TEACHER'S CARE PUBLICATION
38/23, Vaigundaperumal Koil Street,Kancheepuram- 631502
Mobile : 95665 35080, 9786269980 Land Line : 044-2723 5080

2025-2026

PG TRB

G.O No. 219 New Syllabus

COMPUTER SCIENCE

UNIT-3
System Software & Operating System

Your Success is Our Goal...

- System Software
- Basics of Operating Systems
- Process Management
- Threads
- CPU Scheduling
- Deadlocks
- Memory Management

tcaexamguide
Teacher's Care Academy
www.tcaexamguide.com
tcaexamguide@gmail.com

CONTENTS

TEACHER'S CARE ACADEMY, KANCHIPURAM

TNPSC-TRB- COMPUTER SCIENCE -TET COACHING CENTER

HEAD OFFICE: NO. 38/23, VAIGUNDA PERUMAL KOIL, SANNATHI STREET, KANCHIPURAM – 1. CELL: 9566535080

B.Off 2: 65C, Thillai Ngr(West), 4th Cross St, Trichy – 620018
B.Off 3: Vijiyaraghavachariar Memorial Hall(Opp to Sundar Lodge), Salem

Trichy : 76399 67359 **Salem : 93602 68118**

CHAPTER - 1

SYSTEM SOFTWARE

INTRODUCTION

- In our day-to-day life we somehow come across with different types of software that assist us in solving our tasks and help us to increase efficiency in our work. We find software in various electronic gadgets like a Desktop, Laptop, Cellular Phone and what not.
- From operating system software that greets us when we switch on the computer system to the web browser software that is used to explore the electronic content through the internet or the games that we play on our computer to the step count application software on our smart phone, are all instances of software.
- In this technological world, we even come across various software development trends that help our business to expand; we are surrounded by all this software which helps to make our lives simpler.
- By definition, Software is an assembly of data, programs, procedures, instructions, and documentation that perform various predefined tasks on a computer system. They enable users to interact with the computer by processing different type of information.
- Any software works only when it has an assistance of some computer hardware technology. Both the entities need each other and neither one of them can be influentially used on its own.
- The incorporation of the hardware and the software gives control and flexibility to modern-day computing systems. For example, without the help of our web browser software, we will not be able to surf the Internet. Similarly, in the absence of an operating system, no application can run on our computer.
- Today there are ample of superior technologies and software accessible to us that define the way we lead our lives and house our frequently changing needs. There are non-ending numbers of software categorized on the basis of technology, functionality, usage etc.

1.1 DIFFERENT TYPES OF SOFTWARE

- Software is primarily classified into are two major types, namely **System Software** and **Application Software.**

1.1.1 SYSTEM SOFTWARE

- System software helps the user and the hardware device function and interacts with each other. Basically, it is a type of software which is used to manage the department of computer hardware to provide the very basic functionalities that are required by the user. In simple words, we can say that system software works like an inter-mediatory or a middle layer between the user and the hardware.
- System software provides a necessary platform or an environment for the other software to work in. Due to this reason system software plays an important role in handling the overall computer system.
- System software is not just limited to a desktop or a Laptop computer system. It has a broad existence in various digital and electronic devices wherever there is a usage of a computer processor. When a user turns on the computer, it is the system software that gets initialized and gets loaded in the memory of the system.
- The system software runs in the background and is not used by the end-users. This is the reason why system software is also known as 'low-level software'.

Following are the most common examples of system software:

1. **Operating System (OS)**
    a. It is one of the popularly used System Software throughout the digital arena. It is a collection of software that handles resources and provides general services for the other applications that run over them.
    b. Although each Operating System is different based on look and feel as well as functionalities, most of them provide a Graphical User Interface through which a user can manage the files and folders and perform other tasks.
    c. Every device, whether a desktop, laptop or mobile phone requires an operating system to provide the basic functionality to it.
    d. An Operating System essentially determines how a user interacts with the system; therefore, many users prefer to use one specific OS for their device. There are various types of operating system such as single user, multiuser, embedded, real-time, distributed, mobile, etc. It is important to consider the hardware specifications before choosing an operating system.

e. Some examples of Operating systems software are Microsoft Windows, Linux, Mac OS, Android, iOS, Ubuntu, UNIX, etc.

2. Device Drivers

a. It is a type of software that controls the hardware device which is attached to the system. Hardware devices that need a driver to connect to a system include displays, sound cards, printers, mouse, and hard disks.

b. Further, there are two types of device drivers: User Device Driver and Kernel Device Drivers. Some examples of device drivers are VGA Drivers, VGA Drivers, Virtual Device Drivers, BIOS Driver, Display Drivers, Motherboard Drivers, Printer Drivers, ROM Drivers, Sound card Driver, USB Drivers, USB Drivers, etc.

3. Firmware

a. It is the permanent software that is embedded into a read-only memory. It is a set of instructions permanently stored on a hardware device. It provides essential information regarding how the device interacts with other hardware.

b. Firmware can be considered as 'semi-permanent' as it remains permanent unless it is updated using a firmware updater.

c. Some examples of firmware are: BIOS, Computer Peripherals, Consumer Applications, Embedded Systems, UEFI, etc.

4. **Programming Language Translators**

a. These are mediator programs on which software programs rely to translate high-level language code to simpler machine level code.

b. Besides simplifying the code, the translators have the capability to Assign data storage, enlist source code as well as program details, Offer diagnostic reports, Rectify system errors during the runtime.

c. Examples of Programming Language Translators are Interpreter, Compiler and Assemblers.

5. Utility

a. This software is designed to aid in analyzing, optimizing, configuring and maintaining a computer system. It supports the computer infrastructure.

b. This software focuses on how an OS functions and then accordingly it decides its trajectory to smoothen the functioning of the system.

c. Software like antiviruses, disk cleanup & management tools, compression tools, defragmenters, etc., they are all utility tools.

1.1.2. APPLICATION SOFTWARE

- Also known as end-user programs or productivity programs are software that helps the user in completing various tasks. In contrast to system software, this software is specific in their functionality or tasks used by the end-user. This software is placed above the system software.
- Application Software or simply apps can also be referred to as non-essential software as their requirement is highly subjective and their absence does not affect the functioning of the system.
- For example, such as a text editor, online train or a flight booking application, online banking web-based application, a billing application, high end graphics designing application, accounting software, any type of calculator application or even standalone as well as online games software, the various applications that we use in our cellular phones are also the examples of Application Software.

There are various types of application software:

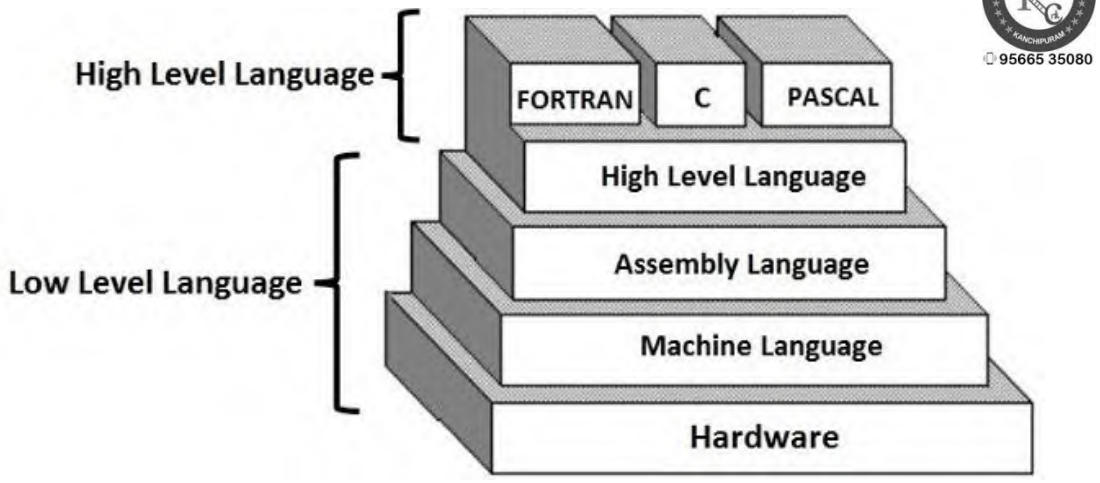
1. **Word Processors:** These applications are widely used to create the documentation. It also helps in storage, formatting, and printing of these documents. Some examples of word processors are Abiword, apple iWork-Pages, Corel WordPerfect, Google Docs, MS Word etc.
2. **Database Software:** This software is used to create and manage a database. It is also known as the Database Management System or DBMS. They help with the organization of data. Some examples of DBMS are Clipper, dBase, FileMaker, FoxPro, MySQL etc.
3. **Multimedia Software:** It is the software that can play, create, or record images, audio or video files. They are used for video editing, animation, graphics, and image editing; some examples of Multimedia Software are Adobe Photoshop, Inkscape, Media Monkey, Picasa, VLC Media Player, Windows Media Player, Windows Movie Maker etc.
4. **Education and Reference Software:** These types of software are specifically designed to facilitate learning on a particular subject. There are various kinds of tutorial software that fall under this category. They are also termed as academic software. Some examples are Delta Drawing, GCompris, Jumpstart titles, KidPix, MindPlay, Tux Paint etc.
5. **Graphics Software:** As the name suggests, Graphics Software has been devised to work with graphics as it helps the user to edit or make changes in visual data or images. It comprises of picture editors and illustration software.

Some examples are Adobe Photoshop, Autodesk Maya, Blender, CorelDRAW, GIMP, Modo, PaintShop Pro etc.

6. **Web Browsers:** These applications are used to browse the internet. They help the user in locating and retrieving data across the web. Some examples of web browsers are Google Chrome, Internet Explorer, Microsoft Edge, Mozilla Firefox, Opera, Safari, UC Browser etc.

1.2 MACHINE, ASSEMBLY AND HIGH-LEVEL LANGUAGES

- A language is the main medium of communicating between the Computer systems and the most common are the programming languages. A language consists of all the instructions to make a request to the system for processing a task.
- Some of the languages like programming language which is a set of codes or instructions used for communicating the machine.
- Machine code is also considered as a computer language that can be used for programming. And also HTML which is a computer language or a markup language but not a programming language.
- Similarly there are different types of languages developed for different types of work to be performed by communicating with the machine. But all the languages that are now available are categorized into two basic types of languages including:
    - Low-level language and
    - High level language



1.2.1 Low Level Language

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

- There are various programs and applications written in low level languages that are directly executable without any interpretation or translation.
- Low level language is also divided into two parts are:
    - Machine language and
    - Assembly language

1.2.1.1 MACHINE LANGUAGE

- Machine Language is the only language that is directly understood by the computer. It does not need any translator program. We also call it machine code and it is written as strings of 1's (one) and 0's (zero).
- When this sequence of codes is fed to the computer, it recognizes the codes and converts it into electrical signals needed to run it.
- For example, a program instruction may look like this: 1011000111101. It is not an easy language for you to learn because of its difficult to understand. It is efficient for the computer but very inefficient for programmers.
- It is considered to the first-generation language. It is also difficult to debug the program written in this language.

Advantages and Limitations of Machine Languages

- Programs written in machine language can be executed very fast by the computer. This is due to the fact that machine instructions are directly understood by the CPU and no translation of the program is required.
- But writing a program in machine language has some disadvantages which are given below:

  1. Machine dependence: Since the Internal design of a computer varies from machine to machine, the machine language is different from computer to computer. Thus a program written in machine language in one computer needs modification for its execution on another computer.

  2. Difficult to the program: A machine language programmer must have thorough knowledge about the hardware structure of the computer.

  3. Error-prone: For writing programs in machine language, a programmer has to remember the OPCODES and has to keep track of the storage locution of data and instructions.

  4. Difficult to modify: It is very difficult to correct or modify machine language programs.

1.2.1.2. ASSEMBLY LANGUAGE

- It uses only letters and symbols. Programming is simpler and less time consuming than machine language programming.

- ❖ It is easy to locate and correct errors in Assembly language. It is also machine-dependent. The programmer must have knowledge of the machine on which the program will run.
- ❖ An assembler is a program that translates an assembly language program into a machine language program.
- ❖ Assembly languages have the following **advantages** over machine languages;

1. Easier to understand and use: Assembly languages are easier to understand and use because mnemonics are used instead of numeric op-codes and suitable names are used for data.

2. Easy to locate and correct errors: While writing programs in assembly language, fewer errors are made and those that are made arc easier to find and correct because of the use of mnemonics and symbolic names.

3. Easier to modify: Assembly language programs are easier for people to modify than machine language programs. This is mainly because they are easier to understand and hence it is easier to locate, correct, and modifies instructions as and when desired.

4. No worry about addresses: The great advantage of assembly language is that it eliminates worry about address for instructions and data.

Disadvantages

1. Machine Dependence: Programs written in assembly language are designed for the specific make and model of the processor being used and are therefore machine-dependent.

2. Knowledge of hardware is required: Since assembly language is machine-dependent, the programmer must be aware of a particular machine's characteristics and requirements as the program is written. Machine and assembly codes are based on the basic design of computers and are referred to as 'low-level language'.

1.2.2 HIGH LEVEL LANGUAGES

- ❖ High Level computer languages are the advanced development languages in the evolution of computer languages. These languages are designed to make the programming easier and less error-free.
- ❖ High level language uses words and commands along with symbols and numbers.
- ❖ The keywords used in High level languages are similar to English words and can be easily understood by Humans when compared to a Low level language.
- ❖ Types of programming languages in High level languages are:
    - C
    - C++

- Java
- Python
- PHP

❖ The above given computer languages list are a few examples but there are many other computer languages. Each of these languages has their own syntax (structured statements) and keywords.

