problem, developing an algorithm for the identified problem and finally implementing the algorithm to develop a computer program.

STEPS FOR PROBLEM SOLVING

When problems are straight forward and easy, we can easily find the solution. But a complex problem requires a methodical approach to find the right solution. In other words, we have to apply problem solving techniques. Problem solving begins with the precise identification of the problem and ends with a complete working solution in terms of a program or software. Key steps required for solving a problem using a computer are discussed below:-

(I) **Analysing the problem :** We need to read and analyse the problem statement carefully in order to list the principal components ofthe problem and decide the core functionalities that our solution should have. By analysing a problem, we would be able to figure out what are the inputs tha tour program should accept and the outputs that itshould produce.

(II) **Developing an Algorithm :** Before writing a program code for a given problem, it is essential to device a solution. The solution is represented in natural language and is called an algorithm.

(III) **Coding :** After finalising the algorithm, we need to convert the algorithm into the format which can be understood by the computer to generate the desired solution. Different high level programming languages can be used for writing a program.

(IV) **Testing and Debugging :** The program created should be tested on various parameters. The program should meet the requirements of the user. It must respond within the expected time. It should generate correct output for all possible inputs. In the presence of syntactical errors, no output will be obtained. In case the output generated is incorrect, then the program should be checked for logical errors, if any.

Software programs goes through testing, updating, troubleshooting, and maintenance during the development process. Usually, software contains errors and bugs, which are removed routinely. Debugging is the process of fixing a bug in the software.

ALGORITHM

An algorithm is an effective method expressed as a finite list of well definedinstructions for calculating a function, starting from an initial state and initial input. The instructions describe a computation, which will eventually produce output, when executed. We can use algorithm to solve any kind of problems. However, before writing a program, we need to write the steps to solve the problem in simple English language. This step-by-step procedure to solve the problem is called algorithm.

The purpose of using an algorithm is to increase the reliability, accuracy and efficiency of obtaining solutions.

- **Characteristics of a good algorithm :**
  - (I) Precision — the steps are precisely stated or defined.
  - (II) Uniqueness — results of each step are uniquely defined and only depend on the input and the result of the preceding steps.
  - (III) Finiteness — the algorithm always stops after a finite number of steps.
  - (IV) Input — the algorithm receives some input.
  - (V) Output — the algorithm produces some output.
- While writing an algorithm, it is required to clearly identify the following:
  - (I) The input to be taken from the user.
  - (II) Processing or computation to be performed to get the desired result.
  - (III) The output desired by the user.

REPRESENTATION OF ALGORITHMS

The software designers or programmers analyse the problem and identify the logical steps that need to be followed to reach a solution by using theiralgorithmic thinking skills. There are two common methods of representing an algorithm —flowchart and pseudocode. This methods can be used to represent an algorithm while keeping in mind the following:

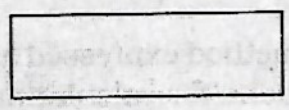
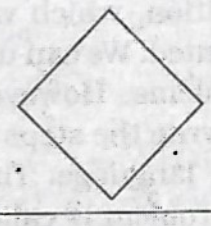
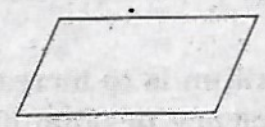
(I) It showcases the logic of the problem solution, excluding any implementational details.

(II) It clearly reveals the flow of control during execution of the program.

- **Flowchart — Visual Representation of Algorithms :**

A flowchart is a visual representation of an algorithm. A flowchart is a diagram made up of boxes, diamonds and other shapes, connected by arrows. Each shape represents a step of the solution process and the arrow represents the order or link among the steps.

Table : Shapes or symbols to draw flow charts :

Flowchart Symbol	Function	Description
	Start/Ends	Also called "Terminator" symbol. It indicates where the flow starts and ends.
	Process	Also called "Action Symbol," it represents a process action, or a single step.
	Decision	A decision or branching point, usually a yes/no or true/false question is asked, and based on the answer, the path gets split into two branches.
	Input/Output	Also called data symbol, this parallelogram shape is used to input or output data
	Arrow	Connector to show order of flow between shapes.

The way of execution of the program shall be categorized into three ways:

(I) sequence statements; (II) selection statements; and (IIII) iteration or looping statements. This is also called as "control structure".

(I) Sequence statements: In this program, all the instructions are executed one after another.

Example

Write an algorithm to print "Good Morning".

Step 1: Start

Step 2: Print "Good Morning"

Step 3: Stop.

Example

Write an algorithm to find area of a rectangle.

Step 1: Start

Step 2: Take length and breadth and store them as L and B?

Step 3: Multiply by L and B and store it in area

Step 4: Print area

Step 5: Stop

In the above mentioned two examples, all the instructions areexecuted one after another. These examples are executed under sequential statement.

(II) **selection statements :** In this program, some portion of the program is executed based upon the conditional test. If the conditional test is true, compiler will execute some part of the program, otherwise it will execute the other part of the program.

Example

Write an algorithm to check whether he is eligible to vote? (more than or equal to 18 years old).

Step 1: Start

Step 2: Take age and store it in age

Step 3: Check age value, if age >= 18 then go to step 4 else step 5

Step 4: Print "Eligible to vote" and go to step 6

Step 5: Print "Not eligible to vote"

Step 6: Stop

Example

Write an algorithm to check whether given number is +ve, -ve or zero.

Step 1: Start

Step 2: Take any number and store it in n.

Step 3: Check n value, if n > 0 then go to step 5 else go to step 4

Step 4: Check n value, if n < 0 then go to step 6 else go to step 7

Step 5: Print "Given number is +ve" and go to step 8

Step 6: Print "Given number is -ve" and go to step 8

Step 7: Print "Given number is zero"

Step 8: Stop

In the above mentioned examples, all the statements are not executed, but

based upon the input, some portions of the algorithm are executed, because we have "true" or "false" situation in the program.

(III) **Iterative statements :** In some programs, certain set of statements are executed again and again based upon conditional test. i.e. executed more than one time. This type of execution is called "looping or iteration".

Example

Write an algorithm to print all natural numbers up to „n?.

Step 1: Start

Step 2: Take any number and store it in n.

Step 3: Store 1 in I

Step 4: Check I value, if I<=n then go to step 5 else go to step 8

Step 5: Print I

Step 6: Increment I value by 1

Step7: Go to step 4

Step 8: Stop

In the above example, steps 4, 5, 6 and 7 are executed more than one time.

- **Pseudocode :** A pseudocode (pronounced Soo-doh-kohd) is another way of representing an algorithm. It is considered as a non-formal language that helps programmers to write algorithm. It is a detailed description of instructions that a computer must follow in a particular order. It is intended for human reading and cannot be executed directly by the computer. No specific standard for writing a pseudocode exists. The word "pseudo" means "not real," so "pseudocode" means "not real code". Following are some of the frequently used keywords while writing pseudocode:

(I)	INPUT	(II)	COMPUTE
(III)	PRINT	(IV)	INCREMENT
(V)	DECREMENT	(VI)	IF/ELSE
(VII)	WHILE	(VIII)	TRUE/FALSE

There is no strict set of standard notations for pseudocode, but some of the most widely recognised are:

(I) INPUT/READ – indicates a user will be inputting something.

(II) OUTPUT/WRITE – indicates that an output will appear on the screen.

(III) WHILE – a loop (iteration that has a condition at the beginning).

(IV) FOR – a counting loop (iteration).

(V) REPEAT – UNTIL – a loop (iteration) that has a condition at the end.

(VI) IF – THEN – ELSE – a decision (selection) in which a choice is made any instructions that occur inside a selection or iteration are usually indented.

Representation of algorithm using Pseudo-code:

Example-

Pseudocode to find out largest of two numbers.

```
Write "please enter 2 numbers"
Read n1,n2
If(n1 > n2)
Set max to n1
Else
Set max to n2
```

Benefits of Pseudocode :

(I) A pseudocode of a program helps in representing the basic functionality of the intended program.

(II) By writing the code first in a human readable language, the programmer safeguards against leaving out any important step.

(III) Besides, for non-programmers, actual programs are difficult to read and understand, but pseudocode helps them to review the steps to confirm that the proposed implementation is going to achieve the desire output.

FLOW OF CONTROL

The flow of control depicts the flow of events as represented in the flow chart. The events can flow in a sequence, or on branch based on a decision or even repeat some part for a finite number of times.

(I) **SEQUENCE :** Such algorithms where all the steps are executed oneafter the other are said to execute in sequence. However, statements in an algorithm may not always execute in a sequence. We may sometimes require the algorithm to either do some routine tasks in a repeated manner or behave differently depending on the outcomes of previous steps.

(II) **SELECTION :** Selecting which path of an algorithm to execute depending on some criteria. For example, if you passed a class in school, then we execute the operations that clap and cheer and play a song. But if you didn't pass the class, then maybe we would say, "Better luck next time, hang in there!"

The program checks one or more conditions and perform operations (sequence of actions) depending on true or false value of the condition. These true or false values are called binary values.

Conditionals are written in the algorithm as follows:

If <condition> then

steps to be taken when the

condition is true/fulfilled

There are situations where we also need totake action when the condition is not fulfilled. To represent that, we can write:

If <condition> is true then

steps to be taken when the condition is true/ fulfilled

otherwise steps to be taken when the condition is false/not fulfilled

In programming languages, 'otherwise' is represented using Else keyword. Hence, a true/false conditional is written using if-else block in actual programs.

(III) **REPETITION :** The kind of statements we use, when we want something to be done repeatedly, for a given number of times. Forexample , suppose 10 cards need to be withdrawn in the card game , then the pseudocode needs to be repeated 10 times to decide the winner.

In programming, repetition is also known as iteration or loop. A loop in an algorithm means execution of some program statements repeatedly till some specified condition is satisfied. Many times, we want to be able to repeat a set of operations a specific number of times or until some condition occurs.

VERIFYING ALGORITHMS

Today software are used in more critical services — like in the medical field or in space shuttles. Such software needs to work correctly in every situation. Therefore, the software designer should make sure that the functioning of all the components are defined correctly, checked and verified in every possible way.

When we have written an algorithm,we want to verify that it is working as expected. To verify, we have to take different input values and go through all the steps of the algorithm to yield the desired output for each input value and may modify or improve as per the need. The method of taking an input and running through the steps of the algorithm is sometimes called dry run. Such a dry run will help us to:

(I) Identify any incorrect steps in the algorithm.

(II) Figure out missing details or specifics in the algorithm.

Suppose we develop some software without verifyingthe underlying algorithm and if there are errors in the algorithm, then the software developed will not run. Hence, it is important to verify an algorithm since the effort required to catch and fix a mistake is minimal.

COMPARISON OF ALGORITHM

There can be more than one approach to solve a problem using computer and hence we can have more than one algorithm.

Algorithms can be compared and analysed on the basis of the amount of processing time they need to run and the amount of memory that is needed to execute the algorithm. These are termed as time complexity and space complexity, respectively. The choice of an algorithm over another is done depending on how efficient they are in terms of proecssing time required (time complexity) and the memory they utilise (space complexity).

CODING

Once an algorithm is finalised, it should be coded in a high-level programming language as selected by the programmer. The ordered set of instructions are written in that programming language by following its syntax. Syntax is the set of rules or grammar that governs the formulation of the statements in the language, such as spellings, order of words, punctuation, etc.

Coding is basically implementing logic/algorithm/ pseudocode (derived from requirement analysis/problem definition) in one of the preferred programming language(C,C++ Java, Javascript, python etc) as per the protocols/rules/syntactic grammar of the choosen language by following the design decisions.

A program written in a high-level language is called source code. We need to translate the source code into machine language using a compiler or an interpreter, so that it can be understood by the computer.

DECOMPOSITION

Sometimes a problem may be complex, that is, its solution is not directly derivable. In such cases, we need to decompose it into simpler parts.

The basic idea of solving a complex problem by decomposition is to 'decompose' or break down a complex problem into smaller sub problems. These sub problems are relatively easier to solve than the original problem. Finally, the subproblems are combined in a logical way to obtain the solution for the bigger, main problem.

Once the individual sub problems are solved, it is necessary to test them for their correctness and integrate them to get the complete solution.

□□□

5 GETTING STARTED WITH PYTHON (NCERT CLASS 11)

INTRODUCTION

An ordered set of instructions to be executed by acomputer to carry out a specific task is called a program, and the language used to specify this set of instructions to the computer is called a programming language.

Computers understand the language of 0s and 1s which is called machine language or low level language. However, it is difficult for humans to write or comprehend instructions using 0s and 1s. This led to the advent of high-level programming languages like Python, C++, Visual Basic, PHP, Java that are easier to manage by humans but are not directly understood by the computer.

A program written in a high-level language is called source code. Language translators like compilers and interpreters are needed to translate the source code into machine language. Python uses an interpreter to convert its instructions into machine language, so that it can be understood by the computer. An interpreter processes the program statements one by one, first translating and then executing. This process is continued until an error is encountered or the whole program is executed successfully. In both the cases, program execution will stop. On the contrary, a compiler translates theentire source code, as a whole, into the object code. After scanning the whole program, it generates error messages, if any.

- **Features of Python**

(I) Python is a high level language. It is a free and open source language.

(II) It is an interpreted language, as Python programs are executed by an interpreter.

(III) Python programs are easy to understand as they have a clearly defined syntax and relatively simple structure.

(IV) Python is case-sensitive. For example, NUMBER and number are not same in Python.

(V) Python is portable and platform independent, means it can run on various operating systems and hardware platforms.

(VI) Python has a rich library of predefined functions.

(VII) Python is also helpful in web development. Many popular web services and applications are built using Python.

(VIII) Python uses indentation for blocks and nested blocks.

- **Working with Python :** To write and run (execute) a Python program, we needto have a Python interpreter installed on our computer or we can use any online Python interpreter. The interpreter is also called Python shell.

The symbol >>> is the Python prompt, which indicates that the interpreter is ready to take instructions. We can type commands or statements on this prompt to execute them using a Python interpreter.

- **Execution Modes:** There are two ways to use the Python interpreter:

(I) **Interactive mode:** To work in the interactive mode, we can simply type a Python statement on the >>> prompt directly. As soon as we press enter, the interpreter executes the statement and displays the result(s). Interactive mode allows execution of individual statement instantaneously. Working in the interactive mode is convenient for testing a single line code for instant execution. But in the interactive mode, we cannot save the statements for future use and we have to retype the statements to run them again.

(II) **Script mode:** we can write a Python program in a file, save it and then use the interpreter to execute it. Python scripts are saved as files where file name has extension ".py". By default, the Python scripts are saved in the Python installation folder. To execute a script, we can either:

(I) Type the file name along with the path at the prompt.

(II) While working in the script mode, after saving the file, click [Run]->[Run Module] from the menu,

(III) The output appears on shell.

PYTHON KEYWORDS

Keywords are reserved words. Each keyword has a specific meaning to the Python interpreter, and we can use a keyword in our program only for the purpose for which it has been defined.

False	Class	finally	Is	Return
None	continue	for	lamda	try
True	Def	from	nonlocal	while
and	Del	global	not	with
As	Elif	if	or	yield
Assert	Else	import	Pass	
Break	except	in	raise	

IDENTIFIERS

In programming languages, identifiers are names used to identify a variable, function, or other entities in a program. The rules for naming an identifier in Python are as follows:

(I) The name should begin with an uppercase or a lowercase alphabet or an underscore sign (_). This may be followed by any combination of characters a–z, A–Z, 0–9 or underscore (_). Thus, an identifier cannot start with a digit.

(II) It can be of any length. (However, it is preferred to keep it short and meaningful).

(III) It should not be a keyword or reserved word given in Table above.

(IV) We cannot use special symbols like !, @, #, $, %,etc., in identifiers.

For example, to find the average of marks obtained by a student in three subjects, we can choose the identifiers as marks1, marks2, marks3 and avg rather than a, b, c, or A, B, C.

avg = (marks1 + marks2 + marks3)/3

VARIABLES

A variable in a program is uniquely identified by a name(identifier). Variable in Python refers to an object — an item or element that is stored in the memory. Value of a variable can be a string (e.g., 'b', 'Global Citizen'), numeric (e.g., 345) or any combination of alphanumeric characters (CD67). In Python we can use an assignment statement to create new variables and assign specific values to them.

Variable declaration is implicit in Python, means variables are automatically declared and defined when they are assigned a value the first time. Variables must always be assigned values before they are used in expressions as otherwise it will lead to an error in the program. Wherever a variable name occurs in an expression, the interpreter replaces it with the value of that particular variable.

COMMENTS

(I) Comments are used to add a remark or a note in the source code. Comments are not executed by interpreter.

(II) Comments are added with the purpose of making the source code easier for humans to understand.

(III) Comments are used primarily to document the meaning and purpose of source code and its input and output requirements, so that we can remember later how it functions and how to use it.

(IV) For large and complex software, comments may require programmers to work in teams and sometimes, a program written by one programmer is required to be used or maintained by another programmer. In such situations, documentations in the form of comments are needed to understand the working of the program.

(V) In Python, a comment starts with # (hash sign).Everything following the # till the end of that line is treated as a comment and the interpreter simply ignores it while executing the statement.

EVERYTHING IS AN OBJECT

Python treats every value or data item whether numeric, string, or other type as an object in the sense that it can be assigned to some variable or can be passed to a function as an argument.

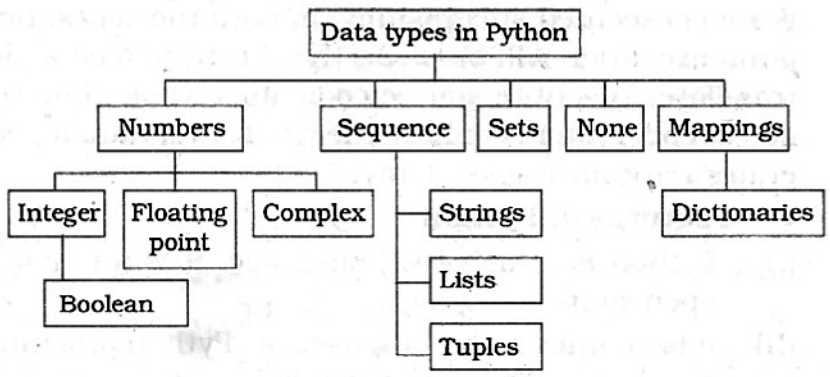
Every object in Python is assigned a unique identity(ID) which remains the same for the lifetime of that object. This ID is akin to the memory address of the object. The function id() returns the identity of an object.

NOTE: In the context of Object Oriented Programming (OOP), objects are a representation of the real world, such as employee, student, vehicle, box, book, etc. In any object oriented programming language like C++, JAVA, etc., each object has two things associated with it: (i) data or attributes and (ii) behaviour or methods. Further there are concepts of class and class hierarchies from which objects can be instantiated.

Python also comes under the category of object oriented programming. However, in Python, the definition of object is loosely casted as some objects may not have attributes or others may not have methods.

DATA TYPES

Every value belongs to a specific data type in Python. Data type identifies the type of data values a variable can hold and the operations that can be performed on that data. Following Figure enlists the data types available in Python.



- **NUMBER:** Number data type stores numerical values only. It isclassified into three different types: int, float and complex.

Type/ Class	Description	Examples
Int	integer numbers	-11, -2, 0, 122, 3
Float	real or floating point numbers	-1.03, 3.0, 15.22
Complex	complex numbers	3 + 4j, 2 – 2j

Boolean data type (bool) is a subtype of integer. Itis a unique data type, consisting of two constants, True and False. Boolean True value is non-zero, non-null and non-empty. Boolean False is the value zero.

Variables of simple data types like integers, float,boolean, etc., hold single values. But such variables are not useful to hold a long list of information, for example, names of the months in a year, names of students in a class, names and numbers in a phone

book or the list of artefacts in a museum. For this, Python provides data types like tuples, lists, dictionaries and sets.

- **Sequence :** A Python sequence is an ordered collection of items,where each item is indexed by an integer. The three types of sequence data types available in Python are Strings, Lists and Tuples.

(I) **String:** It is a group of characters. These characters may be alphabets, digits or special characters including spaces. String values are enclosed either in single quotation marks (e.g., 'Hello') or in double quotation marks (e.g., "Hello"). The quotes are not a part of the string, they are used to mark the beginning and end of the string for the interpreter.

(II) **List:** List is a sequence of items separated by commas and the items are enclosed in square brackets [ ].

(III) **Tuple:** Tuple is a sequence of items separated by commas and items are enclosed in parenthesis ( ). This is unlike list, where values are enclosed in brackets [ ]. Once created, we cannot change the tuple.

- **Set:** Set is an unordered collection of items separated by commas and the items are enclosed in curly brackets { }. A set is similar to list, except that it cannot have duplicate entries. Once created, elements of a set cannot be changed.
- **None:** None is a special data type with a single value. It is used to signify the absence of value in a situation. None supports no special operations, and it is neither same as False nor 0 (zero).
- **Mapping:** Mapping is an unordered data type in Python. Currently, there is only one standard mapping data type in Python called dictionary.

Dictionary in Python holds data items in key-value pairs. Items in a dictionary are enclosed in curly brackets { }. Dictionaries permit faster access to data. Every key is separated from its value using a colon (:) sign. The key: value pairs of a dictionary can be accessed using the key. The keys are usually strings and their values can be any data type. In order to access any value in the dictionary, we have to specify its key in square brackets [ ].

- **Mutable and Immutable Data Types:** Variables whose values can be changed after they are created and assigned are called mutable. Variables whose values cannot be changed after they are created and assigned are called immutable. When an attempt is made to update the value of an immutable variable, the old variable is destroyed and a new variable is created by the same name in memory.
- **Deciding Usage of Python Data Types:**

(I) It is preferred to use lists when we need a simple iterable collection of data that may go for frequent modifications.

(II) Tuples are used when we do not need any change in the data.

(III) When we need uniqueness of elements and to avoid duplicacy it is preferable to use sets.

(IV) If our data is being constantly modified or we need a fast lookup based on a custom key or we need a logical association between the key : value pair, it is advised to use dictionaries.

OPERATORS

An operator is used to perform specific mathematical or logical operation on values. The values that the operators work on are called operands. For example, in the expression 10 + num, the value 10, and the variable num are operands and the + (plus) sign is an operator.

(I) **Arithmetic Operators :** Python supports arithmetic operators that are used to perform the four basic arithmetic operations as well as modular division, floor division and exponentiation.

Operator	Operation	Description	Example

+	Addition	Adds the two numeric values on either side of the operator.
		
		This operator can also be used to concatenate two strings on either side of the operator.
			>>> num1 = 5
			>>> num2 = 6
			>>> num1 + num2
			11
			>>> str1 = "Hello"
			>>> str2 = "India"
			>>> str1 + str2
			'HelloIndia

-	Subtraction	Subtracts the operand on the right from the operand on the left	>>> num1 = 5
			>>> num2 = 6
			>>> num1 - num2
			-1

*	Multiplication	Multiplies the two values on both side of the operator
		
		Repeats the item on left of the operator if first operand is a string and second operand is an integer value
			>>> num1 = 5
			>>> num2 = 6
			>>> num1 * num2
			30
			>>> str1 = 'India'
			>>> str1 * 2
			'IndiaIndia'

/	Division	Divides the operand on the left by the operand on the right and returns the quotient	>>> num1 = 8
			>>> num2 = 4
			>>> num2 / num1
			0.5

%	Modulus	Divides the operand on the left by the operand on the right and returns the remainder.	>>> num1 = 13
			>>> num2 = 5
			>>> num1 % num2
			3

//	Floor Division	Divides the operand on the left by the operand on the right and returns the quotient by removing the decimal part. It is sometimes also called integer division.	>>> num1 = 13
			>>> num2 = 4
			>>> num1 // num2
			3
			>>> num2 // num1
			0

**	Exponent	Performs exponential (power) calculation on operands. That is, raise the operand on the left to the power of the operand on the right	>>> num1 = 3
			>>> num2 = 4
			>>> num1 ** num2
			81

(II) **Relational Operators :** Relational operator compares the values of the operands on its either side and determines the relationship among them.

Operator	Operation	Description	Example
==	Equals to	If the values of two operands are equal, then the condition is True, otherwise it is False	>>> num1 == num2 False >> str1 == str2 False

!=	Not equal to	If values of two operands are not equal, then condition is True, otherwise it is False	>>> num1 != num2
			True
			>>> str1 != str2
			True
			>>> num1 != num3
			False

>	Greater than	If the value of the left-side operand is greater than the value of the rightside operand, then condition is True, otherwise it is False	>>> num1 > num2
			True
			>>> str1 > str2
			True

<	Less than	If the value of the left-side operand is less than the value of the rightside operand, then condition is True,otherwise it is False	>>> num1 < num3
			False
			>>> str2 < str1
			True

>=	Greater than or equal to	If the value of the left-side operand is greater than or equal to the value of the right-side operand, then condition is True, otherwise it is False	>>> num1 >= num2
			True
			>>> num2 >= num3
			False
			>>> str1 >= str2
			True

<=	Less than or equal to	If the value of the left operand is less than or equal to the value of the right operand, then is True otherwise it is False	>>> num1 <= num2
			False
			>>> num2 <= num3
			True
			>>> str1 <= str2
			False

(III) **Assignment Operators:** Assignment operator assigns or changes the value of the variable on its left.

Operator	Description	Example

=	Assigns value from right-side operand to left side operand	>>> num1 = 2
		>>> num2 = num1
		>>> num2
		2
		>>> country = 'India'
		>>> country
		'India'

+=	It adds the value of right-side operand to the left-side operand and assigns the result to the left-side operand
	Note: x += y is same as x = x + y
		>>> num1 = 10
		>>> num2 = 2
		>>> num1 += num2
		>>> num1
		12
		>>> num2
		2
		>>> str1 = 'Hello'
		>>> str2 = 'India'
		>>> str1 += str2
		>>> str1
		'HelloIndia'

-=	It subtracts the value of right-side operand from the left-side operand and assigns the result to left-side operand
	Note: x -= y is same as x = x - y
		>>> num1 = 10
		>>> num2 = 2
		>>> num1 -= num2
		>>> num1
		8

*=	It multiplies the value of right-side operand with the value of left-side operand and assigns the result to left-side operand
	Note: x *= y is same as x = x * y
		>>> num1 = 2
		>>> num2 = 3
		>>> num1 *= 3
		>>> num1
		6
		>>> a = 'India'
		>>> a *= 3
		>>> a
		'IndiaIndiaIndia'

/=	It divides the value of left-side operand by the value of right-side operand and assigns the result to left-side operand
	Note: x /= y is same as x = x / y
		>>> num1 = 6
		>>> num2 = 3
		>>> num1 /= num2
		>>> num1
		2.0

%=	It performs modulus operation using two operands and assigns the result to left-side operand
	Note: x %= y is same as x = x % y
		>>> num1 = 7
		>>> num2 = 3
		>>> num1 %= num2
		>>> num1

//=	It performs floor division using two operands and assigns the result to left-side operand
	Note: x //= y is same as x = x // y
		>>> num1 = 7
		>>> num2 = 3
		>>> num1 //= num2
		>>> num1
		2

**=	It performs exponential (power) calculation on operators and assigns value to the left-side operand
	Note: x **= y is same as x = x ** y
		>>> num1 = 2
		>>> num2 = 3
		>>> num1 **= num2
		>>> num1
		8

(IV) **Logical operator:** There are three logical operators supported by Python.These operators (and, or, not) are to be written in lower case only. The logical operator evaluates to either True or False based on the logical operands on either side. Every value is logically either True or False. By default, all values are True except None, False, 0 (zero), empty collections "", (), [], {}, and few other special values. So if we say num1 = 10, num2 = -20, then both num1 and num2 are logically True.

(V) **Identity Operators :** Identity operators are used to determine whether the value of a variable is of a certain type or not. Identity operators can also be used to determine whether two variables are referring to the same object or not. There are two identity operators.

Operator	Description	Example

is	Evaluates True if the variables on either side of the operator point towards the same memory location and False otherwise. var1 is var2 results to True if id(var1) is equal to id(var2)	>>> num1 = 5
		>>> type(num1) is int
		True
		>>> num2 = num1
		>>> id(num1)
		1433920576
		>>> id(num2)
		1433920576
		>>> num1 is num2
		True

is not	Evaluates to False if the variables on either side of the operator point to the same memory location and True otherwise. var1 is not var2 results to True if id(var1) is not equal to id(var2)	>>> num1 is not num2 False

(VI) **Membership operator:** Membership operators are used to check if a value is a member of the given sequence or not.

Operator	Description	Example

In	Returns True if the variable/value is found in the specified sequence and False otherwise	>>> a = [1,2,3]
		>>> 2 in a
		True
		>>> '1' in a
		False

not in	Returns True if the variable/value is not found in the specified sequence and False otherwise	>>> a = [1,2,3]
		>>> 10 not in a
		True
		>>> 1 not in a
		False

EXPRESSIONS

(I) An expression is defined as a combination of constants, variables, and operators.

(II) An expression always evaluates to a value.

(III) A value or a standalone variable is also considered as an expression but a standalone operator is not an expression.

- **Precedence of Operators :** When an expression contains different kinds of operators, precedence determines which operator should be applied first. Higher precedence operator is evaluated before the lower precedence operator. Most of the operators studied till now are binary operators. Binary operators are operators with two operands. The unary operators need only one operand, and they have a higher precedence than the binary operators. The minus (-) as well as + (plus) operators can act as both unary and binary operators, but not is a unary logical operator.

```
#Depth is using - (minus) as unary operator
Value = -Depth
#not is a unary operator, negates True
print(not(True))
```

- **Precedence of all operators in Python :**

Order of Precedence	Operators	Description
1.	**	Exponentiation (raise to the power)
2.	~ ,+, -	Complement, unary plus and unary minus
3.	* ,/, %, //	Multiply, divide, modulo and floor division
4.	+, -	Addition and subtraction
5.	<= , < , > , >=, == , !=	Relational and Comparison operators
6.	=, %=, /=, //=, -=, +=, *=, **=	Assignment operators
7.	is, is not	Identity operators
8.	in, not in	Membership operators
9.	not	Logical operators
10.	And	Logical operators
11.	or	Logical operators

Note:

(I) Parenthesis can be used to override the precedence of operators. The expression within () is evaluated first.

(II) For operators with equal precedence, the expression is evaluated from left to right.

STATEMENT

In Python, a statement is a unit of code that the Python interpreter can execute.

INPUT AND OUTPUT

In Python, wehave the input() function for taking the user input. The input() function prompts the user to enter data. It accepts all user input as string. The user may enter a number or a string but the input() function treats them as strings only. The syntax for input() is:

input ([Prompt])

Prompt is the string we may like to display on the screen prior to taking the input, and it is optional. When a prompt is specified, first it is displayed on the screen after which the user can enter data. The input() takes exactly what is typed from the keyboard, converts it into a string and assigns it to the variable on left-hand side of the assignment operator (=). Entering data for the input function is terminated by pressing the enter key.

Python uses the print() function to output data tostandard output device — the screen.

The syntax for print() is:

```
print(value [, ..., sep = ' ', end = '\n'])
```

- sep: The optional parameter sep is a separator between the output values. We can use a character, integer or a string as a separator. The default separator is space.
- end: This is also optional and it allows us to specify any string to be appended after the last value. The default is a new line.

TYPE CONVERSION

(I) **Explicit Conversion :** Explicit conversion, also called type casting happens when data type conversion takes place because the programmer forced it in the program. The general form of an explicit data type conversion is: (new_data_type) (expression)

Following are some of the functions in Python that are used for explicitly converting an expression or a variable to a different type.

Function	Description
int(x)	Converts x to an integer
float(x)	Converts x to a floating-point number
str(x)	Converts x to a string representation
chr(x)	Converts ASCII value of x to character
ord(x)	returns the character associated with the ASCII code x

The interpreter cannot convert an integer value to string implicitly. It may appear quite intuitive that the program should convert the integer value to a string depending upon the usage. However, the interpreter may not decide on its own when to convert as there is a risk of loss of information. Python provides the mechanism of the explicit type conversion so that one can clearly state the desired outcome.

Type casting is needed to convert float to string. In Python, one can convert string to integer or float values whenever required.

(II) **Implicit Conversion :**Implicit conversion, also known as coercion, happens when data type conversion is done automatically by Python and is not instructed by the programmer.

The float value not converted to an integer due to type promotion that allows performing operations (whenever possible) by converting data into a wider-sized data type without any loss of information.

DEBUGGING

The process of identifying and removing mistakes also known as bugs or errors, from a program which a programer make during writing a program is called debugging. Errors occurring in programs can be categorised as:

(i) Syntax errors

(ii) Logical errors

(iii) Runtime errors

- **Syntax errors :** Like other programming languages, Python has its own rules that determine its syntax. The interpreter interprets the statements only if it is syntactically (as per the rules of Python) correct. If any syntax error is present, the interpreter shows error message(s) and stops the execution there.
- **Logical errors :** A logical error is a bug in the program that causes it to behave incorrectly. A logical error produces an undesired output but without abrupt termination of the execution of the program.
- **Runtime errors :** A runtime error causes abnormal termination of program while it is executing. Runtime error is when the statement is correct syntactically, but the interpreter cannot execute it. Runtime errors do not appear until after the program starts running or executing.

□□□

6 FLOW OF CONTROL (NCERT CLASS 11)

INTRODUCTION

The order of execution of the statements in a program is known as flow of control. The flow of control can be implemented using control structures.

CONTROL STATEMENTS

Flow control statements are used to control the flow of execution depending upon the specified condition/logic.

Sequential control statement - Sequential execution is when statements are executed one after another in order. We don't need to do anything more for this to happen as python compiler itself do it.

There are three types of control statements:-

(I) Decision Making Statements/If control statement.

(II) Iteration Statements (Loop control statement).

(III) Jump Statements (break, continue, pass).

Decision Making Statement :

Decision making statement used to control the flow of execution of program depending upon condition.

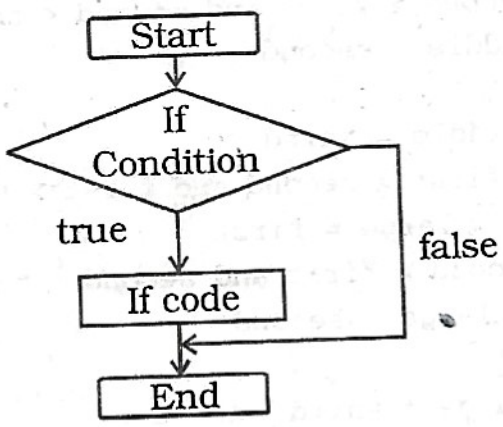
There are three types of decision making statemen:-

(I) if statements.

(II) if-else statements.

(III) Nested if-else statement.

- **if statements :** An if statement is a programming conditional statement that, if proved true, performs a function or displays information.



if statements

Syntax:

```
if(condition):
statement
[statements]
e.g.
noofbooks = 2
if (noofbooks == 2):
print('You have ')
print('two books')
print('outside of if statement')
Output
You have two books.
```

Note : To indicate a block of code in Python, you must indent each line of the block by the same amount. In above e.g. both print statements are part of if condition because of both are at same level indented but not the third print statement.

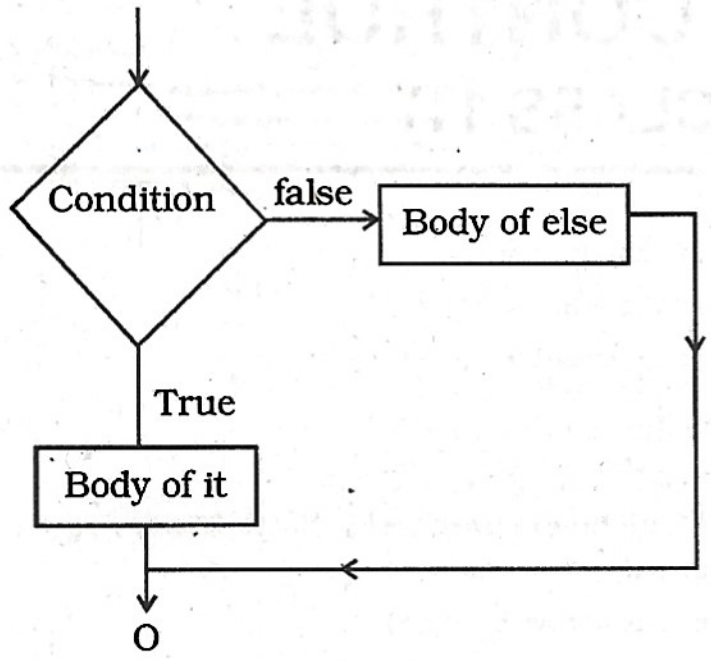
- **if-else statements :**

```
#find absolute value
a=int(input("enter a number"))
if(a<0):
a=a*-1
print(a)
#it will always return value in positive.
```

- **if statements :**

```
Using logical operator in if statement
x=1
 y=2
 if(x==1 and y==2):
print('condition matcing the criteria')
Output :-
condition matcing the criteria
.....
a=100
if not(a == 20):
print('a is not equal to 20')
Output :-
a is not equal to 20.
```

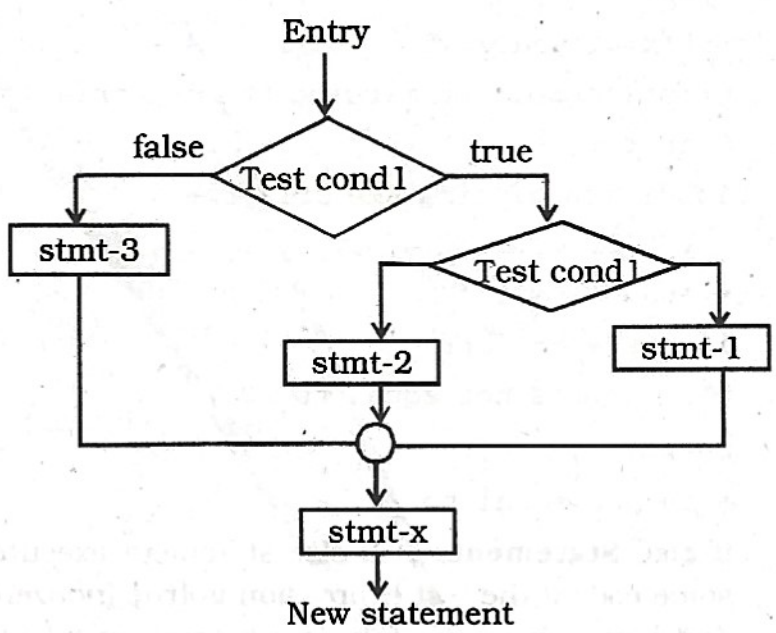
- **if-else Statements :** If-else statement executes some code if the test expression is true (nonzero) and some other code if the test expression is false.



if-else Statements

```
Syntax:
 if(condition):
                    statements
else:
                    statements
e.g.
a=10
 if(a < 100):
    print('less than 100')
 else:
     print('more than equal 100')
 OUTPUT
 less than 100.
```

(III) **Nested if-else statement :** The nested if...else statement allows you to check for multiple test expressions and execute different codes for more than two conditions.