Difference between High-Level and Low-Level Computer languages

S.No	Low Level Computer Language	High Level Computer Language
1.	Low level languages are hard to understand by humans as they use binary numbers and easy to understand by the computer.	High level languages are simple to understand by humans as they use English statements.
2.	Low level languages are programmer-friendly	High level languages are human friendly.
3.	Program execution time (computational speed) is less.	Program execution time is longer.
4.	These are complex to maintain.	These are simple to maintain.
5.	Debugging process is hard.	It is easy to debug in High level languages.
6.	The programs in low level language are not portable from one computer to another.	The programs in a high level language are portable, so we can use them on any computer.
7.	Usage of low level language is less in today's technologies.	We use high level language in today's technologies.

Important Terms Used in Computer Languages

- **Statement:** A statement is telling a computer on how to do a desired action using words or instructions.
- **Syntax:** Syntax is the structured arrangement of statements.
- **Algorithm:** Algorithm is a set of instructions written to solve a problem. It's the logical thought process of a computer.

- **Binary numbers:** Binary numbers are a way of expressing data. The numbers 1 and 0 are called binary numbers. Computers can only understand binary language. Computers can be able to process millions of 1's and 0's.
- **Coding:** Do you know "what is computer code"? Coding is the process of creating computer instructions. Computer codes are also called Programs.
- **Debug:** Debugging is the process of finding and removing errors from a code.
- **Assembler:** An Assembler is a computer program designed in such a way that it converts mnemonics to 0's and 1's.

There are also other types of languages, which include

- **System languages**: These are designed for low-level tasks, like memory and process management.
- **Scripting languages**: These tend to be high-level and very powerful.
- **Domain-specific languages**: These are only used in very specific contexts.
- **Visual languages**: Languages those are not text-based.
- **Esoteric languages**: Languages that are jokes or are not intended for serious use.

1.3 COMPILERS AND INTERPRETERS

- A compiler is defined as software that transforms an entire set of source code into object code and saves it as a file before executing it. Conversely, an interpreter converts and executes source code line by line without saving it and points out errors along the way.
- Compiled languages include C, C++, COBOL, and FORTRAN. Python utilizes an interpreter, as do JavaScript, Perl, and BASIC.

What Is a Compiler?

- A compiler is a piece of software that transforms source code into object code before executing it. Simply put, a high-level language is converted into machine/binary language, and this stage is required to make the program executable. This is because the only language the computer understands is binary.
- As an intermediate phase, certain compilers transform the high-level programming language into assembly language, and others directly convert it to machine code. This conversion of source code into machine code is called compilation. Popular computer languages that use compilers include C, C++, COBOL, and FORTRAN, among others.

How does a compiler work?

- The operation of a compiler can be categorized as follows:

- **Source code creation:** The source code is a piece of code composed in a text editor, and the file extension for the source code is '.c,' if you are using a compiler with the C programming language as an example.
- **Pre-processing**: This source code is initially transmitted to the pre-processor, which expands it. The enlarged code will be provided to the compiler after expansion.
- **Compiling**: The code expanded by the pre-processor is passed to the compiler, which converts it into assembly code.
- **Conversion into object code by an assembler:** Using an assembler, the assembly code is transformed into object code. The object file created by an assembler has the same name as the source file.
- **Linking:** When a program occasionally references functions specified in other files, the linker serves a crucial role. In the compiler workflow, the primary function of a linker is to connect the object code of coding library data alongside the object code of a program.
- **Execution:** The executable file is the final product of the linker.

Pros and cons of a compiler

❖ A compiler offers the following advantages:

- It operates quicker than the interpreter because the source code has already been compiled and the executable file has been generated.
- Client does not need to install a compiler, interpreter, or third-party program to run the executable file of our shared source code.
- On all of your clients' and any other system, executable files generated by the compiler can be launched without needing the source code. This renders your program hack-proof, protected, and confidential.
- The machine code of an executable file generated by a compiler is frequently a well-optimized, native machine command for the intended machine, resulting in accelerated execution.

However, compilers do have a few drawbacks:

- Since its code is optimized for the system on which it was executed, it could trigger system compatibility issues.
- Compilers must generate a new file, which consumes additional memory.
- After perusing the entire code, it returns all available errors at once, making it harder to locate and correct them.
- Unlike an interpreter, we cannot run the source code straight away; we must additionally launch the executable file.

What Is an Interpreter?

- The software that executes the line-by-line conversion of high-level instructions to machine-level programming is called an interpreter. If an error is detected on any line of code, the execution is halted until resolved. Errors are displayed line by line, making the correction of them simpler. However, the program requires a little more time to run successfully.
- Source code lines are compiled beforehand and preserved as machine-independent codes, and this is subsequently attached at runtime and processed by the interpreter.
- The interpreter executes the high-level program code's instructions. For interpreted programs, the source code is always required for processing. Consequently, they tend to execute more slowly than compiled programs. Among the most prominent programming languages, Python utilizes an interpreter. JavaScript, Perl, and BASIC are instances of other popular interpreter-led programming languages.

How does an interpreter work?

- The operation of an interpreter can be broken down as follows:
  - **Source code creation:** This step of the functionality is the same as in the case of a compiler. During runtime, however, the interpreter transforms the source code one line at a time.
  - **Direct interpretation:** An interpreter meticulously translates a high-level language program into machine-level language.
  - **Source code editing**: The interpreter permits program assessment and modification throughout the execution in a side-by-side window.
  - **Execution**: Compared to the compiler, program execution is moderately sluggish since all the linking is done at runtime without a separate linker.

Advantages of interpreters:

- Since it scans the code line by line and sends the error notice straight away, debugging is simpler.
- Those with access to the source code can easily rectify or alter the code if necessary.
- In interpreted languages, the source code can be directly shared and executed on any machine without device incompatibility problems.
- Interpreters do not create new independent files. As a result, it does not consume additional memory.
- The source code is executed without any additional steps; it is run spontaneously.

Disadvantages of interpreters:

- If code execution is interrupted at any point, interpreters restart from the beginning every time the code is executed.
- The interpreter is typically more sluggish than the compiler as it scans, analyses, and transforms each line of code individually.
- A client or anyone with access to the shared source code will need an interpreter installed on their system to execute the code.
- To pass on an interpreted program with others, one must share unsecured and non-private source code.

Compiler vs. Interpreter: Key Differences

❖ The basic differences between compilers and interpreters are listed below.

Feature	Compiler	Interpreter
Translation	Translates the entire source code into machine code at once.	Translates the source code line-by-line during execution.
Execution	The compiled program is executed after the entire translation is done.	Executes the program directly, line by line.
Speed of Execution	Faster, as the program is already translated into machine code.	Slower, as translation and execution occur simultaneously.
Error Detection	Errors are detected after the entire code is compiled, so debugging is done after compilation.	Errors are detected line-by-line, and execution stops at the first error.
Output	Produces an independent executable file (e.g., .exe, binary).	No separate output file; code is executed on the fly.
Portability	The compiled machine code is specific to a particular platform (e.g., Windows, macOS, Linux).	More portable since the same source code can run anywhere the interpreter is available.
Memory Usage	Requires additional memory for storing the executable.	Typically uses less memory since it doesn't produce a separate executable.
Examples	C, C++, Java (compiles to bytecode), Swift, Rust.	Python, JavaScript, Ruby, PHP, MATLAB.

1.4 Loading, Linking and Relocation (UGC-NET-DEC-2013)

- The loader loads the program into the main memory for execution of that program. It loads machine instruction and data of related programs and subroutines into the main memory, this process is known as **loading.** The loader performs loading; hence, the assembler must provide the loader with the object program.

Functions of Loader:

1. **Allocation:** It allocates memory for the program in the main memory.
2. **Linking:** It combines two or more separate object programs or modules and supplies necessary information.
3. **Relocation:** It modifies the object program so that it can be loaded at an address different from the location.
4. **Loading:** It brings the object program into the main memory for execution.

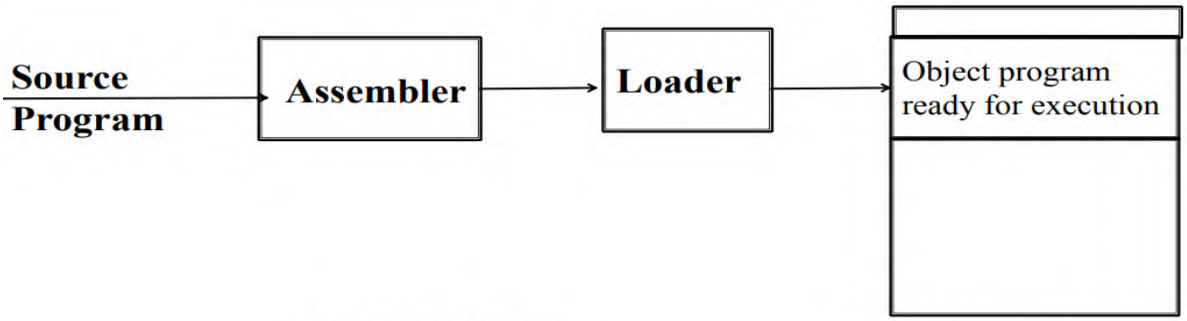


Fig.: Role of loader

- Loader is utility program which takes object code as input prepares it for execution and loads the executable code into the memory. Thus loader is actually responsible for initiating the executions process.

1. **Loading**

- Loading refers to the process of bringing the executable program or code into memory so that it can be executed by the computer's processor. The loader, a part of the operating system, handles this process.
- It prepares the program to run by placing it into memory, assigning it an appropriate address space, and ensuring that the program's code and data are available to the CPU.
    - Steps in Loading:
        - **Load the executable into memory**: The operating system loads the compiled machine code (executable) from disk into RAM.
        - **Allocate space for data and code**: The code section and data section (such as variables) of the program are assigned specific areas in memory.
        - **Set up the execution context**: The loader prepares the environment for execution, including setting up stacks, heaps, and registers.

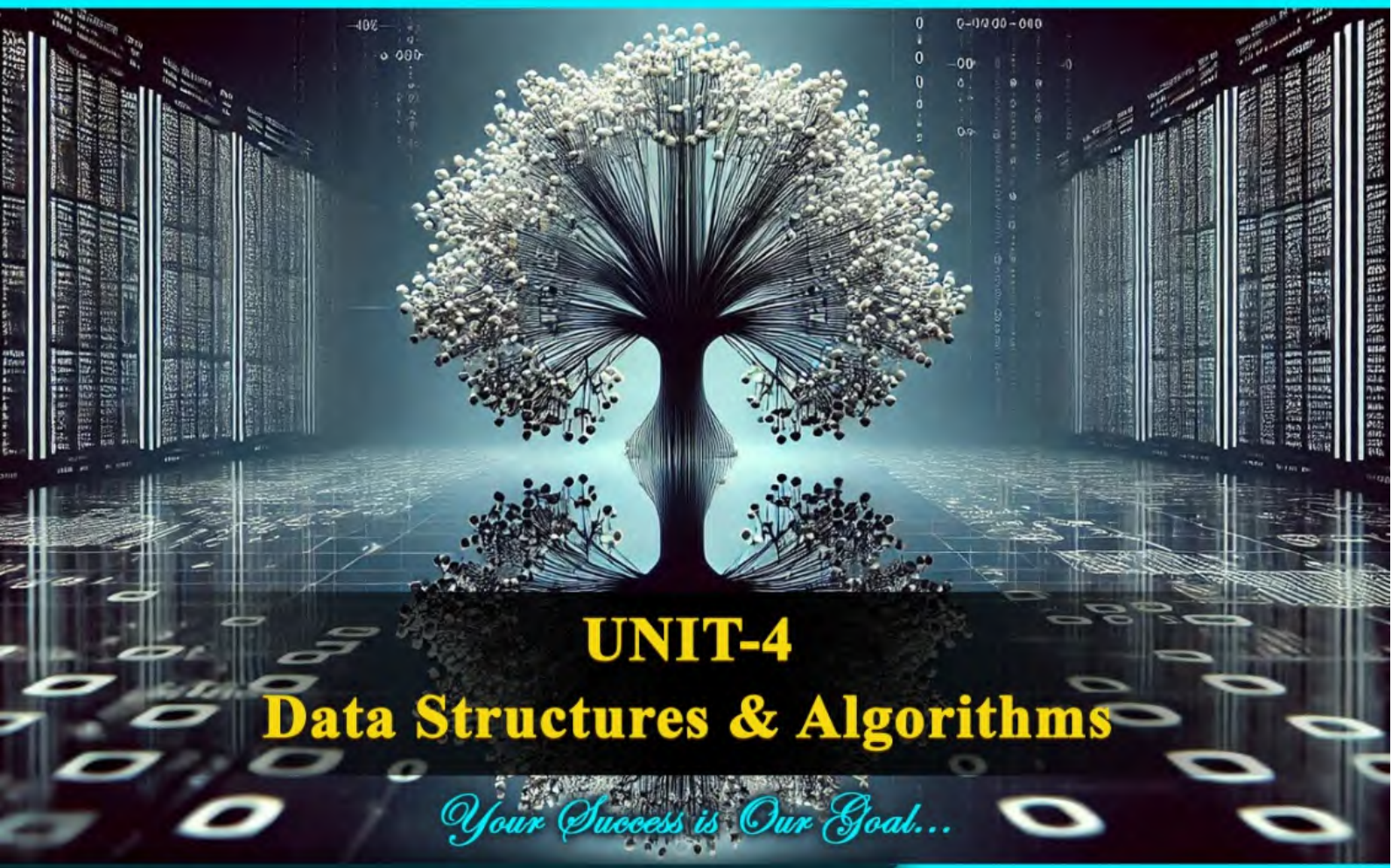
TEACHER'S CARE PUBLICATION
38/23, Vaigundaperumal Koil Street,Kancheepuram- 631502
Mobile : 95665 35080, 9786269980 Land Line : 044-2723 5080

2025-2026

PG TRB

G.O No. 219 New Syllabus

COMPUTER SCIENCE



UNIT-4
Data Structures & Algorithms

Your Success is Our Goal...

- Data Structures
- Trees and Graphs
- Performance Analysis of Algorithms
- Recurrences
- Design Techniques
- Graph Algorithms
- Advanced Algorithms:

tcaexamguide
Teacher's Care Academy
www.tcaexamguide.com
tcaexamguide@gmail.com

UNIT-VI DATA STRUCTURES & ALGORITHMS

TEACHER'S CARE ACADEMY, KANCHIPURAM

TNPSC-TRB- COMPUTER SCIENCE -TET COACHING CENTER

HEAD OFFICE: NO. 38/23, VAIGUNDA PERUMAL KOIL, SANNATHI STREET, KANCHIPURAM – 1. CELL: 9566535080

B.Off 2: 65C, Thillai Ngr(West), 4th Cross St, Trichy – 620018

B.Off 3: 266-C - Advaitha Ashram Road, Opp to New Bus Stand, Salem – 4

Trichy: 76399 67359 Salem: 93602 68118

UNIT IV: DATA STRUCTURES

SYLLABUS

Data Structures: Abstract data types, Arrays and their Applications; Sparse Matrix, Stacks, Queues, Priority Queues, Linked Lists,

Trees and Graph: Trees, Forest, Binary Tree, Threaded Binary Tree, Binary Search Tree, AVL Tree, B Tree, B+ Tree, B* Tree, Graphs, Sorting and Searching Algorithms; Hashing.

Performance Analysis of Algorithms and Recurrences: Time and Space Complexities; Asymptotic Notation, Recurrence Relations.

Design Techniques: Divide and Conquer; Dynamic Programming, Greedy Algorithms, Backtracking, Branch and Bound.

Graph Algorithms: Breadth-First Search, Depth-First Search, Shortest Paths, Maximum Flow, Minimum Spanning Trees.

Advanced Algorithms: Parallel Algorithms Algorithms, Randomized Algorithms.

BOOKS TO STUDY:

1) Classic Data Structures - D.Samanta
2) Data Structures made simple - Sathish Jain, Shashi Singh.
3) Data Types and structures - Gotlieb, C.C. and L.R.Gotlieb.
4) Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, "Fundamentals of Computer Algorithms", 2nd Edition, University Press, 2008.
5) T. H. Cormen, C. L. Leiserson, R. L. Rivest, and C. Stein, "Introduction to Algorithms", MIT Press.

UNIT IV - DATA STRUCTURES

INTRODUCTION

WHAT IS MEANT BY A DATA?

- Data is a Single (or) a set of values. (Or) Facts and statistics collected together for reference or analysis

WHAT IS MEANT BY DATA STRUCTURE?

- It is a logical or mathematical model of a particular organization of data.

(Or)

- Data Structure is a specialized format for organizing and storing data so that it can be accessed and worked with in appropriate ways to make a program efficient.
    - **Data Structure = Organized Data + Allowed Operations.**
- Data structure suitable for Data design application **(PGTRB 2019)**

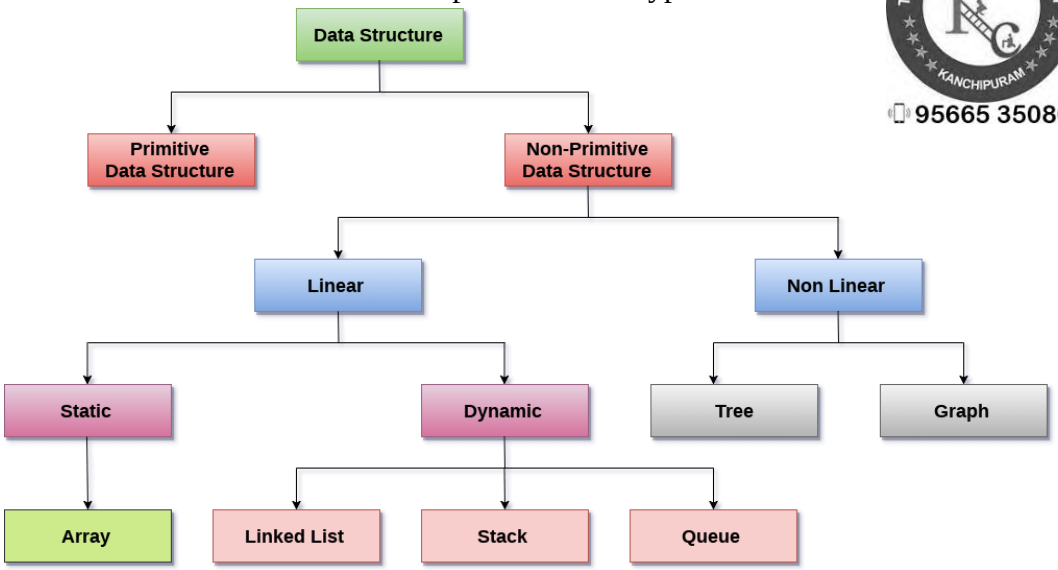
APPLICATIONS OF DATA STRUCTURE

Areas	*Type of Data Structure*
Operating System	*Arrays and Tables*
Data Base Management System	*Array, Tables, B- Trees*
Compiler Design	*Hash Tables (look up an identifier)*
Neural Network	*Graph*
Hierarchical Data Model	*Trees*

Categories of data structures:

- Two types:

1) Linear data structure → Single generic type **(UGC NET 2012)**
2) Non-linear data structure → Multiple Individual type



Extra Bytes:

1) Which one of the following is a nonlinear data structures? (PGTRB 2019)

(A) Arrays (B) Sets

(C) Queue (D) Linked list

Answer: B - Sets

1. ARRAY AND THEIR APPLICATIONS

- jjAn array is a collection of items stored at contiguous memory locations.
- The idea is to store multiple items of the same type together.

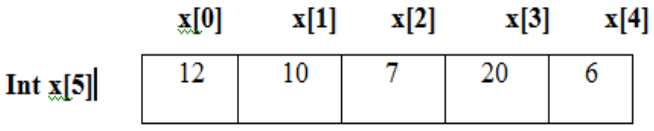


Fig : An array of size 5 containing integers

- This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array).
- Array data structure is typically used to implement hash table **(UGC NET 2023)**

1.1. ARRAYS TERMINOLOGY:

Size:

- Number of elements in an array is called the size of the array. Also called as length or dimension.

Type:

- Type of an away represents the kind of data type. Ex: int, string

Base:

- Base of an array is address of memory location where the first element in the array is located.

Range of index:

- Indices of array elements any charge can be referenced by subscript like Ai or A[i], this subscript is known as index. Index is always as integer value. Every element is identified by a subscripted or indexed variable
- Ex:
  - ✓ Int A[100]; The range of index is from 0 to 9
  - ✓ A:Array[-5….19] of integer: The Points of the rage is -5,-4,-3,….18,19.
  - ✓ Here L is the Lower Bound.
  - ✓ If the range of index varies from L…U then the size of the away can be calculated as Size(A)=U-L+1.

Word:

- It denotes the size of an element. In memory location computer can store an element of word size w. This word size varies from machine to machine such as 1 byte to 8 bytes.

1.2. OPERATIONS ON ARRAY

- The common operations can be performed on an array are
  - ✓ **Traversing-processing each element in the array.**
  - ✓ **Sorting -Organizing the elements in some order.**
  - ✓ **Searching -Finding the location of an element with a given value.**
  - ✓ **Insertion - Adding a new element.**
  - ✓ **Deletion -Removing an element.**
  - ✓ **Merging -Combining two arrays into a single array.**
- Although searching, and traversal of an array is an easy job, insertion and deletion is time consuming. The elements need to be shifted down before insertion and shifted up after deletion.

1. Traversing:

This operation is used visiting all elements in an array.

- Example: Array 'a' contains the following elements:

5	3	4	8	7			

The result of traversing is:

5
3
4
8
7

The algorithm for traversing is as follows:

1) Read the Array elements.
2) Display the elements of the array.

Algorithms: Traverse-array ( )

Input: An array A with elements

Output: According to process ( )

Steps:

1. i=L // start from first location L
2. while i<=U do // U upper bound
   1. Process (A[i])
   2. i=i+1 // move to next position

3. End while
4. Stop

Here process ( ) is an procedure which when called for an element can perform an action

2. Sorting:

- This operation if performed on an array will sort it in a specified order. The algorithm is used to store the elements of an integer array in ascending order or descending order.

Example:

Before Sort						
10	2	3	4	6	9	1

After Sort						
1	2	3	4	6	9	10

Algorithm steps:

1) Read the array elements.
2) Set the first position by comparing the first position of the array element with other array element. If the value is smaller than the first position element, then swap the elements.
3) Set the second position by comparing the elements. If the value is smaller than the second position element, then swap the elements.
4) Set the other positions likewise.

Input: An array with integer data

Output: An array with sorted element in an order according to ORDER ( )

Steps:

1. i= U

2. While i>= L do
    1. j=L // start comparing from first
    2. While j< i do
        1. If ORDER (A[j, A[j+1]) = FALSE // if A[j] and A[j+1] are not in order
            1. Swap (A[j], A[j+1]) // Interchange the elements
        2. Endif
        3. j=j+1 // Go to next statement
    3. Endwhile
    4. i=i-1
3. Endwhile
4. Stop

- Here order ( ) is a procedure to test whether two elements are in order and SWAP ( ) is a procedure to interchange the elements between two consecutive locations.

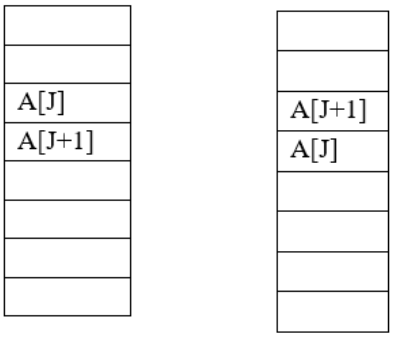


Fig: Swapping of two elements in an array

3. Searching:

- This operation is applied to search an element of interest in an array

Example:

4	5	6	8	9		

Element to search: 8

The given element is present in the position: 4

Algorithm steps:

1) Read the element to search.
2) Compare the element to the array elements.
3) If it matches then, display the position of the array.
4) Otherwise compare the entire array.
5) If match not found display the message "search is unsuccessful, key is not in the array "

Algorithm: Search_array(key)

Input: Key is the element to be searched

Output: Index of key in A or a message on failure

Steps:

1. i=L, found=0, location=0 // found=0 indicates search is not finished and unsuccessful
2. While (i<=U) and (found =0) do
    1. if compare(A[i], key) = true then
        1. Found=1
        2. Location =i
    2. Else
        1. i=i+1

3. End if

3. End while

4. If found=0 then

1. Print “search is unsuccessful; key is not in the array “

5. Else

1. Print “search is successful: key is in the array at location “, location

6. End if

7. Return (location)

8. Stop--

4. Insertion:

- This operation is used to insert an element into an array provided that the array is not full.

Example:

Array

4	5	6	8	9		

Insert -2 at position: 3

4	5	6	8	9		

Shifting one position right

4	5	-2	6	8	9	

After insertion

Fig: Insertion of an element

Algorithm steps:

1) Read the number to insert and the position to insert.
2) Shift the numbers from the specified position, one place to the right from their existing position.
3) Place the number at the vacant place.

Algorithm: insert (key, location)

Input: key is the item; location is the index of the element where it is to be stored.

Output: array enriched with key

Steps:

1. if A[U] # NULL then
    1. print “Array is full, no insertion possible”
    2. Exit

2. Else
    1. i = U
    2. While i> location do
        1. A[i+1]=A[i]
        2. i = i-1
    3. End while
    4. A[location] =key
    5. U=U+1
3. End if
4. Stop

5. Deletion:

This operation is used to delete a particular element from an array. The element will be deleted by overwriting it with its subsequent element and this subsequent element then is to be deleted.

Example: Delete the element in the position: 3

Before deletion:

4	5	6	8	9		

After deletion:

4	5	8	9	0	0

Algorithm steps:

1) Read the position to delete.
2) Shift the numbers placed after the position, where the number is to be deleted.
3) Leave the last position blank.

Algorithm: delete (key)

Input: key is the element to be deleted.

Output: slimed array without key

Steps:

1. i = search_array (a, key)
2. if i=0 then
    1. print “key is not found, no deletion”
    2. Exit
3. Else
    1. While i< U do

1. A[i] = A[i+1]
2. i = i+1

3. End while

4. End if
5. A[U] = NULL
6. U=U-1
7. Stop

6. Merging:

- Merging is an important operation when we need to compact the elements from two different arrays into a single array.

Rules:

1) Copy all the elements of first array into a new array (third array)
2) Copy the second array into the third array after the position, at which the last elements of the first array copied.

Types

- Merging can be done in two ways:
  - Merging without sorting.
  - Merging with sorting.

1
3
7

2
4
8

1
3
7
2
4
8

Fig: merging of A1 and A2 to A

Algorithm: merge (A1, A2: A)

Input: Two arrays A1 [L1…U1], A2 [l2…U2]

Output: Result array A [L…U], where L=L1 and U=U1+(U2-L2+1) when A1 is append after A2

Steps:

1. i1=L1, i2=L2; // initialization of variables
2. L=L1, U=U1+U2 –L2 +1 // initialization of lower and upper bounds of an array

3. i=L
4. Allocate memory for a[L…U]
5. while i1<U do // to copy array A1 into the first part of A
    1. A[i] = A1[i1]
    2. i=i+1, i1=i1+1
6. End while
7. While i2<=U2 do // To copy the array A2 into last part of A
    1. A[i] = A2[i2]
    2. i=i+1, i2=i2+1
8. End while
9. Stop

COMPLEXITY ANALYSIS OF OPERATIONS ON ARRAYS

Time Complexity

Operation	Best Case	Average Case	Worst Case
Traversal	O(1)	O(n)	O(n)
Insertion	O(1)	O(n)	O(n)
Deletion	O(1)	O(n)	O(n)
Search	O(1)	O(n)	O(n)
Update	O(1)	O(1)	O(1)

1.3. TYPES OF ARRAYS

- Types of arrays depend upon the number of dimensions of an array.
- The count of indices or subscripts required to access one element of an array define the dimensions of an array.