Nested if-else statement:

```
Syntax:
If (condition):
                    statements
elif (condition):
                    statements
else:
                    statements
e.g.
num = float(input("Enter a number: "))
if num >= 0:
   if num == 0:
   print("Zero")
else:
    print("Positive number")
else:
    print("Negative number")
OUTPUT
Enter a number: 5
Positive number
Nested if-else Statements (sort 3 numbers)
first = int(input("Enter the first number:
"))
second = int(input("Enter the second number:
"))
third = int(input("Enter the third number:
"))
small = 0
middle = 0
large = 0
if first < third and first < second:
   small = first
if second < third and second < first:
   small = second
else:
   small = third
elif first < second and first < third:
     middle = first
if second > first and second < third:
   middle = second
else:
    middle = third
elif first > second and first > third:
        large = first
if second > first and second > third:
      large = second
else:
    large = third
print("The numbers in accending order are:
", small, middle, large).
Nested if-else Statements (Check leap year
/ divisibility):-
year = int(input("Enter a year: "))
```

```
if (year % 4) == 0:
   if (year % 100) == 0:
   if (year % 400) == 0:
   print("{0} is a leap year".format(year))
else:
 print("{0} is not a leap year".format(year))
else:
 print("{0} is a leap year".format(year))
else:
 print("{0} is not a leap year".format(year)).
```

ITERATION STATEMENTS (LOOPS)

Iteration statements(loop) are used to execute a block of statements as long as the condition is true. Loops statements are used when we need to run same code again and again.

Python Iteration (Loops) statements are of three type :

(I) While Loop

(II) For Loop

(III) Nested For Loops

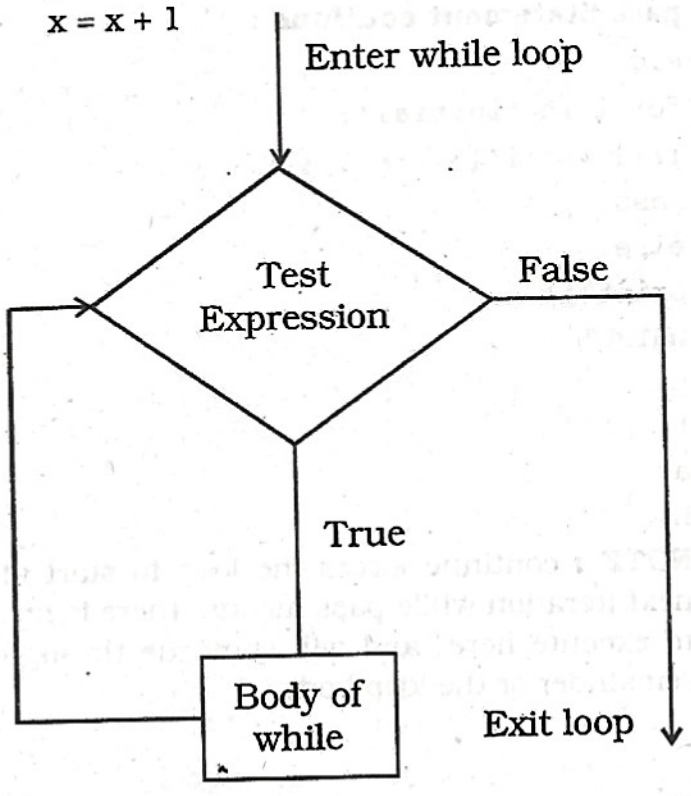
(I) **While Loop :** It is used to execute a block of statement as long as a given condition is true. And when the condition become false, the control will come out of the loop. The condition is checked every time at the beginning of the loop.

Syntax

```
while (condition):
        statement
       [statements]
```

e.g.

```
x = 1
while (x <= 4):
print(x)
x = x + 1
```



- **While Loop continue (While Loop With Else) :**

e.g.

```
x = 1
while (x < 3):
 print('inside while loop value of x is ',x)
x = x + 1
else:
   print('inside else value of x is ', x)
```

Output

inside while loop value of x is 1

inside while loop value of x is 2

inside else value of x is 3

- **While Loop continue (Infinite While Loop) :**

```
e.g.
x = 5
while (x == 5):
print('inside loop')
```

Output

Inside loop

Inside loop

...

...

(II) **For Loop :** It is used to iterate over items of any sequence, such as a list or a string.

Syntax

```
for val in sequence:
   statements
```

e.g.

```
for i in range(3,5):
    print(i)
```

Output

3

4

- For Loop continue:- Example programs

```
for i in range(5,3,-1):
    print(i)
```

Output

5

4

range() Function Parameters

start: Starting number of the sequence.

stop: Generate numbers up to, but not including this number.

step(Optional): Determines the increment between each numbers in the sequence.

- **For Loop continue :** Example programs with range() and len() function

```
fruits = ['banana', 'apple', 'mango']
```

```
for index in range(len(fruits)):
print ('Current fruit :', fruits[index] )
```

range() with len() Function Parameters.

- **For Loop continue :** For Loop With Else

e.g.

```
for i in range(1, 4):
print(i)
else: # Executed because no break in for
print("No Break")
```

Output

1

2

3

No Break

- For Loop continue:- Nested For Loop

e.g.

```
for i in range(1,3):
    for j in range(1,11):
    k=i*j
    print (k,.end=' ')
print()
```

Output

1 2 3 4 5 6 7 8 9 10

2 4 6 8 10 12 14 16 18 20

- **For Loop continue :** Compound Interest calculation:-

```
n=int(input("Enter the principle amount:"))
rate=int(input("Enter the rate:"))
years=int(input("Enter the number of years:"))
for i in range(years):
n=n+((n*rate)/100)
print(n)
```

(III) **Jump Statements :** Jump statements are used to transfer the program's control from one location to another. Means these are used to alter the flow of a loop like - to skip a part of a loop or terminate a loop.

There are three types of jump statements used in python.

(I) break

(II) continue

(III) pass

- break : it is used to terminate the loop.

e.g.

```
for val in "string":
    if val == "i":
     break
    print(val)
print("The end")
```

Output

s

t

r

The end

- **continue :** It is used to skip all the remaining statements in the loop and move controls back to the top of the loop.

e.g.

```
for val in "init":
    if val == "i":
    continue
print(val)
print("The end")
```

Output

n

t

The end

- **pass Statement :** This statement does nothing. It can be used when a statement is required syntactically but the program requires no action.

Use in loop

while True:

pass # Busy-wait for keyboard interrupt (Ctrl+C)

In function : It makes a controller to pass by without executing any code.

e.g.

```
def myfun():
```

pass #if we don't use pass here then error message will be shown print('my program')

OUTPUT

My program

- **pass Statement continue :**

e.g.

```
for i in 'initial':
if(i == 'i'):
pass
else:
print(i)
```

OUTPUT

n

t

a

L

NOTE : continue forces the loop to start at the next iteration while pass means "there is no code to execute here" and will continue through the remainder or the loop body.

❑❑❑

7 FUNCTIONS
(NCERT CLASS 11)

INTRODUCTION

A function is a named sequence of statement(s) that performs a computation. It containsline of code(s) that are executed sequentially from top to bottom by Python interpreter. They are the most important building blocks for any software in Python.

Functions can be categorized as belonging to

(I) Modules

(II) Built in

(III) User Defined

- **Module:-**A module is a file containing Python definitions (i.e. functions) and statements. Standard library of Python is extended as module(s) to a programmer. Definitions from the module can be used within the code of a program. To use these modules in the program, a programmer needs to import the module. Once you import a module, you can reference (use), any of its functions or variables in your code. There are many ways to import a module in your program, the one's which you should know are: import and from.

Import : It is simplest and most common way to use modules in our code. Its syntax is:

```
modulename1 [,modulename2, ——]
```

Example

```
>>> import math
```

On execution of this statement, Python will

(i) search for the file "math.py".

(ii) Create space where modules definition & variable will be created,

(iii) then execute the statements in the module.

Now the definitions of the module will become part of the code in which the module was imported.

To use/ access/invoke a function, you will specify the module name and name of the function- separated by dot (.). This format is also known as dot notation.

Example

```
>>> value= math.sqrt (25) → # dot notation
```

The example uses sqrt( ) function of module math to calculate square root of the value provided in parenthesis, and returns the result which is inserted in the value. The expression (variable) written in parenthesis is known as argument (actual argument). It is common to say that the function takes arguments and return the result.

This statement invokes the sqrt ( ) function. Many function invoke statement(s), such as

```
>>> type ( )
>>> int ( ), etc.
```

From Statement : It is used to get a specific function in the code instead of the complete module file. If we know beforehand which function(s), we will be needing, then we may use from. For modules having large no. of functions, it is recommended to use from instead of import. Its syntax is

```
>>> from modulename import functionname [,
functionname.....]
```

Example

```
>>> from math import sqrt
value = sqrt (25)
```

Here, we are importing sqrt function only, instead of the complete math module. Now sqrt( ) function will be directly referenced to. These two statements are equivalent to previous example.

```
from modulename import *
```

will import everything from the file.

Note : You normally put all import statement(s) at the beginning of the Python file buttechnically they can be anywhere in program.

Functions available in math module:

Name of the function	Description	Example
ceil(x)	It returns the smallest integer not less than x, where *x is a numeric expression.*	math.ceil(-45.17) **-45.0** math.ceil(100.12) **101.0** math.ceil(100.72) **101.0**
floor(x)	It returns the largest integer not greater than x, where *x is a numeric expression.*	math.floor(-45.17) **-46.0** math.floor(100.12) **100.0** math.floor(100.72) **100.0**
fabs(x)	It returns the absolute value of x, *where x is a numeric value.*	math.fabs(-45.17) **45.17** math.fabs(100.12) **100.12** math.fabs(100.72) **100.72**
exp(x)	It returns exponential of x: ex, where x is a numeric expression.	math.exp(-45.17) **2.41500621326e-20** math.exp(100.12) **3.03084361407e+43** math.exp(100.72) **5.52255713025e+43**
log(x)	It returns natural logarithm of x, for x > 0, where *x is a numeric expression.*	math.log(100.12), **4.60636946656** math.log(100.72) **4.61234438974**
log10(x)	It returns base-10 logarithm of x for x > 0, where *x is a numeric expression.*	math.log10(100.12) **2.00052084094** math.log10(100.72) **2.0031157171**
pow(x, y)	It returns the value of xy; *where x and y are numeric expressions.*	math.pow(100, 2) **10000.0** math.pow(100, -2) **0.0001** math.pow(2, 4) **16.0** math.pow(3, 0) **1.0**
sqrt (x)	It returns the square root of x for x > 0, where x is a numeric expression.	math.sqrt(100) **10.0** math.sqrt(7) **2.64575131106**

cos (x)	It returns the cosine of x in radians, *where x is a numeric expression.*	math.cos(3) **-0.9899924966** math.cos(-3) **-0.9899924966** math.cos(0) **1.0** math.cos(math.pi) **-1.0**
sin (x)	It returns the sine of x, in radians, *where x must be a numeric value.*	math.sin(3) **0.14112000806** math.sin(-3) **-0.14112000806** math.sin(0) **0.0**
tan (x)	It returns the tangent of x in radians, *where x must be a numeric value.*	math.tan(3) **-0.142546543074** math.tan(-3) **0.142546543074** math.tan(0) **0.0**
degrees (x)	It converts angle x from radians to degrees, *where x must be a numeric value.*	math.degrees(3) **171.887338539** math.degrees(-3) **-171.887338539** math.degrees(0) **0.0**
radians(x)	It converts angle x from degrees to radians, *where x must be a numeric value.*	math.radians(3) **0.0523598775598** math.radians(-3) **-0.0523598775598** math.radians(0) **0.0**

Some functions from random module are :

Name Of Function	Description	Example
random ( )	It returns a random float x, such that $0 \leq x<1$	>>>random.random ( ) **0.281954791393** >>>random.random ( ) **0.309090465205**

randint (a, b)	It returns a int x between a & b such that
	$a \leq x \leq b$
		>>> random.randint (1,10)
		5
		>>> random.randint (-2,20)
		-1

uniform (a,b)	It returns a floating point number x, such that a <= x < b	>>>random.uniform (5, 10) **5.52615217015**
randrange ([start,] stop [,step])	It returns a random item from the given range ,1000,3)	>>>random.randrange(100

- **Built in Function:-**Built in functions are the function(s) that are built into Python and can be accessed by a programmer. These are always available and for using them, we don't have to import any module (file). Python has a small set of built-in functions as most of the functions have been partitioned to modules. This was done to keep core language precise.

Name	Description	Example
abs (x)	It returns distance between x and zero, where *x is a numeric expression.*	>>>abs(-45) **45** >>>abs(119L) **119**
max(x, y, z, ...)	It returns the largest of its arguments: where x, y and z are numeric variable/expression.	>>>max(80, 100, 1000) **1000** >>>max(-80, -20, -10) **-10**

min(x, y, z, ...)	It returns the smallest of its arguments; where x, y, and z are numeric variable/expression.	>>> min(80, 100, 1000)
		80
		>>> min(-80, -20, -10)
		-80

cmp(x, y)	It returns the sign of the difference of two numbers: -1 if x < y, 0 if x == y, or 1 if x > y, *where x and y are* **1** *numeric variable/expression.*	>>>cmp(80, 100) **-1** >>>cmp(180, 100) **1**

divmod (x,y)	Returns both quotient and remainder by division through a tuple, when x is divided by y; where x & y are variable/expression.	>>> divmod (14,5)
		(2,4)
		>>> divmod (2.7, 1.5)
		(1.0, 1.20000)

len (s)	Return the length (the number of items) of an object. The argument may be a sequence (string, tuple or list) or a mapping (dictionary).	>>> a= [1,2,3]
		>>>len (a)
		3
		>>> b= "Hello"
		>>> len (b)
		5

range (*start*, *stop*[, *step*])	This is a versatile function to create lists containing arithmetic progressions. It is most often used in for loops. The arguments must be plain integers. If the *step* argument is omitted, it defaults to 1. If the *start* argument is omitted, it defaults to 0. The full form returns a list of plain integers [start, start + step, start + 2 * step, ...]. If *step* is positive, the last element is the largest start +,i * step less than *stop*; if *step* is negative, the last element is the smallest start + i * step greater than *stop*. *step* must not be zero (or else Value Error is raised).	>>> range(10)
		[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
		>>> range(1, 11)
		[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
		>>> range(0, 30, 5)
		[0, 5, 10, 15, 20, 25]
		>>> range(0, 10, 3)
		[0, 3, 6, 9]
		>>> range(0, -10, -1)
		[0, -1, -2, -3, -4, -5, -6, -7, -8, -9]
		>>> range(0)
		[]
		>>>range(1, 0)
		[]

round(x [, n])	It returns float x rounded to n digits from the decimal point, *where x and n are numeric expressions.* If n is not provided then x is rounded to 0 decimal digits.	>>>round(80.23456, 2) **80.23** >>>round(-100.000056, 3) **-100.0** >>> round (80.23456) **80.0**

- **Composition:-** Composition is an art of combining simple function(s) to build more complicated ones, i.e., result of one function is used as the input to another.

Example :

Suppose we have two functions fn1 & fn2, such that

a= fn2 (x)

b= fn1 (a)

then call to the two functions can be combined as

b= fn1 (fn2 (x))

Similarly, we can have statement composed of more than two functions. In that result of one function is passed as argument to next and result of the last one is the final result.

Composition is used to package the code into modules, which may be used in many different unrelated places and situations. Also it is easy to maintain the code.

Note: Python also allow us to take elements of program and compose them.

- **User Defined Functions:-**

So far we have only seen the functions which come with Python either in some file(module) or in interpreter itself (built in), but it is also possible for programmer to write their own function(s). These functions can then be combined to form a module which can then be used in other programs by importing them.

To define a function keyword def is used. After the keyword comes an identifier i.e. name of the function, followed by parenthesized list of parameters and the colon whichends up the line. Next follows the block of statement(s) that are the part of function.

BLOCK OF STATEMENTS

A block is one or more lines of code, grouped together so that they are treated as one big sequence of statements while executing. In Python, statements in a block are written with indentation. Usually, a block begins when a line is indented (by four spaces) and all the statements of the block should be at same indent level. A block within block begins when its first statement is indented by four space, i.e., in total eight spaces. To end a block, write the next statement with the same indentation before the block started.

Now, lets move back to function- the Syntax of function is:

```
def NAME ([PARAMETER1, PARAMETER2, .....]):
#Square brackets include
statement(s) #optional part of statement
```

Let's write a function to greet the world:

```
def sayHello (): # Line No. 1
print "Hello World!" # Line No.2
```

The first line of function definition, i.e., Line No. 1 is called header and the rest, i.e. LineNo. 2 in our example, is known as body. Name of the function is sayHello, and empty parenthesis indicates no parameters. Body of the function contains one Python statement, which displays a string constant on screen.

- **Function Header :** It begins with the keyword def and ends with colon and contains the function identification details. As it ends with colon, we can say that what follows next is, block of statements.
- **Function Body :**Consisting of sequence of indented (4 space) Python statement(s), to perform a task. Defining a function will create a variable with same name, but does not generate any result. The body of the function gets executed only when the function is called/invoked. Function call contains the name of the function (being executed) followed by the list of values (i.e. arguments) in parenthesis. These arguments are assigned to parameters from LHS.

```
>>> sayHello ()  # Call/invoke statement of this function
Will produce following on screen
Hello World!
```

Let's know more about def. It is an executable statement. At the time of execution afunction is created and a name (name of the function) is assigned to it. Because it is a statement, def can appear anywhere in the program. It can even be nested.

Example

```
if condition:
    def fun ( ): # function definition one way
    .
    .
    .
  else:
    def fun ( ):          # function definition other way
```

fun ( ) # calls the function selected.

This way we can provide an alternative definition to the function. This is possible because def is evaluated when it is reached and executed.

def fun (a):

- **Let's explore Function body :-** The first statement of the function body can optionally be a string constant, docstring, enclosed in triple quotes. It contains the essential information that someone might need about the function, such as:-
  (I) What function does (without How it does) i.e. summary of its purpose.
  (II) Type of parameters it takes
  (III) Effect of parameter on behavior of functions, etc.

DocString is an important tool to document the program better, and makes it easier tounderstand. We can actually access docstring of a function using __ doc__ (function name). Also, when you used help(), then Python will provide you with docstring of that function on screen. So it is strongly recommended to use docstring ... when you write functions.

Example:

```
def area (radius):
""" calculates area of a circle. docstring begins
require an integer or float value to calculate area.
returns the calculated value to calling function
""" docstring ends
a=radius**2
        return a
```

Function is pretty simple and its objective is pretty much clear from the docString added to the body.

The last statement of the function, i.e. return statement returns a value from thefunction. Return statement may contain a constant/literal, variable, expression or function, if return is used without anything, it will return None. In our example value of a variable area is returned.

Instead of writing two statements in the function, i.e.

```
a = radius **2
return a
```

We could have written

```
radius **2
```

Here the function will first calculate and then return the value of the expression.

It is possible that a function might not return a value, as sayHello( ) was not returning a value. sayHello ( ) prints a message on screen and does not contain a return statement, such functions are called void functions.

Void functions might display something on the screen or have some other effect, but they don't have a return value. If you try to assign the result of such function to a variable, you get a special value called None.

DocString Conventions:

(I) The first line of a docstring starts with capital letter and ends with a period (.)
(II) Second line is left blank (it visually separates summary from other description).
(III) Other details of docstring start from 3rd line.

PARAMETERS AND ARGUMENTS

Parameters are the value(s) provided in the parenthesis when we write function header.These are the values required by function to work. Let's understand this with the help of function written for calculating area of circle.

Radius is a parameter to function area.

If there is more than one value required by the function to work on, then, all of them will be listed in parameter list separated by comma.

Arguments are the value(s) provided in function call/invoke statement. List of arguments should be supplied in same way as parameters are listed. Bounding ofparameters to arguments is done 1:1, and so there should be same number and type ofarguments as mentioned in parameter list.

Example :

```
of argument in function call
>>> area (5)
```

5 is an argument. An argument can be constant, variable, or expression.

SCOPE OF VARIABLES

Scope of variable refers to the part of the program, where it is visible, i.e., area whereyou can refer (use) it. We can say that scope holds the current set of variables and their values. We will study two types of scope of variables- global scope or local scope.

(I) **Global Scope:-** A variable, with global scope can be used anywhere in the program. It can be created by defining a variable outside the scope of any function/block.
(II) **Local Scope:-**A variable with local scope can be accessed only within the function/block that it is created in. When a variable is created inside the function/block, the variable becomes local to it. A local variable only exists while the function is executing.

A global variable remains global, till it is not recreated inside the function/block.

More on defining Functions

It is possible to provide parameters of function with some default value. In case the userdoes not want to provide values (argument) for all of them at the time of calling, we can provide default argument values.

Example :

→ Default value to parameter

?

```
def greet (message,
times=1):
        print message * times
>>> greet ("Welcome") # calling function with one argument value
>>> greet ("Hello", 2) # calling function with both the argument values.
Will result in:
Welcome
HelloHello
```

The function greet () is used to print a message (string) given number of times. If thesecond argument value, is not specified, then parameter times work with the default value provided to it. In the first call to greet (), only one argument value is provided,which is passed on to the first parameter from LHS and the string is printed only onceas the variable times take default value 1. In the second call to greet (), we supply both the argument values a string and 2, saying that we want to print the message twice. So now, parameter times get the value 2 instead of default 1 and the message is printed twice.

As we have seen functions with default argument values, they can be called in with fewer arguments, then it is designed to allow.

Note:

(I) The default value assigned to the parameter should be a constant only.

(II) Only those parameters which are at the end of the list can be given default value.

(III) You cannot have a parameter on left with default argument value, without assigning default values to parameters lying on its right side.

(IV) The default value is evaluated only once, at the point of function definition.

If there is a function with many parameters and we want to specify only some of them in function call, then value for such parameters can be provided by using their name, instead of the position (order)- this is called keyword arguments.

- While using keyword arguments, following should be kept in mind:
  - (I) An argument list must have any positional arguments followed by any keywords arguments.
  - (II) Keywords in argument list should be from the list of parameters name only.
  - (III) No parameter should receive value more than once.
  - (IV) Parameter names corresponding to positional arguments cannot be used as keywords in the same calls.
- Advantages of writing functions with keyword arguments are:
  - (I) Using the function is easier as we do not need to remember about the order of the arguments.
  - (II) We can specify values of only those parameters to which we want to, as – other parameters have default argument values.

In python, as function definition happens at run time, so functions can be bound to other names. This allow us to

(I) Pass function as parameter

(II) Use/invoke function by two names.

Example :

```
def x ( ):
        print 20
        >>> y=x
        >>>x ( )
        >>>y ( )
        20
```

Example :

```
def x ( ):
         print 20 -
        def test (fn):
        for I in range (4):
        fn( )
        >>> test (x)
        20
        20
        20
        20
```

FLOW OF EXECUTION OF PROGRAM CONTAINING FUNCTION CALL

Execution always begins at the first statement of the program. Statements are executed one at a time, in order from top to bottom. Function definition does not alter the flow of execution of program, as the statement inside the function is not executed until the function is called.

On a function call, instead of going to the next statement of program, the control jumps to the body of the function; executes all statements of the function in the order from top to bottom and then comes back to the point where it left off. This remains simple, till a function does not call another function. Simillarly, in the middle of a function, program might have to execute statements of the other function and so on.

Don't worry; Python is good at keeping track of execution, so each time a function completes, the program picks up from the place it left last, until it gets to end of program, where it terminates.

Note :

(I) Python does not allow you to call a function before the function is declared.

(II) When you write the name of a function without parenthesis, it is interpreted as the reference, when you write the function name with parenthesis, the interpreter invoke the function (object). □□□

8 STRINGS
(NCERT CLASS 11)

INTRODUCTION

In python, consecutive sequence of characters is known as a string. An individual character in a string is accessed using a subscript (index). The subscript should always be an integer (positive or negative). A subscript starts from 0.

Important points about accessing elements in the strings using subscripts :

(I) Positive subscript helps in accessing the string from the beginning.

(II) Negative subscript helps in accessing the string from the end.

(III) Subscript 0 or –ve n(where n is length of the string) displays the first element.

(IV) Subscript 1 or –ve (n-1) displays the second element.

Note : Python does not support character data type. A string of size 1 can be treated ascharacters.

CREATING AND INITIALIZING STRINGS

A literal/constant value to a string can be assigned using a single quotes, double quotesor triple quotes.

- Enclosing the string in single quotes:-

Example

```
>>>print ("A friend in need is a friend indeed")
A friend in need is a friend indeed
```

Example

```
>>>print("This book belongs to Raghav\'s sister")
This book belongs to Raghav's sister
```

As shown in example 2, to include the single quote within the string it should bepreceded by a backslash.

- **Enclosing the string in double quotes :**

Example

```
>>>print("A room without books is like a body without a soul.")
A room without books is like a body without a soul.
```

- **Enclosing the string in triple quote :**

Example

```
>>>life="""\" Live as if you were to die tomorrow.
Learn as if you were to live forever.\"
—— Mahatma Gandhi """
>>> print life
" Live as if you were to die tomorrow.
Learn as if you were to live forever."
—— Mahatma Gandhi """
```

Triple quotes are used when the text is multiline.

In the above example, backslash (\) is used as an escape sequence. An escape sequences is nothing but a special character that has a specific function. As shown above, backslash (\) is used to escape the double quote.

Escape sequence	Meaning	Example
\n	New line	>>> print "Hot\nCold" Hot Cold
	Tab space	>>>print "Hot\tCold" Hot Cold

- **By invoking raw_input() method :**

Let's understand the working of raw input() function

Example

```
>>>raw_input()
Right to education
"Right to education"
```

As soon as the interpreter encounters raw_input method, it waits for the user to key in the input from a standard input device (keyboard) and press Enter key. The input is converted to a string and displayed on the screen.

- **By invoking input() method :**

Example

```
>>>str=input("Enter the string")
Enter the string hello
NameError: name 'hello' is not defined
```

Python interpreter was not able associate appropriate data type with the entereddata. So a NameError is shown. The error can be rectified by enclosing the given input i.e. hello in quotes as shown below:-

>>>str=input("Enter the String") Enter the String "hello" >>> print str Hello	>>>str=input("Enter the String") Enter the String'hello' >>> print str Hello

STRINGS ARE IMMUTABLE

Strings are immutable means that the contents of the string cannot be changed after it iscreated. Let us understand the concept of immutability with help of an example.

Example

```
>>>str='honesty'
>>>str[2]='p'
TypeError: 'str' object does not support item assignment
```

Python does not allowthe programmer to change a character in a string. As shown in the above example, str has the value "honesty". An attempt to replace "n" in the string by "p" displays a TypeError.

TRAVERSING A STRING

Traversing a string means accessing all the elements of the string one after the other byusing the subscript. A string can be traversed using: for loop or while loop.

String traversal using for loop	String traversal using while loop
A="Welcome" >>>for i in A: print i W e l c o m e	A="Welcome" >>>i=0 >>>while i<len(A) print A[i] i=i+1 W e l c o m e
A is assigned a string literal "Welcome". On execution of the for loop, the characters in the string are printed till the end of the string is not reached.	A is assigned a string literal "Welcome" i is assigned value 0 The len() function calculates the length of the string. On entering the while loop, the interpreter checks the condition. If the condition is true, it enters the loop. The first character in the string is displayed. The value i is incremented by 1. The loop continues till value i is less than len-1. The loop finishes as soon as the value of I becomes equal to len-1.

STRINGS OPERATIONS

Operator	Description	Example
+ (Concatenation)	The + operator joins the text on both sides of the operator	>>> "Save"+"Earth" "Save Earth" To give a white space between the two words, insert a space before the closing single quote of the first literal.

* (Repetition)	The * operator repeats thestring on the left hand side times the value on right hand side.	>>>3*"Save Earth" "Save Earth Save Earth Save Earth"
in (Membership)	The operator displays 1 if the string contains the given character or the sequence of characters.	>>>A="Save Earth" >>> "S" in A True >>>"Save" in A True >>"SE" in A False
not in	The operator displays 1 if the string does not contain the given character or the sequence of characters. (working of this operator is the reverse of **in** operator discussed above)	>>>"SE" not in "Save Earth" True >>>"Save" not in "Save Earth" False
range (*start*, *stop*[, *step*])	**This function is already Discussed.**	
Slice[n:m]	The Slice[n : m] operator extracts sub parts from the strings.	>>>A="Save Earth" >>> print A[1:3] av The print statement prints the substring starting from subscript 1 and ending at subscript 3 but not including subscript 3

STRING METHODS & BUILT IN FUNCTIONS

Syntax	**Description**	**Example**
len()	Returns the length of the string.	>>>A="Save Earth" >>> print len(A) >>>10
capitalize()	Returns the exact copy of the string with the first letter inupper case	>>>print str.capitalize() Welcome
find(*sub*[, *start*[, *end*]])	The function is used to search the first occurrence of the substring in the given string. It returns the index at which the substring starts. It returns -1 if the substring does occur in the string.	>>>str='mammals' >>>str.find('ma') 0 **On omitting the start parameters, the function starts the search from the beginning.** >>>str.find('ma',2) 3 >>>str.find('ma',2,4) -1 **Displays -1 because the substring could not be found between the index 2 and 4-1** >>>str.find('ma',2,5) 3

isalnum()	Returns True if the string contains only letters and digit. It returns False ,If the string contains any special characterlike _ , @,#,* etc.	>>>str="Save Earth' >>>str.isalnum() False The function returns False as space is an alphanumeric character. >>>"Save1Earth'.isalnum() True

isalpha()
 	
isdigit()
	Returns True if the string contains only letters. Otherwise return False.
	
	Returns True if the stringcontains only numbers. Otherwise it returns False.
		>>> 'Click123'.isalpha()
		False
		>>> 'python'.isalpha()
		True
		>>>print str.isdigit()
		False

lower()	Returns the exact copy of the string with all the letters in lowercase.	>>>print str.lower() "save earth"
islower()	Returns True if the string is in lowercase.	>>>print str.islower() True
isupper()	Returns True if the string is in uppercase.	>>>print str.isupper() False
upper()	Returns the exact copy of the string with all letters in uppercase.	>>>print str.upper() WELCOME

lstrip()	Returns the string after removing the space(s) on the left of the string.	>>> print str
		Save Earth
		>>>str.lstrip()
		'Save Earth'
		>>>str='Teach India Movement'
		>>> print str.lstrip("T")
		each India Movement
		>>> print str.lstrip("Te")
		ach India Movement
		>>> print str.lstrip("Pt")
		Teach India Movement
		If a string is passed as argument to the lstrip() function, it removes those characters from the left of the string.

rstrip()	Returns the string after removing the space(s) on theright of the string.	>>>str="Teach India Movement" >>> print str.rstrip() Teach India Movement

isspace()	Returns True if the string contains only white spaces and False even if it contains one character.	>>> str=' '
		>>> print str.isspace()
		True
		>>> str='p'
		>>> print str.isspace()
		False

istitle()	Returns True if the string is title cased. Otherwise returns False	>>> str='The Green Revolution'
		>>> str.istitle()
		True
		>>> str='The green revolution'
		>>> str.istitle()
		False

replace(old,new)	The function replaces all the occurrences of the old string with the new string	>>>str="hello"
		>>> print str.replace('l','%')
		He%%o
		>>> print str.replace('l','%%')
		he%%%%o

join ()	Returns a string in which the string elements have been joined by a separator.	>>> str1=('jan', 'feb' ,'mar')
		>>>str="&"
		>>> str.join(str1)
		'jan&feb&mar'

swapcase()	Returns the string with case changes	>>> str='UPPER'
		>>> print str.swapcase()
		upper
		>>> str='lower'
		>>> print str.swapcase()
		LOWER

partition(sep)	The function partitions the strings at the first occurrence of separator, and returns the strings partition in three parts i.e. before the separator, the separator itself, and the part after the separator If the separator is not found, returns the string itself, followed by two empty strings	>>> str='The Green Revolution'
		>>> str.partition('Rev')
		('The Green ', 'Rev', 'olution')
		>>> str.partition('pe')
		('The Green Revolution', '', '')
		>>> str.partition('e')
		('Th', 'e', ' Green Revolution')

split([sep[, maxsplit]])	The function splits the string into substrings using the separator. The second argument is optional and its default value is zero. If an integer value N is given for the second argument, the string is split in N+1 strings.	>>>str='The$earth$is$what$we$all $have$in$common.'
		>>> str.split($,3)
		SyntaxError: invalid syntax
		>>> str.split('$',3)
		['The', 'earth', 'is', 'whatweall$have$in$common.']
		>>> str.split('$')
		['The', 'earth', 'is', 'what', 'we', 'all', 'have', 'in', 'common.']
		>>> str.split('e')
		['Th', ' Gr', '', 'n R', 'volution']
		>>> str.split('e',2)
		['Th', ' Gr', 'en Revolution']

Note: In the table given above, len( ) is a built in function and so we don't needimport the string module. For all other functions import string statement is required for their successful execution.

- Some interesting strings constants defined in string module:
  - (I) string.ascii_uppercase:- The command displays a string containing uppercase characters.
  - (II) string.ascii_lowercase:- The command displays a string containing all lowercase characters.
  - (III) string.ascii_letters:- The command displays a string containing both uppercase and lowercase characters.
  - (IV) string.digits:- The command displays a string containing digits.
  - (V) string.hexdigits:- The command displays a string containing hexadecimal characters.
  - (VI) string.octdigits:- The command displays a string containing octal characters.
  - (VII) string.punctuations:- The command displays a string containing all the punctuation characters.
  - (VIII) string.whitespace:- The command displays a string containing all ASCII characters that are considered whitespace. This includes the characters space, tab, linefeed, return, formfeed, and vertical tab.
  - (IX) string.printable:- The command displays a string containing all characters which are considered printable like letters, digits, punctuations and whitespaces.

REGULAR EXPRESSIONS AND PATTERN MATCHING

A regular expression is a sequence of letters and some special characters (also called meta characters). These special characters have symbolic meaning. The sequence formed by using meta characters and letters can be used to represent a group of patterns.

Let's start by understanding some meta characters.

For example

str= "Ram$"

The pattern "Ram$" is known as a regular expression. The expression has the meta character "$". Meta character "$" is used to match the given regular expression at the end of the string. So the regular expression would match the string "SitaRam" or "HeyRam" but will not match the string "Raman".

Consider the following codes:

```
def find():
import re
string1='SitaRam'
if
re.search('Ram$',string1):
print "String Found"
else :
print" No Match"
Output:
String Found
```

```
def find():
import re
string1='SitaRam'
if re.search('Sita$',string1):
print "String Found"
else :
print" No Match"
Output
No Match
```

As shown in the above examples, Regular expressions can be used in python for matching a particular pattern by importing the re module.

Note : re module includes functions for working on regular expression.

Now let's learn how the meta characters are used to form regular expressions.

S.No.	Meta character	Usage	Example
1	[ ]	Used to match a set of characters.	[ram] The regular expression would match any of the characters r, a, or m. [a-z] The regular expression would match only lowercase characters.
2	^	Used to complementing a set of characters	[^ram] The regular expressionwould match any other characters than r, a or m.
3	$	Used to match the end of string only	Ram$ The regular expression would match Ram in SitaRam but will not match Ram in Raman
4	*	Used to specify that the previous character can be matched zero or more times.	wate*r The regular expression would match strings like watr, wateer, wateeer and so on.
5	+	Used to specify that the previous character can be matched one or more times.	wate+r The regular expression would match strings like water, wateer, wateeer andso on.
6	?	Used to specify that the previous character can be matched either once or zero times	wate?r The regular expression would only match strings like watr or water
7	{}	The curly brackets accept two integer value s. The first value specifies the minimum no of occurrences and second value specifies the maximum of occurrences	wate{1,4}r The regular expression would match only strings water, wateer, wateeer or wateeeer

- **Few functions from re module :**

(I) re.compile():- The re.compile( ) function will compile the pattern into pattern objects. After the compilation the pattern objects will be able to access methods for various operations like searching and subsitutions.

(II) re.match():- The match function is used to determine if the regular expression (RE) matches at the beginning of the string.

(III) re.group():- The group function is used to return the string matched the RE.

(IV) re.start():- The start function returns the starting position of the match.

(V) re.end():- The end function returns the end position of the match.

(VI) re.span():- The span function returns the tuple containing the (start, end) positions of the match.

(VII) re.search():- The search function traverses through the string and determines the position where the RE matches the string.

(VIII) Re.findall():- The function determines all substrings where the RE matches, and returns them as a list.

(IX) re.finditer():- The function determines all substrings where the RE matches, and returns them as aniterator.

□□□

9 LIST
(NCERT CLASS 11)

INTRODUCTION

Like a String, list also is sequence data type. It is an ordered set of values enclosed insquare brackets []and are separated by comma. Values in the list can be modified, i.e. it is mutable. As it is set of values, we can use index in square brackets [] to identify a value belonging to it. The values that make up a list are called its elements, and they can be of any type.Like string indices, list indices also start from 0.

We can also say that list data type is a container that holds a number of elements in agiven order. For accessing an element of the list, indexing is used.

Its syntax is:

Variable name [index] (variable name is name of the list).

It will provide the value at "index+1" in the list. Index here, has to be an integer value which can be positive or negative. Positive value of index means counting forward from beginning of the list and negative value means counting backward from end of the list.

Remember the result of indexing a list is the value of type accessed from the list.

Index value	Element of the list
0, -size	1st
1, -size +1	2nd
2, -size +2	3rd
.	
.	
.	
size -2, -2	2nd last
size -1, -1	last

Note that in the above example size is the total number of elements in the list.

Let's look at some example of simple list:

(I) >>>L1 = [1, 2, 3, 4] # list of 4 integer elements.

(II) >>>L2 = ["Delhi", "Chennai", "Mumbai"] #list of 3 string elements.

(III) >>>L3 = [ ] # empty list i.e. list with no element

(IV) >>>L4 = ["abc", 10, 20] # list with different types of elements

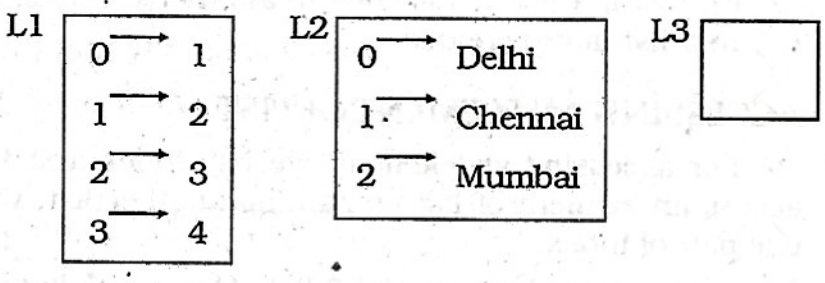
(V) >>>L5 = [1, 2, [6, 7, 8], 3] # A list containing another list known as nested list

To change the value of element of list, we access the element & assign the new value.

```
>>>print L1 # let's get the values of list before change
>>> L1 [2] = 5
>>> print L1 # modified list
[1, 2, 5, 4]
```

Here, 3rd element of the list (accessed using index value 2) is given a new value, soinstead of 3 it will be 5.

State diagram for the list looks like:



Note: List index works the same way as String index, which is:

(I) An integer value/expression can be used as index.

(II) An Index Error appears, if you try and access element that does not exist in the list.

(III) An index can have a negative value, in that case counting happens from the end of the list.

CREATING A LIST

List can be created in many ways:

(I) By enclosing elements in [ ], as we have done in above examples.

(II) Using other Lists.

Example

```
L5=L1 [:]
```

Here L5 is created as a copy of L1.

```
00 >>>print L5
  L6 = L1 [0:2]
  >>>print L6
```

will create L6 having first two elements of L1.

(III) List comprehension

Example

```
>>>n = 5
>>>l = range(n)
>>>print l
[0, 1, 2, 3, 4]
```

Example

```
>>> S= [x**2 for x in range (10)]
>>> print S
    [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

In mathematical terms, S can be defined as S = {x2 for: x in (0.....9)}. So, we can say that list comprehension is short-hand for creating list.

(IV) Using built-in object

L = list ( ) will create an empty list

Example

```
>>>l = list ( )
>>>print l
[ ] # empty list
Or
  L = list (sequence)
```

- A single new list is created every time, you execute [ ]. We have created many different lists each using [ ]. But if a list is assigned to another variable, a new list is not created.

ACCESSING AN ELEMENT OF LIST

For accessing an element, we use index and to access an element of list containing another list, we use pair of index. .

Lets access elements of L5 list. Also a sub-list of list can be accessed using list slice.

List Slices : Slice operator works on list also. We know that a slice of a list is its sub-list. For creating a list slice, we use

[n:m] operator.

```
>>>print L5 [0]
1
>>>print L5 [2]
      [6, 7, 8]
```

as the 3rd element of this list is a list. To access a value from this sub-list, we will use

```
>>>print L5 [2] [0]
6
>>>print L5 [2] [2]
8
```

This will return the part of the list from nth element to mth element, including the firstelement but excluding the last element. So the resultant list will have m-n elements in it.

```
>>> L1 [1:2]
```

will give

```
  [2]
```

Slices are treated as boundaries, and the result will contain all the elements betweenboundaries.

Its Syntax is:

seq = L [start: stop: step]

Where start, stop & step- all three are optional. If you omit first index, slice starts from '0' and omitting of stop will take it to end. Default value of step is 1.

Example

For list L2 containing ["Delhi", "Chennai", "Mumbai"]

```
>>>L2 [0:2]
["Delhi", "Chennai"]
```

Note: Since lists are mutable, it is often recommended to make a copy of it before performing operation that change a list.

TRAVERSING A LIST

Let us visit each element (traverse the list) of the list to display them on screen. This canbe done in many ways:

(i)
```
i = 0
while i < 4:
print L1 [i],
i + = 1
```
will produce following output

1 2 5 4

(ii)
```
for i in L1:
print i,
```
will also produce the same output

(iii)
```
i=0
while i < len [L1]:
print L1 [i],
i + = 1
```
will also produce the same output.

Here len( ) function is used to get the length of list L1. As length of L1 is 4, i will takevalue from 0 to 3.

(iv)
```
for i in range ( len (L1)):
      print L1 [i],
```

Using 2nd way for transversal will only allow us to print the list, but other ways can also be used to write or update the element of the list.

In 4th way, range ( ) function is used to generate, indices from 0 to len -1; with each iteration i gets the index of next element and values of list are printed.

Note: for loop in empty list is never executed.

Example

```
for i in [ ]:
print i
```

Accessing list with negative index

```
i = 1
while i < len (L1):
print L1 [-i],
i += 1
```

In this case, Python will add the length of the list to index and then return the index value and accesses the desired element. In this loop execution for a positive

value of "i" L1 [-i] will result into L1 [len (L1)-i] for i=1, L1 [4-1] will be printed. So resultant of the loop will be 4 5 2.

APPENDING IN THE LIST

Appending a list is adding more element(s) at the end of the list. To add new elementsat the end of the list, Python provides a method append ( ).

Its Syntax is:

List. append (item)

L1. append (70)

This will add 70 to the list at the end, so now 70 will be the 5th element of the list, as it already have 4 elements.

>>> print L1

will produce following on screen

[1, 2, 5, 4, 70]

UPDATING ARRAY ELEMENTS

Updating an element of list is, accomplished by accessing the element & modifying its value in place. It is possible to modify a single element or a part of list. For first type, we use index to access single element and for second type, list slice is used. We have seen examples of updations of an element of list. Lets update a slice.

As lists are sequences, they support many operations of strings. For example, operator +& * results in concatenation & repetition of lists. Use of these operators generate a new list.

It is important to know that '+' operator in lists expects the same type of sequence on both the sides otherwise you get a type error.

If you want to concatenate a list and string, either you have to convert the list to string or string to list.

DELETING ELEMENTS

It is possible to delete/remove element(s) from the list. There are many ways of doing so:

(I) If index is known, we can use pop ( ) or del

(II) If the element is known, not the index, remove ( ) can be used.

(III) To remove more than one element, del ( ) with list slice can be used.

(IV) Using assignment operator.

Let us study all the above methods in details:

(I) Pop ( ): It removes the element from the specified index, and also return the element which wasremoved.

Its syntax is:

List.pop ([index])

Example

>>> L1 = [1, 2, 5, 4, 70, 10, 90, 80, 50]

>>> a= L1.pop (1) # here the element deleted will be returned to 'a'

>>> print L1

[1, 5, 4, 70, 10, 90, 80, 50]

>>> print a

2

If no index value is provided in pop ( ), then last element is deleted.

>>>L1.pop ( )

50

del removes the specified element from the list, but does not return the deleted

value.

>>> del L1 [4]

>>> print L1

[1, 5, 4, 70, 90, 80]

(II) remove ( ) : In case, we know the element to be deleted not the index, of the element, then remove ( ) can be used.

>>> L1. remove (90)

will remove the value 90 from the list

>>> print L1

[1; 5, 4, 70, 80]

(III) del () with slicing : Consider the following example:

Examples

>>> del L1 [2:4]

>>>print L1

[1, 5, 80]

will remove 2nd and 3rd element from the list. As we know that slice selects all the elements up to 2nd index but not the 2nd index element. So 4th element will remain in the list.

>>> L5 [1:2] = [ ]

Will delete the slice

>>>print L5

[1, [6, 7, 8], 3]

Note:

(I) All the methods, modify the list, after deletions.

(II) If an out of range index is provided with del ( ) and pop ( ), the code will result in to run-time error.

(III) del can be used with negative index value also..

Other functions & methods

(I) insert ( ) : This method allows us to insert an element, at the given position specified by its index, and the remaining elements are shifted to accommodate the new element. Insert () requires two arguments-index value and item value.

Its syntax is

list. insert (index, item)

Index specifies the position (starting from 0) where the element is to be inserted. Item is the element to be inserted in the list. Length of list changes after insert operation.

Note: If the index specified is greater then len (list) the object is inserted in the lastand if index is less than zero, the object is inserted at the beginning.

(II) reverse ( ) : This method can be used to reverse the elements of the list in place.

Its syntax is:

list.reverse ( )

Method does not return anything as the reversed list is stored in the same variable.

(III) sort ( ) : For arranging elements in an order Python provides a method sort ( ) and a function sorted ( ). sort ( ) modifies the list in place and sorted ( ) returns a new sorted list.

Its Syntax are:

sort ([cmp [, key [, reverse]]])

sorted (list [, cmp [, key [, reverse]]])

Parameters mentioned in [ ] are optional in both the cases. These parameters allow us to customize the function/method.

cmp, argument allow us to override the default way of comparing elements of list. By default, sort determines the order of elements by comparing the elements in the list against each other. To overside this, we can use a user defined function which should take two values and return -1 for 'less than', 0 for 'equal to' and 1 for 'greater than' . 'Key' argument is preferred over 'cmp' as it produces list faster.

- The parameter 'key' is for specifying a function that transforms each element of listbefore comparison. We can use predefined functions or a user defined function here. If its user defined then, the function should take a single argument and return a key which can be used for sorting purpose.
- Reverse parameter can have a boolean value which is used to specify the order of arranging the elements of list. Value 'True' for reverse will arrange the elements of list in descending order and value 'False' for reverse will arrange the elements in ascending order. Default value of this parameter is False.
- sorted ( ) function also behaves in similar manner except for it produce a new sorted list, so original is not changed. This function can also be used to sort any iterable collection. As sort ( ) method does not create a new list so it can be little faster.
- **List as arguments :** When a list is passed to the function, the function gets a reference to the list. So if the function makes any changes in the list, they will be reflected back in the list.

Note: Operations which create a newlist will not affect the original (argument) list.

MATRIX IMPLEMENTATION USING LIST

We can implement matrix operation using list. Matrix operation can be implemented using nested list. When a list appears as an element of another list, it iscalled a nested list.

Its syntax is:

a=[[random.random() for row in range(number of row)]for col in range(number of column)]

Here random function is used. So we need to import random file.

LIST AS ARGUMENT TO A FUNCTION

Whenever a list is passed as an argument to a function, we have to consider two scenarios:

(A) Elements of the original list may be changed, i.e. changes made to the list in the function are reflected back in the calling function.

For example in the following program list list1 of numbers is passed as an argument to function increment(). This function increases every element of the list by 5.

Program to increment the elements of a list. The list is passed as an argument to a function:-

```
#Function to increment the elements of the list passed as argument
def increment(list2):
    for i in range(0,len(list2)):
        #5 is added to individual elements in the list
        list2[i] += 5
print('Reference of list Inside Function',id(list2))
#end of function
list1 = [10,20,30,40,50] #Create a list
print("Reference of list in Main",id(list1))
print("The list before the function call")
print(list1)
```

increment(list1) #list1 is passed as parameter to function

print("The list after the function call")

print(list1)

Output:

Reference of list in Main 70615968

The list before the function call

[10, 20, 30, 40, 50]

Reference of list Inside Function 70615968 #The id remains same

The list after the function call

[15, 25, 35, 45, 55]

Observe that, when we pass a list as an argument, weactually pass a reference to the list. Hence any change made to list2 inside the function is reflected in the actual list list1.

(B) If the list is assigned a new value inside the function then a new list object is created and it becomes the local copy of the function. Any changes made inside the local copy of the function are not reflected back to the calling function.

FUNCTIONS WITH LISTS

We can pass list value to function. Whatever modification we are doing with infunction will affect list.

Example

Write a program to pass any list and to arrange all numbers in descending order.

```
def arrange (l,n):
for i in range(n-1):
for j in range(n-i-1):
if l[j]>l[j+1]:
temp=l[j]
l[j]=l[j+1]
l[j+1]=temp
```

Output

```
>>>
>>> l=[7,5,8,2,9,10,3]
>>> arrange (l)
>>> print l
[10, 9, 8, 7, 5, 3, 2]
>>>
```

Function pass nested list also:

Example

Write a program to input nXm matrix and find sum of all numbers using function.

Function:

```
def summat(a,m,n):
s=0
for i in range(m):
for j in range(n):
s+=a[i][j]
return s
```

Note: This function is stored in mataddition.py.

- **Function call :**

```
import random
import mataddition
m=input("Enter total number of rows in the first matrix")
n=input("Enter total number of columns in the first matrix")
a=[[random.random()for row in range(m)]for col in range(n)]
for i in range(m):
for j in range(n):
a[i][j]=input()
s=mataddition.summat(a,m,n)
print s
```

Output

```
>>>
Enter total number of rows in the first matrix2
Enter total number of columns in the first matrix2
1
2
3
4
10
>>>
```

□□□

10 TUPLES AND DICTIONARIES (NCERT CLASS 11)

INTRODUCTION OF TUPLES

A tuple is a ordered sequence of values, which can be of any type such as integer, float, string, listand they are indexed by integer. Tuples are just like list, but we can't change values of tuples in place. Thus tuples are immutable. Like list and string, elements of a tuple can be accessed using index values, starting from 0.

Elements of a tuple are enclosed in parenthesis (round brackets) and are separated by commas.

For example:

```
>>> T=10, 20, 30, 40
>>> print T
(10, 20, 30, 40)
```

But in the result, same tuple is printed using parentheses. To create a tuple with single element, we have to use final comma. A value with in the parenthesis is not tuple.

Example

```
>>> T=(10)
>>> type(T)
<type 'int'>
```

Example

```
>>> t=10,
>>> print t
(10,)
```

Example

```
>>> T=(10,20)
>>> type(T)
<type 'tuple'>
```

Example

Tuple with string values

```
>>> T=('sun','mon','tue')
>>> print T
('sun', 'mon', 'tue')
```

Example

Tuples with single character

```
>>> T=('P','Y','T','H','O','N')
>>> print T
('P', 'Y', 'T', 'H', 'O', 'N')
```

- **Tuple Creation :**

If we need to create a tuple with a single element, we need to include a final comma.

Example

```
>>> t=10,
>>> print t
(10,)
```

Another way of creating tuple is built-in function tuple ().

Syntax:

T = tuple()

Example

```
>>> T=tuple()
>>> print T
()
```

- **Add new element to Tuple :**

We can add new element to tuple using + operator.

We can also add new element to tuple by using list. For that we have to convert thetuple into a list first and then use append() function to add new elements to the list. After completing the addition, convert the list into tuple. Following example illustrates how to add new elements to tuple using a list.

- **Tuple Assignment:**

Assignment of tuple is a useful feature in Python.It allows a tuple of variables on the left side of the assignment operator to be assigned respective values from a tuple on the right side. The number of variables on the left should be same as the number of elements in the tuple.. For example;

```
>>> A=10
>>> B=20
>>> print A,B
10 20
>>> T=A
>>> A=B
>>> B=T
>>> print A,B
20 10
```

But in python, tuple assignment is more elegant:

Example

```
>>> T1=(10,20,30)
>>> T2=(100,200,300,400)
>>> print T1
(10, 20, 30)
>>> print T2
(100, 200, 300, 400)
>>> T1,T2=T2,T1 # swap T1 and T2
>>> print T1
(100, 200, 300, 400)
>>> print T2
(10, 20, 30)
```

The left side is a tuple of variables, while the right side is a tuple of expressions. Each value is assigned

to its respective variable. All the expressions on the right side are evaluated before any of the assignments.

The number of variables on the left and the number of values on the right have to be the same:

If two tuples are in the left side and three tuples are in right side. Then, weget errors. Thus, it is required to have same number of tuples in both sides to get the correct result.

- **Tuple Slices**

Slice operator works on Tuple also. This is used to display more than one selected value on the output screen. Slices are treated as boundaries and the result will contain all the elements between boundaries.

Syntax is:

Seq = T [start: stop: step]

Where start, stop & step all three are optional. If we omit first index, slice starts from '0'. On omitting stop, slice will take it to end. Default value of step is 1.

Example

```
>>> T=(10,20,30,40,50)
>>> T1=T[2:4]
>>> print T1
(30, 40)
```

In the above example, starting position is 2 and ending position is 3(4-1), so the selected elements are 30 & 40.

```
>>> T[:]
    (10, 20, 30, 40, 50)
```

Will produce a copy of the whole tuple.

```
>>> T[::2]
    (10, 30, 50)
```

Will produce a Tuple with every alternate element.

```
>>> T[:3]
    (10, 20, 30)
```

Will produce 0 to 2(3-1)

```
>>> T[2:]
    (30, 40, 50)
```

Will produce from 2 to end.

- **Nested Tuples :**

A tuple inside another tuple is called a nested tuple. For example:-If in any program, roll number, name and marks(in percentage) of students are saved in a tuple. To store details of many such students we can create a nested tuple.

TUPLE METHODS AND BUILT-IN FUNCTIONS

Python provides many functions to work on tuples. Following table list some of the commonly used tuple methods and built-in functions:-

Method	Description	Example

tuple()	Creates an empty tuple if no argument is passed	>>> tuple1 = tuple()
		>>> tuple1
		( )

	Creates a tuple if a sequence is passed as argument	>>> tuple1 = tuple('aeiou')#string
		>>> tuple1
		('a', 'e', 'i', 'o', 'u')

		>>> tuple2 = tuple([1,2,3]) #list
		>>> tuple2
		(1, 2, 3)

		>>> tuple3 = tuple(range(5))
		>>> tuple3
		(0, 1, 2, 3, 4)

count()	Returns the number of times the given element appears in the tuple	>>> tuple1 = (10,20,30,10,40,10,50)
		>>> tuple1.count(10)
		3>
		>> tuple1.count(90)
		0

index()	Returns the index of the first occurrence of the element in the given tuple	>>> tuple1 = (10,20,30,40,50)
		>>> tuple1.index(30)
		2>
		>> tuple1.index(90)
		ValueError: tuple.index(x): x not in tuple

sorted()	Takes elements in the tuple and returns a new sorted list. It should be noted that, sorted() does not make any change to the original tuple	>>> tuple1 = ("Rama","Heena","Raj", "Mohsin","Aditya")
		>>> sorted(tuple1)
		['Aditya', 'Heena', 'Mohsin', 'Raj','Rama']

min()	Returns minimum or smallest element of the tuple	>>> tuple1 = (19,12,56,18,9,87,34)
		>>> min(tuple1)
		9

max()	Returns maximum or largest elementof the tuple	>>> max(tuple1) 87
sum()	Returns sum of the elements of the tuple	>>> sum(tuple1) 235

TUPLE OPERATIONS

(I) **Concatenation:** Python allows us to join tuples using concatenationoperator depicted by symbol +. We can also create a new tuple which contains the result of this concatenation operation.

(II) **Repetition:** Repetition operation is depicted by the symbol *. It is used to repeat elements of a tuple. We can repeat the tuple elements. The repetition operator requires the first operand to be a tuple and the second operand to be an integer only.

(III) **Membership:** The in operator checks if the element is present in the tuple and returns True, else it returns False.

INTRODUCTION OF DICTIONARIES

The data type dictionary fall under mapping. It is amapping between a set of keys and a set of values. The key-value pair is called an item. A key is separated from its value by a colon(:) and consecutive items are separated by commas. Items in dictionaries are unordered, so we may not get back the data in the same order in which we had entered the data initially in the dictionary.

- **Creating a Dictionary :** To create a dictionary, the items entered are separated bycommas and enclosed in curly braces. Each item is a keyvalue pair, separated through colon (:). The keys in the dictionary must be unique and should be of any immutable data type, i.e., number, string or tuple. The values can be repeated and can be of any data type.
- **Accessing Items in a Dictionary :** The items of a sequence (string, list and tuple) are accessed using a technique called indexing. The items of a dictionary are accessed via the keys rather than via their relative positions or indices. Each key serves as the index and maps to a value.

DICTIONARIES ARE MUTABLE

Dictionaries are mutable which implies that the contents of the dictionary can be changed after it has been created.

- **Adding a new item :** We can add a new item to the dictionary as shown inthe following example:

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
    'Sangeeta':85}
>>> dict1['Meena'] = 78
>>> dict1
  {'Mohan': 95, 'Ram': 89, 'Suhel': 92,
  'Sangeeta': 85, 'Meena': 78}
```

- **Modifyingan Existing Item:** The existing dictionary can be modified by justoverwriting the key-value pair. Example to modify a given item in the dictionary:

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
        'Sangeeta':85}
        #Marks of Suhel changed to 93.5
>>> dict1['Suhel'] = 93.5
>>> dict1
      {'Mohan': 95, 'Ram': 89, 'Suhel': 93.5,
      'Sangeeta': 85}
```

DICTIONARY OPERATIONS

(I) **Membership:** The membership operator in checks if the key is present in the dictionary and returns True, else it returns False.

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
    'Sangeeta':85}
>>> 'Suhel' in dict1
    True
```

The not in operator returns True if the key is not present inthe dictionary, else it returns False.

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
    'Sangeeta':85}
>>> 'Suhel' not in dict1
    False
```

TRAVERSING A DICTIONARY

We can access each item of the dictionary or traverse adictionary using for loop.

```
>>> dict1 = {'Mohan':95,'Ram':89,'Suhel':92,
      'Sangeeta':85}
```

DICTIONARY METHODS AND BUILT-IN FUNCTIONS

Python provides many functions to work on dictionaries. Following table lists some of the commonly used dictionary methods.

Method	Description	Example

len()	Returns the length or number of key: value pairs of the dictionary passed as the argument	>>> dict1 ={'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> len(dict1)
		4

dict()	Creates a dictionary from a sequence of key-value pairs	pair1 = [('Mohan',95),('Ram',89), ('Suhel',92),('Sangeeta',85)]
		>>> pair1
		[('Mohan', 95), ('Ram', 89), ('Suhel', 92), ('Sangeeta', 85)]
		>>> dict1 = dict(pair1)
		>>> dict1
		{'Mohan': 95, 'Ram': 89, 'Suhel': 92, 'Sangeeta': 85}

keys()	Returns a list of keys in the dictionary	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.keys()
		dict_keys(['Mohan', 'Ram', 'Suhel', 'Sangeeta'])

values()	Returns a list of values in the dictionary	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.values()
		dict_values([95, 89, 92, 85])

items()	Returns a list of tuples(key –value) pair	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.items()
		dict_items([('Mohan', 95), ('Ram', 89), ('Suhel', 92), ('Sangeeta', 85)])

.get()	Returns the value corresponding to the key passed as the argument
	
	If the key is not present in the dictionary it will return None
		>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.get('Sangeeta')
		85
		>>> dict1.get('Sohan')
		>>>

update()	appends the key-value pair of the dictionary passed as the argument to the key-value pair of the given dictionary	>>> dict1 = {'Mohan':95, 'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict2 = {'Sohan':79,'Geeta':89}
		>>> dict1.update(dict2)
		>>> dict1
		{'Mohan': 95, 'Ram': 89, 'Suhel': 92, 'Sangeeta': 85, 'Sohan': 79, 'Geeta': 89}
		>>> dict2
		{'Sohan': 79, 'Geeta': 89}

del()	Deletes the item with the given key
	To delete the dictionary from the memory we write:
	del Dict_name
		>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> del dict1['Ram']
		>>> dict1
		{'Mohan':95,'Suhel':92, 'Sangeeta': 85}
		
		>>> del dict1 ['Mohan']
		>>> dict1
		{'Suhel': 92, 'Sangeeta': 85}
		>>> del dict1
		>>> dict1
		NameError: name 'dict1' is not defined

clear()	Deletes or clear all the items of the dictionary	>>> dict1 = {'Mohan':95,'Ram':89, 'Suhel':92, 'Sangeeta':85}
		>>> dict1.clear()
		>>> dict1
		{}

MANIPULATING DICTIONARIES

we have learnt how to create a dictionary and apply various methods to manipulate it. The following program show the application of those manipulation methods on dictionaries.

Create a dictionary 'ODD' of odd numbers between 1 and10, where the key is the decimal number and the value is the corresponding number in words. Perform the following operations on this dictionary:

(a) Display the keys
(b) Display the values
(c) Display the items
(d) Find the length of the dictionary
(e) Check if 7 is present or not
(f) Check if 2 is present or not
(g) Retrieve the value corresponding to the key 9
(h) Delete the item from the dictionary corresponding to the key 9

```
>>> ODD
= {1:'One',3:'Three',5:'Five',7:'Seven',9:'Nine'}
>>> ODD
{1: 'One', 3: 'Three', 5: 'Five', 7: 'Seven', 9: 'Nine'}
```

(a) Display the keys

```
>>> ODD.keys()
dict_keys([1, 3, 5, 7, 9])
```

(b) Display the values

```
>>> ODD.values()
dict_values(['One', 'Three', 'Five', 'Seven', 'Nine'])
```

(c) Display the items

```
>>> ODD.items()
dict_items([(1, 'One'), (3, 'Three'), (5, 'Five'), (7, 'Seven'),
(9, 'Nine')])
```

(d) Find the length of the dictionary

```
>>> len(ODD)
5
```

(e) Check if 7 is present or not

```
>>> 7 in ODD
True
```

(f) Check if 2 is present or not

```
>>> 2 in ODD
False
```

(g) Retrieve the value corresponding to the key 9

```
>>> ODD.get(9)
'Nine'
```

(h) Delete the item from the dictionary corresponding to the key 9