Types of arrays

1) One-dimensional Array
2) Two-dimensional Array
3) Three-dimensional Array

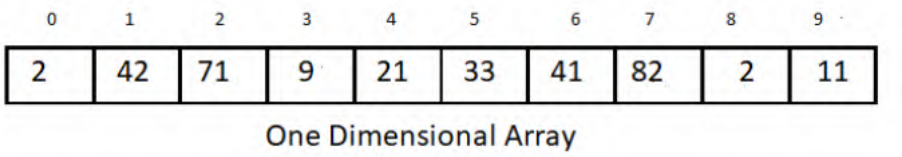
- Two dimensional and three dimensional arrays are also called multi-dimensional arrays.

1. One-dimensional Array

- In a one-dimensional array the elements are stored in contiguous memory locations where each element is accessed by using a single index value. It is a linear data structure storing all the elements in sequence.

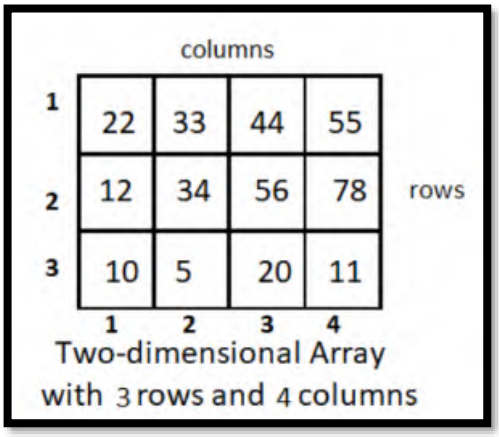
2.Two-dimensional Array

- In types of arrays, a two dimensional array is a tabular representation of data where elements are stored in rows and columns.



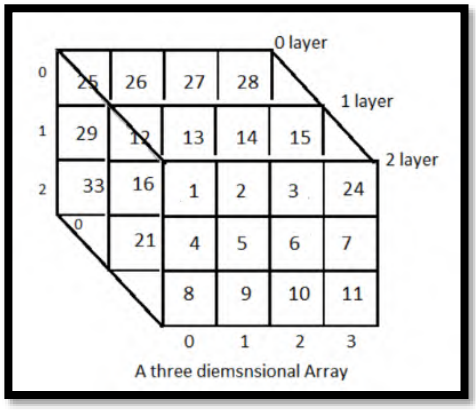
One Dimensional Array

- A two dimensional array is actually a collection of M X N elements which has M rows and N columns. To access any element in a two-dimensional array two subscripts are required for defining position of an element in a specific row and column.
- The first index is for row number and second is for column index. In the example shown the first index values row=2 column=3 is used to access element 56.



3.Three-dimensional Array

- In types of arrays, a three-dimensional array is an extension to the two dimensional array with addition of depth.
- It can be seen as a cube that has rows, columns and depth as third dimension. To access any element in a three-dimensional array three subscripts are required for position of element in a specific row, column and depth.
- The first index is for depth (dimension or layer), second is for row index and third is for column. In the example shown the index values (2,0,3) is used to access element 24.



1.4. APPLICATIONS OF ARRAYS IN DATA STRUCTURES

- **Storing and accessing data:** Arrays are used to store and retrieve data in a specific order. For example, an array can be used to store the scores of a group of students, or the temperatures recorded by a weather station.
- **Sorting:** Arrays can be used to sort data in ascending or descending order. Sorting algorithms such as bubble sort, merge sort, and quicksort rely heavily on arrays.
- **Searching:** Arrays can be searched for specific elements using algorithms such as linear search and binary search.
- **Matrices:** Arrays are used to represent matrices in mathematical computations such as matrix multiplication, linear algebra, and image processing.
- **Stacks and queues:** Arrays are used as the underlying data structure for implementing stacks and queues, which are commonly used in algorithms and data structures.
- **Graphs:** Arrays can be used to represent graphs in computer science. Each element in the array represents a node in the graph, and the relationships between the nodes are represented by the values stored in the array.
- **Dynamic programming:** Dynamic programming algorithms often use arrays to store intermediate results of sub problems in order to solve a larger problem.
- **Image processing:** Arrays can be used to represent and process images. Each element in the array represents a pixel in the image, and operations can be performed on the array to manipulate the image.
- **Numerical computations:** The application of an array is extensive in numerical computations, such as in linear algebra and signal processing. For example, a matrix can be represented as a two-dimensional array, and operations like matrix multiplication can be performed efficiently using arrays.
- **Games and simulations:** Arrays can be used to represent game boards, game pieces, and game states. They are also used in simulations to store and manipulate data over time.

1.5. ADVANTAGES & DISADVANTAGES OF ARRAYS IN DATA STRUCTURES

ADVANTAGES

- **Efficient access:** Arrays offer fast and efficient access to elements because each element can be accessed directly through its index. This makes array traversal quick and straightforward.
- **Versatility:** Arrays can be used to store any type of data like integers, characters, and strings. They can also be used to store user-defined data types, such as structures and classes.

- **Flexibility:** Arrays are used to implement other data structures like stacks, queues, trees, graphs, etc.
- **Easy to remember:** Arrays represent multiple data items of the same type using a single name. Therefore, it's easier to remember an array name than remembering the names of several variables.

DISADVANTAGES

- **Fixed-size:** The size of an array is fixed at the time of its creation, which means that once the array is created, its size cannot be changed. This can be a limitation in situations where the size of the data is not known in advance.
- **Memory wastage:** There will be a wastage of memory if we store less number of elements than the declared size because there is static memory allocation in arrays.
- **Inefficient insertion and deletion:** Arrays store data in contiguous memory locations, which makes deletion and insertion very difficult to implement. All the elements after insertion or deletion must be shifted to accommodate the new element or fill in the gap. This can be a time-consuming process, especially for large arrays.
- **Homogeneous data:** Arrays can only store elements of the same data type, which can be a limitation in situations where the user needs to store data of different types.
- **No built-in support for dynamic resizing:** While some programming languages provide built-in support for dynamic resizing of arrays, many do not. In those cases, the developer may have to implement their own resizing logic, which can be complex and error-prone.

EXCERICE 1:

1) **What is Data Structure?**
   a) Address of the variable
   b) Subset of all variables
   c) The memory representation of data
   d) The type of the variable

2) **Which of the following is a collection of heterogeneous elements?**
   a) Array
   b) Structure
   c) Stack
   d) Queue

3) **_____is a linear data structure**
   a) Tree
   b) Array
   c) Graph
   d) None of these.

4) **The Smallest element of an array index is _____**
   a) Smallest Bound
   b) Lower Bound
   c) First Bound
   d) Higher Bound

TCA

5) **Two Dimensional Array are also called a**

a) Table Array
b) Matrix Array
c) Both A & B
d) None of above

6) **An Array of n elements will be declared in c as**

a) Array [n+1]
b) Array [n-1]
c) Array [n]
d) Array

7) **The First Element of a 0 based array can be accessed by**

a) Array [0]
b) Array [n-1]
c) Array
d) Both A & C

8) **Which of the following data structure can't store the nonhomogeneous data elements?**

a) Arrays
b) Stacks
c) Records
d) None of the above

9) **In an array range specifies_____**

a) Scope of the Array
b) Number of the Elements in the Array
c) The Group of the Array
d) Size-1 of the array

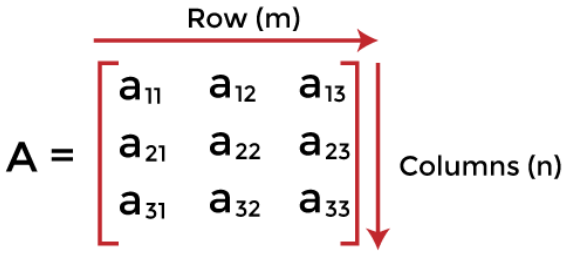
10) **Which of the following data structures are indexed structures?**

a) Linear arrays
b) Linked lists
c) Both (A) & (B)
d) None of above

2.SPARSE MATRIX

2.1. WHAT IS A MATRIX?

- A matrix can be defined as a two-dimensional array having 'm' rows and 'n' columns. A matrix with m rows and n columns is called m × n matrix. It is a set of numbers that are arranged in the horizontal or vertical lines of entries.
- For example:



2.2. WHAT IS SPARSE MATRIX?

- A matrix is a two-dimensional data object made of m rows and n columns, therefore having total m x n values. If most of the elements of the matrix have 0 value, then it is called a sparse matrix.

- There are the following benefits of using the sparse matrix -
    1) **Storage:** We know that a sparse matrix contains lesser non-zero elements than zero, so less memory can be used to store elements. It evaluates only the non-zero elements.
    2) **Computing time:** In the case of searching in sparse matrix, we need to traverse only the non-zero elements rather than traversing all the sparse matrix elements. It saves computing time by logically designing a data structure traversing non-zero elements.

2.3. REPRESENTATION OF SPARSE MATRIX

- Now, let's see the representation of the sparse matrix. The non-zero elements in the sparse matrix can be stored using triplets that are rows, columns, and values.
- There are two ways to represent the sparse matrix that are listed as follows: -
    1) Array representation
    2) Linked list representation

2.3.1. ARRAY REPRESENTATION OF THE SPARSE MATRIX

- Representing a sparse matrix by a 2D array leads to the wastage of lots of memory. This is because zeroes in the matrix are of no use, so storing zeroes with non-zero elements is wastage of memory. To avoid such wastage, we can store only non-zero elements. If we store only non-zero elements, it reduces the traversal time and the storage space.
- In 2D array representation of sparse matrix, there are three fields used that are named as

ROW	COL	VALUE

- **Row**: It is the index of a row where a non-zero element is located in the matrix.
- **Column:** It is the index of the column where a non-zero element is located in the matrix.
- **Value**: It is the value of the non-zero element that is located at the index (row, column).

Example

- Let's understand the array representation of sparse matrix with the help of the example given below
- Consider the sparse matrix

Sparse Matrix →

	0	1	2	3
0	0	4	0	5
1	0	0	3	6
2	0	0	2	0
3	2	0	0	0
4	1	0	0	0

- In the above figure, we can observe a 5x4 sparse matrix containing 7 non-zero elements and 13 zero elements. The above matrix occupies 5x4 = 20 memory space. Increasing the size of matrix will increase the wastage space.
- **The tabular representation of the above matrix is given below**

Table Structure

Row	Column	Value
0	1	4
0	3	5
1	2	3
1	3	6
2	2	2
3	0	2
4	0	1
5	4	7

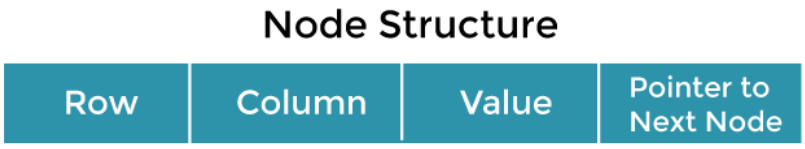
- In the above structure, first column represents the rows, the second column represents the columns, and the third column represents the non-zero value. The first row of the table represents the triplets. The first triplet represents that the value 4 is stored at 0th row and 1st column. Similarly, the second triplet represents that the value 5 is stored at the 0th row and 3rd column. In a similar manner, all triplets represent the stored location of the non-zero elements in the matrix.
- The size of the table depends upon the total number of non-zero elements in the given sparse matrix. Above table occupies 8x3 = 24 memory space which is more than the space occupied by the sparse matrix. So, what's the benefit of using the sparse matrix? Consider the case if the matrix is 8*8 and there are only 8 non-zero elements in the matrix, then the space occupied by the sparse matrix would be 8*8 = 64, whereas the space occupied by the table represented using triplets would be 8*3 = 24.

2.3.2. LINKED LIST REPRESENTATION OF THE SPARSE MATRIX

- In a linked list representation, the linked list data structure is used to represent the sparse matrix. The advantage of using a linked list to represent the sparse matrix is that the complexity of inserting or deleting a node in a linked list is lesser than the array.
- Unlike the array representation, a node in the linked list representation consists of four fields. The four fields of the linked list are given as follows -

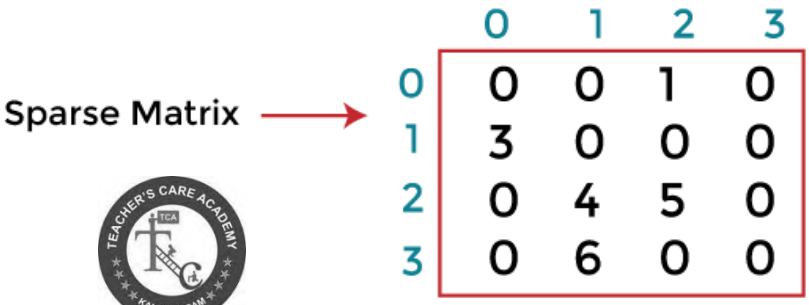
- **Row:** It represents the index of the row where the non-zero element is located.
- **Column:** It represents the index of the column where the non-zero element is located.
- **Value:** It is the value of the non-zero element that is located at the index (row, column).
- **Next node**: It stores the address of the next node.

- The node structure of the linked list representation of the sparse matrix is shown in the below image



- Example
- Let's understand the linked list representation of sparse matrix with the help of the example given below

Consider the sparse matrix



- In the above figure, we can observe a 4x4 sparse matrix containing 5 non-zero elements and 11 zero elements. Above matrix occupies 4x4 = 16 memory space. Increasing the size of matrix will increase the wastage space.
- The linked list representation of the above matrix is given below -

- In the above figure, the sparse matrix is represented in the linked list form. In the node, the first field represents the index of the row, the second field represents the index of the column, the third field represents the value, and the fourth field contains the address of the next node.
- In the above figure, the first field of the first node of the linked list contains 0, which means 0^{th} row, the second field contains 2, which means 2^{nd} column, and the third field contains 1 that is the non-zero element. So, the first node represents that element 1 is stored at the 0^{th} row-2^{nd} column in the given sparse matrix. In a similar manner, all of the nodes represent the non-zero elements of the sparse matrix.

2.4. DENSE MATRIX

- A dense matrix is the opposite of a sparse matrix. In a dense matrix, most of the elements are non-zero. This means that the matrix is fully populated, and there are relatively few zero elements.

2.4.1. CHARACTERISTICS OF DENSE MATRICES:

- **Fully Populated:** In a dense matrix, almost all elements have meaningful values, and there are very few zero elements.
- **Memory Intensive:** Dense matrices require a significant amount of memory to store all their elements, especially for large matrices.
- **Regular Structure**: Dense matrices have a regular structure where every row has the same number of elements and every column has the same number of elements.
- Example of a Dense Matrix:

  Consider the following 3x3 dense matrix:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

2.4.2. OPERATIONS ON DENSE MATRICES:

- **Arithmetic Operations:** Addition, subtraction, multiplication, and division of dense matrices involve performing corresponding operations on each pair of elements.
- **Linear Algebra Operations:** Dense matrices are used extensively in linear algebra operations such as matrix inversion, determinant calculation, and solving systems of linear equations.
- **Matrix Decompositions:** Techniques like LU decomposition, QR decomposition, and Singular Value Decomposition (SVD) are commonly applied to dense matrices.

2.4.3. USE CASES:

- **Image Processing:** Dense matrices are often used to represent images, where each pixel corresponds to an element in the matrix.
- **Numerical Simulations:** Dense matrices are employed in simulations of physical systems, financial modeling, and other numerical computations.
- **Deep Learning:** In neural networks, dense layers perform matrix-vector multiplications, where both weights and inputs are represented as dense matrices.

EXCERICE 2:

1) **What is a sparse matrix?**
   a) A matrix with a large number of non-zero elements
   b) A matrix with a large number of zero elements
   c) A matrix with only one row
   d) A matrix with only one column

2) **Which of the following is a common representation format for sparse matrices?**
   a) Dense matrix
   b) Coordinate List (COO)
   c) Binary tree
   d) Linked list

3) **Which data structure is commonly used to store non-zero elements in a sparse matrix?**
   a) List of Lists (LIL)
   b) Array
   c) Binary tree
   d) Stack

4) **Which of the following operations is typically more efficient on a sparse matrix compared to a dense matrix?**
   a) Matrix addition
   b) Matrix multiplication
   c) Matrix transposition
   d) Matrix inversion

5) **Which of the following statements about converting a dense matrix to a sparse matrix is true?**
   a) The resulting sparse matrix will have more non-zero elements than the dense matrix.
   b) The process involves identifying and storing only the non-zero elements.
   c) The resulting sparse matrix will have a larger memory footprint than the dense matrix.
   d) The process requires converting all elements to zero.

6) **In which application area are sparse matrices commonly used?**
   a) Image processing
   b) Cryptography
   c) Real-time gaming
   d) Audio processing

7) **Which data structure provides fast access to individual elements in a sparse matrix?**
   a) Array
   b) Linked List
   c) Dictionary
   d) Stack

8) **Which application domain extensively uses sparse matrices for computations?**
   a) Image processing
   b) Web development
   c) Natural language processing
   d) Mobile app development

9) **What is the time complexity of converting a sparse matrix to triples form?**
   a) O (n)
   b) O (n log n)
   c) O (m + n)
   d) O (m * n)

10) In triples form representation of a sparse matrix, what information is stored for each non-zero element?

a) Row index, column index, and value
b) Row index, column index, and matrix dimension
c) Element position and value
d) Element position and matrix dimension

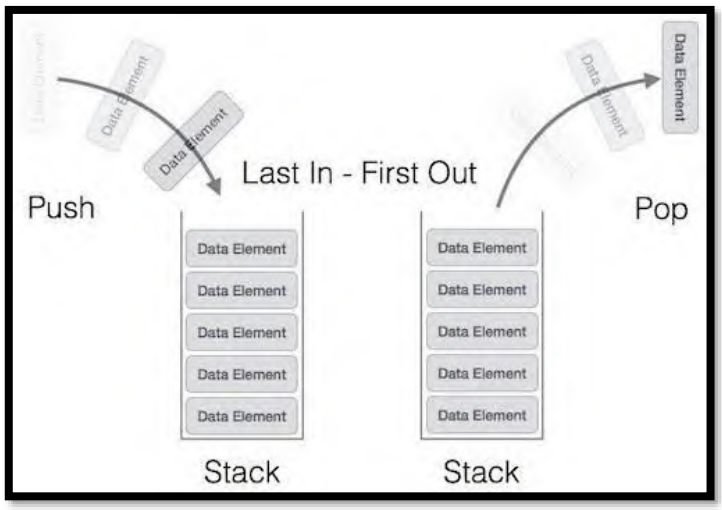
3. STACK

3.1. INTRODUCTION

- Stack is a linear data structure and which items are added or from which items can be deleted at one end only. This end is called as top.
- Stack follows the LIFO policy because the element which is added last will be removed first.
- Ex:
  - Printer tray
  - Shipment in a cargo.
  - Order supply in a restaurant.

3.1.1. DEFINITION

- A stack is an ordered collection of homogeneous data element where the insertion & deletion operation take place at one end only. **(PGTRB 2022)**
- The insertion and deletion operations in case of stack are specially termed as PUSH and POP.
- Push() → Insert new elements into the Stack
- Pop() → Remove an element from the stack
- Where the operations are performed is known as TOP of the stack.
- An element in a stack is termed as ITEM.
- The maximum number of elements that a stack can accommodate is termed as SIZE.



TEACHER'S CARE PUBLICATION
38/23, Vaigundaperumal Koil Street,Kancheepuram- 631502
Mobile : 95665 35080, 9786269980 Land Line : 044-2723 5080

2025-2026

PG TRB

G.O No. 219 New Syllabus

COMPUTER SCIENCE

UNIT-5
Data Communication & Computer Networks

Your Success is Our Goal...

- Data Communication
- Computer Networks
- Network Models
- Functions of OSI
- Functions of TCP/IP Layers
- World Wide Web (WWW)

TEACHER'S CARE ACADEMY, KANCHIPURAM

TNPSC-TRB- COMPUTER SCIENCE -TET COACHING CENTER

HEAD OFFICE: NO. 38/23, VAIGUNDA PERUMAL KOIL, SANNATHI STREET, KANCHIPURAM – 1. CELL: 9566535080

B.Off 2: 65C, Thillai Ngr(West), 4th Cross St, Trichy – 620018

B.Off 3: Vijiyaraghavachariar Memorial Hall(Opp to Sundar Lodge), Salem

Trichy : 76399 67359 **Salem : 93602 68118**

UNIT-5

DATA COMMUNICATION AND COMPUTER NETWORKS

SYLLABUS

Data Communication: Components of a Data Communication System, Simplex, Half- Duplex and Duplex Modes of Communication; Analog and Digital Signals; Noiseless and Noisy Channels; Bandwidth, Throughput and Latency; Digital and Analog Transmission; Data Encoding and Modulation Techniques; Broadband and Baseband Transmission; Multiplexing, Transmission Media, Transmission Errors, Error Handling Mechanisms.

Computer Networks: Network Topologies, Local Area Networks, Metropolitan Area Networks, Wide Area Network, Wireless Networks, Internet.

Network Models: Layered Architecture, OSI Reference Model and its Protocols; TCP/IP Protocol Suite, Physical, Logical, Port and Specific Addresses; Switching Techniques.

Functions of OSI and TCP/IP Layers: Framing, Error Detection and Correction; Flow and Error Control; Sliding Window Protocol, HDLC, Multiple Access – CSMA/CD, CSMA/CA, Reservation, Polling, Token Passing, FDMA, CDMA, TDMA, Network Devices, Backbone Networks, Virtual LANs.

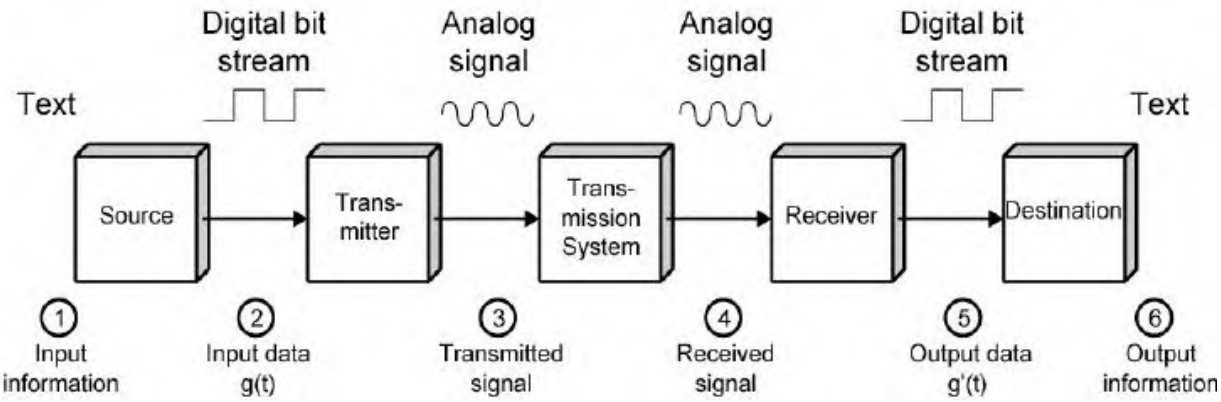
World Wide Web (WWW): Uniform Resource Locator (URL), Domain Name Service (DNS), Resolution–Mapping Names to Addresses and Addresses to Names; Electronic Mail Architecture, SMTP, POP and IMAP; TELNET and FTP.

CHAPTER 1

DATA COMMUNICATION

1.1 COMPONENTS OF DATA COMMUNICATION SYSTEM

- Sending or Receiving information, such as speaking, writing, telephone lines , computers or using some other medium is communication.
- The **communication system** basically deals with the transmission of information from one point to another using the well-defined steps which are carried out in sequential manner. The system for data transmission makes use of the sender and destination address, In this other so many elements are also there that allows it to transfer data from one set of point to another set of point after dividing the **elements of communication system** in groups and these interface elements acts as the main **component for data communication**.



- A data communication system comprises several components that work together to facilitate the exchange of data between two or more devices. These components ensure that data is transmitted accurately, efficiently, and securely. The main components of a data communication system include:

1. Message:

- The message is the information or data that needs to be transmitted from the source to the destination.
- It can take various forms, such as text, numbers, images, audio, video, or any other type of digital data.

2. Sender/Transmitter:

- The sender or transmitter is the device that originates and sends the message.
- It converts the message into electrical, electromagnetic, or optical signals suitable for transmission over the communication channel.

3. Receiver:

- The receiver is the device that receives the transmitted signals from the sender.
- It converts the received signals back into the original message format for interpretation by the destination device.

4. Communication Channel/Medium:

- The communication channel or medium is the physical or logical pathway through which the signals travel from the sender to the receiver.
- It can be wired (e.g., twisted-pair cables, coaxial cables, optical fibers) or wireless (e.g., radio waves, microwaves, infrared).
- The choice of communication channel depends on factors such as distance, data rate, cost, and environmental conditions.

5. Protocol:

- A protocol is a set of rules, conventions, and standards that governs how data is transmitted and received between devices in a network.
- It defines parameters such as data format, error detection and correction, flow control, and addressing.
- Protocols ensure interoperability and reliable communication between different devices and systems.

6. Encoder/Decoder:

- Encoders and decoders are responsible for converting the message into a format suitable for transmission and then back into its original form upon reception.
- They may perform functions such as data compression, encryption, modulation, and demodulation.

7. Modem (Modulator-Demodulator):

- A modem is a device that modulates digital signals into analog signals for transmission over analog communication channels and demodulates analog signals back into digital signals upon reception.
- It enables communication between digital devices over analog communication networks such as telephone lines.

8. Switching and Routing Equipment:

- Switches and routers are network devices that direct data traffic between multiple devices and networks.
- They ensure that data packets are delivered to their intended destinations efficiently and securely.
- Switches operate at the data link layer (Layer 2) of the OSI model, while routers operate at the network layer (Layer 3).

9. Multiplexers/Demultiplexers:

- Multiplexers combine multiple signals or data streams into a single transmission channel, allowing for more efficient use of bandwidth.
- Demultiplexers separate the combined signals back into their original individual signals upon reception.

10. Terminal Equipment:

- Terminal equipment includes devices such as computers, terminals, printers, scanners, and other end-user devices that generate or consume data.
- They interact with the communication system to send or receive messages.
- These components work together to enable effective communication and data exchange between devices across various communication networks and systems. Each component plays a specific role in the transmission, reception, processing, and delivery of data within the communication system.