```
>>> del ODD[9]
>>> ODD
{1: 'One', 3: 'Three', 5: 'Five', 7: 'Seven'}
```

11 SOCIETAL IMPACT
(NCERT CLASS 11)

INTRODUCTION

In recent years, the world around us has seen a lot of changes due to use of 'Digital Technologies'. These changes have made a dramatic impact on our lives, making things more convenient, faster, and easier to handle. The introduction of personal computers (PCs) and Internet followed by smartphones has brought these technologies to the common man.

While we reap the benefits of digital technologies, these technologies can also be misused. Let's look at the impact of these technologies on our society and the best practices that can ensure a productive and safe digital environment for us.

DIGITAL FOOTPRINTS

A digital footprint is data that is left behind when users have been online. There are two types of digital footprints which are active and passive.

(I) An active digital footprint is where the user has deliberately shared information about themselves either by using social media sites or by using websites.

Examples of active digital footprints

- Posting on Instagram,Facebook, Instagram, Twitter, and other social media platforms.
- Filling out online forms, i.e. when signing up to receive emails or texts
- Agreeing to install cookies on our devices when prompted by the browser

(II) A passive digital footprint is made when information is collected from the user without the person knowing this is happening.

Examples of passive digital footprints

- Websites that install cookies in our device without disclosing it to us.
- Apps and websites that use geolocation to pinpoint our location .
- Social media news channels and advertisers that use our likes, shares, and comments to profile us and to serve up advertisements based on our interests.

How digital footprint is being used for marketing purposes?

- Digital footprints are also known as cyber shadow, electronic footprint, or digital shadow are generally collected with the help of tracking cookies. These cookies are created while using popular sites. Whatever we search is stored in these along with our dates, GPS relevant data. These are shared by actual site we are visiting to the popular sites.
- Popular sites in tern analyze these data and revert back in the form of advertise later on. For e.g. we search for a flight from x location to y location for a particular date. Next day if we open search engine ,ads automatically popups even if we have booked out tickets.

For the following four reasons we should care about managing our digital footprint :

(I) To protect our reputation.
(II) To make safe personal information.
(III) To prevent financial loss.
(IV) To preserve our freedom

Risk due to digital footprint :

- Privacy concern
- Scam
- Identity theft
- Fake websites

How to manage digital footprints ?

(I) Enter name into several search engines.
(II) Double-check privacy settings, but don't trust them.
(III) Create strong, memorable passwords.
(IV) Keep all our software up to date.
(V) Review our mobile use. Delete useless files(temp.).
(VI) Build reputation through behavior.

DIGITAL SOCIETY AND NETIZEN

Digital society reflects the growing trend of using digital technologies in all spheres of human activities. But while online, all of us need to be aware of how to conduct ourselves, how best to relate with others and what ethics, morals and values to maintain.Being a good netizen means practicing safe, ethical and legal use of digital technology. A responsible netizen must abide by net etiquettes, communication etiquettes and social media etiquettes.

- **Net or communication etiquettes :** Netiquette is short for "Internet etiquette" or communication etiquettes over internet. It is Just like etiquette - a code of polite behavior in society, netiquette is a code of good behavior on the Internet. It includes several aspects of the Internet, social media, email, online chat, web forums, website comments, multiplayer gaming, and other types of online communication.

Do	Don't
❖ Keep Messages and Posts Brief. ❖ Use Discretion. ❖ Protect Personal Information. ❖ Obey Copyright Laws. ❖ Help Others. ❖ Respect other people's privacy. ❖ Verify facts before reposting. ❖ Check messages and respond promptly. ❖ Thank others who help you online.	❖ posting inflammatory /offensive comments shout. ❖ respond to Internet Trollers. ❖ Post private or embarrassing images/comments. ❖ Name-call or express offensive opinions. ❖ Exclude people or talk behind their backs. ❖ Stick to the topic. ❖ spam others by sending large amounts of unsolicited email.

DATA PROTECTION

Refers to the practices, safeguards, and binding rules put in place to protect our personal information and ensure that it remain in control. In short, we should be able to decide whether or not we want to share some information, who has access to it, for how long, for what reason, and be able to modify some of this information, and more.

Consequences of Unprotected Data/Data breaches:-

(I) Suffer from security breach/attack.
(II) Physical data loss.
(III) Hit with a virus .
(IV) Targeted by hackers .
(V) Suffer from DDoS(Distributed denial of service).
(VI) Lose of money.
(VII) Intellectual property at risk .
(VIII) Damage downtime.

How we can protect our personal data online:-

(I) Through Encrypt our Data.
(II) Keep Passwords Private.
(III) Don't Overshare on Social Networking Sites.
(IV) Use Security Software.
(V) Avoid Phishing Emails.
(VI) Be Wise About Wi-Fi .
(VII) Be Alert to Impersonators .
(VIII) Safely Dispose of Personal Information.

INTELLECTUAL PROPERTY (IP)

Is a property created by a person or group of persons using their own intellect for ultimate use in commerce and which is already not available in the public domain.

Examples of IP Property which are, an invention relating to a product or any process, a new design, a literary or artistic work and a trademark (a word, a symbol and / or a logo, etc.),

Intellectual Property Right (IPR) is the statutory right granted by the Government, to the owner(s) of the intellectual property or applicant(s) of an intellectual property (IP) to exclude others from exploiting the IP commercially for a given period of time, in lieu of the discloser of his/her IP in an IPR application.

Why should an IP be protected?

- IP is an assets and can be exploited by the owner for commercial gains any manner.
- IP owner may intend to stop others from manufacturing and selling products and services which are dully protected by him.
- IP owner can sell and/or license the IP for commercial gains.
- IP can be used to establish the goodwill and brand value in the market.
- IP can be mention in resumes of it's creator and thus show competence of it's creator.
- IPR certificate establishes legal and valid ownership about an intellectual property.

Kinds of IPRs

- Patent (to protect technologies - The Patent Act).
- Trade Mark (to protect words, signs, logos, labels – The Trade Mark Act).
- Design (to protect outer ornamental configuration –The Designs Act).
- Geographical Indications (GI) (to protect region specific product –The Geographical Indications of Goods Act).
- Copyright (to protect literary and artistic work – The Copyright Act).

IPRs are protected in accordance with the provisions of legislations of a country specific. In India, IPRs can be protected and monopolized as per the act. Some of them are :

1. The Patent Act, 1970
2. The Designs Act, 2000
3. The Trade Mark Act, 1999
4. The Geographical Indications of Goods Act, 1999,
5. The Copyright Act, 1957
6. Protection of Integrated Circuits Layout and Designs Act, 2000
7. Protection of Plant Varieties and Farmers Rights Act, 2001, and also Trade Secret

PLAGIARISM

It is "the act of presenting the words, ideas, images, sounds, or the creative expression of others as it is your creation or your own." The word plagiarism is derived from the Latin word plagiare, which means to kidnap or abduct.

Why is it important to understand Plagiarism?

- Plagiarism is stealing of intellectual property.
- Plagiarism is cheating.
- Plagiarism is an Academic offence.
- Plagiarism is Academic theft!

Two Types of Plagiarism:

Intentional Plagiarism	Unintentional Plagiarism
Copying other's work	Not knowing how to acknowledge or incorporate sources of information through proper paraphrasing, summarizing and quotation
Borrowing/buying assignments	Careless copying or cutting and pasting from electronic databases
Cut , paste from electronic resources	Quoting excessively
Downloading essays/text from the Internet and presenting as our own work	Failure to use our own "voice"

How to avoid plagiarism

1. Use your own ideas
2. Cite the sources-When someone else's ideas are used, always acknowledge the sources and tell your reader WHERE THE IDEAS ARE FROM.
3. Rewrite other's ideas in your own words
4. Take careful notes
5. Develop your writing skills

SOFTWARE LICENSE

A software license is a document that provides legally binding guidelines to the person who holds it for the use and distribution of software. It typically provide end users with the right to make one or more copies of the software without violating copyrights. It also defines the responsibilities of the parties entering into the license agreement and may impose restrictions on how the software can be used. Software licensing terms and conditions usually include fair use of the software, the limitations of liability, warranties and disclaimers and protections.

Benefits of Using Licensed Software:-

- Using Unlicensed Software Against the Law.
- The Right Software License Can Save our Money.
- We can Receive Around-The-Clock License Support.

Software copyright is used by software developers/ software companies/proprietary software companies to prevent the unauthorized copying of their softwares. Free and open source licenses also rely on copyright law to enforce their terms.

Reason for copyright our software

(I) Our work(software development) is an asset Protect our rights.

(II) It protects our software structures.

(III) It protects software code,sequencing and organization.

(IV) It enhances protection against license agreements.

- Difference between licensing and copyright:- Copyright is a type of intellectual property protection and licensing is a kind of risk control measure that can be applied to control copyright loss exposure, so the licensor (copyright owner) can grant permission that usually takes the form of a licensing agreement to use its copyrighted material. This agreement specifies the exact material to be used, purposes the work could be used for and the duration of the license.
- Free and Open Source software(FOSS) FOSS is a kind of software that all allows users to not only freely run the program for any purpose, but also provides users access to its source code. Moreover, it also allows us to modify as we wish, as well as freely distribute copies of the original version or their altered version.

Following criteria must be met for FOSS:

- Source code must be included.
- Anyone must be allowed to modify the source code.
- Modified versions can be redistributed.
- The license must not require the exclusion of other.
- It must be free.

Example of Free and Open source software:-

- As Operating system – linux,Ubuntu
- As dbms – mysql,mongodb,SQlite
- As Programming language – java,php,python
- As internet browser/webserver – chromium,firfox/ apache http server,apache tomcat.

Types of Software based on use:

(I) **Free Software :** Free Software are those which are freely accessible, freely accessible, freely used, changed,improved, copied and distributed. It provides all types of freedom. The term 'Free' means'Freedom' at very little or No cost. The Source Code is also available with Free Software.

(II) **Open Source Software :** Open Source Software can be freely used, changed, improved, copied and Re-distributed but it may have some cost for the

media and support for further development.Source Code is also available with OSS. It can be modified and redistributed with some guidelines. The License may restrict source-code from being distributed and modification to maintain the Author's integrity. A software which is FREE as well as OPEN, called Free & Open Source Software (FOSS) or Free Libre & Open Source Software (FLOSS).

CYBER CRIME

Any crime that involves a computer and a network is called a "Computer Crime" or "Cyber Crime. Or in other term ,it is a crime in which a computer is the object of the crime (hacking, phishing, spamming) or is used as a tool to commit an offense (child pornography, hate crimes).

STEPS TO PROTECT YOURSELF AGAINST CYBER CRIME

1. Make sure your security software is current – and update it regularly.
2. Lock or log off your computer when you step away.
3. Go offline when you don't need an internet connection.
4. Consider sharing less online.
5. Think twice about using public Wi-Fi.
6. When in doubt, don't click.

Types of Cyber Crime

(I) A computer is the target of the attack—for example, a data breach on a bank site.

(II) A computer is the weapon for an attack—for example, a denial of service (DoS) attack.

(III) A computer is an accessory to a criminal act—for example, digital identity theft which leads to theft of funds from a bank account.

HACKING

Hacking is the process of gaining unauthorized access into a computing device, or group of computer systems. This is done through cracking of passwords and codes which gives access to the systems.

Difference between hacker and cracker is that a cracker breaks the security of computer systems, and a hacker is a person who likes to explore computer systems and master them.

Types of Hackers

(I) Black hat hackers or crackers are individuals with extraordinary computing skills, resorting to malicious / destructive activities. Black hat hackers use their knowledge and skill for their own personal gains probably by hurting others.

(II) White hat hackers are those individuals who use their hacking skills for defensive purposes. This means that the white hat hackers use their knowledge and skill for the good of others and for the common good. Ethical hacking also known as penetration testing or white-hat hacking, involves the same tools, tricks, and techniques that hackers use,but with one major difference that Ethical hacking is legal.

(III) **Grey-Hat Hackers :** These are individuals who work both offensively and defensively at different times. Their behavior can't be predicted. Sometimes they use their skills for the common good while in some other times he uses them for their personal gains.

Hacking Process :

- Foot Printing - Whois lookup,NS lookup,IP lookup
- Scanning - Port Scanning,Network Scanning
- Gaining Access-Password Attacks,Social Engineering,Viruses
- Maintaining Access - Os BackDoors,Trojans,Clears Tracks.

Required Skills of an Ethical Hacker:

(I) **Microsoft:** skills in operation, configuration and management.

(II) **Linux:** knowledge of Linux/Unix;security setting, configuration, services.

(III) **Network Protocols:** TCP/IP; how they function and can be manipulated.

(IV) **Firewalls:** configurations, and operation of intrusion detection systems.

(V) **Project Management:** leading, planning, organizing, and controlling a penetration testing team.

(VI) **Routers:** knowledge of routers, routing protocols, access control lists.

(VII) Mainframes

CYBERBULLYING

It is the use of technology to harass, threaten or humiliate a target. Examples of cyberbullying is sending mean texts, posting false information about a person online, or sharing embarrassing photos or videos.

Cyberbullying differs from in-person bullying :

- **More difficult to recognize** – Bullying conducted via text or online medium can more easily go unnoticed.
- **More relentless** – Cyberbullying doesn't end at school, and can reach at child home.
- **More enduring** –It leaves a paper trail that can follow both the bully and the victim for years.

Different Types of Cyber Bullying

- **Doxing** – publishing revealing personal information about an individual online, for purposes of defaming,humiliating, or harassing the victim .
- **Harassment** – posting threatening,hurtful, or intimidating messages online, or sending them directly to someone, with the intention of harming that person.
- **Impersonation** – creating fake accounts or gaining access to a person's real social media accounts

and posting things to damage the victim's reputation.

- **Cyberstalking** – tracking and monitoring a person's online activity, and using the internet to stalk or harass an individual.

How to Prevent Cyber Bullying?

- Be aware of child's online activities
- Watch for the following signs of cyberbullying in children:
  - Refusal to allow to see what they are doing online
  - Avoidance of discussing what they are doing online
  - Sudden, unexplained increase or decrease in online activity
  - Deactivating social media accounts
  - Emotional responses (including sadness, anger, happiness) linked to their device usage Adults should also teach children to recognize and be aware of the signs of cyberbullying themselves..

CYBER LAW

Cyber law as it is the part of the legal systems that deals with the cyberspace,Internet and with the legal issues. It covers a broad area, like freedom of expressions, access to and utilization of the Internet, and online security or online privacy. Generically, it is known as the law of the web.

- **What is the importance of Cyber Law?**

Cyber law plays a very important role in this new epoch of technology. It is important as it is concerned to almost all aspects of activities and transactions that take place either on the internet or other communication devices. Whether we are aware of it or not, but each action and each reaction in Cyberspace has some legal and Cyber legal views.

- The Information Technology Act of India, 2000 According to Wikipedia "The Information Technology Act, 2000 (also known as ITA-2000, or the IT Act) is an act of the Indian Parliament (no 21 of 2000), it was notified on 17th October 2000. It is the most important law in India that deals with the digital crimes or cyber crimes and electronic commerce. It is based on the United Nations Model Law on Electronic Commerce 1996 (UNCITRAL Model) recommended by the General Assembly of United Nations by a resolution dated 30 January 1997.
- **Some key points of the Information Technology (IT) Act 2000 are as follows :**
  - (I) Act has given birth to new business to companies to issue digital certificates by becoming the Certifying Authorities.
  - (II) This Act allows the government to issue notices on internet through egovernance.
  - (III) E-mail is now considered as a valid and legal form of communication.
  - (IV) Digital signatures are given legal validity within the Act.
  - (V) The communication between the companies or between the company and the government can be done through internet.
  - (VI) Addressing the issue of security is the most important feature of this Act. It introduced the construct of digital signatures that verifies the identity of an individual on internet.
  - (VII) In case of any harm or loss done to the company by criminals, the Act provides a remedy in the form of money to the company.

The Information Technology Act, 2000 provides legal recognition to the transaction done via an electronic exchange of data and other electronic means of communication or electronic commerce transactions.Some of sections under it act 2000 are given below:-

SECTION	OFFENCE	PENALITY
67A	Publishing images containing sexual acts	Imprisonment up to seven years, or/and with fine up to Rs. 1,000,000
67B	Publishing child porn or predating children online	Imprisonment up to five years, or/and with fine up to Rs.1,000,000 on first conviction. Imprisonment up to seven years, or/and with fine up to Rs.1,000,000 on second conviction.
67C	Failure to maintain records	Imprisonment up to three years, or/and with fine
68	Failure/refusal to comply with orders	Imprisonment up to three years, or/and with fine up to Rs.200,000
69	Failure/refusal to decrypt data	Imprisonment up to seven years and possible fine
70	Securing access or attempting to secure access to a protected system	Imprisonment up to ten years, or/and with fine.
71	Misrepresentation	Imprisonment up to three years, or/and with fine up to Rs.100,000

E-WASTE

Whenever an electronic device covers up its working life, or becomes non-usable due to technological advancements or becomes non-functional, it is not used anymore and comes under the category of e-waste or electronic waste. As the technology is changing day by day, more and more electronic devices are becoming non-functional and turning into e-waste. Managing such non-functional electronic devices is termed as e-waste management.

Ewaste Hazards

(I) **On environment**

- Acidification of soil
- Air pollution
- Pollution of ground water
- Landfills with lead and heavy metals

(II) **On Human Health :**

- Lung cancer
- DNA damage
- Asthmatic bronchitis
- Chronic damage to the brain
- Damage to heart, liver and spleen

- E-waste management can be defined as the practical and holistic approach and the founding pillar of cutting down waste from our mother earth. It is reusing and recycling of e-waste which is no longer in use and can be salved for some of its components. We are on the verge of a technological breakthrough with the introduction of AI and we need to dispose off toxic e-waste from our home before we pile up more and more e-waste. We are is dire need of introducing a customer awareness campaign because of lack of interest and knowledge regarding e-waste.
- Proper disposal of used electronic gadgets:- E-waste is a growing problem for us in India. As an 132cr strong economy, we produce e-waste in large quantities. It is very important to dispose off waste in a pragmatic manner.
- Ways to dispose off e-waste:
  1. Give Back to Your Electronic Companies and Drop Off Points
  2. Visit Civic Institutions
  3. Donating Your Outdated Technology
  4. Sell Off Your Outdated Technology
  5. Give Your Electronic Waste to a Certified E-Waste Recycler.

Awareness of Health concerns related to the usage of technology

(I) **Physical Problems:**

- **Repetitive Strain Injury:** the pain exists even when resting and that the lightest work becomes hard to do.
- **Carpal Tunnel Syndrome:** This is an illness caused by injuries that occur due to force on the median nerve found in the wrist.Its symptoms can occur as tingling in hands and fingers and the feeling of lethargy, sudden pain in wrists and arms and sometimes even in shoulders, neck and in the body.
- **Computer Vision Syndrome:** Experts stated that people blink their eyes more frequently while using a computer than they do at other times and that they face some problems related to this situation.
- **Radiation:** Computer screens produce radiations of various types. There have always been doubts that Individuals will have illnesses such as headaches and inattentiveness.
- Sleeping Disorders and Decrease in Productivity.
- Loss of Attention and Stress.

(II) **Psychological Problems:**

- Fear of technology
- Computer anxiety
- Internet addiction
- **Egosurfing:** An illness of regularly searching for one's own name on the web and checking what information is available about one's own on the net.
- **Infornography:** The word, derived from pornography and information, describes the state of "trying to soothe hunger for information on the net."
- **Blog streaking:** A desire to spread information online that shouldn't be known by everybody.
- **Youtube-Narcissism:** Constantly uploading one's own videos in order to introduce and make himself or herself known tooth ers.
- **Google-Stalking:** Trying to get information about all his or her relatives or acquaintances in the web.
- **Photolurking:** Looking at the photo albums of others' on the net.
- **Wikipediholism:** Contributing to the internet encyclopedia, Wikipedia, sending some one's own writings, and revising the present texts.

❑❑❑

12 EXCEPTION HANDLING IN PYTHON (NCERT CLASS 12)

INTRODUCTION

Software programs and applications do not always work flawlessly. When we write a program, we can make mistakes that cause errors when we run it. In Python, you may encounter two types of mistakes: syntax errors and exceptions. Before enabling the rest of the program to run, you may wish to test a specific block of code to ensure it works properly. Python try except blocks allow you to test your code and handle exceptions if they occur.

An exception is defined as an unexpected condition in a program that causes the program's flow to be interrupted.

When the Python interpreter attempts to execute invalid code, it throws an exception, and if the exception is not handled, it disturbs the regular flow of the program's instructions and outputs a traceback. Exceptions are a form of error in which the code has the correct syntax but contains a fault. There are many different types of exceptions, but some of the most prevalent are: ArithmeticError, ImportError, ZeroDivisionError, NameError, and TypeError.

As Python developers, we must consider various exception circumstances and incorporate error management into your code. Python, fortunately, includes a sophisticated error handling system. Python applications may determine the error type at run time and act accordingly by using structured exception handling and a collection of pre-defined exceptions. These actions can include adopting a different route, using default settings, or urging for accurate input.

SYNTAX ERRORS

(I) Syntax errors are detected when we have not followed the rules of the particular programming language while writing a program. These errors are also known as parsing errors.

(II) On encountering a syntax error, the interpreter does not execute the program unless we rectify the errors, save and rerun the program. When a syntax error is encountered while working in shell mode, Python displays the name of the error and a small description about the error.

(III) So, a syntax error is reported by the Python interpreter giving a brief explanation about the error and a suggestion to rectify it.

(IV) Similarly, when a syntax error is encountered while running a program in script mode, a dialog box specifying the name of the error and a small description about the error is displayed.

EXCEPTIONS

An exception is a Python object that represents an error. When an error occurs during the execution of a program, an exception is said to have been raised. Such an exception needs to be handled by the programmer so that the program does not terminate abnormally.

(I) Built-in Exceptions:- Commonly occurring exceptions are usually defined in the compiler/ interpreter. These are called built-in exceptions. Some of the commonly occurring built-in exceptions that can be raised in Python are explained in Table below.

S.No	Name of the Built-in Exception	Explanation
1.	SyntaxError	It is raised when there is an error in the syntax of the Python code.
2.	ValueError	It is raised when a built-in method or operation receives an argument that has the right data type but mismatched or inappropriate values.
3.	IOError	It is raised when the file specified in a program statement cannot be opened.
4.	KeyboardInterrupt	It is raised when the user accidentally hits the Delete or Esc key while executing a program due to which the normal flow of the program is interrupted.
5.	ImportError	It is raised when the requested module definition is not found.
6.	EOFError	It is raised when the end of file condition is reached without reading any data by input().
7.	ZeroDivisionError	It is raised when the denominator in a division operation is zero.
8.	IndexError	It is raised when the index or subscript in a sequence is out of range.

9.	NameError	It is raised when a local or global variable name is not defined.
10.	IndentationError	It is raised due to incorrect indentation in the program code.
11.	TypeError	It is raised when an operator is supplied with a value of incorrect data type.
12.	OverFlowError	It is raised when the result of a calculation exceeds the maximum limit for numeric data type.

A programmer can also create custom exceptions to suit one's requirements. These are called user-defined exceptions.

RAISING EXCEPTIONS

Each time an error is detected in a program, the Python interpreter raises (throws) an exception. Raising an exception involves interrupting the normal flow execution of program and jumping to that part of the program (exception handler code) which is written to handle such exceptional situations.

(I) **The raise Statement:** The raise statement can be used to throw an exception. The syntax of raise statement is :

raise exception-name[(optional argument)]

The argument is generally a string that is displayed when the exception is raised.

In addition to the error message displayed, Python also displays a stack Traceback. This is a structured block of text that contains information about the sequence of function calls that have been made in the branch of execution of code in which the exception was raised.

(II) **The assert Statement:** An assert statement in Python is used to test an expression in the program code. If the result after testing comes false, then the exception is raised. This statement is generally used in the beginning of the function or after a function call to check for valid input. The syntax for assert statement is:

assert Expression[,arguments]

On encountering an assert statement, Python evaluates the expression given immediately after the assert keyword. If this expression is false, an AssertionError exception is raised which can be handled like any other exception.

```
Program:- Use of assert statement
    print("use of assert statement")
    def negativecheck(number):
assert(number>=0), "OOPS... Negative Number"
        print(number*number)
print (negativecheck(100))
print (negativecheck(-350))
```

In case the number gets a negative value, AssertionError will be thrown, and subsequent statements will not be executed. Hence, on passing a negative value (-350) as an argument, it results in AssertionError and displays the message "OOPS.... Negative Number".

Handling Exceptions

Each and every exception has to be handled by the programmer to avoid the program from crashing abruptly. This is done by writing additional code in a program to give proper messages or instructions to the user on encountering an exception. This process is known as exception handling.

- **Need for Exception Handling:** Exception handling is being used not only in Python programming but in most programming languages like C++, Java, Ruby, etc. It is a useful technique that helps in capturing runtime errors and handling them so as to avoid the program getting crashed.

Important points regarding exceptions and their handling:

(I) Python categorises exceptions into distinct types so that specific exception handlers (code to handle that particular exception) can be created for each type.

(II) Exception handlers separate the main logic of the program from the error detection and correction code. The segment of code where there is any possibility of error or exception, is placed inside one block. The code to be executed in case the exception has occurred, is placed inside another block. These statements for detection and reporting the exception do not affect the main logic of the program.

(III) The compiler or interpreter keeps track of the exact position where the error has occurred.

(IV) Exception handling can be done for both user-defined and built-in exceptions.

- **Process of Handling Exception:** When an error occurs, Python interpreter creates an object called the exception object. This object contains information about the error like its type, file name and position in the program where the error has occurred. The object is handed over to the runtime system so that it can find an appropriate code to handle this particular exception. This process of creating an exception object and handing it over to the runtime system is called throwing an exception.

The runtime system searches the entire program for a block of code, called the exception handler that can handle the raised exception.

A runtime system refers to the execution of the statements given in the program. It is a complex mechanism consisting of hardware and software that comes into action as soon as the program, written in any programming language, is put for execution.

A runtime system first searches for the method in which the error has occurred and the exception has been raised. If not found, then it searches the method from which this method (in which exception was raised) was called. This hierarchical search in reverse order continues till the exception handler is found. This entire list of methods is known as call stack.

When a suitable handler is found in the call stack, it is executed by the runtime process. This process of executing a suitable handler is known as catching the exception. If the runtime system is not able to find an appropriate exception after searching all the methods in the call stack, then the program execution stops.

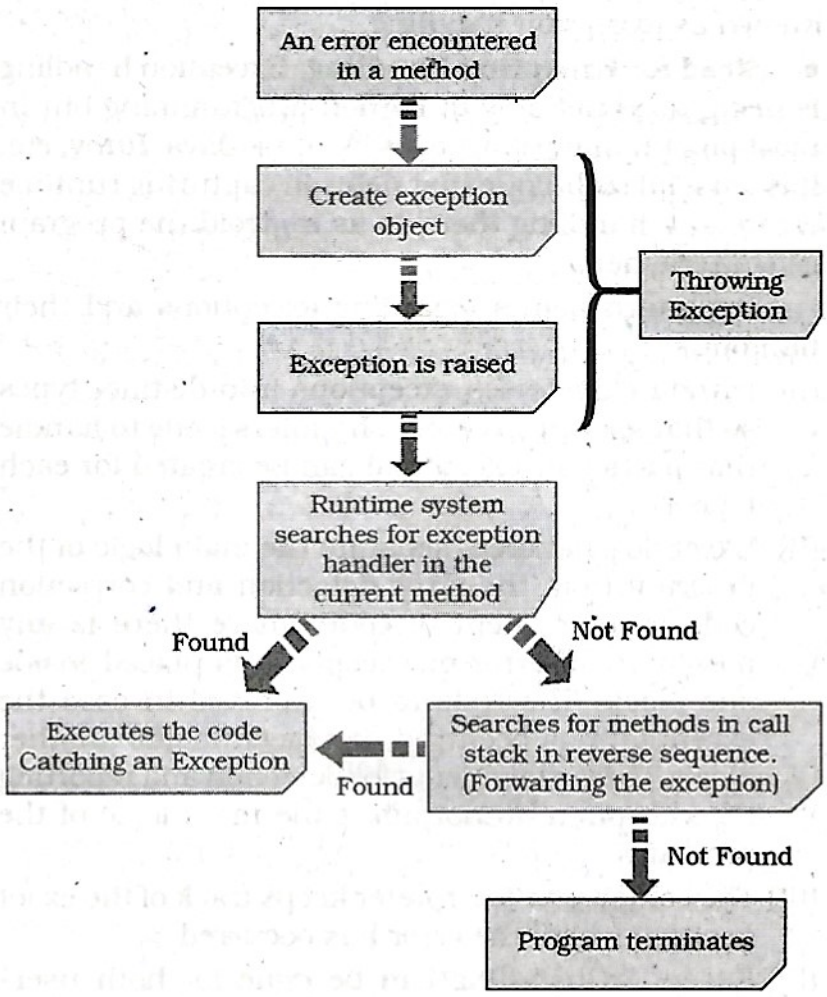


FIG:- Steps of handling exception

- **Catching Exceptions :** An exception is said to be caught when a code that is designed to handle a particular exception is executed. Exceptions, if any, are caught in the try block and handled in the except block.

While writing or debugging a program, a user might doubt an exception to occur in a particular part of the code. Such suspicious lines of codes are put inside a try block.

Every try block is followed by an except block. The appropriate code to handle each of the possible exceptions (in the code inside the try block) are written inside the except clause.

The syntax of try ... except clause is as follows:

```
try:
[ program statements where exceptions might occur]
except [exception-name]:
[ code for exception handling if the exception-name error is encountered]
```

- **try...except...else clause:** We can put an optional else clause along with the try...except clause. An except block will be executed only if some exception is raised in the try block.But if there is no error then none of the except blocks will be executed. In this case, the statements inside the else clause will be executed.

FINALLY CLAUSE

The try statement in Python can also have an optional finally clause. The statements inside the finally block are always executed regardless of whether an exception has occurred in the try block or not. It is a common practice to use finally clause while working with files to ensure that the file object is closed. If used, finally should always be placed at the end of try clause, after all except blocks and the else block.

Use of finally clause

```
print ("Handling exception using try...except...else...finally")
try:
numerator=50
denom=int(input("Enter the denominator: "))
quotient=(numerator/denom)
print ("Division performed successfully")
except ZeroDivisionError:
print ("Denominator as ZERO is not allowed")
except ValueError:
print ("Only INTEGERS should be entered")
else:
print ("The result of division operation is ", quotient)
finally:
print ("OVER AND OUT")
```

In the above program, the message "OVER AND OUT" will be displayed irrespective of whether an exception is raised or not.

(I) Recovering and continuing with finally clause:- If an error has been detected in the try block and the exception has been thrown, the appropriate except block will be executed to handle the error. But if the exception is not handled by any of the except clauses, then it is re-raised after the execution of the finally block.

After execution of finally block, Python transfers the control to a previously entered try or to the next higher level default exception handler. In such a case, the statements following the finally block is executed. That is, unlike except, execution of the finally clause does not terminate the exception. Rather, the exception continues to be raised after execution of finally.

□□□

13 FILE HANDLING IN PYTHON
(NCERT CLASS 12)

INTRODUCTION

Programs which we have done so far, are the ones which run, produce some output and end. Their data disappears as soon as they stop running. Next time when you use them, you again provide the data and then check the output. This happens because the data entered is stored in primary memory, which is temporary in nature.

What if the data with which, we are working or producing as output is required for later use? Result processing done in Term Exam is again required for Annual Progress Report. Here if data is stored permanently, its processing would be faster. This can be done, if we are able to store data in secondary storage media i.e. Hard Disk, which we know is permanent storage media.

Data is stored using file(s) permanently on secondary storage media. Data in Word processing applications, Spreadsheets, Presentation applications, etc. all of them created data files and stored your data, so that you may use the same later on. Apart from this you were permanently storing your python scripts (as .py extension) also.

A file (i.e. data file) is a named place on the disk where a sequence of related data is stored. In python files are simply stream of data, so the structure of data is not stored in the file, along with data. Basic operations performed on a data file are:

(I) Naming a file
(II) Opening a file
(III) Reading data from the file
(IV) Writing data in the file
(V) Closing a file

Using these basic operations, we can process file in many ways, such as :

(I) Creating a file
(II) Traversing a file for displaying the data on screen
(III) Appending data in file
(IV) Inserting data in file
(V) Deleting data from file
(VI) Create a copy of file
(VII) Updating data in the file, etc.

Python allow us to create and manage two types of file :

(I) Text
(II) Binary

- **TEXT FILE :** A text file is usually considered as sequence of lines. Line is a sequence of characters (ASCII), stored on permanent storage media. Although default character coding in python is ASCII but using constant u with string, supports Unicode as well. As we talk of lines in text file, each line is terminated by a special character, known as End of Line (EOL). From strings we know that \n is newline character. So at the lowest level, text file will be collection of bytes. Text files are stored in human readable form and they can also be created using any text editor.
- **Binary File :** A binary file contains arbitrary binary data i.e. numbers stored in the file, can be used for numericaloperation(s). So when we work on binary file, we have to interpret the raw bit pattern(s) read from the file into correct type of data in our program. It is perfectly possible to interpret a stream of bytes originally written as string, as numeric value. But we know that will be incorrect interpretation of data and we are not going to get desired output after the file processing activity. So in the case of binary file it is extremely important that we interpret the correct data type while reading the file. Python provides special module(s) for encoding and decoding of data for binary file.

To handle data files in python, we need to have a file object. Object can be created by using open() function or file() function. To work on file, first thing we do is open it. This is done by using built in function open(). Using this function a file object is created which is then used for accessing various methods and functions available for file manipulation.

Syntax of open() function is:-

file_object = open(filename [, access_mode] [,buffering])

open() requires three arguments to work,i.e.,

(I) first one (filename) is the name of the file on secondary storage media, which can be string constant or a variable. The name can include the description of path, in case, the file does not reside in the same folder / directory in which we are working. We will know more about this in later section of chapter.
(II) The second parameter (access_mode) describes how file will be used throughout the program. This is an optional parameter and the default access_mode is reading.
(III) The third parameter (buffering) is for specifying how much is read from the file in one read. The function will return an object of file type using which we will manipulate the file, in our program.

When we work with file(s), a buffer (area in memory where data is temporarily stored before being written to file), is automaticallyassociated with file when we open the file. While writing the content in the file, first it goes to buffer and once the buffer is full, data is written to the file. Also when file is closed, any unsaved data is transferred tofile. flush() function is used to force transfer of data from buffer to file.

FILE ACCESS MODES

r will open the text file for reading only and rb will do the same for binary format file. This is also thedefault mode. The file pointer is placed at the beginning for reading purpose, when we open a file in thismode.

w will open a text file for writing only and wb for binary format file. The file pointer is again placed at the beginning. A non existing file will be created using this mode. Remember if we open an already existing file (i.e. a file containing data) in this mode then the file will be overwritten as the file pointer will be at the beginning for writing in it.

a mode allow us to append data in the text file and ab in binary file. Appending is writing data at the end of the file. In this mode, file pointer is placed at the end in an existing file. It can also be used for creating a file, by opening a non existing file using this mode.

r+ will open a text file and rb+ will open a binary file, for both reading and writing purpose. The file pointer is placed at the beginning of the file when it is opened using r+ / rb+ mode.

w+ opens a file in text format and wb+ in binary format, for both writing and reading. File pointer will be placed at the beginning for writing into it, so an existing file will be overwritten. A new file can also be created using this mode.

a+ opens a text file and ab+ opens a binary file, for both appending and reading. File pointer is placed at the end of the file, in an already existing file. Using this mode a non existing file may be created.

Example usage of open:-

file= open("Sample.txt","r+")

will open a file called Sample.txt for reading and writing purpose. Here the name (by which it exists onsecondary storage media) of the file specified is constant. We can use a variable instead of a constant as name of the file. Sample file, if already exists, then it has to be in the same folder where we are working now, otherwise we have to specify the complete path. It is not mandatory to have file name with extension. In the example .txt extension is used for our convenience of identification. As it is easy to identify the file as text file. Similarly for binary file we will use . dat extension.

Other function, which can be used for creation of a file is file(). Its syntax and its usage is same as open().

Apart from using open() or file() function for creation of file, with statement can also be used for same purpose. Using with ensures that all the resources allocated to file objects gets deallocated automatically once we stop using the file. Its syntax is :

with open() as fileobject :

Example:

with open("Sample.txt","r+") as file :

file manipulation statements

Let's know about other method's and function's which can be used with file object.

fileobject. close() will be used to close the file object, once we have finished working on it. The method will free up all the system resources used by the file, this means that once file is closed, we will not be able to use the file object any more. Before closing the file any material which is not written in file, will be flushed off. So it is good practice to close the file once we have finished using it. In case, if we reassign the file object to some other file, then python will automatically close the file.

Methods for reading data from the file are:

- readline() will return a line read, as a string from the file. First call to function will return first line, second call next line and so on. Remember file object keeps the track of from where reading / writing of data should happen. For readline() a line is terminated by \n (i.e. new line character). The new line character is also read from the file and post-fixed in the string. When end of file is reached, readline() will return an empty string.

  It's syntax is

  fileobject.readline()

  Since the method returs a string it's usage will be

  >>>x = file.readline()

  or

  >>>print file.readline()

  For reading an entire file using readline(), we will have to loop over the file object. This actually is memory efficient, simple and fast way of reading the file. Let's see a simple example of it

  >>>for line in file:

  ... print line

  Same can be achieved using other ways of looping.

- readlines()can be used to read the entire content of the file. You need to be careful while using it w.r.t. size of memory required before using the function. The method will return a list of strings, each separated by \n.

  It's syntax is:

  fileobject.readlines()

  as it returns a list, which can then be used for manipulation.

- read() can be used to read specific size string from file. This function also returns a string read from the file. At the end of the file, again an empty string will be returned.

Syntax of read() function is

fileobject.read([size])

Here size specifies the number of bytes to be read from the file. So the function may be used to read specific quantity of data from the file. If the value of size is not provided or a negative value is specified as size then entire file will be read. Again take care of memory size available before reading the entire content from the file.

Let's see the usage of various functions for reading data from file. Assuming we have a file data.txt containing hello world.\n this is my first file handling program.\n I am using python language.

Example of readlines():

```
>>>lines = []
>>>lines = file.readlines()
```

If we print element of lines (which can be done by iterating the contents of lines) we will get:

```
hello world.
this is my first file handling program.
I am using python language.
```

Can you notice, there are two blank lines in between every string / sentence. Find out the reason for it.

Example of using read():

```
lines = []
content = file.read() # since no size is given, entire file will be read
lines = content.splitlines()
print lines
```

will give you a list of strings:

['hello world.', 'this is my first file handling program.', 'I am using python language.']

For sending data in file, i.e. to create / write in the file, write() and writelines() methods can be used.write() method takes a string (as parameter) and writes it in the file. For storing data with end of line character, you will have to add \n character to end of the string. Notice addition of \n in the end of every sentence while talking of data.txt. As argument to the function has to be string, for storing numeric value, we have to convert it to string.

Its syntax is

fileobject.write(string)

Example

```
>>>f = open('test1.txt','w')
>>>f.write("hello world\n")
>>>f.close()
```

For numeric data value conversion to string is required.

Example

```
>>>x = 52
>>>file.write(str(x))
```

For writing a string at a time, we use write() method, it can't be used for writing a list, tuple etc. into a file.Sequence data type can be written using writelines() method in the file. It's not that, we can't write a string using writelines() method.

It's syntax is:

fileobject.writelines(seq)

So, whenever we have to write a sequence of string / data type, we will use writelines(), instead of write().

Example:

```
f = open('test2.txt','w')
str = 'hello world.\n this is my first file handling program.\n I am using python language"
f.writelines(str)
f.close()
```

let's consider an example of creation and reading of file in interactive mode

```
>>>file = open('test.txt','w')
>>>s = ['this is 1stline','this is 2nd line']
>>>file.writelines(s)
>>>file.close()
>>>file.open('test.txt') # default access mode is r
>>>print file.readline()
>>>file.close()
```

Will display following on screen

this is 1st line this is 2nd line

Let's walk through the code. First we open a file for creation purpose, that's why the access mode is w. Inthe next statement a list of 2 strings is created and written into file in 3rd statement. As we have a list of 2 strings, writelines() is used for the purpose of writing. After writing the data in file, file is closed.

In next set of statements, first one is to open the file for reading purpose. In next statement we are readingthe line from file and displaying it on screen also. Last statement is closing the file.

Although, we have used readline() method to read from the file, which is suppose to return a line i.e. string at a time, but what we get is, both the strings. This is so, because writelines() does not add any EOL.

Character to the end of string. You have to do it. So to resolve this problem, we can create s using following statement

```
s = ['this is 1st line\n', 'this is 2nd line\n']
```

Now using readline(), will result into a string at a time.

All reading and writing functions discussed till now, work sequentially in the file. To access the contents of file randomly - seek and tell methods are used.

- tell() method returns an integer giving the current position of object in the file. The integer returned specifies the number of bytes from the beginning of the file till the current position of file object.

It's syntax is

fileobject.tell()

- fileobject.seek(offset [, from_what])

here offset is used to calculate the position of fileobject in the file in bytes. Offset is added to from_what (reference point) to get the position. Following is the list of from_what values:

Value	Reference point
0	beginning of the file
1	current position of file
2	end of file

Let's read the second word from the test1 file created earlier. First word is 5 alphabets, so we need to move to 5th byte. Offset of first byte starts from zero.

```
f = open('test1.txt','r+')
f.seek(5)
fdata = f.read(5)
print fdata
f.close()
```

will display world on screen.

Let's write a function to create and display a text file using one stream object.

```
def fileHandling():
file = open("story.txt","w+")
while True:
line = raw_input("enter sentence :")
file.write(line)
choice = raw_input("want to enter more data in file Y / N")
if choice.upper() == 'N' : break
file.seek(0)
lines = file.readlines()
file.close()
for l in lines:
print l
```

in this function after opening the file, while loop allow us to store as many strings as we want in the file.

once that is done, using seek() method file object is taken back to first alphabet in the file. From where we read the complete data in list object.

We know that the methods provided in python for writing / reading a file works with string parameters.

So when we want to work on binary file, conversion of data at the time of reading, as well as writing is required. Pickle module can be used to store any kind of object in file as it allows us to store python objects with their structure. So for storing data in binary format, we will use pickle module.

First we need to import the module. It provides two main methods for the purpose, dump and load. For creation of binary file we will use pickle.dump() to write the object in file, which is opened in binary access mode. Syntax of dump() method is:-

dump(object, fileobject)

Example:

```
def fileOperation1():
import pickle
l = [1,2,3,4,5,6]
file = open('list.dat', 'wb') # b in access mode is for binary file
pickle.dump(l,file) # writing content to binary file
file.close()
```

Example:

Example of writing a dictionary in binary file:

```
MD = {'a': 1, 'b': 2, 'c': 3}
file = open('myfile.dat', 'wb')
pickle.dump(MD,file)
file.close()
```

Once data is stored using dump(), it can then be used for reading. For reading data from file we will:

use pickle.load() to read the object from pickle file.

Syntax of load() is :

object = load(fileobject)

Note : we need to call load for each time dump was called.

```
# read python dict back from the file
ifile = open('myfile.dat', 'rb')
MD1 = pickle.load(ifile) # reading data from binary file
ifile.close()
print MD1
```

Results into following on screen:

{'a': 1, 'c': 3, 'b': 2}

To distinguish a data file from pickle file, we may use a different extension of file. .pk / .pickle are commonly used extension for same.

Files are always stored in current folder / directory by default. The os (Operating System) module ofpython provides various methods to work with file and folder / directories. For using these functions, we have to import os module in our program. Some of the useful methods, which can be used with files in os module are as follows:

1. getcwd() to know the name of current working directory.
2. path.abspath(filename) will give us the complete path name of the data file.
3. path.isfile(filename) will check, whether the file exists or not.
4. remove(filename) will delete the file. Here filename has to be the complete path of file.

5. rename(filename1,filename2) will change the name of filename1 with filename2.

Once the file is opened, then using file object, we can derive various information about file. This is doneusing file attributes defined in os module. Some of the attributes defined in it are:-

1. file.closed returns True if file is closed
2. file.mode returns the mode, with which file was opened.
3. file.name returns name of the file associated with file object while opening the file.

We use file object(s) to work with data file, similarly input/output from standard I/O devices is also performed using standard I/O stream object. Since we use high level functions for performing input/output through keyboard and monitor such as - raw_input(), input() and print statement we were not required to explicitly use I/O stream object. But let's learn a bit about these streams also.

The standard streams available in python are :

(I) Standard input stream
(II) Standard output stream and
(III) Standard error stream

These standard streams are nothing but file objects, which get automatically connected to your program'sstandard device(s), when you start python. In order to work with standard I/O stream, we need to import sys module. Methods which are available for I/O operations in it are:-

(I) read() for reading a byte at a time from keyboard
(II) write() for writing data on terminal i.e. monitor

Their usage is

```
import sys
print ' Enter your name :'
name = ''
while True:
c = sys.stdin.read()
if c == '\n' :
break
name = name + c
sys.stdout.write( 'your name is ' + name)
same can be done using high level methods also
name = raw_input('Enter your name :')
print 'your name is ',name
```

As we are through with all basic operations of file handling, we can now learn the various processes which can be performed on file(s) using these operations.

(I) Creating a file

Option 1 : An empty file can be created by using open() statement. The content in the file can be stored later on using append mode.

Option 2 : create a file and simultaneously store / write the content also.

Algorithm

1. Open a file for writing into it
2. Get data to be stored in the file (can be a string at a time or complete data)
3. Write it into the file (if we are working on a string at a time, then multiple writes will be required.)
4. Close the file

Code :

```
def fileCreation():
ofile = open("story.txt","w+")
choice = True
while True:
line = raw_input("enter sentence :")
ofile.write(line)
choice = raw_input("want to enter more data in file Y / N")
if choice == 'N' : break
ofile.close()
```

At the run time following data was provided

```
this is my first file program
writing 2nd line to file
this is last line
```

(II) Traversal for display

Algorithm

1. Open file for reading purpose.
2. Read data from the file (file is sequentially accessed by any of the read methods).
3. Display read data on screen.
4. Continue step 2 & 3 for entire file.
5. Close the file.

Program Code:

```
def fileDisplaying1():
for l in open("story.txt","r").readlines():
print l
file.close()
```

The above code will display

this is my first file programwriting 2nd line to filethis is last line.

(III)Creating a copy of file

Algorithm

1. Open the source file for reading from it.
2. Open a new file for writing purpose
3. Read data from the first file (can be string at a time or complete data of file.)
4. Write the data read in the new file.
5. Close both the files.

Program Code for creating a copy of existing file:

```
def fileCopy():
ifile = open("story.txt","r")
ofile = open("newstory.txt","w")
l = file.readline()
while l:
ofile.write(l)
```

```
l = file.readline()
ifile.close()
ofile.close()
```

Similarly a binary file can also be copied. We don't need to use dump() & load()methods for this. As wejust need to pass byte strings from one file to another.

(IV) Deleting content from the file

It can be handled in two ways

Option 1 (for small file, which can fit into memory i.e. a few Mb's)

Algorithm

1. Open the source file for reading from it.
2. Open a new file for writing purpose
3. Read data from the first file (can be string at a time or complete data of file.)
4. Write the data read in the new file.
5. Close both the files.
6. Open same file for writing into it
7. Write the modified list into file.
8. Close the file.

Program code for deleting second word from story.txt is:

```
def filedel():
    with open('story.txt','r') as file :
        l = file.readlines()
    file.close()
    print l
    del l[1]
    print l
    file.open('story.txt','w')
    file.writelines(l)
    file.close()
```

Similarly we can delete any content, using position of the data. If position is unknown then serach thedesired data and delete the same from the list.

Option 2 (for large files, which will not fit into memory of computer. For this we will need two files)

Algorithm

1. Get the data value to be deleted
2. Open the file for reading purpose
3. Open another (temporary) file for writing into it
4. Read a string (data value) from the file
5. Write it into temporary file, if it was not to be deleted.
6. The process will be repeated for entire file (in case all the occurrence of data are to be deleted)
7. Close both the files.
8. Delete the original file
9. Rename the temporary file to original file name.

Program code for deletion of the line(s) having word (passed as argument) is :

```
import os
def fileDEL(word):
file = open("test.txt","r")
newfile = open("new.txt","w")
while True:
line = file.readline()
if not line :
break
            else :
            if word in line :
                pass
            else :
            print line
            newfile.write(line)
        file.close()
        newfile.close()
    os.remove("test.txt")
    os.rename("new.txt","test.txt")
```

(V) Inserting data in a file

It can happen in two ways. Insertion at the end of file or insertion in the middle of the file.

Option 1 Insertion at the end. This is also known as appending a file.

Algorithm

1. open the file in append access mode.
2. Get the data value for to be inserted in file, from user.
3. Write the data in the file
4. Repeat set 2 & 3 if there is more data to be inserted (appended)
5. Close the file.

Program code for appending data in binary file:-

```
def binAppend():
import pickle
file = open('data.dat','ab')
while True:
y = int(raw_input())
pickle.dump(y,file)
ans = raw_input('want to enter more data Y
/ N')
if ans.upper()== 'N' : break
file.close()
```

Option 2 Inserting in the middle of file

There is no way to insert data in the middle of file. This is a limitation because of OS. To handle such insertions re-writing of file is done.

Algorithm

1. Get the data value to to be inserted with its position.
2. Open the original file in reading mode.
3. Open another (temporary) file for writing in it.
4. Start reading original file sequentially, simultaneously writing it in the temporary file.

5. This is to be repeated till you reach the position of insertion of new data value.
6. Write the new value in temporary file.
7. Repeat step 4 for remaining data of original file.
8. Delete original file
9. Change the name of temporary file to original file.

Code for inserting data in the file with the help of another file

It will be similar to the code used for deletion of content using another file. Here instead of not writing the content add it in the file.

An alternative to this is, first read the complete data from file into a list. Modify the list and rewrite the modified list in the file.

(VI) Updating a file

File updation can be handled in many ways. Some of which are

Option 1 - Truncate write

Algorithm

1. Open the file for reading from it
2. Read the content of file in an object (variable) usually list
3. Close the file
4. Get the details of data to be modified
5. Update the content in the list
6. Re open the file for writing purpose (we know that now opening the file for writing will truncate the existing file).
7. Write the list back to the file.

Program Code for this will be similar to following:

```
with open("sample.txt","r") as file:
content = file.read()
file.close()
content.process()
wit h open ("sample.txt","w") as file :
file.writelines(content)
file.close()
```

Option 2 - Write replace

Algorithm

Open the original file for reading

Open temporary file for writing

Read a line / record from the file

If this was not to be modified copy it in temporary file otherwise copy the modified line / record in the temporary file.

Repeat previous two steps for complete file.

This way of processing a file using python has been handled earlier.

Option 3 - In place updation

Algorithm

1. Open file for reading and writing purpose
2. Get the details of data value to be modified
3. Using linear search, reach to the record / data to be modified
4. Seek to the start of the record
5. Re write the data
6. Close the file.

Updating a text file in this manner is not safe, as any change you make to the file may overwrite thecontent you have not read yet. This should only be used when the text to be replaced is of same size. In place updation of a binary file is not possible. As this requires placing of fileobject to the beginning of the record, calculating size of data in dump file is not possible. So updating a data file using third option is not recommended in python.

Let's create a data file storing students record such as Admission number, Name, Class and Total marks.

Data to be stored contains numeric data, hence will be stored in binary file. We will use dictionary data type to organize this information.