1.2. MODES OF COMMUNICATION (TRANSMISSION MODE)

- There are several modes of communication that define how data is transmitted between two communicating parties. These modes determine the direction of data flow and the interaction between the sender and receiver. The most common modes of communication include:

1. Simplex Mode:

- In simplex mode, communication occurs in only one direction, either from the sender to the receiver or from the receiver to the sender.
- The sender or transmitter can only transmit data, while the receiver can only receive data.
- Examples of simplex mode include television broadcasting and keyboard input to a computer.

2. Half-Duplex Mode:

- In half-duplex mode, communication can occur in both directions, but not simultaneously.
- A communication channel is shared between the sender and receiver, allowing both parties to transmit and receive data.
- However, only one party can transmit at a time, while the other party listens or receives.
- Walkie-talkies and two-way radios operate in half-duplex mode.

3. Full-Duplex Mode:

- In full-duplex mode, communication occurs in both directions simultaneously.
- Each party can transmit and receive data independently without any interference.
- Dedicated channels are used for transmitting and receiving, allowing for simultaneous communication.
- Examples of full-duplex mode include telephone conversations and most modern computer networks.

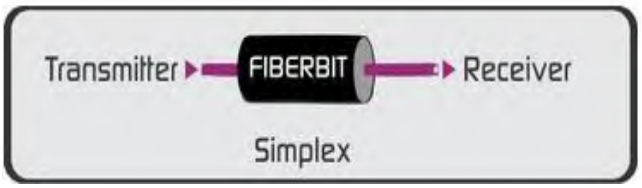
4. Asynchronous Mode:

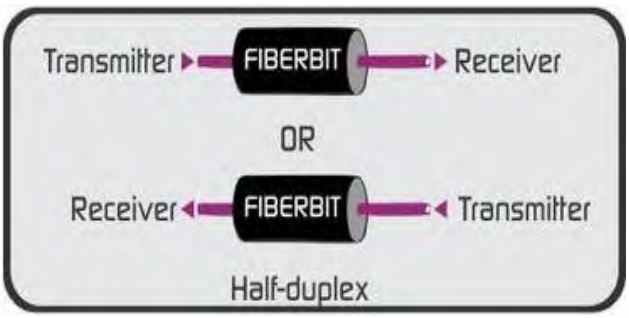
- In asynchronous mode, data transmission is not synchronized with a common clock signal between the sender and receiver.
- Each data character is preceded by start and stop bits to indicate the beginning and end of the data transmission.
- Asynchronous communication is commonly used in serial communication interfaces, such as RS-232.

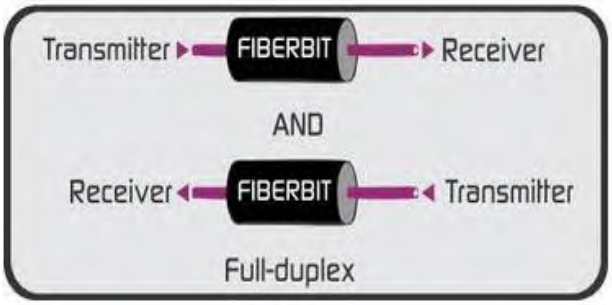
5. Synchronous Mode:

- In synchronous mode, data transmission is synchronized with a common clock signal shared between the sender and receiver.
- Data is transmitted in blocks or frames, with each frame preceded by synchronization bits or headers.
- Synchronous communication allows for higher data rates and more efficient bandwidth utilization.
- Examples of synchronous communication protocols include Ethernet and SONET/SDH.

- transfer speed, reliability, and cost considerations.







BASIS FOR COMPARISON	SIMPLEX	HALF DUPLEX	FULL DUPLEX
Direction of Communication	Communication is unidirectional.	Communication is two-directional but, one at a time.	Communication is two directional and done simultaneously.
Send/Receive	A sender can send data but, cannot receive.	A sender can send as well as receive the data but one at a time.	A sender can send as well as receive the data simultaneously.
Performance	The half-duplex and full duplex yields better performance than the Simplex.	The full duplex mode yields higher performance than half duplex.	Full duplex has better performance as it doubles the utilization of bandwidth.
Example	Keyboard and monitor.	Walkie-Talkies.	Telephone.

1.3. METHODS OF TRANSMITTING DATA

- **Serial Communication** and **Parallel Communication** are two fundamental methods of transmitting data between devices or components in a computer system. Here's a breakdown of both:

1.3.1. Serial Communication

- In serial communication, data is transmitted one bit at a time over a single channel or wire. This method is often used for long-distance communication. **(PG TRB 2022)**

Key Characteristics:

1. **Bit-by-Bit Transmission**: Data is sent in a sequence of bits (one bit at a time) through a single communication channel.
2. **Single Channel**: Typically uses only one wire or communication path for transmitting data.
3. **Slower Speed (in terms of bit rate)**: Because data is transmitted one bit at a time, serial communication is typically slower than parallel communication.
4. **Long-Distance Transmission**: Serial communication is more efficient for long distances because there is less signal degradation and less interference when compared to parallel communication.

5. **Cost-Effective**: Fewer wires or cables are required, making it cheaper and simpler to implement for longer distances.
6. **Examples**:
    - USB (Universal Serial Bus)
    - RS-232
    - I2C
    - SPI
    - Ethernet (in certain applications)

Types of Serial Communication:

- **Half-Duplex**: Data can be sent and received, but not at the same time. (e.g., walkie-talkies).
- **Full-Duplex**: Data can be sent and received simultaneously. (e.g., telephone lines, modern Ethernet).

PG TRB (2022) Question

Q1. Serial transmission rate is specified in:

A) bits per second B) bits per minute
C) bits per hour D) bits per millisecond

Ans: A) bits per second

1.3.2. Parallel Communication

- In parallel communication, multiple bits of data are transmitted simultaneously over multiple channels or wires. This allows for faster transmission but is limited by distance and other factors.

Key Characteristics:

1. **Multiple Bits at Once**: Data is transmitted multiple bits at a time (e.g., 8 bits in parallel) over multiple channels or wires.
2. **Multiple Channels**: Requires more wires, one for each bit of data.
3. **Faster Speed (for short distances)**: Since multiple bits are transmitted simultaneously, parallel communication can be faster for short distances.
4. **Signal Degradation Over Distance**: As the distance increases, signal degradation and interference can occur, making parallel communication less effective for long distances.
5. **More Complex and Expensive**: Requires more physical connections (wires or traces), which makes it more costly and complex for long distances.
6. **Examples**:
    - **Parallel ports** (e.g., old printer ports)
    - **Internal computer buses** (e.g., CPU to memory, PCI buses)

- **SATA** (used in internal drives, though modern versions have switched to serial).

Types of Parallel Communication:

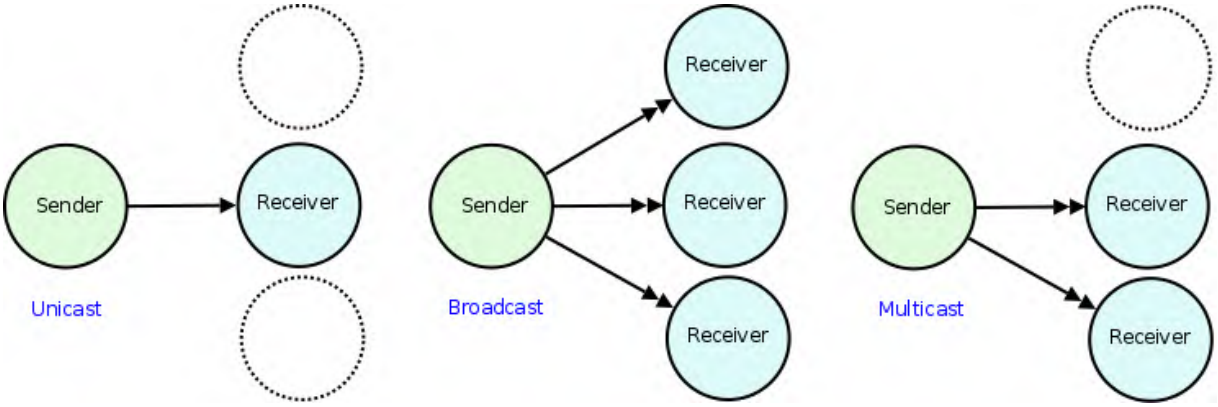
- **Synchronous**: Data is transferred in sync with a clock signal.
- **Asynchronous**: Data is transferred without the need for a clock signal, relying on start and stop bits for synchronization.

1.3.3. Serial vs. Parallel Communication

Feature	Serial Communication	Parallel Communication
Data Transfer	One bit at a time	Multiple bits at once
Number of Wires	One (single wire for data transmission)	Multiple wires (one per bit)
Speed	Slower (limited by the clock speed)	Faster (more bits transmitted at once)
Distance	Suitable for long-distance transmission	Suitable for short-distance transmission
Complexity	Simpler, fewer wires	More complex, more wires
Cost	Lower cost (less wiring needed)	Higher cost (due to more wiring)
Signal Interference	Less interference over long distances	More susceptible to interference and noise
Use Cases	USB, RS-232, Ethernet, Serial Ports	Parallel ports, internal buses (e.g., PCI)

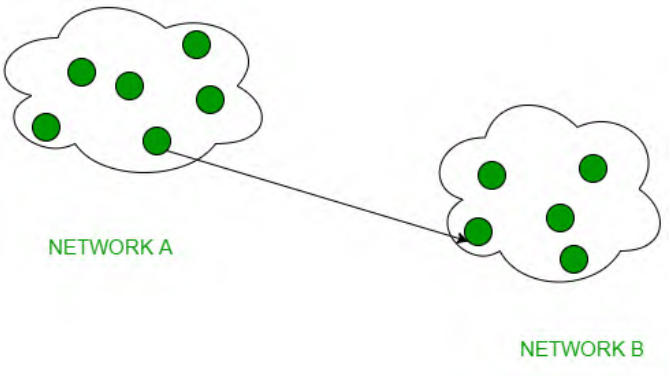
1.4. WAY OF COMMUNICATION

- The way of communication can be either of the following
  - Unicast (one to one communication)
  - Broadcast (One to all communication)
  - Multicast (One to many Communication)



1. Unicast

- This type of information transfer is useful when there is a participation of single sender and single recipient. So, in short, you can term it as a one-to-one transmission.
- For example, a device having IP address 10.1.2.0 in a network wants to send the traffic stream(data packets) to the device with IP address 20.12.4.2 in the other network, then unicast comes into the picture. This is the most common form of data transfer over the networks.

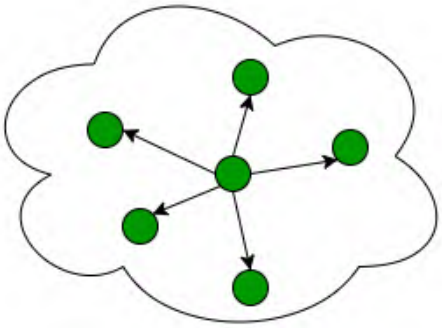


2. Broadcast

- Broadcasting transfer (one-to-all) techniques can be classified into two types :

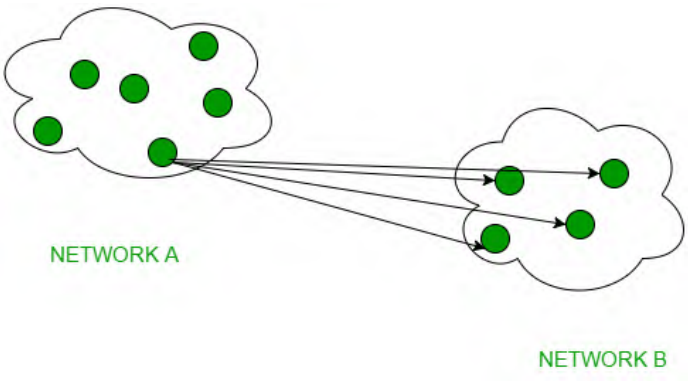
a. Limited Broadcasting

- Suppose you have to send stream of packets to all the devices over the network that you reside, this broadcasting comes handy.
- For this to achieve, it will append 255.255.255.255 (all the 32 bits of IP address set to 1) called as **Limited Broadcast Address** in the destination address of the datagram (packet) header which is reserved for information transfer to all the recipients from a single client (sender) over the network.



b. Direct Broadcasting

- This is useful when a device in one network wants to transfer packet stream to all the devices over the other network.
- This is achieved by translating all the Host ID part bits of the destination address to 1, referred as **Direct Broadcast Address** in the datagram header for information transfer.



- This mode is mainly utilized by television networks for video and audio distribution.

  One important protocol of this class in Computer Networks is **Address Resolution Protocol (ARP)** that is used for resolving IP address into physical address which is necessary for underlying communication.

3. Multicast

- In multicasting, one/more senders and one/more recipients participate in data transfer traffic. In this method traffic recline between the boundaries of unicast (one-to-one) and broadcast (one-to-all).
- Multicast lets server's direct single copies of data streams that are then simulated and routed to hosts that request it. IP multicast requires support of some other protocols like **IGMP (Internet Group Management Protocol), Multicast routing** for its working. Also in Classful IP addressing **Class D** is reserved for multicast groups.

4. Point-to-Point Communication:

- In point-to-point communication, data is transmitted between two individual nodes, typically over a dedicated communication link.
- Examples include serial communication between two devices using cables or wireless communication between two mobile devices.

5. Client-Server Communication:

- In client-server communication, one or more client devices request services or resources from a central server.
- The server responds to client requests by providing the requested data or performing the requested tasks.
- This model is prevalent in networked environments, such as web browsing, email services, and file sharing.

6. Peer-to-Peer (P2P) Communication:

- Peer-to-peer communication enables direct interaction between individual nodes without the need for a central server.
- Each node can act as both a client and a server, exchanging data and resources with other peers on the network.
- P2P networks are commonly used for file sharing, distributed computing, and decentralized applications.