```
from pickle import load, dump
import os
import sys
def bfileCreate(fname):.
l = []
sd = {1000:['anuj',12,450]}
with open(fname,'wb') as ofile :
while True :
dump(sd,ofile)
ans = raw_input("want to enter more data Y
/ N")
if ans.upper() == 'N' : break
x = int(raw_input("enter admission number
of student"))
l = input("enter name class and marks of
student enclosed in []")
sd[x] = l
ofile.close()
def bfileDisplay(fname):
if not os.path.isfile(fname) :
print "file does not exist"
else:
ifile = open(fname,'rb')
try :
while True:
sd = {}
sd = load(ifile)
print sd
except EOFError:
                    pass

ifile.close()
```

Use the code to store records of your class mates. Once the file is created, use bfileDisplay() to see the result. Do you find some problem in the content displayed? Find and resolve the problem? □□□

14 STACK (NCERT CLASS 12)

INTRODUCTION

A stack is a data structure whose elements are accessed according to the Last-In First-Out (LIFO) principle.

This is because in a stack, insertion and deletion of elements can only take place at one end, called top of the stack. Consider the following examples of stacks:

1. Ten glass plates placed one above another. (The plate that is kept last has to be taken out first)
2. The tennis balls in a container. (You cannot remove more than one ball at a time)
3. A pile of books
4. A stack of coins

In the above picture coins are kept one above the other and if any additional coin is to be added, it can beadded only on the top. If we want to remove any coin from the stack, the coin on the top of the stack has to be removed first. That means, the coin that was kept last in the stack has to be taken out first.

The two operations performed on the stack are:

1. Push: Adding(inserting) new element on to the stack.
2. Pop: Removing (deleting) an element from the stack

PUSH OPERATION:-

Adding new element to the stack list is called push operation. When the stack is empty, the value of top is -1. Basically, an empty stack is initialized with an invalid subscript.Whenever a Push operation is performed, the top is incremented by one and then the new value is inserted on the top of the list till the time the value of top is less than or equal to the size of the stack. Let us first have a look at the logic to program the Push operation for a stack through the following algorithm:

1. Start
2. Initialize top with -1.

Step 3: Input the new element.
Step 4: Increment top by one.
Step 5: stack[top]=new element
Step 6: Print "Item Inserted"
Step 7: Stop

POP OPERATION

Removing existing elements from the stack list is called pop operation. Here we have to check if the stack is empty by checking the value of top. If the value of top is -1, then the stack is empty and such a situation is called Underflow. Otherwise Pop operation can be performed in the stack. The top is decremented by one if an element is deleted from the list.

The algorithm for pop operation is as follows:

Step 1: Start
Step 2: If the value of top is -1 go to step 3 else go to step 4
Step 3: Print "Stack Empty" and go to step 7
Step 4: Deleted item = Stack[top]
Step 5: Decrement top by 1
Step 6: print "Item Deleted"
Step 7: Stop

TRAVERSAL IN A ATACK

Traversal is moving through the elements of the stack. If you want to display all the elements of the stack,the algorithm will be as follows:

Step 1: Start
Step 2: Check the value of top. If top=-1 go to step 3 else go to step 4
Step 3: Print "Stack Empty" and go to step 7
Step 4: Print the top element of the stack.
Step 5: Decrement top by 1
Step 6: If top= -1 go to step 7 else go to step 4
Step 7: Stop

In Python, we already have pop() and append() functions for popping and adding elements on to the stack.Hence, there is no need to write the code to add and remove elements in the stack. Consider the following programs that perform the Push and Pop operation on the stack through append() and pop().

Program to implement stack (without classes):-

```
s=[]
c="y"
while (c=="y"):
print "1. PUSH"
print "2. POP "
```

```
    print "3. Display"
    choice=input("Enter your choice: ")
    if (choice==1):
    a=input("Enter any number :")
    s.append(a)
    elif (choice==2):
    if (s==[]):
    print "Stack Empty"
    else:
    print "Deleted element is : ",s.pop()
    elif (choice==3):
    l=len(s)
    for i in range(l-1,-1,-1):
    print s[i]
    else:
    print("Wrong Input")
    c=raw_input("Do you want to continue or
not? ")
```

Output:

```
>>>
```

1. PUSH
2. POP
3. Display

Enter your choice: 2
Stack Empty
Do you want to continue or not? y

1. PUSH
2. POP
3. Display

Enter your choice: 1
Enter any number :100
Do you want to continue or not? y

1. PUSH
2. POP
3. Display

Enter your choice: 1
Enter any number :200
Do you want to continue or not? y

1. PUSH
2. POP
3. Display

Enter your choice: 1
Enter any number :300
Do you want to continue or not? y

1. PUSH
2. POP
3. Display

Enter your choice: 3
300
200
100
Do you want to continue or not? y

1. PUSH
2. POP
3. Display

Enter your choice: 2
Deleted element is : 300
Do you want to continue or not? y

1. PUSH
2. POP
3. Display

Enter your choice: 3
200
100
Do you want to continue or not? n

```
>>>
```

The same program can also be implemented using classes as shown below:

Program to implement a stack(Using classes)

```
s=[]
def push(self):while (c=="y"):
    a=input("Enter any number :")
     stack.s.append(a)
def display(self):
l=len(stack.s)
a=stack()
c="y"
while (c=="y"):
print "1. PUSH"
print "2. POP "
print "3. Display"
choice=input("Enter your choice: ")
if (choice==1):
a.push()
elif (choice==2):
if (a.s==[]):
print "Stack Empty"
else:
print "Deleted element is : ",a.s.pop()
elif (choice==3):
a.display()
else:
print("Wrong Input")
c=raw_input("Do you want to continue or
not? ") output:
```

Output:

```
>>>
```

1. PUSH
2. POP
3. Display

Enter your choice: 1
Enter any number :100
Do you want to continue or not? y

1. PUSH
2. POP

```
3. Display
Enter your choice: 1
Enter any number :200
Do you want to continue or not? y
1. PUSH
2. POP
3. Display
Enter your choice: 3
200
100
Do you want to continue or not? y
1. PUSH
2. POP
3. Display
Enter your choice: 2
Deleted element is : 200
Do you want to continue or not? y
1. PUSH
2. POP
3. Display
Enter your choice: 2
Deleted element is : 100
Do you want to continue or not: y
1. PUSH
2. POP
3. Display
Enter your choice: 2
Stack Empty
Do you want to continue or not? n
>>>
```

EXPRESSION

An .expression is a combination of variables,constants and operators. Expressions can be written in Infix, Postfix or Prefix notations.

Infix Expression: In this type of notation, the operator is placed between the operands. For example: A+B, A*(B+C), X*Y/Z, etc.

Postfix Expression: In this type of notation, the operator is placed after the operands.

For example: AB+, ABC+*,XYZ/*, etc.

Prefix Expression: In this type of notation, the operator is placed before the operands.

For example: +AB,*A+BC,*X/YZ, etc.

- **Conversion of an infix expression to postfix expression:-**

The following algorithm shows the logic to convert an infix expression to an equivalent postfix expression:

Step 1: Start

Step 2: Add "("(left parenthesis) and ")" (right parenthesis) to the start and end of the expression(E).

Step 3: Push "("left parenthesis onto stack.

Step 4: Check all symbols from left to right and repeat step 5 for each symbol of 'E' until the stack become empty.

Step 5: If the symbol is:

i) an operand then add it to list.

ii) a left parenthesis "("then push it onto stack.

iii) an operator then:

a) Pop operator from stack and add to list which has the same or higher precedence than the incoming operator.

b) Otherwise add incoming operator to stack.

iv) A right parenthesis")" then:

a) Pop each operator from stack and add to list until a left parenthesis is encountered.

b) Remove the left parenthesis.

Step 6: Stop.

- **Evaluation of Postfix Expression**

The algorithm to evaluate a postfix expression is as follows:

Step 1: Start

Step 2: Check all symbols from left to right and repeat steps 3 & 4 for each symbol of expression 'E' until all symbols are over.

i) If the symbol is an operand, push it onto stack.

ii) If the symbol is an operator then

a) Pop the top two operands from stack and apply an operator in between them.

b) Evaluate the expression and place the result back on stack.

Step 3: Set result equal to top element on the stack.

Step 4: Stop

Infix, Prefix and Postfix Notations

Type of Expression	Description	Example
Infix	Operators are placed in between the operands	x * y + z 3 *(4 + 5) (x + y)/(z * 5)
Prefix (Polish)	Operators are placed before the corresponding operands	+z*xy *3+45 /+xy*z5
Postfix (Reverse Polish)	Operators are placed after the corresponding operands	xy*z+ 345+* xy+z5*/

15 QUEUE
(NCERT CLASS 12)

INTRODUCTION

A queue is a container of elements, which are inserted and removed according to the first-in first-out(FIFO) principle.

In a queue, persons who stand in the queue will carry out their work one by one. That means those who stands first in the queue will be allowed to carry out his work first and the person who stands at the second position will be allowed to carry out his work second only. At the same time those who come late will be joining the queue at the end. In simple terms it is called 'first come first out'.

Technically speaking a queue is a linear list, to keep an ordered collection of elements / objects. The principle operations, which can be performed on it are:-

(I) Addition of elements &

(II) Removal of elements.

Addition of element is known as INSERT operation, also known as enqueu-ing. It is done using rear terminal position, i.e. tail end. Removal of element is known as DELETE operation also know as dequeueing. It is done using front terminal position, i.e. head of the list. As the two operations in the queue are performed from different ends, we need to maintain both the access points. These access points are known as FRONT, REAR. FRONT is first element of the queue and REAR is last element. As queue is FIFO implementation, FRONT is used for delete operation and REAR is used for insert operation.

Following are some applications of queue in computers:

(I) In a single processor multi tasking computer, job(s) waiting to be processed form a queue. Same happens when we share a printer with many computers.

(II) Compiling a HLL code.

(III) Using down load manager, for multiple files also uses queue for ordering the files.

(IV) In multiuser OS - job scheduling is done through queue.

QUEUE OPERATIONS

Various operations, which can be performed on a queue are:

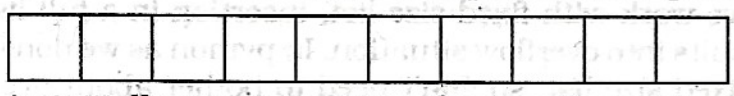
(I) Create a queue having a data structure to store linear list with ordering of elements.

(II) Insert an element will happen using REAR, REAR will be incremented to hold the new value in queue.

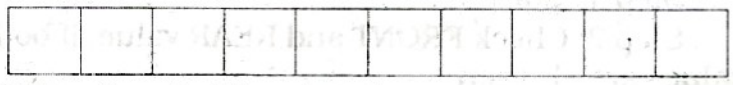
(III) Delete an element will happen using FRONT and FRONT will also be incremented to be able to accessnext element.

Let's understand this with the help of an example:

1. We will use list data type to implement the queue.



2. As initially queue is empty, front and rear should not be able to access any element. The situation can be represented by assigning -1 to both REAR and FRONT as initial value.



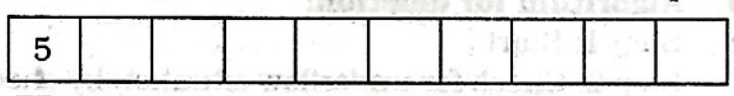
F(front) = -1, R(rear) = -1

Once the queue is created, we will perform various operations on it. Following is the list of operations with its affect on queue:

INSERT(5)

F = F + 1

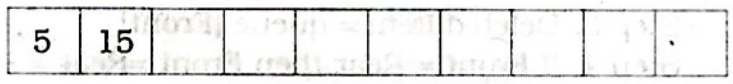
R = R + 1 (as this is the first element of the queue)



FR

INSERT(15)

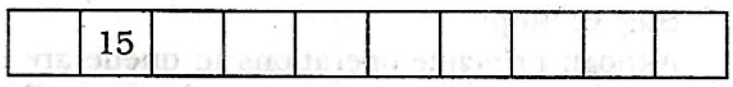
R = R + 1



F R

DELETE()

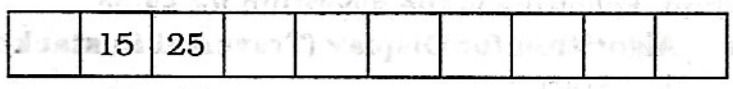
F = F + 1



FR

INSERT(25)

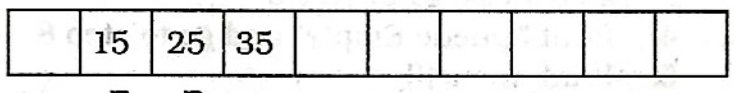
R = R + 1



F R

INSERT(35)

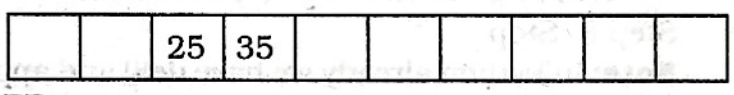
R = R + 1



F R

DELETE()

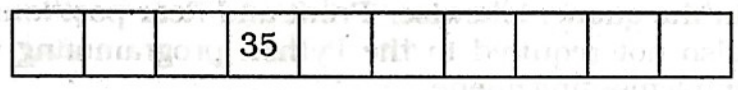
F = F + 1



FR

DELETE()

F + F + 1



FR

DELETE()

As the queue is empty, this is an exception to be handled. We can always say that deletion is attempted from an empty queue, hence not possible. The situation is known as underflow situation. Similarly when we work with fixed size list, insertion in a full list results into overflow situation. In python as we don'thave fixed size list, so don't need to bother about overflow situation. Following are the formal steps for INSERT and DELETE operations.

- **Algorithm for insertion:**

Step 1: Start

Step 2: Check FRONT and REAR value, if both the values are -1, then

FRONT and REAR are incremented by 1

other wise

Rear is incremented by one.

Step 3: Add new element at Rear. (i.e.) queue[Rear]=new element.

Step 4: Stop

- **Algorithm for deletion:**

Step 1: Start

Step 2: Check for underflow situation by checking value of Front = -1

If it is display appropriate message and stop

Otherwise

Step 3: Deleted item = queue [Front]

Step 4: If Front = Rear then Front =Rear = -1

Otherwise

Front is incremented by one

Step 5: Print "Item Deleted"

Step 6: Stop

Althogh principle operations in queue are Insert and Delete, but as a learner, we need to know the contents of queue at any point of time. To handle such requirement we will add traversal operation in our program. Following is the algorithm for same.

- **Algorithm for Display (Traversal in stack):**

1. Start
2. Store front value in I
3. Check I position value, if I value is -1 go to step 4 else go to step 5
4. Print "Queue Empty" and go to step 8
5. Print queue[I]
6. I is incremented by 1
7. Check I position value, if I value is equal to rear+1 go to step 8 else go to step 5.

Step 8: Stop

Note: In Python already we have del() and append() functions for deletion of elements at the front andaddition of elements at the rear. Hence, no need of writing special function for add and remove elements in the queue. Likewise, 'Front and Rear positions' are also not required in the Python programming while implementing queue.

Example:

Write a program to implement Queue using list.

Code: (without using class)

```
a=[]
c='y'
while c=='y':
print "1. INSERT"
print "2. DELETE "
print "3. Display"
choice=input("enter your choice ")
if (choice==1):
b=input("enter new number ")
a.append(b)
elif (choice==2):
if (a==[]):
print("Queue Empty")
else:
print "deleted element is:",a[0]
del a[0]
elif (choice==3):
l=len(a)
for i in range(0,1):
print a[i]
else:
print("wrong input")
c=raw_input("do you want to continue or not
")
```

Program: (Using class)

```
class queue:
q=[]
def insertion(self):
a=input("enter any number: ")
queue.q.append(a)
def deletion(self):
if (queue.q==[]):
print "Queue empty"
else:
print "deleted element is: ",queue.q[0]
del queue.q[0]
def display(self):
l=len(queue.q)
for i in range(0,1):
print queue.q[i]
a=queue()
c="y"
while (c=="y"):
print "1. INSERTION"
print "2. DELETION "
print "3. DISPLAY"
choice=input("enter your choice: ")
if (choice==1):
a.insertion()
elif (choice==2):
a.deletion()
elif (choice==3):
a.display()
else:
print("wrong input")
c=raw_input("do you want to continue or not
:")
```

□□□

16 SORTING (NCERT CLASS 12)

INTRODUCTION

Sorting is the process of ordering or arranging a given collection of elements in some particular order.

If the collection is of strings, we can sort it in an alphabetical order (a-z or z-a) or according to the length of the string.

BUBBLE SORT

It sorts a given list of elements by repeatedly comparing the adjacent elements and swapping them if they are unordered.

In algorithm, every iteration through each element of a list is called a pass. For a list with n elements, the bubble sort makes a total of n – 1 passes to sort the list.

In each pass, the required pairs of adjacent elements of the list will be compared. In order to arrange elements in ascending order, the largest element is identified after each pass and placed at the correct position in the list.

This can be considered as the largest element being 'bubbled up'. Hence the name Bubble sort. This sorted element is not considered in the remaining passes and thus the list of elements gets reduced in successive passes.

Algorithm : Bubble Sort

BUBBLESORT(numList, n)
Step 1: SET i = 0
Step 2: WHILE i< n REPEAT STEPS 3 to 8
Step 3: SET j = 0
Step 4: WHILE j< n-i-1,REPEAT STEPS 5 to 7
Step 5: IF numList[j] > numList[j+1] THEN
Step 6: swap(numList[j],numList[j+1])
Step 7: SET j=j+1
Step 8: SET i=i+1

Program : Implementation of bubble sort using Python.

```
def bubble_Sort(list1):
n = len(list1)
for i in range(n): # Number of passes
for j in range(0, n-i-1):
# size -i-1 because last i elements are already sorted
#in previous passes
if list1[j] > list1[j+1] :
# Swap element at jth position with (j+1)th position
list1[j], list1[j+1] = list1[j+1], list1[j]
numList = [8, 7, 13, 1, -9, 4]
bubble_Sort(numList)
print ("The sorted list is :")
for i in range(len(numList)):
print (numList[i], end=" ")
```

Output:
The sorted list is :
-9 1 4 7 8 13

SELECTION SORT

To sort a list having n elements, the selection sort makes (n-1) number of passes through the list. The list is considered to be divided into two lists — the left list containing the sorted elements, and the right list containing the unsorted elements. Initially, the left list is empty, and the right list contains all the elements.

For arranging elements in ascending order, in the first pass, all the elements in the unsorted list are traversed to find the smallest element.

The smallest element is then swapped with the leftmost element of the unsorted list. This element occupies the first position in the sorted list, and it is not considered in further passes.

In the second pass, the next smallest element is selected from the remaining elements in the unsorted list and swapped with the leftmost element of the unsorted list.

This element occupies the second position in the sorted list, and the unsorted list reduces by one element for the third pass.

This process continues until n-1 smallest elements are found and moved to their respective places. The nth element is the last, and it is already in place.

Algorithm 5.2: Selection Sort

SELECTIONSORT(numList, n)
Step 1: SET i=0
Step 2: WHILE i< n REPEAT STEPS 3 to 11
Step 3: SET min = i, flag = 0
Step 4: SET j= i+1
Step 5: WHILE j<n, REPEAT STEPS 6 to 10
Step 6: IF numList[j] < numList[min] THEN
Step 7: min = j
Step 8: flag = 1
Step 9: IF flag = 1 THEN
Step 10: swap(numList[i],numList[min])
Step 11: SET i=i+1

Program : Implementation of selection sort using Python :

```
def selection_Sort(list2):
flag = 0 #to decide when to swap
n=len(list2)
for i in range(n): # Traverse through all list elements
min = i
for j in range(i + 1, len(list2)): #the left elements
#are already sorted in previous passes
if list2[j] < list2[min]: # element at j is smaller
#than the current min element
min = j
flag = 1
if flag == 1 : # next smallest element is found
list2[min], list2[i] = list2[i], list2[min]
numList = [8, 7, 13, 1, -9, 4]
selection_Sort(numList)
print ("The sorted list is :")
for i in range(len(numList)):
print (numList[i], end=" ")
```

Output:

The sorted list is :

-9 1 4 7 8 13

INSERTIONSORT

Insertion sort is another sorting algorithm that can arrange elements of a given list in ascending or descending order. Like Selection sort, in Insertion sort also, the list is divided into two parts - one of sorted elements and another of unsorted elements. Each element in the unsorted list is considered one by one and is inserted into the sorted list at its appropriate position. In each pass, the sorted list is traversed from the backward direction to find the position where the unsorted element could be inserted. Hence the sorting method is called insertion sort.

In pass 1, the unsorted list has n-1 elements and the sorted list has a single element (say element s). The first element of the unsorted list (say element e) is compared with the element s of sorted list. If element e is smaller than element s, then element s is shifted to the right making space for inserting element e. This shifting will now make sorted list of size 2 and unsorted list of size n-2.

In pass 2, the first element (say element e) of unsorted list will be compared with each element of sorted list starting from the backward direction till the appropriate position for insertion is found. The elements of sorted list will be shifted towards right making space for the element e where it could be inserted.

This continues till all the elements in unsorted lists are inserted at appropriate positions in the sorted list. This results into a sorted list in which elements are arranged in ascending order.

Algorithm 5.3: Insertion Sort

INSERTIONSORT(numList, n)

Step 1: SET i=1

Step 2: WHILE i< n REPEAT STEPS 3 to 9

Step 3: temp = numList[i]

Step 4:SET j = i-1

Step 5: WHILE j> = 0 and numList[j]>temp,REPEAT STEPS 6 to 7

Step 6: numList[j+1] = numList[j]

Step 7: SET j=j-1

Step 8: numList[j+1] = temp #insert temp at position j

Step 9: set i=i+1

Program Implementation of insertion sort using Python :

```
def insertion_Sort(list3):
n= len(list3)
for i in range(n): # Traverse through all elements
temp = list3[i]
j = i-1
while j >=0 and temp< list3[j] :
list3[j+1] = list3[j]
j = j-1 list3[j+1] = temp
numList = [8, 7, 13, 1, -9, 4]
insertion_Sort(numList)
print ("The sorted list is :")
for i in range(len(numList)):
print (numList[i], end=" ")
```

Output:

The sorted list is :

-9 1 4 7 8 13

TIME COMPLEXITY OF ALGORITHMS

The following tips will guide us in estimating the time complexity of an algorithm:-

(I) Any algorithm that does not have any loop will have time complexity as 1 since the number of instructions to be executed will be constant, irrespective of the data size. Such algorithms are known as Constant time algorithms.

(II) Any algorithm that has a loop (usually 1 to n) will have the time complexity as n because the loop will execute the statement inside its body n number of times. Such algorithms are known as Linear time algorithms.

(III) A loop within a loop (nested loop) will have the time complexity as n2. Such algorithms are known as Quadratic time algorithms.

(IV) If there is a nested loop and also a single loop, the time complexity will be estimated on the basis of the nested loop only.

According to the above rules, all the sorting algorithms namely, bubble sort, selection sort and insertion sort have a time complexity of n^2. □□□

17 SEARCHING
(NCERT CLASS 12)

INTRODUCTION

Searching means locating a particular element in a collection of elements. Search result determines whether that particular element is present in the collection or not. If it is present, we can also find out the position of that element in the given collection. Searching is an important technique in computer science. In order to design algorithms, programmers need to understand the different ways in which a collection of data can be searched for retrieval.

LINEAR SEARCH

It is an exhaustive searching technique where every element of a given list is compared with the item to be searched (usually referred to as 'key'). So, each element in the list is compared one by one with the key. This process continues until an element matching the key is found and we declare that the search is successful. If no element matches the key and we have traversed the entire list, we declare the search is unsuccessful i.e., the key is not present in the list. This item by item comparison is done in the order, in which the elements are present in the list, beginning at the first element of the list and moving towards the last. Thus, it is also called sequential search or serial search. This technique is useful for collection of items that are small in size and are unordered.

Algorithm 6.1 : Linear Search

LinearSearch(numList, key, n)
Step 1: SET index = 0
Step 2: WHILE index < n, REPEAT Step 3
Step 3: IF numlist[index]= key THEN
PRINT "Element found at position", index+1
STOP
ELSE
index = index+1
Step 4: PRINT "Search unsuccessful"

The algorithm had to make only 1 comparison to display 'Element found at position 1'. Thus, if the key to be searched is the first element in the list, the linear search algorithm will always have to make only 1 comparison. This is the minimum amount of work that the linear search algorithm would have to do.

Program : Linear Search

```
def linear Search (list, key) :#function to perform the search
        for index in range(0, len(list)):
              in list [index] == key :    #key is present
                  return index+1   #position of key in list
    return None     #key is not in list
#end of function
list1 = []        #create an empty lsit
maximum = int(input ("How many elements in your list?"))
print ('Enter each element and press enter:')
for i in range (0, maximum):
    n = int (input ())
    list1.append(n)                  #append element to the list
print ("The List contents are :".list)

key = int (input ("Enter the number to be searched:")
position = linear Search (list1.key)
if position is non:
    print ('Number'.key. 'is not present in the list')
else
    print ('Number'.key, 'is present at position'.position)
```

Output
How many elements in your list? 4
Enter each element and press enter:
12
23
3
-45
The List contents are: [12, 23, 3, -45]
Enter the number to be searched:23
Number 23 is present at position 2

BINARY SEARCH

The binary search is a search technique that makes use of the ordering of elements in the list to quickly search a key. For numeric values, the elements in the list may be arranged either in ascending or descending order of their key values. For textual data, it may be arranged alphabetically starting from a to z or from z to a.

In binary search, the key to be searched is compared with the element in the middle of a sorted list. This could result in either of the three possibilities:

(I) the element at the middle position itself matches the key
or
(II) the element at the middle position is greater than the key
or
(III) the element at the middle position is smaller than the key.

If the element at the middle position matches the key, we declare the search successful and the searching process ends.

Algorithm 6.2: Binary Search

BinarySearch(numList, key)
Step 1: SET first = 0, last = n-1
Step 2: Calculate mid = (first+last)//2
Step 3: WHILE first <= last REPEAT Step 4
Step 4: IF numList[mid] = key
PRINT "Element found at position", " mid+1
STOP
ELSE
IF numList[mid] > key, THEN last = mid-1
ELSE first = mid + 1
Step 5: PRINT "Search unsuccessful"

Note : The binary search algorithm does not change the list. Rather, after every pass of the algorithm, the search area gets reduced by half. That is, only the index of the element to be compared with the key changes in each iteration.

Program:- Binary search:

```
def binary Search (list, key):
        first = 0
        last = len (list) - 1
        while (first <=last):
                mid = (first + last)//2
                    if list[mid] == key:
                     return mid
                elif key > list[mid]:
                     first = mid + 1
                elif key < list (mid) :
                     last = mid - 1
            return - 1
    list = []          #create an empty list
    print ("Create a lsit by entering elements in ascening order")
    print ("press enter after each element, press - 999 to stop")
    num = int (input ())
    while num! = -999:
          list.append (num)
          num = int (input ())
    n = int (input ("Enter the key to be search:")
    pos = binary Search (list 1, n)
    if (pos ! = -1):
      print (n, "is found at position",post+ 1)
    else:
      print (n, "is not found in the list")
```

Output:

Create a list by entering elements in ascending order
press enter after each element, press -999 to stop
1
3
4
5
-999
Enter the number to be searched: 4
4 is found at position 3
Second run of the program with different data:
Create a list by entering elements in ascending order
press enter after each element, press -999 to stop
12
8
3
-999
Enter the number to be searched: 4
4 is not found in the list

- **Applications of Binary Search**

(I) Binary search has numerous applications including – searching a dictionary or a telephone directory, finding the element with minimum value or maximum value in a sorted list, etc.

(II) Modified binary search techniques have far reaching applications such as indexing in databases, implementing routing tables in routers, data compression code, etc.

SEARCH BY HASHING

Hashing is a technique which can be used to know the presence of a key in a list in just one step.

A hash function takes elements of a list one by one and generates an index value for every element. This will generate a new list called the hash table. Each index of the hash table can hold only one item and the positions are indexed by integer values starting from 0. Note that the size of the hash table can be larger than the size of the list.

A simple hash function that works with numeric values is known as the remainder method. It takes an element from a list and divides it by the size of the hash table. The remainder so generated is called the hash value.

h(element) = element % size(hash table)

COLLISION

The hashing technique works fine if each element of the list maps to a unique location in the hash table. Consider a list [34, 16, 2, 26, 80]. While applying the hash function say, list [i]%10, two elements (16 and 26) would have a hash value 6. This is a problematic situation, because according to our definition, two or more elements cannot be in the same position in the list. This situation is called collision in hashing.
We must have a mechanism for placing the other items with the same hash value in the hash table. This process is called collision resolution.

If every item of the list maps to a unique index in the hash table, the hash function is called a perfect hash function. If a hash function is perfect, collision will never occur.

Apart from modulo division method, hash functions may be based on several other techniques like integer division, shift folding, boundary folding, mid-square function, extraction, radix transformation, etc.

The time taken by different hash functions may be different, but it remains constant for a particular hash function. The advantage of hashing is that the time required to compute the index value is independent of the number of items in the search list. It is to remember that the cost of computing a hash function must be small enough to make a hashing-based searching more efficient than other search methods. ❐❐❐

18 UNDERSTANDING DATA
(NCERT CLASS 12)

INTRODUCTION

In general; data is a collection of characters; numbers, and other symbols that represents values of some situations or variables. Data is plural and singular of the word data is "datum". Using computers, data are stored in electronic forms because data processing becomes faster and easier as compared to manual data processing done by people. The Information and Communication Technology (ICT) revolution led by computer, mobile and Internet has resulted in generation of large volume of data and at a very fast pace. The following list contains some examples of data that we often come across.

(I) Name, age, gender, contact details, etc., of a person
(II) Transactions data generated through banking, ticketing, shopping, etc. whether online or offline.
(III) Images, graphics, animations, audio, video .
(IV) Documents and web pages
(V) Online posts, comments and messages
(VI) Signals generated by sensors
(VII) Satellite data including meteorological data, communication data, earth observation data, etc.

IMPORTANCE OF DATA

Data allows organizations to more effectively determine the cause of problems. Data allows organizations to visualize relationships between what is happening in different locations, departments, and systems.

Besides business, following are some other scenarios where data are also stored and analysed for making decisions:

(I) The electronic voting machines are used for recording the votes cast. Subsequently, the voting data from all the machines are accumulated to declare election results in a short time as compared to manual counting of ballot papers.
(II) Scientists record data while doing experiments to calculate and compare results.
(III) Pharmaceutical companies record data while trying out a new medicine to see its effectiveness.
(IV) Libraries maintain data about books in the library and the membership of the library.
(V) The search engines give us results after analysing large volume of data available on the websites across World Wide Web (www).
(VI) Weather alerts are generated by analysing data received from various satellites.

TYPES OF DATA

As data come from different sources, they can be in different formats. Two broad categories in which data can be classified on the basis of their format are:

(I) **Structured Data:** Data which is organised and can be recorded in a well defined format is called structured data.Structureddata is usually stored in computer in a tabular (in rows and columns) format where each column represents different data for a particular parameter called attribute/ characteristic/variable and each row represents data of an observation for different attributes.
(II) **Unstructured Data :** Data which are not in the traditional row and column structure is called unstructured data. Examples of unstructured data include web pages consisting of text as well as multimedia contents (image, graphics, audio/ video)'. Other examples include text documents, business reports, books, audio/video files, social media messages.

Unstructured data are sometimes described with the help of some other data called metadata. Metadata is basically data about data. For example, we describe different parts of an email as subject, recipient, main body, attachment, etc. These are the metadata for the email data. Likewise, we can have some metadata for an image file as image size (in KB or MB), image type (for example, JPEG, PNG), image resolution, etc.

DATA COLLECTION

For processing data, we need to collect or gather data first. We can then store the data in a file or database for later use. Data collection here means identifying already available data or collecting from the appropriate sources.

DATA STORAGE

Data storage is the process of storing data on storage devices so that data can be retrieved later. There are numerous digital storage devices available in the market like, Hard Disk Drive (HDD), Solid State Drive (SSD), CD/DVD, Tape Drive, Pen Drive, Memory Card, etc.

We store data like images, documents, audios/ videos, etc. as files in our computers. Likewise, school/ hospital data are stored in data files. We use computers to add, modify or delete data in these files or process these data files to get results. However, file processing has certain limitations, which can be overcome through Database Management System (DBMS).

DATA PROCESSING

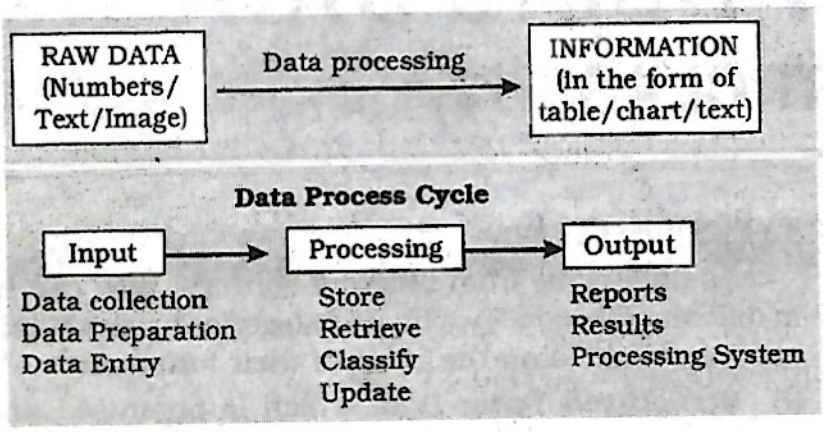


***Fig** - Steps in data processing*

STATISTICAL TECHNIQUES FOR DATA PROCESSING

Summarisation methods are applied on tabular data for its easy comprehension. Commonly used statistical techniques for data summarisation are given below:

(I) **Measures of Central Tendency:** A measure of central tendency is a single value that gives us some idea about the data. Three most common measures of central tendency are the mean, median, and mode.

(a) **Mean :** Mean is simply the average of numeric values of an attribute. Mean is also called average. Suppose there are data on weight of 60 students in a class. Instead of looking at each of the data values, we can calculate the average to get an idea about the average weight of students in that class.

Definition: Given n values $x_1, x_2, x_3,...x_n$, mean is computed as :-

$$\frac{\sum_{i}^{n} xi.}{n}$$

Mean is not a suitable choice if there are outliers in the data. To calculate mean, the outliers or extreme values should be removed from the given data and then calculate mean of the remaining data.

Note: An outlier is an exceptionally large or small value, in comparison to other values of the data. Usually, outliers are considered as error since they can influence/affect the average or other statistical calculation based on the data.

(b) **Median :** Median is also computed for a single attribute/variable at a time. When all the values are sorted in ascending or descending order, the middle value is called the Median. When there are odd number of values, then median is the value at the middle position. If the list has even number of values, then median is the average of the two middle values. Median represents the central value at which the given data is equally divided into two parts.

(c) **Mode :** Value that appears most number of times in the given data of an attribute/variable is called Mode. It is computed on the basis of frequency of occurrence of distinct values in the given data. A data set has no mode if each value occurs only once. There may be multiple modes in the data if more than one values have same highest frequency. Mode can be found for numeric as well as non-numeric data.

(II) **Measures of Variability:** The measures of variability refer to the spread or variation of the values around the mean. They are also called measures of dispersion that indicate the degree of diversity in a data set. They also indicate difference within the group. Two different data sets can have the same mean, median or mode but completely different levels of dispersion, or vice versa. Common measures of dispersion or variability are Range and Standard Deviation.

(a) **Range:** It is the difference between maximum and minimum values of the data (the largest value minus the smallest value). Range can be calculated only for numerical data. It is a measure of dispersion and tells about coverage/spread of data values.

Let M be the largest or maximum value and S is the smallest or minimum value in the data, then Range is the difference between two extreme values i.e. M – S or Maximum – Minimum.

(b) **Standard deviation:** It refers to differences within the group or set of data of a variable. Like Range, it also measures the spread of data. However, unlike Range which only uses two extreme values in the data, calculation of standard deviation considers all the given data. It is calculated as the positive square root of the average of squared difference of each value from the mean value of data. Smaller value of standard deviation means data are less spread while a larger value of standard deviation means data are more spread.

Given n values x1, x2, x3,...xn, and their mean x, the standard deviation, represented as ó (greek letter sigma) is computed as:

$$\sigma = \sqrt{\frac{\sum_{i-1}^{n} (X_1 - \overline{X})^2}{n}}$$

It is important to understand statistical techniques so that one can decide which statistical technique to use to arrive at a decision. Different programming tools are available for efficient analysis of large volumes of data. These tools make use of statistical techniques for data analysis. One such programming tool is Python and it has libraries specially built for data processing and analysis.

19 DATABASE CONCEPT
(NCERT CLASS 12)

INTRODUCTION

A Database is a collection of DATA/INFORMATION that is organized so that it can be easily accessed, managed and updated.

In Database, Data is organized into rows, columns and tables, and it is indexed to make it easier to find relevant information. It works like a container which contains the various object like Tables, Queries, Reports etc. in organized way.

A database has the following properties:

(I) A database is a representation of some aspect of the real world also called miniworld. Whenever there are changes in this mini world they are also reflected in the database.

(II) It is designed, built and populated with data for specific purpose.

(III) It can be of any size and complexity.

(IV) It can be maintained manually or it may be computerized.

Need for a Database : Data is stored in the form of files. A number of application programs are written by programmers to insert, delete, modify and retrieve data from these files. New application programs will be added to the system as the need arises.

FILE SYSTEM

A file can be understood as a container to store data in a computer. Files can be stored on the storage device of a computer system. Contents of a file can betexts, computer program code, comma separated values (CSV), etc. Likewise, pictures, audios/videos, web pages are also files.

Files stored on a computer can be accessed directly and searched for desired data. But to access data of a file through software, for example, to display monthly attendance report on school website, one has to write computer programs to access data from files.

Limitations of a File System :

Following are some of the limitations of file system:

(I) **Difficulty in Access :** Files themselves do not provide any mechanism to retrieve data. Data maintained in a file system are accessed through application programs. While writing such programs, the developer may not anticipate all the possible ways in which data may be accessed. So, sometimes it is difficult to access data in the required format and one has to write application program to access data.

(II) **Data Redundancy :** Redundancy means same data are duplicated in different places (files). Redundancy leads to excess storage use and may cause data inconsistency also.

(III) **Data Inconsistency :** Data inconsistency occurs when same data maintained in different places do not match.

(IV) **Data Isolation :** In a more complex system where data files are generated by different person at different times, files being created in isolation may be of different formats. In such case, it is difficult to write new application programs to retrieve data from different files maintained at multiple places, as one has to understand the underlying structure of each file as well.

(V) **Data Dependence :** Data are stored in a specific format or structure in a file. If the structure or format itself is changed, all the existing application programs accessing that file also need to be changed. Otherwise, the programs may not work correctly. This is data dependency. Hence, updating the structure of a data file requires modification in all the application programs accessing that file.

(VI) **Controlled Data Sharing :** Controlled access data sharing requires a request for access to the dataset to be approved. The requirements vary, but usually limit data sharing to researchers with a specific, relevant research question.It is very difficult to enforce this kind of access control in a file system while accessing files through application programs.

DATABASE MANAEMENT SYSTEM

- A DBMS refers to a software that is responsible for storing, maintaining and utilizing database in an efficient way.
- A Database along with DBMS software is called Database System.
- Example of DBMS software are Oracle, MS SQL Server, MS Access, Paradox, DB2 and MySQL etc.
- MySQL is open source and freeware DBMS.

WHY DO WE NEED DATABASE ?

(I) **To manage large chunks of data:** if size of data increases into thousands of records, it will simply create a problem to manage.Database can manage large amount of data.

(II) **Accuracy:** Through validation rule in database , data accuracy can be maintained.

(III) **Ease of updating data:** With the database, we can flexibly update the data according to our convenience. Moreover, multiple people can also edit data at same time.

(IV) **Security of data:** With databases we have security groups and privileges to restrict access.

(V) **Data integrity:** In databases, we can be assured of accuracy and consistency of data due to the built in integrity checks and access controls.

Advantages of Database System :

(I) Databases reduces Redundancy It removes duplication of data because data are kept at one place and all the application refers to the centrally maintained database.

(II) Database controls Inconsistency When two copies of the same data do not agree to each other, then it is called Inconsistency. By controlling redundancy, the inconsistency is also controlled.

(III) Database facilitate Sharing of Data Data stored in the database can be shared among several users.

(IV) Database ensures Security Data are protected against accidental or intentional disclosure to unauthorized person or unauthorized modification.

(V) Database maintains Integrity It enforces certain integrity rules to insure the validity or correctness of data. For ex. A date can't be like 31/31/2000.

Data Model- Way of data representation :

Data model is a model or presentation which shows How data is organized or stored in the database. A data is modeled by one of the following given-

- **Relational Data Model :** In this model data is organized into Relations or Tables (i.e. Rows and Columns). A row in a table represents a relationship of data to each other and also called a Tuple or Record. A column is called Attribute or Field.
- **Network Data Model" :** In this model, data is represented by collection of records and relationship among data is shown by Links.
- **Hierarchical Data Model :** In this model, Records are organized as Trees. Records at top level is called Root record and this may contains multiple directly linked children records.
- **Object Oriented Data Model :** In this model, records are represented as a objects. The collection of similar types of object is called class.

RELATIONAL DATABASE

A relational database is a collective set of multiple data sets organized by tables, records and columns. Relational database establish a well-defined relationship between database tables. Tables communicate and share information, which facilitates data searcheability, organization and reporting. A Relational database use Structured Query Language (SQL), which is a standard user application that provides an easy programming interface for database interaction.

RELATIONAL DATABASE TERMS

(I) **Relation (Table) :** A Relation or Table is Matrix like structure arranged in Rows and Columns. It has the following properties-

- **Atomicity :** Each column assigned a unique name and must have atomic(indivisible) value i.e. a value that can not be further subdivided.
- **No duplicity :** No two rows of relation will be identical i.e. in any two rows value in at least one column must be different. All items in a column are homogeneous i.e.same data type.
  v
- Ordering of rows and column is immaterial.

(II) Domain :It is collection of values from which the value is derived for a column.

(III) Tuple / Entity / Record - Rows of a table is called Tuple or Record.

(IV) Attribute/ Field- Column of a table is called Attribute or Field.

(V) Degree - Number of columns (attributes) in a table.

(VI) Cardinality - Number of rows (Records) in a table.

Three Important Properties of a Relation:-In relational data model, following three properties are observed with respect to a relation which makes a relation different from a data file or a simple table.

Property 1: imposes following rules on an attribute of the relation.

- Each attribute in a relation has a unique name.
- Sequence of attributes in a relation is immaterial.

Property 2: governs following rules on a tuple of a relation.

- Each tuple in a relation is distinct. For example, data values in no two tuples of relation ATTENDANCE can be identical for all the attributes. Thus, each tuple of a relation must be uniquely identified by its contents.
- Sequence of tuples in a relation is immaterial. The tuples are not considered to be ordered, even though they appear to be in tabular form.

Property 3: imposes following rules on the state of a relation.

- All data values in an attribute must be from the same domain (same data type).
- Each data value associated with an attribute must be atomic (cannot be further divisible into meaningful subparts). For example, GPhone of relation GUARDIAN has ten digit numbers which is indivisible.
- No attribute can have many data values in one tuple. For example, Guardian cannot specify multiple contact numbers under GPhone attribute.

- A special value "NULL" is used to represent values that are unknown or non-applicable to certain attributes. For example, if a guardian does not share his or her contact number with the school authorities, then GPhone is set to NULL (data unknown).

KEYS IN A DATABASE

Key plays an important role in relational database; it is used for identifying unique rows from table & establishes relationship among tables on need.

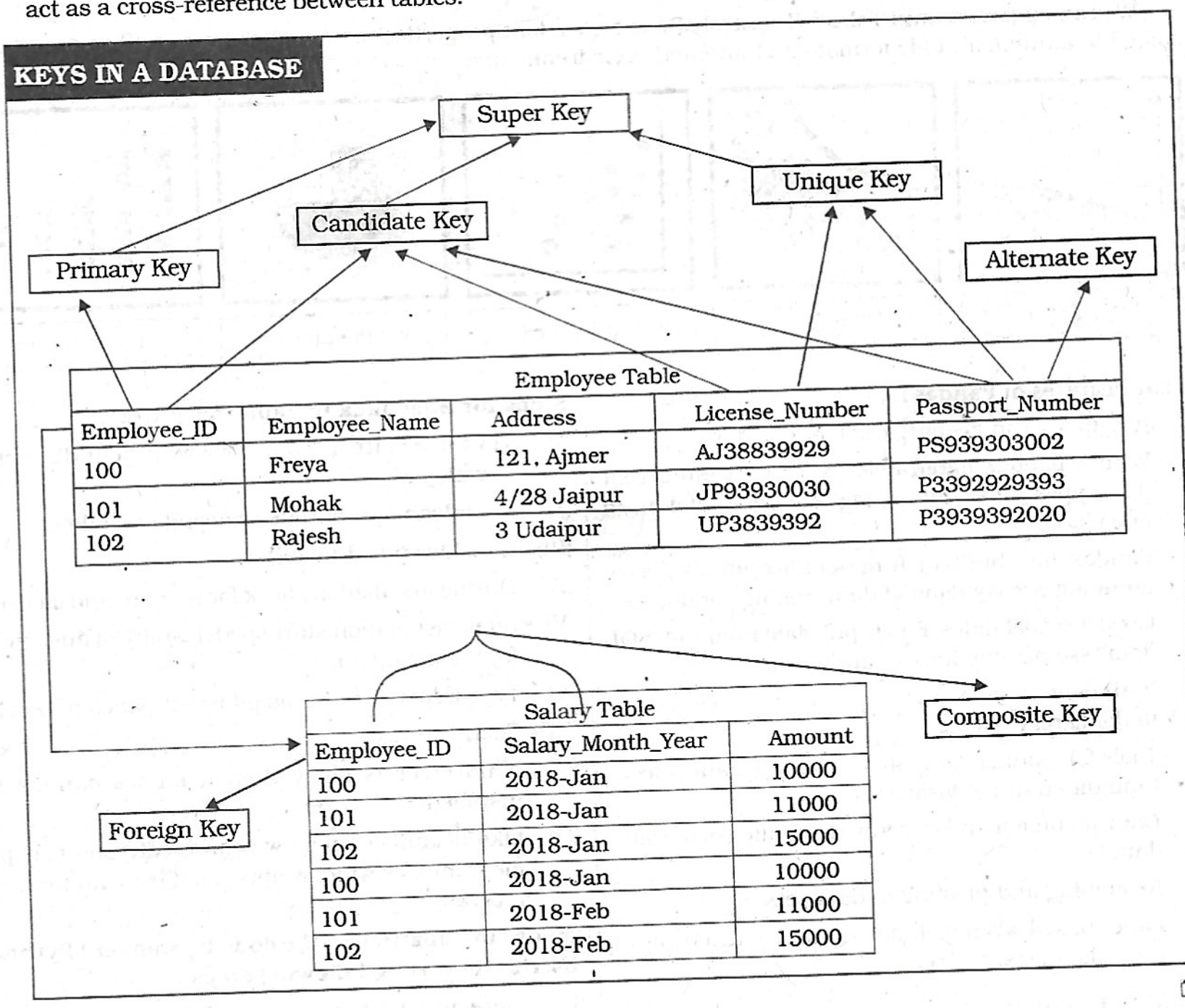
Types of keys in DBMS :

(I) **Primary Key** – A primary is a column or set of columns in a table that uniquely identifies tuples (rows) in that table.

(II) **Candidate Key** – It is an attribute or a set of attributes or keys participating for Primary Key, to uniquely identify each record in that table.

(III) **Alternate Key** – Out of all candidate keys, only one gets selected as primary key, remaining keys are known as alternate or secondary keys.

(IV) **Foreign Key** – Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.



Employee Table

Employee_ID	Employee_Name	Address	License_Number	Passport_Number
100	Freya	121, Ajmer	AJ38839929	PS939303002
101	Mohak	4/28 Jaipur	JP93930030	P3392929393
102	Rajesh	3 Udaipur	UP3839392	P3939392020

Salary Table

Employee_ID	Salary_Month_Year	Amount
100	2018-Jan	10000
101	2018-Jan	11000
102	2018-Jan	15000
100	2018-Jan	10000
101	2018-Feb	11000
102	2018-Feb	15000

❑❑❑

20 DATA HANDLING USING PANDAS (NCERT CLASS 12)

PYTHON LIBRARY – PANDAS

It is a most famous Python package for data science, which offers powerful and flexible data structures that make data analysis and manipulation easy.Pandas makes data importing and data analyzing much easier. Pandas builds on packages like NumPy and matplotlib to give us a single & convenient place for data analysis and visualization work.

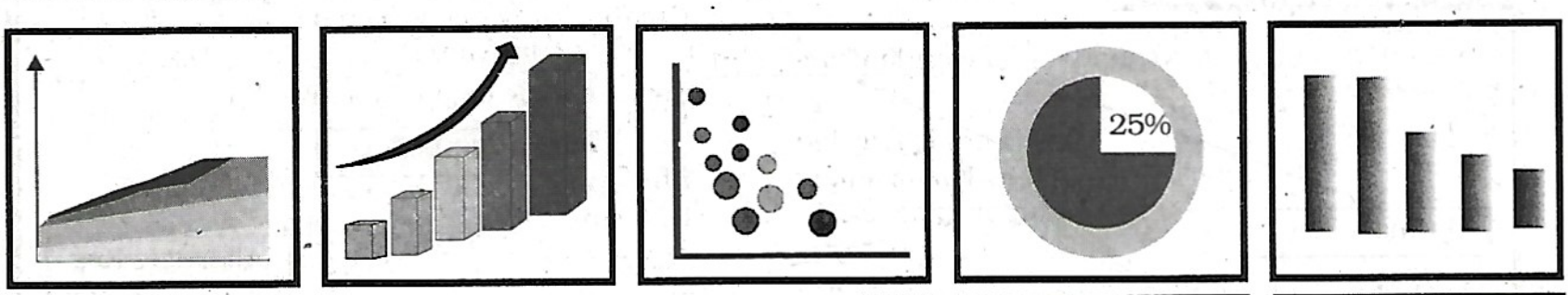
Python Library – Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.It is used to:-

(I) Develop publication quality plots with just a few lines of code.

(II) Use interactive figures that can zoom, pan, update.

We can customize and Take full control of line styles, font properties, axes properties as well as export and embed to a number of file formats and interactive environments.



Area Plot | Histogram | Scatter Plot | Pie Plot | Bar Graph

Basic Features of Pandas :

- It help a lot in keeping track of our data.
- With a pandas dataframe, we can have different data types (float, int, string, datetime, etc) all in one place.
- Pandas has built in functionality for like easy grouping & easy joins of data, rolling windows.
- Good IO capabilities; Easily pull data from a MySQL database directly into a data frame.
- With pandas, you can use patsy for R-style syntax in doing regressions.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, indexing and subsetting of large data sets.

Pandas – Installation/Environment Setup

Pandas module doesn't come bundled with Standard Python. If we install Anaconda Python package Pandas will be installed by default.

Steps for Anaconda installation & Use

(I) visit the site https://www.anaconda.com/download/

(II) Download appropriate anaconda installer.

(III) After download install it.

(IV) During installation check for set path and all user.

(V) After installation start spyder utility of anaconda from start menu.

(VI) Type import pandas as pd in left pane(temp.py) .

(VII) Then run it.

(VIII) If no error is show then it shows pandas is installed.

(IX) Like default temp.py we can create another .py file from new window option of file menu for new program.

Pandas installation can be done in Standard Python distribution,using following steps :

(I) There must be service pack installed on our computer if we are using windows.If it is not installed then we will not be able to install pandas in existing Standard Python(which is already installed).So install it first(google it).

(II) We can check it through properties option of my computer icon.

(III) Now install latest version(any one above 3.4) of python.

(IV) Now move to script folder of python distribution in command prompt (through cmd command of windows).

(V) Execute following commands in command prompt serially.

>pip install numpy

>pip install six

>pip install pandas

Wait after each command for installation Now we will be able to use pandas in standard python distribution.

(VI) Type import pandas as pd in python (IDLE) shell.

(VII) If it executed without error(it means pandas is installed on your system).

Data Structures in Pandas

Two important data structures of pandas are– Series, DataFrame.

(I) **Series :** Series is like a one-dimensional array like structure with homogeneous data. For example, the following series is a collection of integers.

78	45	12	89	56

Basic feature of series are :

- Homogeneous data
- Size Immutable
- Values of Data Mutable.

(II) **DataFrame :** DataFrame is like a two-dimensional array with heterogeneous data.

SR. No.	Admn No	Student Name	Class	Section	Gender	Date of Birth
1	001284	Nidhi Mandal	I	A	Girl	07/08/2010
2	001285	Soumyadip Bhattacharya	I	A	Boy	24/02/2011
3	001286	Shreyaang Shandilya	I	A	Boy	29/12/2010

Basic feature of DataFrame are :

- Heterogeneous data
- Size Mutable
- Data Mutable.

Pandas Series

It is like one-dimensional array capable of holding data of any type (integer, string, float, python objects, etc.). Series can be created using constructor.

Syntax :- pandas.Series(data, index, dtype, copy)

Creation of Series is also possible from – ndarray, dictionary, scalar value.

Series can be created using:-

(I) Array

(II) Dict

(III) Scalar value or constant

Pandas Series :

Create an Empty Series

e.g.

```
import pandas as pseries
s = pseries.Series()
print(s)
Output
Series([], dtype: float64)
```

Create a Series from ndarray :

Without index
e.g.

```
import pandas as pd1
import numpy as np1
data = np1.array(['a','b','c','d'])
s = pd1.Series(data)
print(s)
```

Output

```
1       a
2       b
3       c
4       d
dtype: object
```

Note: default index is starting from 0

Without index position
e.g.

```
import pandas as p1
import numpy as np1
data = np1.array(['a','b','c','d'])
s = pd1.Series(data,index=[100, 101, 102, 103])
print(s)
```

Output

```
100     a
101     b
102     c
103d    dtype:
object
```

Note: index is starting from 100

Create a Series from dict :

Eg. 1(without index)

```
import pandas as pd1
import numpy as np1
data = {'a' : 0., 'b' : 1., 'c' : 2.}
s = pd1.Series(data)
print(s)
```

Output

```
a      0.0
b      1.0
c      2.0
dtype: float64
```

Eg.2 (with index)

```
import pandas as pd1
import numpy as np1
data = {'a' : 0., 'b' : 1., 'c' : 2.}
s = pd1.Series(data,index=['b','c','d','a'])
print(s)
```

Output

```
b      1.0
c      2.0
d      NaN
a      0.0
dtype: float64
```

Create a Series from Scalar :

e.g

```
import pandas as pd1
import numpy as np1
s = pd1.Series(5, index=[0, 1, 2, 3])
print(s)
```

Output

0 5

1 5

2 5

3 5

dtype: int64

Note : here 5 is repeated for 4 times (as per no of index).

Maths operations with Series :

e.g.

```
import pandas as pd1
s = pd1.Series([1,2,3])
t = pd1.Series([1,2,4])
u=s+t #addition operation print (u)
u=s*t # multiplication operation
print (u)
```

OUTPUT

0 2

1 4

2 7

dtype: int64

0 1

1 4

2 12

dtype: int64

Head function

e.g

```
import pandas as pd1
s = pd1.Series([1,2,3,4,5],index =
['a','b','c','d','e'])
print (s.head(3))
```

Output

a.1

b. 2

c. 3

dtype: int64

Return first 3 elements

Tail function :

e.g

```
import pandas as pd1
s = pd1.Series([1,2,3,4,5],index =
['a','b','c','d','e'])
print (s.tail(3))
```

Output

c. 3

d. 4

e. 5

dtype: int64

Return last 3 elements

Accessing Data from Series with indexing and slicing

e.g.

e.g.

```
import pandas as pd1
s = pd1.Series([1,2,3,4,5],index =
['a','b','c','d','e'])
print (s[0]) # for 0 index position
print (s[:3]) # for first 3 index values
print (s[-3:]) # slicing for last 3 index values
```

Output

```
1
a.    1
b.    2
c.    3
dtype: int64  c       3
d.    4
e.    5
dtype: int64
```

Retrieve Data Using Label as (Index):

e.g.

```
import pandas as pd1
s = pd1.Series([1,2,3,4,5],index =
['a','b','c','d','e'])
print (s[['c','d']])
```

Output

c 3

d 4

dtype: int64

Retrieve Data from selection :

There are three methods for data selection:

(I) loc gets rows (or columns) with particular labels from the index.

(II) iloc gets rows (or columns) at particular positions in the index (so it only takes integers).

(III) ix usually tries to behave like loc but falls back to behaving like iloc if a label is not present in the index. ix is deprecated and the use of loc and iloc is encouraged instead.

Retrieve Data from selection :

e.g.

```
>>>s = pd.Series (np.nan,
index = [49, 48, 47, 46, 45, 1, 2, 3, 4, 5]
>>>s.iloc[:3]# slice the first three rows
49   NaN
48   NaN
47   NaN
>>>s.loc[:3] # slice up to and including label 3
49   NaN
48   NaN
47   NaN
46   NaN
45   NaN
1    NaN
2    NaN
3    NaN
```

```
>>>s.ix[:3]# the integer is in the index so
s.ix[:3] works like loc
49   NaN
48   NaN
47   NaN
46   NaN
45   NaN
1    NaN
2    NaN
3    NaN
```

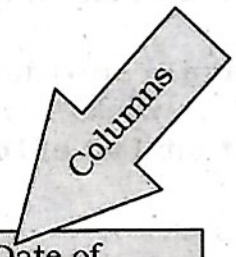
Pandas DataFrame :

It is a two-dimensional data structure, just like any table (with rows & columns).

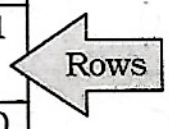
Basic Features of Data Frame

- Columns may be of different types
- Size can be changed (Mutable)
- Labeled axes (rows/columns)
- Arithmetic operations on rows and columns

Structure



SR. No.	Admn No	Student Name	Class	Section	Gender	Date of Birth
1	001284	Nidhi Mandal	I	A	Girl	07/08/2010
2	001285	Soumyadip Bhattacharya	I	A	Boy	24/02/2011
3	001286	Shreyaang Shandilya	I	A	Boy	29/12/2010



It can be created using constructor
pandas.DataFrame(data, index, columns, dtype, copy).

Create an Empty DataFrame :

e.g.

```
import pandas as pd1
df1 = pd1.DataFrame()
print(df1)
OUTPUT:
Empty
DataFrame
Columns: [ ]
Index: [ ]
Create a DataFrame from Lists :-
e.g.1
```

```
import pandas as pd1
data1 = [1,2,3,4,5]
df1 = pd1.DataFrame(data1)
print (df1)
OUTPUT:
0
0 1
1 2
2 3
3 4
4 5
e.g.2.
```

```
import pandas as pd1
data1                                    =
[['Freya',10],['Mohak',12],['Dwivedi',13]]
d                        f                       1
=pd1.DataFrame(data1,columns=['Name','Age'])
print (df1)
```

OUTPUT:

	NAME	AGE
1	Freya	10
2	Mohak	12
2	Dwivedi	13

Write below for numeric value as float

```
df1 = pd1.DataFrame(data,columns
    = ['Name','Age'],dtype=float)
```

Create a DataFrame from Dict of ndarrays / Lists :

```
e.g.1
import pandas as pd1
data1      =      {'Name':['Freya',
'Mohak'],'Age':[9,10]}
df1 = pd1.DataFrame(data1)
print (df1)
```

Output

```
Name Age
1 Freya 9
2 Mohak 10
```

```
Write below as 3rd statement in above prog
for indexing
df1  =  pd1.DataFrame(data1,  index=
['rank1','rank2', 'rank3','rank4'])
```

Create a DataFrame from List of Dicts :

```
e.g.1
import pandas as pd1
data1 = [{'x': 1, 'y': 2},{'x': 5, 'y': 4,
'z': 5}]
df1 = pd1.DataFrame(data1)
print (df1)
```

Output

```
x y z
0 1 2 NaN
1 5 4 5.0
```

Write below as 3rd stmnt in above program for indexing

```
df  =  pd.DataFrame(data,  index=['first',
'second'])
```

Create a DataFrame from Dict of Series :

```
e.g.1
import pandas as pd1
d1  =  {'one'  :  pd1.Series([1,  2,  3],
index=['a', 'b', 'c']),
'two' : pd1.Series([1, 2, 3, 4], index=['a',
'b', 'c', 'd'])}
df1 = pd1.DataFrame(d1)
print (df1)
```

Output

```
one two
a 1.0 1
b 2.0 2
c 3.0 3
d NaN 4
```

Column Selection -> print (df ['one'])

Adding a new column by passing as Series: -> df1['three']=pd1.Series([10,20,30],index=['a','b','c'])

Adding a new column using the existing columns values df1['four']=df1['one']+df1['three']

Create a DataFrame from .txt file:

Having a text file './inputs/dist.txt' as:

```
1 1 12.92
1 2 90.75
1 3 60.90
2 1 71.34
```

Pandas is shipped with built-in reader methods. For example the pandas.read_table method seems to be a good way to read (also in chunks) a tabular data file.

```
import pandas
df = pandas.read_table('./input/dists.txt',
delim_whitespace=True,
names=('A', 'B', 'C'))
```

will create a DataFrame objects with column named A made of data of type int64, B of int64 and C of float64.

Create a DataFrame from csv(comma separated value) file / import data from cvs file :

e.g.

Suppose filename.csv file contains following data

```
Date,"price","factor_1","factor_2"
2012-06-11,1600.20,1.255,1.548
2012-06-12,1610.02,1.258,1.554
import pandas as pd
# Read data from file 'filename.csv'
# (in the same directory that your python program
is based)
# Control delimiters, rows, column names with
read_csv data = pd.read_csv("filename.csv")
# Preview the first 1 line of the loaded data
data.head(1)
```

Column addition :

```
df = pd.DataFrame({"A": [1, 2, 3], "B": [4, 5, 6]})
c = [7,8,9]
df['C'] = c
```

Column Deletion :

```
del df1['one'] # Deleting the first column using DEL
function
df.pop('two') #Deleting another column using POP
function
```

Rename columns :

```
df = pd.DataFrame({"A": [1, 2, 3], "B": [4, 5, 6]})
>>> df.rename(columns={"A": "a", "B": "c"})
```

```
a c
0 1 4
1 2 5
2 3 6
```

Row Selection, Addition, and Deletion:

```
#Selection by Label
import pandas as pd1
d1 = {'one' : pd1.Series([1, 2, 3], index=['a', 'b', 'c']),
'two' : pd1.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df1
= pd1.DataFrame(d1)
print (df1.loc['b'])
Output
one 2.0
two 2.0
Name: b, dtype: float64
#Selection by integer location
import pandas as pd1
d1 = {'one' : pd1.Series([1, 2, 3], index=['a', 'b', 'c']),
'two' : pd1.Series([1, 2, 3, 4], index=['a', 'b', 'c','d'])}
df1 = pd1.DataFrame(d1)
print (df1.iloc[2])
Output
one 3.0
two 3.0
Name: c, dtype: float64
```

Slice Rows :

Multiple rows can be selected using ' : 'operator.

```
print (df1[2:4])
```

Addition of Rows :

```
import pandas as pd1
df1 = pd1.DataFrame([[1, 2], [3, 4]], columns =
['a','b'])
df2 = pd1.DataFrame([[5, 6], [7, 8]], columns =
['a','b'])
 df1 = df1.append(df2)
print (df1)
Deletion of Rows
# Drop rows with label 0
df1 = df1.drop(0)
```

Iterate over rows in a dataframe :

```
e.g.
import pandas as pd1
import numpy as np1
raw_data1 = {'name': ['freya', 'mohak'],
'age': [10, 1],
'favorite_color': ['pink', 'blue'],
'grade': [88, 92]}
df1 = pd1.DataFrame(raw_data1, columns =
['name', 'age', 'favorite_color', 'grade'])
for index, row in df1.iterrows():
print (row["name"], row["age"])
Output
freya 10
mohak 1
```

Head & Tail

head() returns the first n rows (observe the index values). The default number of elements to display is five, but you may pass a custom number. tail() returns the last n rows .e.g.

```
import pandas as pd
import numpy as np
#Create a Dictionary of series
d = {'Name':pd.Series(['Tom','James','Ricky',
'Vin','Steve', 'Smith','Jack']), 'Age':pd.Series
([25,26,25, 23,30,29,23]), 'Rating':pd.Series
([4.23,3.24,3.98, 2.56,3.20,4.6,3.8])}
#Create a DataFrame
df = pd.DataFrame(d)
print ("Our data frame is:")
 print df
print ("The first two rows of the data frame is:")
print df.head(2)
```

Indexing a DataFrame using .loc[ ] :

This function selects data by the label of the rows and columns.

```
#import the pandas library and aliasing as pd
import pandas as pd
import numpy as np
df = pd.DataFrame(np.random.randn(8, 4),
index = ['a','b','c','d','e','f','g','h'],
columns = ['A', 'B', 'C', 'D'])
#select all rows for a specific column
print df.loc[:,'A'].
```

Accessing a DataFrame with a boolean index :

In order to access a dataframe with a boolean index, we have to create a dataframe in which index of dataframe contains a boolean value that is "True" or "False".

```
# importing pandas as pd
import pandas as pd
# dictionary of lists
dict = {'name':["Mohak", "Freya", "Roshni"],
'degree': ["MBA", "BCA", "M.Tech"],
 'score':[90, 40, 80]}
# creating a dataframe with boolean index
df = pd.DataFrame(dict, index = [True, False, True])
# accessing a dataframe using .loc[] function
print(df.loc[True])
#it will return rows of Mohak and Roshni
only(matching true only)
```

import csv file in Pandas DataFrame :

```
e.g.
import pandas as pd
# Takes the file's folder
filepath = r"csv file path"
 # read the CSV file
df = pd.read_csv(filepath)
# print the first five rows
print(df.head())
```

Export Pandas DataFrame to a CSV File

```
e.g.
import pandas as pd
cars = {'Brand': ['Honda Civic','Toyota
Corolla','Ford Focus','Audi A4'],
'Price':[22000,25000,27000,35000]
 }
df = pd.DataFrame(cars, columns= ['Brand',
'Price'])
df.to_csv (r'C:\export_dataframe.csv',
index = False, header=True)
print (df)
```

□□□

21 STRUCTURED QUERY LANGUAGE (SQL) (NCERT CLASS 12)

INTRODUCTION

There are many Relational Database Management Systems (RDBMS) such as MySQL, Microsoft SQL Server, PostgreSQL, Oracle, etc. that allow us to create a database consisting of relations. These RDBMS also allow us to store, retrieve and manipulate data on that database through queries.Now, we will learn how to create, populate and query databases using MySQL.

STRUCTURED QUERY LANGUAGE (SQL)

SQL (Structured Query Language) is a standard language for accessing and manipulating databases. SQL commands are used to create, transform and retrieve information from Relational Database Management Systems and also used to create interface between user and database. By using SQL commands, one can search any data in the database and perform other functions like, create tables, add records, modify data, remove rows, drop table etc. SQL commands are used to implement the following;

(I) SQL can retrieve data from a database
(II) SQL can insert records in a database
(III) SQL can update records in a database
(IV) SQL can delete records from a database
(V) SQL can create new databases
(VI) SQL can create new tables in a database
(VII) SQL can create views in a database

- **Installing MySQL**

MySQL is an open source RDBMS software which can be easily downloaded from the official website https://dev.mysql.com/downloads. After installing MySQL, start MySQL service. The appearance of mysql> prompt means that MySQL is ready to accept SQL statements.

Following are some important points to be kept in mind while using SQL:

(I) SQL is case insensitive. For example, the column names 'salary' and 'SALARY' are the same for SQL.
(II) Always end SQL statements with a semicolon (;).
(III) To enter multiline SQL statements, we don't write ";" after the first line. We press the Enter key to continue on the next line. The prompt mysql> then changes to "->", indicating that statement is continued to the next line. After the last line, put ";" and press enter.

DATA TYPES AND CONSTRAINTS IN MySQL

Database consists of one or more relations and each relation (table) is made up of attributes (column). Each attribute has a data type. We can also specify constraints for each attribute of a relation.

- **Data type of Attribute:** Data type of an attribute indicates the type of data value that an attribute can have. It also decides the operations that can be performed on the data of that attribute. For example, arithmetic operations can be performed on numeric data but not on character data. Commonly used data types in MySQL are numeric types, date and time types, and string types as shown in following Table:-

Data type	Description
CHAR(n)	Specifies character type data of length n where n could be any value from 0 to 255. CHAR is of fixed length, means, declaring CHAR (10) implies to reserve spaces for 10 characters. If data does not have 10 characters (e.g., 'city' has four characters), MySQL fills the remaining 6 characters with spaces padded on the right.
VARCHAR(n)	Specifies character type data of length where n could be any value from 0 to 65535. But unlike CHAR, VARCHAR(n) is a variable-length data type. That is, declaring VARCHAR (30) means a maximum of 30 characters can be stored but the actual allocated bytes will depend on the length of entered string. So 'city' in VARCHAR (30) will occupy space needed to store 4 characters only.
INT	INT specifies an integer value. Each INT value occupies 4 bytes of storage. The range of unsigned values allowed in a 4 byte integer type are 0 to 4,294,967,295. For values larger than that, we have to use BIGINT, which occupies 8 bytes.
FLOAT	Holds numbers with decimal points. Each FLOAT value occupies 4 bytes.
DATE	The DATE type is used for dates in 'YYYY-MM-DD' format. YYYY is the 4 digit year, MM is the 2 digit month and DD is the 2 digit date. The supported range is '1000-01-01' to '9999-12-31'.

• **Constraints :** Constraints are the certain types of restrictions on the data values that an attribute can have. Following table lists some of the commonly used constraints in SQL. They are used to ensure correctness of data. However, it is not mandatory to define constraints for each attribute of a table.

Constraint	Description
NOT NULL	Ensures that a column cannot have NULL values where NULL means missing/ unknown/not applicable value.
UNIQUE	Ensures that all the values in a column are distinct/unique
DEFAULT	A default value specified for the column if no value is provided
PRIMARY KEY	The column which can uniquely identify each row/record in a table.
FOREGIN KEY	The column which refers to value of an attribute defined as primary key in another table

SQL FOR DATA DEFINITION

In order to be able to store data we need to first define the relation schema. Defining a schema includes creating a relation and giving name to a relation, identifying the attributes in a relation, deciding upon the datatype for each attribute and also specify the constraints as per the requirements. Sometimes, we may require to make changes to the relation schema also. SQL allows us to write statements for defining, modifying and deleting relation schemas. These are part of Data Definition Language (DDL).

• **CREATE Database :** To create a database, we use the CREATE DATABASE statement as shown in the following syntax: CREATE DATABASE databasename;

Note: In LINUX environment, names for database and tables are case-sensitive whereas in WINDOWS, there is no such differentiation. However, as a good practice, it is suggested to write database/table name in the same letter cases that were used at the time of their creation.

A DBMS can manage multiple databases on one computer. Therefore, we need to select the database that we want to use.

To know the names of existing databases, we use the statement SHOW DATABASES. From the listed databases, we can select the database to be used. Once the database is selected, we can proceed with creating tables or querying data.

• **CREATE Table :** After creating a database, we need to define relations in this database and specify attributes for each relation along with data type and constraint (if any) for each attribute. This is done using the CREATE TABLE statement.

Syntax:

```
CREATE TABLE tablename(
attributename1 datatype constraint,
attributename2 datatype constraint,
:
attributenameN datatype constraint);
```

It is important to observe the following points with respect to the CREATE TABLE statement:

(I) The number of columns in a table defines the degree of that relation, which is denoted by N.

(II) Attribute name specifies the name of the column in the table.

(III) Datatype specifies the type of data that an attribute can hold.

(IV) Constraint indicates the restrictions imposed on the values of an attribute. By default, each attribute can take NULL values except for the primary key.

Note : "," is used to separate two attributes and each statement terminates with a semi-colon (;). The arrow (->) is an interactive continuation prompt. If we enter an unfinished statement, the SQL shell will wait for us to enter the rest of the statement.

• **Describe Table :** We can view the structure of an already created table using the DESCRIBE statement or DESC statement.

Syntax:

```
DESCRIBE tablename;
```

• **ALTER Table :** After creating a table, we may realise that we need to add/remove an attribute or to modify the datatype of an existing attribute or to add constraint in attribute. In all such cases, we need to change or alter the structure (schema) of the table by using the alter statement.

(I) **Add primary key to a relation :** The following MySQL statement adds a primary key to the GUARDIAN relation:

```
mysql> ALTER TABLE GUARDIAN ADD PRIMARY
KEY (GUID);
Query OK, 0 rows affected (1.14 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(II) **Add foreign key to a relation :** Once primary keys are added, the next step is to add foreign keys to the relation (if any). Following points need to be observed while adding foreign key to a relation:

(a) The referenced relation must be already created.

(b) The referenced attribute(s) must be part of the primary key of the referenced relation.

(c) Data types and size of referenced and referencing attributes must be the same.

Syntax:

```
ALTER TABLE table_name ADD FOREIGN
KEY(attribute name)
REFERENCES referenced_table_name
(attribute name);
```

(III) **Add constraint UNIQUE to an existing attribute:**

Syntax:

ALTER TABLE table_name ADD UNIQUE (attribute name);

(IV) Add an attribute to an existing table :-Sometimes, we may need to add an additional attribute in a table. It can be done using the ADD attribute statement as shown in the following Syntax:

ALTER TABLE table_name ADD attribute name DATATYPE;

(V) **Modify datatype of an attribute :** We can change data types of the existing attributes of a table using the following ALTER statement.