7. Inter-Process Communication (IPC):

- IPC facilitates communication between different processes or threads running on the same or different devices within a network.
- Methods such as pipes, sockets, shared memory, and message passing are used for IPC in operating systems and distributed computing environments.

8. Remote Procedure Call (RPC):

- RPC allows a process to execute code or invoke procedures on a remote system as if they were local.
- It abstracts the complexities of network communication, enabling seamless interaction between distributed components.
- RPC is widely used in client-server applications, distributed systems, and remote administration tasks.

CHAPTER 1.1, 1.2, 1.3 & 1.4 MCQ

1. Which component is responsible for converting digital signals to analog signals for transmission over analog communication channels?

A) Modem B) Router
C) Switch D) Gateway

2. Which component of data communication is responsible for encoding data into signals suitable for transmission?

A) Sender B) Receiver
C) Transmission medium D) Modem

3. Which communication mode is best suited for applications where real-time bidirectional communication is required?

(a) Simplex (b) Half Duplex
(c) Full Duplex (d) Multiplexing

4. Which communication mode does not require a common clock signal between the sender and receiver?

(a) Asynchronous (b) Synchronous
(c) Full Duplex (d) Simplex

5. Which communication mode is more suitable for high-speed data transfer applications that require precise timing and synchronization?

(a) Asynchronous (b) Synchronous
(c) Full Duplex (d) Simplex

6. Which type of communication is often used for sending important network control messages, such as routing updates and network management commands?

(a) Unicast (b) Multicast
(c) Broadcast (d) AnyCast

7. Which type of communication is more bandwidth-efficient when transmitting data to multiple recipients located in different parts of the network?

(a) Unicast (b) Multicast
(c) Broadcast (d) AnyCast

8. Which communication mode is commonly used in applications where data transfer rates are relatively low and timing precision is not critical?

(a) Asynchronous (b) Synchronous

(c) Full Duplex (d) Simplex

9. Which of the following is an example of a point-to-point communication link?

(a) Sending an email to multiple recipients

(b) Making a phone call from one phone to another

(c) Broadcasting a message on social media

(d) Sharing a file on a network drive

10. Which of the following is an example of client-server communication?

(a) Sending an email to multiple recipients

(b) Broadcasting a message on social media

(c) Accessing a website hosted on a remote server

(d) Sharing files on a peer-to-peer network

11. What is the primary characteristic of a Peer-to-Peer (P2P) network?

(a) Centralized control (b) Client-server architecture

(c) Decentralized architecture (d) Hierarchical structure

12. Communication between a computer and a keyboard involves _____ transmission

(a) Simplex (b) Automatic

(c) Full-duplex (d) Half-duplex

13. Communication channel is shared by all the machines on the network in

(a) Multicast network (b) Broadcast network

(c) Unicast network (d) None of these

14. In a peer to peer network, who controls the devices?

(a) First computer (b) Last computer

(c) No device controls (d) All devices control

15. Choose the correct statements about a Client-Server networking model.

(a) Server software is costly to acquire and renew

(b) Dedicated admin people are required to monitor and grant resources to normal computers or nodes. So this client-server network is highly secure.

(c) The hardware of a Server PC is costly

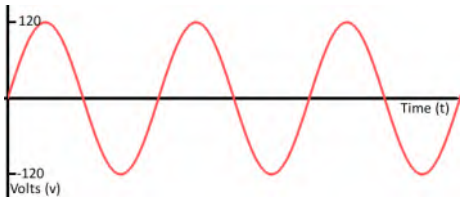
(d) All the above

1.5. SIGNAL

- A signal is an electromagnetic or electrical current that carries data from one system or network to another.
- In electronics, a signal is often a time-varying voltage that is also an electromagnetic wave carrying information, though it can take on other forms, such as current.
- There are two main types of signals used in electronics: analog and digital signals.

a.) Analog Signal

- Because a signal varies over time, it's helpful to plot it on a graph where time is plotted on the horizontal, *x*-axis, and voltage on the vertical, *y*-axis.
- Looking at a graph of a signal is usually the easiest way to identify if it's analog or digital; a time-versus-voltage graph of an analog signal should be **smooth** and **continuous**.



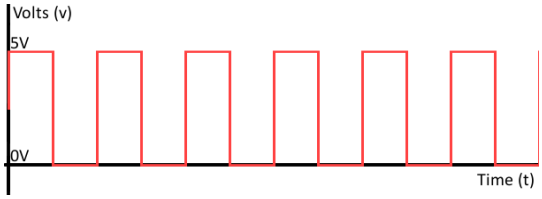
- While these signals may be limited to a **range** of maximum and minimum values, there are still an infinite number of possible values within that range.

For example:

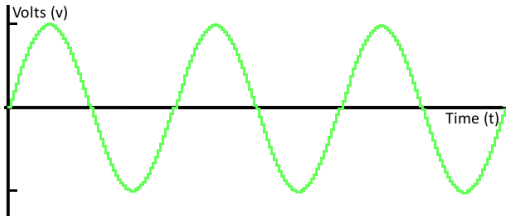
- The analog voltage coming out of your wall socket might be clamped between -120V and +120V, but, as you increase the resolution more and more, you discover an infinite number of values that the signal can actually be (like 64.4V, 64.42V, 64.424V, and infinite, increasingly precise values).

b.) Digital Signals

- Digital signals must have a finite set of possible values.
- The number of values in the set can be anywhere between two and a-very-large-number-that's-not-infinity.
- Most commonly digital signals will be one of **two values** -- like either 0V or 5V.
- Timing graphs of these signals look like **square waves**.



- Or a digital signal might be a discrete representation of an analog waveform. Viewed from afar, the wave function below may seem smooth and analog, but when you look closely there are tiny discrete **steps** as the signal tries to approximate values:



- That's the big difference between analog and digital waves. Analog waves are smooth and continuous, digital waves are stepping, square, and discrete.

c.) Key Differences:

- An analog signal is a continuous signal whereas Digital signals are time separated signals.
- Analog signal is denoted by sine waves while It is denoted by square waves
- Analog signal uses a continuous range of values that help you to represent information on the other hand digital signal uses discrete 0 and 1 to represent information.
- The analog signal bandwidth is low while the bandwidth of the digital signal is high.
- Analog instruments give considerable observational errors whereas Digital instruments never cause any kind of observational errors.
- Analog hardware never offers flexible implementation, but Digital hardware offers flexibility in implementation.
- Analog signals are suited for audio and video transmission while Digital signals are suited for Computing and digital electronics.

d.) Characteristics Of Analog Signal

Here, are essential characteristics of Analog Signal

- These type of electronic signals are time-varying
- Minimum and maximum values which is either positive or negative.
- It can be either periodic or non-periodic.
- Analog Signal works on continuous data.
- The accuracy of the analog signal is not high when compared to the digital signal.
- It helps you to measure natural or physical values.
- Analog signal output form is like Curve, Line, or Graph, so it may not be meaningful.

e.) Characteristics of Digital Signals

Here, are essential characteristics of Digital signals

- Digital signal are continuous signals
- This type of electronic l signals can be processed and transmitted better compared to analog signal.
- Digital signals are versatile, so it is widely used.
- The accuracy of the digital signal is better than that of the analog signal.

f.) Difference Between Analog and Digital Signal

Here are important differences between Analog and Digital Signal:

Analog	Digital
An analog signal is a continuous signal that represents physical measurements.	Digital signals are time separated signals which are generated using digital modulation.
It is denoted by sine waves	It is denoted by square waves
It uses a continuous range of values that help you to represent information.	Digital signal uses discrete 0 and 1 to represent information.
Temperature sensors, FM radio signals, Photocells, Light sensor, Resistive touch screen are examples of Analog signals.	Computers, CDs, DVDs are some examples of Digital signal.
The analog signal bandwidth is low	The digital signal bandwidth is high.
Analog signals are deteriorated by noise throughout transmission as well as write/read cycle.	Relatively a noise-immune system without deterioration during the transmission process and write/read cycle.
Analog hardware never offers flexible implementation.	Digital hardware offers flexibility in implementation.
It is suited for audio and video transmission.	It is suited for Computing and digital electronics.
Processing can be done in real-time and consumes lesser bandwidth compared to a digital signal.	It never gives a guarantee that digital signal processing can be performed in real time.
Analog instruments usually have s scale which is cramped at lower end and gives considerable observational errors.	Digital instruments never cause any kind of observational errors.
Analog signal doesn't offer any fixed range.	Digital signal has a finite number, i.e., 0 and 1.

g.) Advantages of Analog Signals

Here, are pros/benefits of Analog Signals

- Easier in processing
- Best suited for audio and video transmission.
- It has a low cost and is portable.
- It has a much higher density so that it can present more refined information.
- Not necessary to buy a new graphics board.
- Uses less bandwidth than digital sounds
- Provide more accurate representation of a sound
- It is the natural form of a sound.

h.) Advantages of Digital Signals

Here, are pros/advantages of Digital Signals:

- Digital data can be easily compressed.
- Any information in the digital form can be encrypted.
- Equipment that uses digital signals is more common and less expensive.
- Digital signal makes running instruments free from observation errors like parallax and approximation errors.
- A lot of editing tools are available
- You can edit the sound without altering the original copy
- Easy to transmit the data over networks

i.) Disadvantages of Analog Signals

Here are cons/drawback of Analog Signals:

- Analog tends to have a lower quality signal than digital.
- The cables are sensitive to external influences.
- The cost of the Analog wire is high and not easily portable.
- Low availability of models with digital interfaces.
- Recording analog sound on tape is quite expensive if the tape is damaged
- It offers limitations in editing
- Tape is becoming hard to find
- It is quite difficult to synchronize analog sound
- Quality is easily lost
- Data can become corrupted
- Plenty of recording devices and formats which can become confusing to store a digital signal
- Digital sounds can cut an analog sound wave which means that you can't get a perfect reproduction of a sound
- Offers poor multi-user interfaces

j.) Disadvantage of Digital Signals

- Sampling may cause loss of information.
- A/D and D/A demands mixed-signal hardware
- Processor speed is limited
- Develop quantization and round-off errors
- It requires greater bandwidth
- Systems and processing is more complex.

1.5.1. DIGITAL AND ANALOG TRANSMISSION

- Digital and analog transmission are two methods used to transmit data over communication channels. These methods differ in how they represent and transmit data signals. Here's a comparison between digital and analog transmission:

a. Digital Transmission:

- **Representation**: Digital transmission uses discrete, binary signals (0s and 1s) to represent data. Each binary digit (bit) represents a specific voltage level or symbol.
- **Signal Characteristics:** Digital signals are characterized by distinct voltage levels or symbols, typically represented by square waveforms. These signals have well-defined thresholds for interpreting 0s and 1s.
- **Noise Immunity:** Digital signals are less susceptible to noise and interference compared to analog signals. They can be regenerated and cleaned up using repeaters and error correction techniques, resulting in improved signal integrity.
- **Bandwidth Efficiency:** Digital transmission allows for higher data transmission rates and greater bandwidth efficiency compared to analog transmission. Multiple digital signals can be transmitted simultaneously over the same communication channel using techniques like multiplexing.
- **Examples:** Ethernet, Fiber Optic Communication, Digital Subscriber Line (DSL), Satellite Communication.

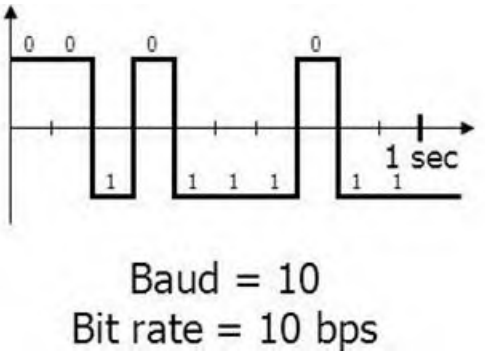
b. Analog Transmission:

- **Representation:** Analog transmission uses continuous, varying signals to represent data. These signals can take on an infinite number of values within a specified range.
- **Signal Characteristics:** Analog signals have continuously varying voltage levels or waveforms that represent the original data. They can be affected by noise, distortion, and attenuation during transmission.
- **Noise Sensitivity:** Analog signals are more susceptible to noise and distortion compared to digital signals. Noise and interference can degrade the quality of the signal and result in errors during data transmission.

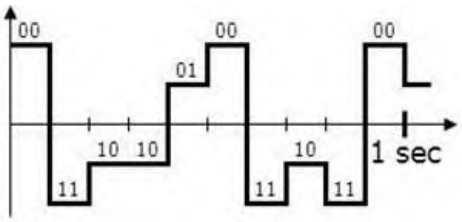
- **Bandwidth Usage:** Analog transmission typically requires more bandwidth to transmit the same amount of information compared to digital transmission. This limits the data transmission rates and efficiency of analog communication systems.
- **Examples:** Analog Telephone Systems (POTS), AM/FM Radio Broadcasting, Analog Television (TV) Broadcasting.
- In summary, digital transmission offers advantages such as noise immunity, higher data rates, and greater bandwidth efficiency, making it the preferred choice for modern communication systems. However, analog transmission is still used in certain applications where continuous signals are required or legacy systems are in place.

1.5.2. BIT RATE AND BAUD RATE

- Bit rate and baud rate are not always the same.
- The bit rate is the number of bits transmitted per second,
- The baud rate is the number of signal units transmitted per second and one signal unit is able to represent one or more bits.
- Therefore, baud rate is always less than or equal to the bit rate but never greater.