Syntax:

ALTER TABLE table_name MODIFY attribute DATATYPE;

(VI) **Modify constraint of an attribute :** When we create a table, by default each attribute takes NULL value except for the attribute defined as primary key. We can change an attribute's constraint from NULL to NOT NULL using an alter statement.

Syntax:

ALTER TABLE table_name MODIFY attribute DATATYPE NOT NULL;

Note: We have to specify the data type of the attribute along with constraint NOT NULL while using MODIFY.

(VII) **Add default value to an attribute :** If we want to specify default value for an attribute, then use the following syntax:

ALTER TABLE table_name MODIFY attribute DATATYPE DEFAULT default_value;

Note: We have to specify the data type of the attribute along with DEFAULT while using MODIFY.

(VIII) **Remove an attribute :** Using ALTER, we can remove attributes from a table, as shown in the following syntax:

ALTER TABLE table_name DROP attribute;

(IX) **Remove primary key from the table :** Sometime there may be a requirement to remove primary key constraint from the table. In that case, Alter table command can be used in the following way:

Syntax:

ALTER TABLE table_name DROP PRIMARY KEY;

- **DROP Statement :**Sometimesa table in a database or the database itself needs to be removed. We can use a DROP statement to remove a database or a table permanently from the system. However, one should be very cautious while using this statement as it cannot be undone.

Syntax to drop a table:

DROP TABLE table_name;

Syntax to drop a database:

DROP DATABASE database_name;

Note: Using the DROP statement to remove a database will ultimately remove all the tables within it.

SQL FOR DATA MANIPULATION:-

Data Manipulation using a database means either insertion of new data, removal of existing data or modification of existing data in the database.

- **INSERTION of Records :** INSERT INTO statement is used to insert new records in a table. Its syntax is:

INSERT INTO tablename

VALUES(value 1, value 2,....);

Here, value 1 corresponds to attribute 1, value 2 corresponds to attribute 2 and so on.

Note that we need not to specify attribute names in the insert statement if there are exactly the same numbers of values in the INSERT statement as the total number of attributes in the table.

Caution: While populating records in a table with foreign key, ensure that records in referenced tables are already populated.

SQL FOR DATA QUERY

SQL provides efficient mechanisms to retrieve data stored in multiple tables in MySQL database (or any other RDBMS). The SQL statement SELECT is used to retrieve data from the tables in a database and is also called a query statement.

- **SELECT Statement :**

The SQL statement SELECT is used to retrieve data from the tables in a database and the output is also displayed in tabular form.

Syntax:

SELECT attribute1, attribute2, ...

FROM table_name

WHERE condition;

Here, attribute1, attribute2, ... are the column names of the table table_name from which we want to retrieve data. The FROM clause is always written with SELECT clause as it specifies the name of the table from which data is to be retrieved. The WHERE clause is optional and is used to retrieve data that meet specified condition(s).

To select all the data available in a table, we use the following select statement:

SELECT * FROM table_name;

- **QUERYING using Database OFFICE**

Organisations maintain databases to store data in the form of tables.

(I) **Retrieve selected columns :** The following query selects employee numbers of all the employees:

mysql> SELECT EmpNo FROM EMPLOYEE;

(II) **Renaming of columns :-** In case we want to rename any column while displaying the output, it can be done by using the alias 'AS'. The following query selects Employee name as Name in the output for all the employees:

```
mysql> SELECT EName as Name FROM EMPLOYEE;
```

(III) **Distinct Clause :** By default, SQL shows all the data retrieved through query as output. However, there can be duplicate values. The SELECT statement when combined with DISTINCT clause, returns records without repetition (distinct records). For example, while retrieving a department number from employee relation, there can be duplicate values as many employees are assigned to the same department. To select unique department number for all the employees, we use DISTINCT as shown below:

```
mysql> SELECT DISTINCT DeptId FROM EMPLOYEE;
```

(IV) **WHERE Clause :** The WHERE clause is used to retrieve data that meet some specified conditions. In the OFFICE database, more than one employee can have the same salary.

Note :- "=" operator is used in the WHERE clause. Other relational operators (<, <=, >, >=, !=) can be used to specify such conditions. The logical operators AND, OR, and NOT are used to combine multiple conditions.

(V) **Membership operator IN :** The IN operator compares a value with a set of values and returns true if the value belongs to that set. The above query can be rewritten using IN operator as shown below:

```
mysql> SELECT
```

(VI) **ORDER BY Clause :** ORDER BY clause is used to display data in an ordered form with respect to a specified column. By default, ORDER BY displays records in ascending order of the specified column's values. To display the records in descending order, the DESC (means descending) keyword needs to be written with that column.

(VII) **Handling NULL Values :** SQL supports a special value called NULL to represent a missing or unknown value. For example, the Gphone column in the GUARDIAN table can have missing value for certain records. Hence, NULL is used to represent such unknown values. It is important to note that NULL is different from 0 (zero).

Also, any arithmetic operation performed with NULL value gives NULL. For example: 5 + NULL = NULL because NULL is unknown hence the result is also unknown. In order to check for NULL value in a column, we use IS NULL operator.

(VIII) **Substring pattern matching :** Many a times we come across situations where we do not want to query by matching exact text or value. Rather, we are interested to find matching of only a few characters or values in column values.

The LIKE operator makes use of the following two wild card characters:

- % (per cent)- used to represent zero, one, or multiple characters.
- _ (underscore)- used to represent exactly a single character .

DATA UPDATION AND DELETION

Updation and deletion of data are also part of SQL Data Manipulation Language (DML).

- **Data Updation** :-This command is used to implement modification of the data values.

Syntax:

UPDATE <table name>

SET <column name1>=new value, <column name>=new value etc

[WHERE <condition>];

- **Data Deletion** :-DELETE statement is used to delete/remove one or more records from a table.

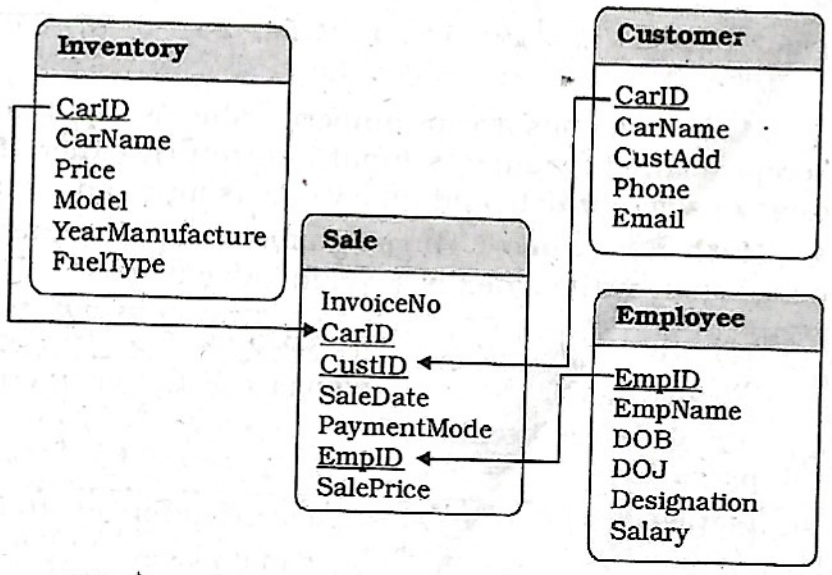
Syntax:

DELETE FROM table_name

WHERE condition;

FUNCTIONS IN SQL

In this section, we will understand how to use single row functions, multiple row functions, group records based on some criteria, and working on multiple tables using SQL.



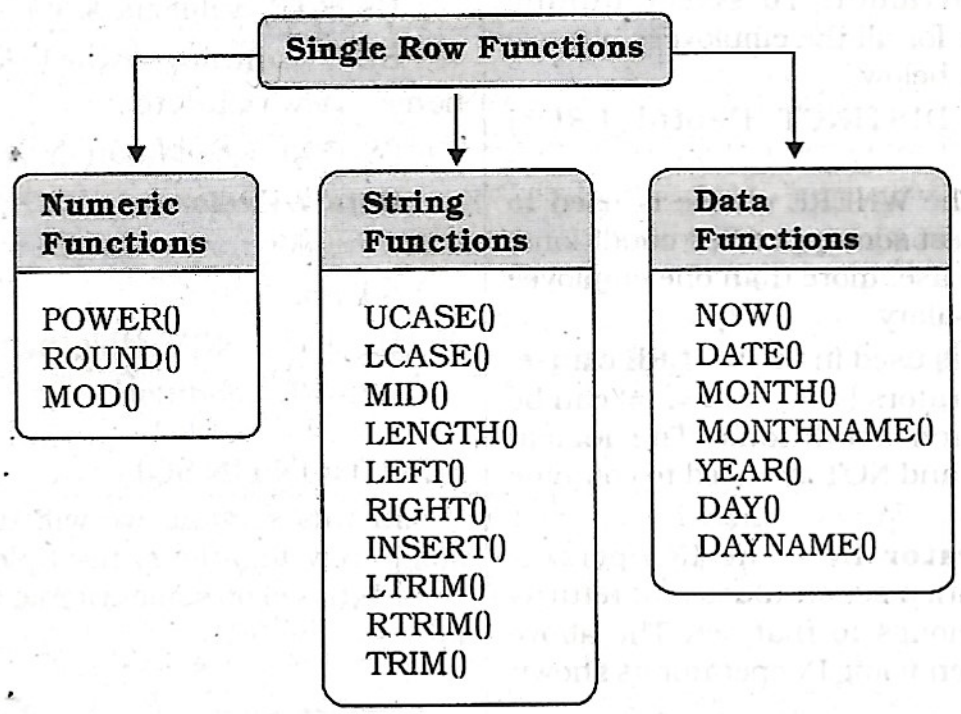
Let us create a database called CARSHOWROOM having the schema as shown in Figure above. It has the following four relations:

1) **INVENTORY :** Stores name, price, model, year of manufacturing, and fuel type for each car in inventory of the showroom,
2) **CUSTOMER :** Stores customer id, name, address, phone number and email for each customer,
3) **SALE :** Stores the invoice number, car id, customer id, sale date, mode of payment, sales person's employee id and selling price of the car sold,
4) **EMPLOYEE :** Stores employee id, name, date of birth, date of joining, designation and salary of each employee in the showroom.

We know that a function is used to perform some particular task and it returns zero or more values as a result. Functions are useful while writing SQL queries also. Functions can be applied to work on single or multiple records (rows) of a table. Depending on their application in one or multiple rows, SQL functions are categorised as Single Row functions and Aggregate functions.

- **Single Row Functions :**These are also known as Scalar functions. Single row functions are applied on a single value and return a single value.

Below Figure lists different single row functions under three categories — Numeric (Math), String, Date and Time.



Math Functions accept numeric value as input and return a numeric value as a result. String Functions accept character value as input and return either character or numeric values as output. Date and Time functions accept date and time value as input and return numeric or string or Date and Time as output.

- **Math Functions :** Three commonly used numeric functions are POWER(), ROUND() and MOD(). Their usage along with syntax is given in Table below:-

Function	Description	Example with output
POWER(X,Y) can also be written as POW(X,Y)	Calculates X to the power Y.	mysql> SELECT POWER(2,3); Output: 8

ROUND(N,D)	Rounds off number N to D number of decimal places.
	Note: If D=0, then it rounds off the number to the nearest integer.
		mysql>SELECT ROUND(2912.564, 1);
		Output:
		2912.6
		mysql> SELECT ROUND(283.2);
		Output:
		283

MOD(A, B)	Returns the remain der after dividing number A by number B.	mysql> SELECT MOD(21, 2); Output: 1

- **String Functions :** String functions can perform various operations on alphanumeric data which are stored in a table. They can be used to change the case (uppercase to lowercase or vice-versa), extract a substring, calculate the length of a string and so on.String functions and their usage are shown in Table below:-

Function	Description	Example with output
UCASE(string) OR UPPER(string)	converts string into uppercase.	mysql> SELECT UCASE ("Informatics Practices"); Output: INFORMATICS PRACTICES
LOWER(string) OR LCASE(string)	converts string into lowercase.	mysql> SELECT LOWER ("Informatics Practices"); Output: informatics practices

MID(string, pos, n)
OR
SUBSTRING
(string, pos, n)
OR
SUBSTR
(string, pos, n)
	Returns a substring of size n starting from the specified position (pos) of the string. If n is not specified, it returns the substring from the position pos till end of the string.	mysql> SELECT MID("Informatics", 3, 4);
		Output: form
		mysql> SELECT MID('Informatics',7);
		Output:
		atics

LENGTH(string)	Return the number of characters in the specified string.	mysql> SELECT LENGTH("Informatics"); Output: 11

LEFT(string, N)
 	
RIGHT(string, N)
	Returns N number of characters from the left side of the string.
	
	Returns N number of characters from the right side of the string.
		mysql> SELECT LEFT("Computer", 4);
		Output:
		Comp
		mysql> SELECT RIGHT("SCIENCE", 3);
		NCE

INSTR (string, substring)	Returns the position of the first occurrence of the substring in the given string. Returns 0, if the substring is not present in the string.	mysql> SELECT INSTR("Informatics", "ma"); Output: 6
LTRIM(string)	Returns the given string after removing leading white space characters.	mysql> SELECT LENGTH(" DELHI"), LENGTH(LTRIM(" DELHI")); Output: +---+---+ | 7 | 5 | +---+---+ 1 row in set (0.00 sec)
RTRIM(string)	Returns the given string after removing trailing white space characters.	mysql>SELECT LENGTH("PEN ") LENGTH(RTRIM("PEN)); Output: +---+---+ | 5 | 3 | +---+---+ 1 row in set (0.00 sec)
TRIM(string)	Returns the given string after removing both leading and trailing white space characters.	mysql> SELECT LENGTH(" MADAM "), LENGTH(TRIM(" MADAM ")); Output: +---+---+ | 9 | 5 | +---+---+ 1 row in set (0.00 sec)

- **Date and Time Functions :** There are various functions that are used to perform operations on date and time data. Some of the operations include displaying the current date, extracting each element of a date (day, month and year), displaying day of the week and so on. Below Table explains various date and time functions.

Function	Description	Example with output
NOW()	It returns the current system date and time.	mysql> SELECT NOW(); Output: 2019-07-11 19:41:17
DATE() MONTH(date)	It returns the date part from the given date/time expression. It returns the month in numeric form from the date.	mysql> SELECT DATE(NOW()); Output: 2019-07-11 mysql> SELECT MONTH(NOW()); Output: 7
MONTHNAME(date)	It returns the month name from the specified date.	mysql> SELECT MONTHNAME("2003-11-28"); Output: November
YEAR(date)	It returns the year from the date.	mysql> SELECT YEAR("2003-10- 03"); Output: 2003
DAY(date)	It returns the day part from the date.	mysql> SELECT DAY("2003-03- 24"); Output: 24
DAYNAME(date)	It returns the name of the day from the date.	mysql> SELECT DAYNAME("2019-07-11"); Output: Thursday

- **Aggregate functions :** These functions are also called Multiple Row functions. These functions work on a set of records as a whole and return a single value for each column of the records on which the function is applied.

Differences between Single and Multiple Row Functions :

Single Row Function	Multiple Row Function
1. It operates on a single row at a time	1. It operates on groups of rows.
2. It returns one result per row.	2. It returns one result for a group of rows.
3. It can be used in Select, Where, and Order by clause.	3. It can be used in the select clause only.
4. Math, String and Date functions are examples of single row functions.	4. Max(), Min(), Avg(), Sum(), Count() and Count(*) are examples of multiple row functions.

Aggregate Functions in SQL :

Function	Description	Example with output
MAX(column)	Returns the largest value from the specified column.	mysql> SELECT MAX(Price) FROM INVENTORY; Output: 673112.00
MIN(column)	Returns the smallest value from the specified column.	mysql> SELECT MIN(Price) FROM INVENTORY; Output: 355205.00
AVG(column)	Returns the average of the values in the specified column.	mysql> SELECT AVG(Price) FROM INVENTORY; Output: 576091.625000

SUM(column)	Returns the sum of the values for the specified column.	mysql> SELECT SUM(Price) FROM INVENTORY; Output: 4608733.00
COUNT(*)	Returns the number of records in a table. **Note:** In order to display the number of records that matches a particular criteria in the table, we have to use COUNT(*) with WHERE clause.	mysql> SELECT COUNT(*) from MANAGER; +----------+ | count(*) | +----------+ | 4 | +----------+ 1 row in set (0.00 sec)
COUNT(column)	Returns the number of values in the specified column ignoring the NULL values. **Note:** In this example, let us consider a MANAGER table having two attributes and four records.	mysql> SELECT * from MANAGER; +------ +----------+ | MNO | MEMNAME | +------ +----------+ | 1 | AMIT | | 2 | KAVREET | | 3 | KAVITA | | 4 | NULL | +------+----------+ 4 rows in set (0.00 sec) mysql> SELECT COUNT(MEMNAME) FROM MANAGER; +----------------+ | COUNT(MEMNAME) | +----------------+ | 3 | +----------------+ 1 row in set (0.01 sec)

GROUP BY CLAUSE IN SQL

At times we need to fetch a group of rows on the basis of common values in a column. This can be done using a group by clause. It groups the rows together that contains the same values in a specified column. We can use the aggregate functions (COUNT, MAX, MIN, AVG and SUM) to work on the grouped values. HAVING Clause in SQL is used to specify conditions on the rows with Group By clause.

OPERATIONS ON RELATIONS

We can perform certain operations on relations like Union, Intersection and Set Difference to merge the tuples of two tables. These three operations are binary operations as they work upon two tables.

These operations can only be applied if both the relations have the same number of attributes and corresponding attributes in both tables have the same domain.

- **UNION (∪) :** This operation is used to combine the selected rows of two tables at a time. If some rows are same in both the tables, then result of the Union operation will show those rows only once.
- **INTERSECT (∩) :** Intersect operation is used to get the common tuples from two tables and is represented by symbol ?.
- **MINUS (-) :** This operation is used to get tuples/rows which are in the first table but not in the second table and the operation is represented by the symbol - (minus).
- **Cartesian Product (X) :** Cartesian product operation combines tuples from two relations. It results in all pairs of rows from the two input relations, regardless of whether or not they have the same values on common attributes. It is denoted as 'X'.

The degree of the resulting relation is calculated as the sum of the degrees of both the relations under consideration. The cardinality of the resulting relation is calculated as the product of the cardinality of relations on which cartesian product is applied.

USING TWO RELATIONS IN QUERY

Till now we have written queries in SQL using a single relation only. In this section, we will learn to write queries using two relations.

- **Cartesian product on two tables :** When more than one table is to be used in a query, then we must specify the table names by separating commas in the FROM clause. On execution of such a query, the DBMS (MySql) will first apply cartesian product on specified tables to have a single table.
- **JOIN on two tables :** JOIN operation combines tuples from two tables on specified conditions. This is unlike cartesian product which make all possible combinations of tuples. While using the JOIN clause of SQL, we specify conditions on the related attributes of two tables within the FROM clause. Usually, such an attribute is the primary key in one table and foreign key in another table.

Following are some of the points to be considered while applying JOIN operations on two or more relations:

(I) If two tables are to be joined on equality condition on the common attribute, then one may use JOIN with ON clause or NATURAL JOIN in FROM clause. If three tables are to be joined on equality condition, then two JOIN or NATURAL JOIN are required.

(II) In general, N-1 joins are needed to combine N tables on equality condition.

(III) With JOIN clause, we may use any relational operators to combine tuples of two tables.

□□□

22 COMPUTER NETWORK
(NCERT CLASS 12)

INTRODUCTION

We often need peripheral devices and data to be shared among various computers. In fact, in your school'scomputer lab, you must have seen one printer which is connected to only one computer, serving to the needs of all the computers in the lab. How does this happen? This happens because all your lab's computers and peripherals are forming a network. They are interconnected with each other enabling you to send and receive data from one computer to another. Hence it can be said that two computers are interconnected if they are able to exchange information.

A network is any collection of independent computers that communicate with one another over ashared network medium. In simple terms, a computer network is a collection of two or more computers linked together for the purpose of sharing information and resources. When these computers are joined in a network, people can share files and peripherals such as modems, printers, backup drives, or CD-ROM drives. Each computer on the network is called a node and hence, a network is a series of points or nodes interconnected by communication paths (transmission media). A network can be as small and simple as two computers that share a printer or as complex as the world's largest network, the Internet. When networks at multiple locations are connected using services available from phone companies, people can send e-mail, share links to the global Internet, or conduct video conferences in real time with other remote users. As companies rely on applications like electronic mail and database management for core business operations, computer networking becomes increasingly more important.

NEED FOR NETWORKING

Why has networking evolved as an indispensible part of technology today? To find answer to thisquestion, let us have a look at the advantages of networking:

- Resource sharing - files and peripherals:-

i) Sharing of files and software

A network enables users to share data files with each other. For e.g. different departments of an organization may be seperated physically, being at distant places, but their data could be stored on a central computer which can be accessed by computers located in different departments. In this way latest data can be made available at all times to all users. Files and folders can be backed up to local or remote shares. Software can be installed centrally rather than on each machine which proves to be much cheaper than buying licenses for every machine:

ii) Sharing Peripherals

Laser printers and large storage media are quite expensive. Networks enable us to share such resources and hence reduce the operational cost of any organization. For e.g. a company with about fifty computers can share resources such as printers, scanners, hard disks etc, thereby reducing the cost considerably. Even fax systems can be integrated within a network. Audio and video content can also be streamed to multiple devices.

iii) Sharing storage

On a network, one can access data from any machine. Hence storage can be distributed and thus database load can be shared on the network. This even proves to be cost effective. A file can even have copies on two or three machines.

- **Improving communication**

A computer network can provide a powerful, fast and reliable communication medium among the users of various computers on the network. Using a network, it is easy for two or more people, of saydifferent departments or different branches to prepare a presentation together in spite of being located in different cities. In fact the best example in this context can be of the use of internet (discussed later in the chapter). With the help of internet we can communicate efficiently and easily via email, instant messaging, chat rooms, telephone, video telephone calls, and video conferencing.

- **Access to remote database:-**

This is another major use of network. It is easy for an average person to access any remote database, say for example airline reservations and thereby book tickets. Likewise databases of trains, online universities, hotels etc can be accessed as per the requirement. Remote-control/access programs can be used to troubleshoot problems or show new users how to perform a task.

Like two sides of a coin, networking also has some disadvantages associated with it. The disadvantages are as follows:

(I) Threat to data

A computer network may be used by unauthorized users to steal or corrupt the data and even to deploy computer viruses or computer worms on the network. File security has to be taken care of especially if connected to WANs.

(II) Difficult to set up

The systems on a network are more sophisticated and complex to run. Sometimes setting up a network, especially larger networks may turn out to be a difficult task. If systems are badly managed services can become unusable. In addition to this, larger networks may also be very costly to set up and maintain. Often a specialist may be needed to run and maintain the network.

EVOLUTION OF NETWORKING

Networking started way back in 1969 with the development of the first network called the ARPANET. The U.S. department of defence sponsored a project named ARPANET (Advanced Research Projects Agency Network) whose goal was to connect computers at different universities and U.S. defence. Soon engineers, scientists, students and researchers who were part of this system began exchanging data and messages on it. Gradually they could play long distance games and also socialize with people. Hence ARPANET expanded rapidly. In mid 80s, the National Science Foundation created a new high capacity network called NSFnet which allowed only academic research on its network. So many private companies built their own networks, which were later interconnected along with ARPANET and NSFnet to form Internet - a network formed by linking two or more networks.

TYPES OF NETWORKS

A network may be a small group of interlinked computers to a chain of a few hundred computers ofdifferent types (for example personal computers, minicomputers , mainframes etc.). These computers may be localised or spread around the world. Thus networks vary in terms of their size and complexity. Various types of networks are discussed below:

PAN (Personal Area Network)

A Personal Area Network is a computer network organized around an individual person. Personal area networks typically involve a mobile computer, a cell phone and/or a handheld computing device such as a PDA. You can use these networks to transfer files including email and calendar appointments, digital photos and music. Personal area networks can be constructed with cables or be wireless. USB and FireWire technologies often link together a wired PAN, while wireless PANs typically use bluetooth or sometimes infrared connections. Bluetooth PANs generally cover a range of less than 10 meters (about 30 feet). PANs can be viewed as a special type (or subset) of local area network (LAN) that supports one person instead of a group.

LAN (Local Area Network)

In a LAN, network devices are connected over a relatively short distance. They are generally privately owned networks within a single building or campus, of up to a few kilometres in size. LANs can be small, linking as few as three computers, but often link hundreds of computers used by thousands of people. This means that many users can share expensive devices, such as laser printers, as well as data on the LAN. Users can also use the LAN to communicate with each other, by sending mails or engaging in chat sessions.

Nowadays we also have WLAN (Wireless LAN) which is based on wireless network. One LAN can even be connected to other LANs over any distance via telephone lines and radio waves. However there is also a limit on the number of computers that can be attached to a single LAN. The development of standard networking protocols and media has resulted in worldwide proliferation of LANs throughout business and educational organizations.

MAN (Metropolitan Area Network)

This is basically a bigger version of LAN and normally uses similar technology. It might cover fewbuildings in a city and might either be private or public. This is a network which spans a physical area (in the range of 5 and 50 km diameter) that is larger than a LAN but smaller than a WAN. MANs are usually characterized by very high-speed connections using optical fibres or other digital media and provides uplink services to wide area networks (WANs) and the Internet. For example in a city, a MAN, which can support both data and voice might even be related to local cable television network.

The MAN, its communications links and equipment are generally owned by either a consortium of users or by a single network provider who sells the service to the users. Since MAN adopts technologies from both LAN and WAN to serve its purpose, it is also frequently used to provide a shared connection to other networks using a link to a WAN.

WAN (Wide Area Network)

As the term implies, WAN spans a large geographical area, often a country or a continent and uses various commercial and private communication lines toconnect computers. Typically, a WAN combines multiple LANs that are geographically separated. This is accomplished by connecting the different LANs using services such as dedicated leased phone lines, dial-up phone lines, satellite links, high speed fibre optic cables and data packet carrier services. Wide area networking can be as simple as a modem and remote access server for employees to dial into, or it can be as complex as hundreds of branch offices globally linked using special routing protocols and filters to minimize the expense of sending data sent over vast distances.

Let us take an example of your local telephone exchange which is a part of WAN. The computers at

each telephone exchange connect to other exchanges to allow you to talk to people all over the world. The internet is the largest WAN , spanning the entire earth.

NETWORK DEVICES

To communicate data through different transmission media and to configure networks with different functionality, we require different devices like Modem, Hub, Switch, Repeater, Router, Gateway, etc.

- **Modem** : Modem stands for 'MOdulator DEModulator'. It refers to a device used for conversion between analog signals and digital bits. We know computers store and process data in terms of 0s and 1s. However, to transmit data from a sender to a receiver, or while browsing the internet, digital data are converted to an analog signal and the medium (be it free-space or a physical media) carries the signal to the receiver. There are modems connected to both the source and destination nodes. The modem at the sender's end acts as a modulator that converts the digital data into analog signals. The modem at the receiver's end acts as a demodulator that converts the analog signals into digital data for the destination node to understand.
- **Ethernet Card :** Ethernet card, also known as Network Interface Card (NIC card in short) is a network adapter used to set up a wired network. It acts as an interface between computer and the network. It is a circuit board mounted on the motherboard of a computer. The Ethernet cable connects the computer to the network through NIC. Ethernet cards can support data transfer between 10 Mbps and 1 Gbps (1000 Mbps). Each NIC has a MAC address, which helps in uniquely identifying the computer on the network.
- **RJ45 :** RJ 45 or Registered Jack-45 is an eight-pin connector that is used exclusively with Ethernet cables for networking. It is a standard networking interface that can be seen at the end of all network cables. Basically, it is a small plastic plug that fits into RJ-45 jacks of the Ethernet cards present in various computing devices.
- **Repeater :** Data are carried in the form of signals over the cable. These signals can travel a specified distance (usually about 100 m). Signals lose their strength beyond this limit and become weak. In such conditions, original signals need to be regenerated.

A repeater is an analog device that works with signals on the cables to which it is connected. The weakened signal appearing on the cable is regenerated and put back on the cable by a repeater.

- **Hub :** An Ethernet hub is a network device used to connect different devices through wires. Data arriving on any of the lines are sent out on all the others. The limitation of Hub is that if data from two devices come at the same time, they will collide.
- **Switch :** A switch is a networking device that plays a central role in a Local Area Network (LAN). Like a hub, a network switch is used to connect multiple computers or communicating devices. When data arrives, the switch extracts the destination address from the data packet and looks it up in a table to see where to send the packet. Thus, it sends signals to only selected devices instead of sending to all. It can forward multiple packets at the same time. A switch does not forward the signals which are noisy or corrupted. It drops such signals and asks the sender to resend it.

Ethernet switches are common in homes/offices to connect multiple devices thus creating LANs or to access the Internet.

- **Router :** A router is a network device that can receive the data, analyse it and transmit it to other networks. A router connects a local area network to the internet. Compared to a hub or a switch, a router has advanced capabilities as it can analyse the data being carried over a network, decide/alter how it is packaged, and send it to another network of a different type. For example, data has been divided into packets of a certain size. Suppose these packets are to be carried over a different type of network which cannot handle bigger packets. In such a case, the data is to be repackaged as smaller packets and then sent over the network by a router.

A router can be wired or wireless. A wireless router can provide Wi-Fi access to smartphones and other devices. Usually, such routers also contain some ports to provide wired Internet access. These days, home Wi-Fi routers perform the dual task of a router and a modem/ switch. These routers connect to incoming broadband lines, from ISP (Internet Service Provider), and convert them to digital data for computing devices to process.

- **Gateway :** As the term "Gateway" suggests, it is a key access point that acts as a "gate" between an organisation's network and the outside world of the Internet. Gateway serves as the entry and exit point of a network, as all data coming in or going out of a network must first pass through the gateway in order to use routing paths. Besides routing data packets, gateways also maintain information about the host network's internal connection paths and the identified paths of other remote networks. If a node from one network wants to communicate with a node of a foreign network, it will pass the data packet to the gateway, which then routes it to the destination using the best possible route.

For simple Internet connectivity at homes, the gateway is usually the Internet Service Provider that provides access to the entire Internet. Generally, a router is configured to work as a gateway device in

computer networks. But a gateway can be implemented completely in software, hardware, or a combination of both. Because a network gateway is placed at the edge of a network, the firewall is usually integrated with it.

NETWORK TOPOLOGIES

The arrangement of computers and other peripherals in a network is called its topology. Common network topologies are Mesh, Ring, Bus, Star and Tree.

- **Mesh Topology :** In this networking topology, each communicating device is connected with every other device in the network. Such a network can handle large amounts of traffic since multiple nodes can transmit data simultaneously. Also, such networks are more reliable in the sense that even if a node gets down, it does not cause any break in the transmission of data between other nodes. This topology is also more secure as compared to other topologies because each cable between two nodes carries different data. However, wiring is complex and cabling cost is high in creating such networks and there are many redundant or unutilised connections.
- **Ring Topology :** In ring topology each node is connected to two other devices, one each on either side, as shown in Figure 10.16. The nodes connected with each other thus forms a ring. The link in a ring topology is unidirectional. Thus, data can be transmitted in one direction only (clockwise or counterclockwise).
- **Bus Topology :** In bus topology, each communicating device connects to a transmission medium, known as bus. Data sent from a node are passed on to the bus and hence are transmitted to the length of the bus in both directions. That means, data can be received by any of the nodes connected to the bus.

In this topology, a single backbone wire called bus is shared among the nodes, which makes it cheaper and easier to maintain. Both ring and bus topologies are considered to be less secure and less reliable.

- **Star Topology :** In star topology, each communicating device is connected to a central node, which is a networking device like a hub or a switch.

Star topology is considered very effective, efficient and fast as each device is directly connected with the central device. Although disturbance in one device will not affect the rest of the network, any failure in a central networking device may lead to the failure of complete network.

The central node can be either a broadcasting device means data will be transmitted to all the nodes in the network, or a unicast device means the node can identify the destination and forward data to that node only.

- **Tree or Hybrid Topology :** It is a hierarchical topology, in which there are multiple branches and each branch can have one or more basic topologies like star, ring and bus. Such topologies are usually realised in WANs where multiple LANs are connected. Those LANs may be in the form of a ring, bus or star.

In this type of network, data transmitted from source first reaches the centralised device and from there the data passes through every branch where each branch can have links for more nodes.

IDENTIFYING NODES IN A NETWORKED COMMUNICATION

Each node in a network should be uniquely identified so that a network device can identify the sender and receiver and decide a routing path to transmit data. Let us explore further and know how each node is distinguished in a network.

- **MAC Address :** MAC stands for Media Access Control. The MAC address, also known as the physical or hardware address, is a unique value associated with a network adapter called a NIC. The MAC address is engraved on NIC at the time of manufacturing and thus it is a permanent address and cannot be changed under any circumstances. The machine on which the NIC is attached, can be physically identified on the network using its MAC address.

Each MAC address is a 12-digit hexadecimal numbers (48 bits in length), of which the first six digits (24 bits) contain the manufacturer's ID called Organisational Unique Identifier (OUI) and the later six digits (24 bits) represents the serial number assigned to the card by the manufacturer.

- **IP Address :** IP address, also known as Internet Protocol address, is also a unique address that can be used to uniquely identify each node in a network. The IP addresses are assigned to each node in a network that uses the Internet Protocol for communication. Thus, if we know a computer's IP address, we can communicate with that computer from anywhere in the world. However, unlike MAC address, IP address can change if a node is removed from one network and connected to another network.

The initial IP Address called version 4 (IPV4 in short), is a 32 bit numeric address, written as four numbers separated by periods, where each number is the decimal (base-10) representation for an 8-bit binary (base-2) number and each can take any value from 0 - 255.

With more and more devices getting connected to the Internet, it was realised that the 32-bit IP address will not be sufficient as it offers just under 4.3 billion unique addresses. Thus, a 128 bits IP address, called IP version 6 (IPV6 in short) was proposed. An IPv6 address is represented by eight groups of hexadecimal (base-16) numbers separated by colons.

INTERNET, WEB AND THE INTERNET OF THINGS

The Internet is a system of linked networks that are worldwide in scope and facilitate data communicationservices such as remote login, file transfer, electronic mail, the World Wide Web and news groups. The Internet is made up of many networks each run by a different companies and are interconnected atpeering points. It is really a network of networks spread across the globe, all of which are connected to each other.This super network is a glorified WAN in many respects. It connects many smaller networks together and allows all the computers to exchange information with each other through a common set of rules for communication. These rules are called protocols and the internet uses Transmission Control Protocol/Internet Protocol (TCP/IP). Programs such as web browsers, File Transfer Protocol (FTP) clients, and email clients are some of the most common ways through which the users work on the Internet.

With the meteoric rise in demand for connectivity, the Internet has become a communications highway for millions of users. The Internet was initially restricted to military and academic institutions, but now it is a full-fledged conduit for any and all forms of information and commerce. Internet websites now provide personal, educational, political and economic resources to every corner of the planet.

- **The World Wide Web (WWW) :** The World Wide Web (WWW) or web in short, is an ocean of information, stored in the form of trillions of interlinked web pages and web resources. The resources on the web can be shared or accessed through the Internet.

Sir Tim Berners-Lee — a British computer scientist invented the revolutionary World Wide Web in 1990 by defining three fundamental technologies that lead to creation of web:

(I) **HTML** – HyperText Markup Language. It is a language which is used to design standardised Web Pages so that the Web contents can be read and understood from any computer. Basic structure of every webpage is designed using HTML.

(II) **URI** – Uniform Resource Identifier. It is a unique address or path for each resource located on the web. It is also known as Uniform Resource Locator (URL). Every page on the web has a unique URL. Examples are: https://www.mhrd.gov.in,http://www.ncert.nic.in,http://www.airindia.in, etc. URL is sometimes also called web address. However, a URL is not only the domain name. It contains other information that completes a web address.

(III) **HTTP** – The HyperText Transfer Protocol is a set of rules which is used to retrieve linked web pages across the web. The more secure and advanced version is HTTPS.

Many people confuse the web with the Internet. The Internet as we know is the huge global network of interconnected computers, which may or may not have any file or webpage to share with the world. The web on the other hand is the interlinking of collection of Webpages on these computers which are accessible over the Internet.

DOMAIN NAME SYSTEM

(I) DNS stands for Domain Name System.

(II) DNS is a directory service that provides a mapping between the name of a host on the network and its numerical address.

(III) DNS is required for the functioning of the internet.

(IV) Each node in a tree has a domain name, and a full domain name is a sequence of symbols specified by dots.

(V) DNS is a service that translates the domain name into IP addresses. This allows the users of networks to utilize user-friendly names when looking for other hosts instead of remembering the IP addresses.

(VI) For example, suppose the FTP site at EduSoft had an IP address of 132.147.165.50, most people would reach this site by specifying ftp.ABCSoft.com. Therefore, the domain name is more reliable than IP address.

- **DNS Server :** Instead of remembering IP addresses, we assign a domain name to each IP. But, to access a web resource, a browser needs to find out the IP address corresponding to the domain name entered. Conversion of the domain name of each web server to its corresponding IP address is called domain name resolution. It is done through a server called DNS server. Thus, when we enter a URL on a web browser, the HTTP protocol approaches a computer server called DNS server to obtain the IP address corresponding to that domain name. After getting the IP address, the HTTP protocol retrieves the information and loads it in our browser.

A DNS server maintains a database of domain names and their corresponding IP addresses. To understand how the domain name resolution works, we have to understand how and where the DNS servers are kept. The DNS servers are placed in hierarchical order. At the top level, there are 13 servers called root servers. Then below the root servers there are other DNS servers at different levels. A DNS server may contain the IP address corresponding to a domain or it will contain the IP address of other DNS servers, where this domain entry can be searched.

□□□

23 DATA COMMUNICATION (NCERT CLASS 12)

CONCEPT OF COMMUNICATION

The term "Data Communication" comprises two words: Data and Communication. Data can be any text, image, audio, video, and multimedia files. Communication is an act of sending or receiving data. Thus, data communication refers to the exchange of data between two or more networked or connected devices. These devices must be capable of sending and receiving data over a communication medium. Examples of such devices include personal computers, mobile phones, laptops, etc.

COMPONENTS OF DATA COMMUNICATION

The communication media is also called transmission media.

(I) **Sender :** A sender is a computer or any such device which is capable of sending data over a network. It can be a computer, mobile phone, smartwatch, walkie-talkie, video recording device, etc.

(II) **Receiver :** A receiver is a computer or any such device which is capable of receiving data from the network. It can be any computer, printer, laptop, mobile phone, television, etc. In computer communication, the sender and receiver are known as nodes in a network.

(III) **Message :** It is the data or information that needs to be exchanged between the sender and the receiver. Messages can be in the form of text, number, image, audio, video, multimedia, etc.

(IV) **Communication media :** It is the path through which the message travels between source and destination. It is also called medium or link which is either wired or wireless. For example, a television cable, telephone cable, ethernet cable, satellite link, microwaves, etc.

(V) **Protocols :** It is a set of rules that need to be followed by the communicating parties in order to have successful and reliable data communication. You have already come across protocols such as Ethernet and HTTP.

MEASURING CAPACITY OFCOMMUNICATION MEDIA

In data communication, the transmission medium is also known as channel. The capacity of a channel is the maximum amount of signals or traffic that a channel can carry. It is measured in terms of bandwidth and data transfer rate as described below:

(I) **Bandwidth :** Bandwidth of a channel is the range of frequencies available for transmission of data through that channel. Higher the bandwidth, higher the data transfer rate. Normally, bandwidth is the difference of maximum and minimum frequency contained in the composite signals. Bandwidth is measured in Hertz (Hz).

$$1\ KHz = 1000\ Hz$$

$$1\ MHz = 1000\ KHz = 1000000\ Hz$$

(II) **Data Transfer Rate :** Data travels in the form of signals over a channel. One signal carries one or more bits over the channel. Data transfer rate is the number of bits transmitted between source and destination in one second. It is also known as bit rate. It is measured in terms of bits per second (bps). The higher units for data transfer rates are:

$$1\ Kbps = 210\ bps = 1024\ bps$$

$$1\ Mbps = 220\ bps = 1024\ Kbps$$

$$1\ Gbps = 230\ bps = 1024\ Mbps$$

$$1\ Tbps = 240\ bps = 1024\ Gbps$$

TYPES OF DATA COMMUNICATION

Data communication happens in the form of signals between two or more computing devices or nodes. The transfer of data happens over a point-to-pointor multipoint communication channel. Data communication between different devices are broadly categorised into 3 types: Simplex communication, Half-duplex communication, and Full-duplex communication.

(I) **SIMPLEX COMMUNICATION :**It is a one way or unidirectional communication between two devices in which one device is sender and other one is receiver. For example, data entered through a keyboard or audio sent to a speaker are one way communications.With the advent of IoT, controlling home appliances is another example of simplex communication.

(II) **Half-duplex Communication :** It is two way or bidirectional communication between two devices in which both the devices can send and receive data or control signals in both directions, but not at the same time.While one device is sending data, the other one will receive and vice-versa.

Basically, it is a simplex channel where the direction of transmission can be switched. Application of such type of communication can be found in walkie-talkie where one can press the push-to-talk button and talk. This enables the transmitter and turns off the receiver in that device and others can only listen.

(III) **Full-duplex Communication :** It is two way or bidirectional communication in which both devices can send and receive data simultaneously.

This type of communication channel is employed to allow simultaneous communication, for example, in our mobile phones and landline telephones. The capacity of the transmission link is shared between the signals going in both directions. This can be done either by using two physically separate simplex lines — one for sending and other for receiving, or the capacity of the single channel is shared between the signals travelling in different directions.

SWITCHING TECHNIQUE

Switching techniques are used to efficiently transmit data across the network. The two types of switchingtechniques are employed nowadays to providecommunication between two computers on a network are:Circuit Switching and Packet Switching.

(I) **Circuit Switching :** Circuit switching is a technique in which a dedicated and complete physical connection is establishedbetween two nodes and through this dedicated communication channel, the nodes may communicate. The circuit guarantees the full bandwidth of the channel and remains connected for the duration of the communication session. Even if no communication is taking place in a dedicated circuit, that channel still remains unavailable to other users (idle channels).

The defining example of a circuit-switched network is the early analogue telephone network. When a call is made from one telephone to another, switches within the telephone exchange create a continuous wire circuit between the two telephones, for as long as the call lasts.

(II) **Packet Switching :** Packet switching is a switching technique in which packets (discrete blocks of data of fixed size and of anycontent, type or structure) are routed between nodes over data links shared with other traffic. The term "packets" refers to the fact that the data stream from your computer is broken up into packets of about 200 bytes (on average), which are then sent out onto the network. Each packet contains a "header" with information necessary for routing the packet from source to destination. Each packet in a data stream is independent.

The main advantage of packet-switching is that the packets from many different sources can share a line, allowing for very efficient use of the communication medium. With current technology, packets are generally accepted onto the network on a first-come, first-served basis.

If the network becomes overloaded, packets are delayed or discarded ("dropped"). This method of data transmission became the fundamental networking technology behind the internet and most Local Area Networks.

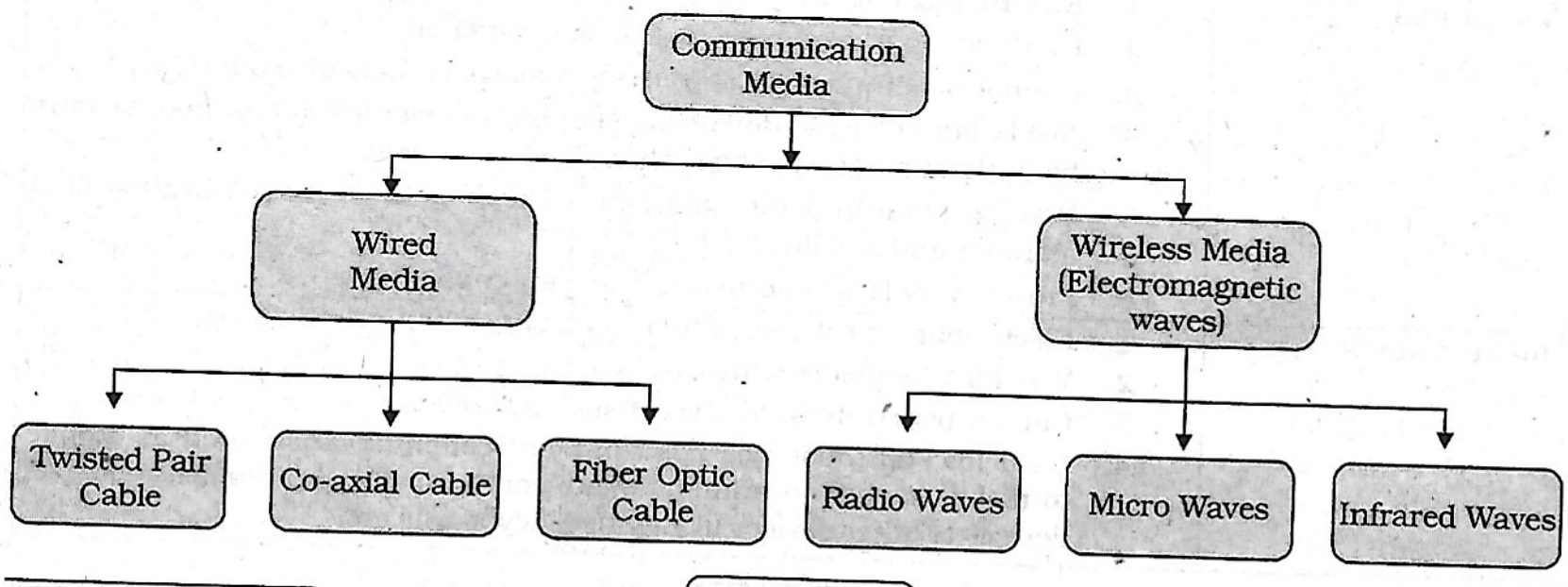
TRANSMISSION MEDIA :

A transmission medium can be anything that can carry signals or data between the source (transmitter) and destination (receiver). For example, as we switch on a ceiling fan or a light bulb, the electric wire is the medium that carries electric current from switch to the fan or bulb.

In data communication, transmission media are the links that carry messages between two or more communicating devices. Transmission can be classified as guided or unguided.

In guided transmission, there is a physical link made of wire/cable through which data in terms of signals are propagated between the nodes. These are usually metallic cable, fiber-optic cable, etc. They are also known as wired media.

In unguided transmission, data travels in air in terms of electromagnetic waves using an antenna. They are also known as wireless media.



Dish-shaped antennas are used for sending and receiving data at longer distances. These antennas are mounted on taller buildings so that it would be in line-of-sight. Waves gradually become weaker and weaker after travelling a certain distance through the air. Therefore repeaters are installed to regenerate the signals of the same energy.

- **Wired Transmission Media :** Any physical link that can carry data in the form of signals belongs to the category of wired transmission media. Three commonly used guided/wired media for data transmission are, twisted pair, coaxial cable, and fiber optic cable. Twisted-pair and coaxial cable carry the electric signals whereas the optical fiber cable carries the light signals.
- **Twisted Pair Cable :** A twisted-pair consists of two copper wires twisted like a DNA helical structure. Both the copper wires are insulated with plastic covers. Usually, a number of such pairs are combined together and covered with a protective outer wrapping.

Each of the twisted pairs act as a single communication link. The use of twisted configuration minimises the effect of electrical interference from similar pairs close by. Twisted pairs are less expensive and most commonly used in telephone lines and LANs. These cables are of two types: Unshielded twisted-pair (UTP) and Shielded twisted-pair (STP).

- **Coaxial cable :** Coxial cable is another type of data transmission medium. It is better shielded and has more bandwidth than a twisted pair. It has a copper wire at the core of the cable which is surrounded with insulating material. The insulator is further surrounded with an outer conductor (usually a copper mesh). This outer conductor is wrapped in a plastic cover. The key to success of coaxial cable is its shielded design that allows the cable's copper core to transmit data quickly, without interference of environmental factors. These types of cables are used to carry signals of higher frequencies to a longer distance.
- **Optical Fibre :** The optical fiber cable carries data as light, which travels inside a thin fiber of glass. Optic fiber uses refraction to direct the light through the media. A thin transparent strand of glass at the centre is covered with a layer of less dense glass called cladding. This whole arrangement is covered with an outer jacket made of PVC or Teflon. Such types of cables are usually used in backbone networks. These cables are of light weight and have higher bandwidth which means higher data transfer rate. Signals can travel longer distances and electromagnetic noise cannot affect the cable. However, optic fibers are expensive and unidirectional. Two cables are required for full duplex communication.
- **Wireless Transmission Media :** In wireless communication technology, information In wireless communication technology, information travels in the form of electromagnetic signals through air. Electromagnetic spectrum of frequency ranging from 3 KHz to 900 THz is available for wireless communication.

Classification of transmission waves and their properties :-

Transmission Waves	Properties
Radio Waves	1. Waves of frequency range 3 KHz - 1 GHz 2. Omni-directional, these waves can move in all directions 3. Radio waves of frequency 300KHz-30MHz can travel long distance 4. Susceptible to interference 5. Radio waves of frequency 3-300KHz can penetrate walls 6. These waves are used in AM and FM radio, television, cordless phones.
Microwaves	1. Electromagnetic waves of frequency range 1GHz - 300GHz. 2. Unidirectional, can move in only one direction. 3. Cannot penetrate solid objects such as walls, hills or mountains. 4. Needs line-of-sight propagation i.e. both communicating antenna must be in the direction of each other. 5. Used in point-to-point communication or unicast communication such as radar and satellite. 6. Provide very large information-carrying capacity.
Infrared waves	1. Electromagnetic waves of frequency range 300GHz - 400THz. 2. Very high frequency waves. 3. Cannot penetrate solid objects such as walls. 4. Used for short-distance point-to-point communication such as mobile-to-mobile, mobile-to-printer, remote-control-to-TV, and Bluetooth-enabled devices to other devices like mouse, keyboards etc.

Wireless technologies allow communication between two or more devices in short to long distance without requiring any physical media. There are many types of wireless communication technologies such as Bluetooth, WiFi, WiMax etc.

The electromagnetic spectrum range (3KHz to 900THz) can be divided into 4 categories - Radio waves, Microwaves, Infrared waves and Visible or Light waves, according to their frequency ranges.

- **Wireless Technologies :**

(I) **Bluetooth :** Bluetooth is a short-range wireless technology that can be used to connect mobile-phones, mouse, headphones, keyboards, computers, etc. wirelessly over a short distance. One can print documents with bluetoothenabled printers without a physical connection. All these bluetooth-enabled devices have a low cost transceiver chip. This chip uses the unlicensed frequency band of 2.4 GHz to transmit and receive data. These devices can send data within a range of 10 meters with a speed of 1 - 2 Mbps.

In Bluetooth technology, the communicating devices within a range of 10 meters build a personal area network called piconet. The devices in a piconet work in a master-slave configuration. A master device can communicate with up to 7 active slave devices at the same time.

Bluetooth technology allows up to 255 devices to build a network. Out of them, 8 devices can communicate at the same time and remaining devices can be inactive, waiting for a response command from the master device.

(II) Wireless LAN:- This is another way of wireless communication. Wireless LAN is a local area network (LAN), and it is a popular way to connect to the Internet. The international organisation IEEE assigns numbers to each different standards of LAN. The wireless LAN is number as 802.11, and it is popularly known as Wi-Fi.

These networks consist of communicating devices such as laptops and mobile phones, as well as the network device called APs (access points) which is installed in buildings or floors.

An access point is a device that is used to create a wireless local area network, by connecting to a wired router, switch, or hub. The APs are connected to a wired network, and all the devices communicate or access the Internet through an access point.

Wi-Fi gives users the flexibility to move around within the network area while being connected to the network. Following are some of the benefits of WLAN:

(i) Wireless connections can be used to extend or replace an existing wired infrastructure.

(ii) Resulted in increased access for mobile devices.

(iii) Provides easy access to the Internet in public places.

MOBILE TELECOMMUNICATION TECHNOLOGIES

Today the mobile phone network is the most used network in the world. The ability to be connected to the network on-the-go makes it very convenient to communicate with people via call or instant messages. It is also handy to access the Internet using the mobile phone network through wireless connection. Besides, the Internet of Things (IoT) is letting us control and communicate with other smart devices as well.

The architecture of the mobile network has rapidly evolved over the last few decades. The different landmark achievements in mobile communication technologies are classified as different generations. They are identified as 1G, 2G, 3G, 4G, and 5G.

The first generation (1G) mobile network system came around 1982. It was used to transmit only voice calls. The analog signals were used to carry voices between the caller and receiver.

The second generation (2G) mobile network system came around 1991. Instead of analog signals, voice calls were transmitted in digital form thus providing improved call quality. This increased capacity allowed more people to talk simultaneously, and led to improved security as the signals could be encrypted. It also enabled an additional service to send SMS and MMS (Multimedia messages).

The third generation (3G) mobile network technology was developed during late 90s, but it was introduced commercially around 2001. It offered both digital voice and data services. 3G provided Internet access via the same radio towers that provide voice service to the mobile phone. It facilitated greater voice and data capacity. Therefore, more simultaneous calls could happen in the same frequency range and also a significantly faster data transfer speed.

Demand for faster data is always increasing and thus 4G mobile networks were developed and now 5G networks have also come into being. 4G is much faster than 3G and this has revolutionised the field of telecommunication by bringing the wireless experience to a new level altogether. 4G systems support interactive multimedia, voice, video, wireless internet and other broadband services. Technologically, 4G is very different compared to 3G.

The fifth generation or 5G is currently under development. It is expected to be a milestone development for the success of IoT and Machine to Machine (M2M) communications. Machine to machine (M2M) is direct communication between devices — wired and wireless. 5G is expected to allow data transfer in Gbps, which is much faster than 4G. It is expected to be able to support all the devices of the future such as connected vehicles and the Internet of Things.

PROTOCOL

In communication, Protocol is a set of standard rules that the communicating parties — the sender, the receiver, and all other intermediate devices need to follow. We know that the sender and receiver can be

parts of different networks, placed at different geographic locations. Besides, the data transfer rates in different networks can vary, requiring data to be sent in different formats.

- **Need for Protocols :** We need protocols for different reasons such as flow control, access control, addressing, etc. Flow control is required when the sender and receiver have different speeds of sending and receiving the data.

Access control is required to decide which nodes in a communication channel will access the link shared among them at a particular instant of time. Otherwise, the transmitted data packets will collide if computers are sending data simultaneously through the same link resulting in the loss or corruption of data.

Protocols also define :

(I) how computers identify one another on a network.

(II) the form to which the data should be converted for transit.

(III) how to decide whether the data received is for that node or to be forwarded to another node.

(IV) ensuring that all the data have reached the destination without any loss.

(V) how to rearrange the packets and process them at the destination.

- **HyperText Transfer Protocol (HTTP) :** HTTP stands for HyperText Transfer Protocol. It is the primary protocol used to access the World Wide Web. Tim Berners-Lee led the development of HTTP at CERN in 1989 in collaboration with Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C).

HTTP is a request-response (also called client-server) protocol that runs over TCP. The common use of HTTP is between a web browser (client) and a web server (server). HTTP facilitates access of hypertext from the World Wide Web by defining how information are formatted and transmitted, and how the Web servers and browsers should respond to various commands.

A web page is written using a markup language like HTML and is stored on a web server for access via its URL. Once a user opens a web browser and types in the URL of the intended web page, a logical communication link between the user machine (client) and the web server is created using HTTP.

- **File Transfer Protocol (FTP) :** File Transfer Protocol (FTP) is the protocol used for transferring files from one machine to another. Like HTTP, FTP also works on a client-server model.

File transfer between two systems seems simple and straightforward because FTP takes care of issues between two communicating devices, such as:

(i) use of different conventions while naming files.

(ii) representation of text and data in different formats.

(iii) having different directory structure

- **Point to Point Protocol (PPP) :** PPP is a communication protocol which establishes a dedicated and direct connection between two communicating devices. This protocol defines how two devices will authenticate each other and establish a direct link between them to exchange data. For example, two routers with direct connection communicate using PPP. The Internet users who connect their home computers to the server of an Internet Service Provider (ISP) through a modem also use PPP.

The communicating devices should have duplex modes for using this protocol. This protocol maintains data integrity ensuring that the packets arrive in order. It intimates the sender about damage or lost packets and asks to resend it.

- **Simple Mail Transfer Protocol (SMTP) :** SMTP is a protocol used for email services. It uses information written on the message header (like an envelope on a letter sent by post), and is not concerned with the content of the email message. Each email header contains email addresses of recipients. The email containing header and body are entered into a queue of outgoing mails.

The SMTP sender program takes mails from the outgoing queue and transmits them to the destination(s). When the SMTP sender successfully delivers a particular mail to one or more destinations, it removes the corresponding receiver's email address from the mail's destination list. When that mail is delivered to all the recipients, it is removed from the outgoing queue. The SMTP receiver program accepts each mail that has arrived and places it in the appropriate user mailbox.

- **Transmission Control Protocol (TCP)/ Internet Protocol (IP) :**

TCP/IP stands for Transmission Control Protocol/ Internet Protocol. It is a set of standardised rules that uses a client-server model of communication in which a user or machine (a client) requests a service by a server in the network.

The IP protocol ensures that each computer or node connected to the Internet is assigned an IP address, which is used to identify each node independently. It can be considered to be the adhesive that holds the whole Internet together.

TCP ensures that the message or data is broken into smaller chunks, called IP packets. Each of these packets are routed (transmitted) through the Internet, along a path from one router to the next, until it reaches the specified destination. TCP guarantees the delivery of packets on the designated IP address.It is also responsible for ordering the packets so that they are delivered in sequence.

There are many redundant connection paths in the Internet, with backbones and ISPs connecting to each other in multiple locations. So, there are manypossible paths between two hosts. Hence, two packets of the same message can take two different routes depending on congestion and other factors in different possible routes. When all the packets finally reach the destination machine, they are reassembled into the original message at the receiver's end. □□□

24 SECURITY ASPECT
(NCERT CLASS 12)

THREATS AND PREVENTION

Network security is concerned with protection of our device as well as data from illegitimate access or misuse. Threats include all the ways in which one can exploit any vulnerability or weakness in a network or communication system in order to cause harm or damage one's reputation.

MALWARE

Malware is a short term used for Malicious software. It is any software developed with an intention to damage hardware devices, steal data, or cause any other trouble to the user. Various types of malware have been created from time-to-time, and large-scale damages have been inflicted. Many of these malware programs have been identified and counter measures have been initiated. However, different types of malware keep on coming on a regular basis that compromise the security of computer systems and cause intangible damages. Besides, each year, malware incur financial damages worth billions of dollars worldwide. Viruses, Worms, Ransomware, Trojans, and Spyware are some of the kinds of malware.

(I) **VIRUS :** If you observe that your system
(a) takes longer time to load applications
(b) shows unpredictable program behaviour
(c) shows inexplicable changes in file sizes
(d) has inability to boot,
(e) has strange graphics appearing on your screen

This could be because of your computer being infected by a virus.

Virus is a malicious program that attaches itself to the host program. It is designed to infect the host program and gain control over the system without the owner's knowledge. The virus gets executed eachtime the host program is executed. Also it has the tendency to replicate. They can spread through external media such as CDs, browsing infected internet sites and from email attachments.

Types of Viruses

(a) **File Virus :** These viruses infect and replicate when it gets attached to MS-DOS program files with EXE or COM extensions.

(b) **Boot sector virus :** These viruses infect the boot sector of floppy disks or hard drives. Boot sector of a drive contains program that participates in booting the system. A virus can infect the system by replacing or attaching itself to these programs.

(c) **Macro virus :** These viruses infect and replicate using the MS Office program suite, mainly MS Word and MS Excel. The virus inserts unwanted words or phrases in the document.

(II) **WORM :** Worm is also a malicious program like a virus. But unlike viruses, it does not need to attach itself to a host program. A worm works by itself as an independent object.It uses security holes in a computer networks to replicate itself. A copy of the worm scans the network for another machine that has a specific security hole. It copies itself to the new machine using the security hole, and then starts replicating from there, as well.

(III) **Ransomware :** It is a type of malware that targets user data. It either blocks the user from accessing their own data or threatens to publish the personal data online and demands ransom payment against the same. Some ransomware simply block the access to the data while others encrypt data making it very difficult to access.

It worked by encrypting data and demanding ransom payments in the Bitcoin cryptocurrency. It literally made its victims "cry" and hence the name.

(IV) **Trojan horse :** A Trojan horse is a program that contains hidden malicious functions. Trojan Horses trick users intoinstalling them by appearing to be legitimate programs. Once installed on a system, they reveal their true nature and cause damage. Some Trojan horses will contact a central server and report back information such as passwords, user IDs, and captured keystrokes. Trojans lack a replication routine and thus are not viruses by definition.

(V) **SPYWARE :** It is a type of malware that spies on a person or an organisation by gathering information about them, without the knowledge of the user. It records and sends the collected information to an external entity without consent or knowledge of the user.

Spyware usually tracks internet usage data and sells them to advertisers. They can also be used to track and capture credit card or bank account information, login and password information or user's personal identity.

(VI) **ADWARE:** An Adware is a malware that is created to generate revenue for its developer. An adware displays online advertisements using pop-ups, web pages, or installation screens. Once an adware has infected a substantial number of computer systems, it generates revenue either by displaying

advertisements or using "pay per click" mechanism to charge its clients against the number of clicks on their displayed ads. Adwareis usually annoying, but harmless. However, it often paves way for other malware by displaying unsafe links as advertisements.

(VII) **Keyloggers :** A keylogger can either be malware or hardware. The main purpose of this malware is to record the keys pressed by a user on the keyboard. A keylogger makes logs of daily keyboard usage and may send it to an external entity as well. In this way, very sensitive and personal information like passwords, emails, private conversations, etc. can be revealed to an external entity without the knowledge of the user. One strategy to avoid the threat of password leaks by keyloggers is to use a virtual keyboard while signing into your online accounts from an unknown computer.

- **Modes of Malware distribution :**

A malware once designed, can take many routes to reach your computer. Some of the common distribution channels for malware are:

(I) **Downloaded from the Internet :** Most of the time, malware is unintentionally downloaded into the hard drive of a computer by the user. Of course, the malware designers are smart enough to disguise their malware, but we should be very careful while downloading files from the Internet (especially those highlighted as free stuff).

(II) **Spam Email :** We often receive an unsolicited email with embedded hyperlinks or attachment files. These links or attached files can be malware.

(III) **Removable Storage Devices :** Often, the replicating malware targets the removable storage media like pen drives, SSD cards, music players, mobile phones, etc. and infect them with malware that gets transferred to other systems that they are plugged into.

(IV) **Network Propagation :** Some malware like Worms have the ability to propagate from one computer to another through a network connection.

- **Combating Malware**

Common signs of some malware infection include the following:

(I) frequent pop-up windows prompting you to visit some website and/or download some software;

(II) changes to the default homepage of your web browser;

(III) mass emails being sent from your email account;

(IV) unusually slow computer with frequent crashes;

(V) unknown programs startup as you turn on your computer;

(VI) programs opening and closing automatically;

(VII) sudden lack of storage space, random messages, sounds, or music start to appear;

(VIII) programs or files appear or disappear without your knowledge.

ANTIVIRUS

Antivirus is a software, also known as anti-malware. Initially, antivirus software was developed to detect and remove viruses only and hence the name anti-virus. However, with time it has evolved and now comes bundled with the prevention, detection, and removal of a wide range of malware.

Methods of Malware Identification used by Antivirus:-

(I) **Signature-based detection :** In this method, an antivirus works with the help of a signature database known as "Virus Definition File (VDF)". This file consists of virus signatures and is updated continuously on a real-time basis. This makes the regular update of the antivirus software a must. If there is an antivirus software with an outdated VDF, it is as good as having no antivirus software installed, as the new malware will infect the system without getting detected. This method also fails to detect malware that has an ability to change its signature (polymorphic) and the malware that has some portion of its code encrypted.

(II) **Sandbox detection :** In this method, a new application or file is executed in a virtual environment (sandbox) and its behavioural fingerprint is observed for a possible malware. Depending on its behaviour, the antivirus engine determines if it is a potential threat or not and proceeds accordingly. Although this method is a little slow, it is very safe as the new unknown application is not given access to actual resources of the system.

(III) **Data mining techniques :** This method employs various data mining and machine learning techniques to classify the behaviour of a file as either benign or malicious.

(IV) **Heuristics :** Often, a malware infection follows a certain pattern. Here, the source code of a suspected program is compared to viruses that are already known and are in the heuristic database. If the majority of the source code matches with any code in the heuristic database, the code is flagged as a possible threat.

(V) **Real-time protection :** Some malware remains dormant or gets activated after some time. Such malware needs to be checked on a real-time basis. In this technique, the anti-malware software keeps running in the background and observes the behavior of an application or file for any suspicious activity while it is being executed i.e. when it resides in the active (main) memory of the computer system.

SPAM

The term spam means endless repetition of worthless text. In other words, unwanted messages or mails are known as Spam. At times internet is flooded with multiple copies of the same message, it is nothing but spam. Most spam is commercial advertising. In addition to wasting people's time, spam also eats up a lot of network bandwidth.

HTTP vs HTTPS

Both the HTTP (Hyper Text Transfer Protocol) and its variant HTTPS (Hyper Text Transfer Protocol Secure) are a set of rules (protocol) that govern how data can be transmitted over the WWW (World Wide Web). In other words, they provide rules for the client web browser and servers to communicate.

HTTP sends information over the network as it is. It does not scramble the data to be transmitted, leaving it vulnerable to attacks from hackers. Hence, HTTP is sufficient for websites with public information sharing like news portals, blogs, etc. However, when it comes to dealing with personal information, banking credentials and passwords, we need to communicate data more securely over the network using HTTPS. HTTPS encrypts the data before transmission. At the receiver end, it decrypts to recover the original data. The HTTPS based websites require SSL Digital Certificate.

FIREWALL

A firewall is hardware or software based network security system. It prevents unauthorized access(hackers, viruses, worms etc.) to or from a network.

Firewalls are used to prevent unauthorized internet users to access private networks connected to the Internet. All data entering or leaving the Intranet pass through the firewall, which examines each packet and blocks those that do not meet the specified security criteria.

A firewall examines all traffic routed between the two networks to see if it meets certain criteria. If it does, it is routed between the networks, otherwise it is stopped. A firewall filters both inbound and out bound traffic. A firewall may allow all traffic through unless it meets certain criteria, or it may deny all traffic unless it meets certain criteria.

Types of Firewall :

(I) **Network Firewall :** If the firewall is placed between two or more networks and monitors the network traffic between different networks, it is termed as Network Firewall.

(II) **Host-based Firewall :** If the firewall is placed on a computer and monitors the network traffic to and from that computer, it is called a host-based firewall.

COOKIES

When the user browses a website, the web server sends a text file to the web browser.This small text file is a cookie. Generally a cookie contains the name of the website from which it has come from and a unique ID tag.

Some cookies last only until the browser is closed. They are not stored on your hard drive. They are usually used to track the pages that you visit so that information can be customised for you for that visit. On the other hand, some cookies are stored on your hard drive until you delete them or they reach their expiry date. These may, for example, be used to remember your preferences when you use the website.

HACKERS AND CRACKERS

The term hacking was first used at M.I.T during 1950s and 1960s. The term was used for people whoengaged themselves in harmless technical experiments and fun learning activities.

A computer enthusiast, who uses his computer programming skills to intentionally access a computer without authorization is known as hacking. The computer enthusiast involved in this activity is known as a hacker. A hacker accesses the computer without the intention of destroying data or maliciously harming the computer.

Another term commonly used with hacking is cracking. Cracking can be defined as a method by which a person who gains unauthorized access to a computer with the intention of causing damage.

Depending on the intent, there are different types of hackers :

(I) **White Hats: Ethical Hacker :** If a hacker uses its knowledge to find and help in fixing the security flaws in the system, its termed as White Hat hacker. These are the hackers with good intentions. They are actually security experts. Organisations hire ethical or white hat hackers to check and fix their systems for potential security threats and loopholes. Technically, white hats work against black hats.

(II) **Black Hats: Crackers :** If hackers use their knowledge unethically to break the law and disrupt security by exploiting the flaws and loopholes in a system, then they are called black hat hackers.

(III) **Grey Hats :** The distinction between different hackers is not always clear. There exists a grey area in between, which represents the class of hackers that are neutral, they hack systems by exploiting its vulnerabilities, but they don't do so for monetary or political gains. The grey hats take system security as a challenge and just hack systems for the fun of it.

NETWORK SECURITY THREATS

(I) **Denial of Service :** Denial of Service (DoS) is a scenario, wherein an attacker (Hacker) limits or stops an authorised user to access a service, device, or any such resource by overloading that resource with illegitimate requests. The DoS attack floods the victim resource with traffic, making the resource appear busy. If attackers carry out a DoS attack on a website, they will flood it with a very large number of network packets by using different IP addresses. This way, the web server would be overloaded and will not be able to provide service to a legitimate user. The users will think that the website is not working, causing damage to the victim's organisation. Same way, DoS attacks can be done on resources like email servers, network storage, disrupting connection between two machines or disrupting the state of information (resetting of sessions).

If a DoS attack makes a server crash, the server or resource can be restarted to recover from the attack. However, a flooding attack is difficult to recover from, as there can be some genuine legitimate requests in it as well.

A variant of DoS, known as Distributed Denial of Service (DDoS) is an attack, where the flooded requests come from compromised computer (Zombies) systems distributed across the globe or over a very large area. The attacker installs a malicious software known as Bot on the Zombie machines, which gives it control over these machines. Depending upon the requirement and availability, the attacker activates a network of these Zombie computers known as Bot-Net to carry out the DDoS attack. While as a simple DoS attack may be countered by blocking requests or network packets from a single source, DDoS is very difficult to resolve, as the attack is carried from multiple distributed locations.

(II) **Intrusion Problems :** Network Intrusion refers to any unauthorised activity on a computer network. These activities may involve unauthorised use of network resources (DoS) or threatening the security of the network and the data. Network intrusion is a very serious problem and the network administrator needs to devise strategy and implement various security measures to protect the network.

(a) **Asymmetric Routing :** The attacker tends to avoid detection by sending the intrusion packets through multiple paths, thereby bypassing the network intrusion sensors.

(b) **Buffer Overflow Attacks :** In this attack, the attacker overwrites certain memory areas of the computers within the network with code (set of commands) that will be executed later when the buffer overflow (programming error) occurs. Once the malicious code is executed, an attacker can initiate a DoS attack or gain access to the network.

(c) **Traffic Flooding :** It is one of the most trivial methods of network intrusion. It involves flooding the network intrusion detection system with message packets. This huge load leaves the network detection system incapable of monitoring the packets adequately. The hacker takes advantage of this congested and chaotic network environment to sneak into the system undetected.

(III) **Snooping :** Snooping means secretly listening to a conversation. In the context of networking, it refers to the process of secret capture and analysis of network traffic. It is a computer program or utility that has a network traffic monitoring capability. In this attack, the hacker taps or listens to a channel of communication by picking all of the traffic passing through it. Once the network packets are analysed by the snooping device or software, it reproduces the exact traffic packets and places them back in the channel, as if nothing has happened. So, if the data that is being sent over the network is not encrypted, it is vulnerable to snooping and eventuallymay cause serious damage, depending upon the type of information leak. However, snooping is not always an attack, at times it is also used by network administrators for troubleshooting various network issues. Snooping is also known as Sniffing.

Various snooping software exist that act as network traffic analyser. Besides, various network hubs and switches have a SPAN (Sniffer Port Analyser) port function for snooping.

(IV) **Eavesdropping :** The term eavesdropping has been derived from the literal practice of secretly listening to the conversations of people by standing under the eaves of a house. Unlike snooping, where the network traffic can be stored for later analysis, eavesdropping is an unauthorised real-time interception or monitoring of private communication between two entities over a network. Also, the targetsphone calls (VoIP), instant messages, video conference, fax transmission, etc. In older days, eavesdropping was performed on the conventional telephone line and was known as wiretapping. Digital devices like laptops and cell phones that have a built-in microphone or camera can be easily hacked and eavesdropped using rootkit malware.

□□□

25 PROJECT BASED LEARNING (NCERT CLASS 12)

INTRODUCTION

Through project based learning, students learn to organise their project and use their time effectively for successful completion of the project.

APPROACHES FOR SOLVING PROJECTS

The approach followed for the development and completion of a project plays a pivotal role in project based learning. There are several approaches to execute a project such as modular approach, top down approach and bottom up approach. A structured or a modular approach to a project means that a project is divided into various manageable modules and each of the modules has a well-defined task to be performed with a set of inputs. This would lead to a set of outputs which when integrated leads to the desired outcome.

Different steps involved in project based learning are :

(I) **Identification of a project :** The project idea may come through any real-life situation. For example, one could think of doing a project for organising a seminar. One needs to understand the usefulness of the project and its impact. Students must be encouraged to undertake interdisciplinary projects.

(II) **Defining a plan :** Normally for any kind of project, there are several project members involved in it. One project leader has to be identified. The roles of project leader and each project member have to be clearly defined. Students who are performing a project must be assigned with specific activities. The various tools for executing these activities must be known. To obtain a better solution, one should always think of the extreme situations.

(III) **Fixing of a time frame and processing :** Every project is a time relevance project. A student must understand the importance of time frame for completion of the project. All the activities which are performed in the projects require a certain amount of time. Every project must be well structured and at the same time it must be flexible in its time frame.

(IV) **Providing guidance and monitoring a project :** Many times, the participants in the project get stuck up with a particular process and it becomes impossible to proceed further. In such a case, they need guidance, which can be obtained from various resources such as books, websites and experts in the field. While it is essential that the project leader should ensure monitoring of the project, the guide teacher also helps in monitoring the project.

(V) **Outcome of a project :** One needs to understand thoroughly the outcome of a project. The outcome can be single, or it can be multiple. The output of a project can be peer reviewed and can be modified as per the feedback from the guide teacher or other users.

TEAMWORK

Many real-life tasks are very complex and require a lot of individuals to contribute in achieving them. Efforts made by individuals collectively to accomplish a task is called teamwork.

Following are the components of teamwork :

(I) **Communicate with Others :** When a group of individuals perform one job, it is necessary to have effective communication between the members of the team. Such communication can be done via e-mails, telephones or by arranging group meetings. This helps the team members to understand each other and sort out their problems to achieve the goal effectively.

(II) **Listen to Others :** It is necessary to understand the ideas of others while executing a job together. This can be achieved when the team members listen to each other in group meetings and follow steps that are agreed upon.

(III) **Share with Others :** Ideas, images and tools need to be shared with each other in order to perform a job. Sharing is an important component of teamwork. Any member of the team who is well versed in a certain area should share the expertise and experience with others to effectively achieve the goal within the time frame.

(IV) **Respect for Others :** Every member of the team must be treated respectfully. All the thoughts and ideas that are put forth in the group meetings may be respected and duly considered. Not respecting the views of a particular member may cause problems and that particular team member may not give his best.

(V) **Help Others :** A helping hand from every member is a key to success. Sometimes help from people who are not a part of the team is also obtained in order to accomplish a job.

(VI) **Participate :** All the team members must be encouraged by each other to participate in completing the project and also in discussions in group meetings. Also, every member should take an active participation so that they feel their importance in the team.

❑❑❑