Baud = 10
Bit rate = 10 bps



Baud = 10
Bit rate = 20 bps

- **Bit Rate:** The number of bits transmitted per second
  - Bit rate = baud rate * number of bits per single unit
  - Ex: if 10 baud/Sec and there are 2 bits/baud what is bit rate?
  - 10*2=20 bits/Sec
- **Baud Rate:** Number of Signal unit transmitted per second.
  - Baud rate= bit rate / number of bits per single unit
  - Ex: if 20 bits/Sec and the 2 bits per baud what is baud rate?
  - 20/2=10 baud/Sec

Example:

1. If there are 4 bits per baud and there are 40 pits transferred per second , what will be the baud rate? 40/4=10 Baud/sec

2. If there are 10 baud per second and each baud contains 4 bit then what is bit rate? 10*4=40 bps

3. If there are 10 baud per second and the bit rate is 40 bps then how many signal elements are there? 40/10=4 bits/baud.

Example:

1.) An analog signal has a bit rate of 8000 bps and a baud rate of 1000. Then analog signal has _____ signal elements and carry _____ data elements in each signal.

(A) 256, 8 bits (B) 128, 4 bits

(C) 256, 4 bits (D) 128, 8 bits

Answer: (A)

Explanation:

Analog signal has a bit rate of 8000 bps and a baud rate of 1000. So, each signal will clearly carry bit rate / baud rate bits.

i.e. 8000 / 1000 = 8 bits and 2^8 = 256 signal.

So, option (A) is correct.

CHAPTER 1.5 MCQ

1. The speech signal is obtained after

(a) Analog to digital conversion (b) Digital to analog conversion

(c) Modulation (d) Quantization

2. What type of data is typically represented by analog signals?

(a) Text documents (b) Images and graphics

(c) Audio and video (d) Binary code

3. What type of data is typically represented by digital signals?

(a) Analog sound waves (b) Continuous video streams

(c) Text documents and numerical data (d) Analog voltage levels

4. Which of the following statements is true regarding noise immunity?

(a) Analog signals are more immune to noise than digital signals

(b) Digital signals are more immune to noise than analog signals

(c) Analog and digital signals have equal immunity to noise

(d) Noise does not affect either analog or digital signals

5. Before data can be transmitted, they must be transformed to _____.

(a) periodic signals (b) electromagnetic signals

(c) aperiodic signals (d) low-frequency sine waves

6. A periodic signal completes one cycle in 0.001 s. What is the frequency?

(a) 1 Hz (b) 100 Hz (c) 1 KHz (d) 1 MHz

7. In a frequency-domain plot, the horizontal axis measures the _____.

(a) Peak amplitude (b) Frequency

(c) Phase (d) Slope

8. If the bandwidth of a signal is 5 KHz and the lowest frequency is 52 KHz, what is the highest frequency?

(a) 5 KHz (b) 10 KHz (c) 47 KHz (d) 57 KHz

9. What is the bandwidth of a signal that ranges from 1 MHz to 4 MHz?

(a) 4 MHz (b) 1 KHz

(c) 3 MHz (d) None of the above

10. As frequency increases, the period _____.

(a) Decreases (b) Increases

(c) Remains the same (d) Doubles

11. _____ is a type of transmission impairment in which the signal loses strength due to the resistance of the transmission medium.

(a) Attenuation (b) Distortion

(c) Noise (d) Decibel

12. _____ is a type of transmission impairment in which the signal loses strength due to the different propagation speeds of each frequency that makes up the signal.

(a) Attenuation (b) Distortion

(c) Noise (d) Decibel

13. _____ is a type of transmission impairment in which an outside source such as crosstalk corrupts a signal.

(a) Attenuation (b) Distortion

(c) Noise (d) Decibel

14. A _____ sine wave is not useful in data communications; we need to send a _____ signal.

(a) Composite; single-frequency (b) Single-frequency; composite

(c) Single-frequency; double-frequency (d) None of the above

15. Frequency and period are _____

(a) Inverse of each other (b) Proportional to each other

(c) The same (d) None of the above

TEACHER'S CARE PUBLICATION
38/23, Vaigundaperumal Koil Street,Kancheepuram- 631502
Mobile : 95665 35080, 9786269980 Land Line : 044-2723 5080

2025-2026

PG TRB

G.O No. 219 New Syllabus

COMPUTER SCIENCE

UNIT-8
Web Development

Your Success is Our Goal...

- HTML
- Elements of HTML
- CSS
- Menu and Division
- JavaScript

tcaexamguide
Teacher's Care Academy
www.tcaexamguide.com
tcaexamguide@gmail.com

UNIT-VIII WEB DEVELOPMENT

TEACHER'S CARE ACADEMY, KANCHIPURAM

TNPSC-TRB- COMPUTER SCIENCE -TET COACHING CENTER

HEAD OFFICE: NO. 38/23, VAIGUNDA PERUMAL KOIL, SANNATHI STREET, KANCHIPURAM – 1. CELL: 9566535080

B.Off 2: 65C, Thillai Ngr(West), 4th Cross St, Trichy – 620018
B.Off 3: Vijiyaraghavachariar Memorial Hall(Opp to Sundar Lodge), Salem

Trichy: 76399 67359 **Salem: 93602 68118**

UNIT VIII: WEB DEVELOPMENT

SYLLABUS

HTML: Introduction –Basic Tags-formatting and fonts, Working with color.

Elements of HTML: Working with Tables, working with Images, Working with Links, List and Tables, Frame and Frameset, Forms and Controls

CSS: Introduction to Cascading Style Sheet, using CSS background images, color and properties, manipulating texts using fonts, border and boxes margins, padding lists, positioning using CSS.

Menu and Division: Types of Style Sheets, Class and ID, selector, Inline Menu, DIV and CSS layout

JavaScript: Introduction to JavaScript, Understanding Variables, Loops and Arrays, Functions, working with alert, confirm and prompt boxes, Creating Rollover image, Working with Operators, Events.

BOOKS TO STUDY:

1) Microsoft Step By Step Html And Xh,Faithe Wempen Prentice Hall Of India Private Limited New Delhi 2006
2) Jon Duckett, "Beginning HTML, XHTML, CSS and Java script", Wiley Publishing.
3) Beginning HTML and CSS by Rob Larsen
4) HTML, CSS, and JavaScript by Julie C Meloni & Jennifer Kyrnin
5) David R.Brooks, "An Introduction to HTML and JavaScript for Scientists and Engineers

1.HTML (HYPERTEXT MARKUP LANGUAGE)

1.INTRODUCTION

1.1.HTML BASICS

- HTML (Hypertext markup language) is the basic programming language of the World Wide Web.
- We can create a simple web page in just a few minutes.
- Creating a web page with HTML is much simpler than writing 3 computer program in a language like VB or C++
- We can convert plain text into attractive formatting.
- We can add graphics and hyperlinks.

1.2. UNDERSTANDING HTML

- A web page is a plain text file that has been encoded using HTML so that it appears nicely formatted in a web browser.

Hyper text	Text that we click to jump from document to document. This is a reference to the ability of web pages to link one another.
Markup	Tags that apply layout and formatting conventions to plain text. Literally , the plain text is “marked-up “with the tags.
Language	A reference to the fact that HTML is considered a programming language.

- HTML is an interpreted programming language.
- Each time out web browser opens a web page, it processed the HTML code within the file.

1.3. BRIEF HISTORY OF HTML

- In the late 1980's, a physicist, Tim Berners-Lee who was a contractor at CERN, proposed a system for CERN researchers. In 1989, he wrote a memo proposing an internet based hypertext system.
- Tim Berners-Lee is known as the father of HTML. The first available description of HTML was a document called "HTML Tags" proposed by Tim in late 1991. The latest version of HTML is HTML5, which we will learn later in this tutorial.

1.4. HTML VERSIONS

- Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:
  - HTML 1.0: The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in1991.

- HTML 2.0: This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.
- HTML 3.2: HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.
- HTML 4.01: HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.
- HTML5 : HTML5 is the newest version of HyperText Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG (Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

1.5. OBJECTIVES OF HTML

- Recognize the components of an HTML file and create such a file
- Link to local files and Web pages from their Web pages.
- Add graphics and sound to their Web pages using HTML.
- Create different kinds of lists to their Web page using HTML.
- Create a multi-column and multi-row tables using HTML.
- Set background colors and graphics for Web pages.
- Evaluate Web page design and consider design issues that affect web pages.
- Add links to sites from their Web pages.
- Tailor web design and development to Interlink Environment.
- HTML is the *lingua franca* of the Internet. Publishing HTML-formatted documents on the Internet via the World Wide Web proved to be the answer to these needs.
- HTML is a subset of the Standard Generalized Markup Language (SGML). SGML is an international standard (ISO 8879) published in 1986 as a format for structuring and marking up documents.

1.6. FEATURES OF HTML

- It is a very easy and simple language. It can be easily understood and modified.
- It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.

- It is a markup language, so it provides a flexible way to design web pages along with the text.
- It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
- It is platform-independent because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
- It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive.
- HTML is a case-insensitive language, which means we can use tags either in lower-case or upper-case.

EXERCISE 1:

1) **What is the main purpose of HTML?**
   a) To style web pages
   b) To add interactivity
   c) To create the structure of web pages
   d) To manage databases

2) **Which of the following is NOT a feature of HTML?**
   a) Platform-independent
   b) Easy to learn
   c) Supports database management
   d) Works on all browsers

3) **HTML is classified as a:**
   a) Programming language
   b) Styling language
   c) Markup language
   d) Scripting language

4) **Which version of HTML introduced semantic tags like `<article>` and `<section>`?**
   a) HTML 3.2
   b) HTML 4.01
   c) HTML5
   d) XHTML

5) **HTML stands for -**
   a) High Text Machine Language
   b) Hyper Text and links Markup Language
   c) Hyper Text Markup Language
   d) None of these

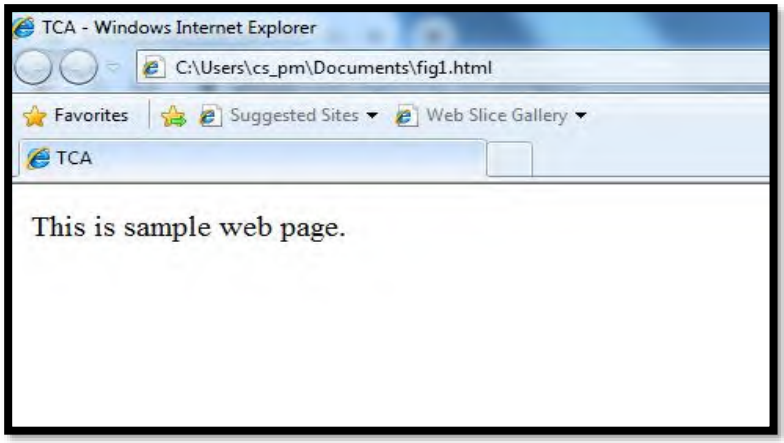
2.BASIC TAGS OF HTML

2.1. TAGS:

- HTML tags are the fundamental elements of HTML used for defining the structure of the document. These are letters or words enclosed by angle brackets (< and >).
- Usually, most of the HTML tags contain an opening and a closing tag. Each tag has a different meaning, and the browser reads the tags and displays the contents enclosed by them accordingly.
- For example, if we wrap any text in the paragraph ( ) tag, the browser displays it as a separate paragraph. In this chapter, we will discuss all the basic tags in HTML.

2.2. BASIC WEB PAGE

```
<HTML>
<HEAD>
<TITLE> TCA </TITLE>
</HEAD>
<BODY>
 This is my first web page.
</BODY>
</HTML>
```



2.2.1.HTML Tag

- As shown in Figure <HTML> is a starting tag. To delimit the text inside, add a closing tag by adding a "/" to the starting tag. Most but not all tags have a closing tag. It is necessary to write the code for an HTML page between <HTML> and </HTML>.
- Think of tags as talking to the browser or, better still, giving it instructions. What you have just told the browser is 'this is the start of an HTML document' (<HTML>) and 'this is the end of an HTML document' (</HTML>). Now you need to put some matter in between these two markers.

2.2.2. HEAD Tag

- It should be the first element inside the <html> element, which contains the metadata (information about the document). It must be closed before the body tag opens.
- Every HTML document is segregated into a HEAD and BODY. The information about the document is kept within <HEAD> tag. The BODY contains the page content.

2.2.3. TITLE Tag

- The only thing you have to concern yourselves with in the HEAD tag (for now) is the TITLE tag.
- The bulk of the page will be within the BODY tag, as shown in Figure.

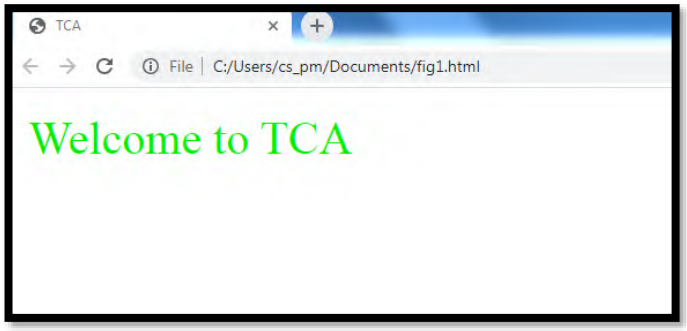
```
<HEAD>
   <TITLE> TCA </TITLE>
</HEAD>
```

- Here the document has been given the title TCA. It is a good practice to give a title to the document created.
- What you have made here is a skeleton HTML document. This is the minimum required information for a web document and all web documents should contain these basic components. Secondly, the document title is what appears at the very top of the browser window.

2.2.4. BODY Tag

- If you have a head, you need a body. All the content to be displayed on the web page has to be written within the body tag. So whether text, headlines, textbox, checkbox or any other possible content, everything to be displayed must be kept within the BODY tag as shown in Figure.
- Whenever you make a change to your document, just save it and hit the Reload/Refresh button on your browser. In some instances, just hitting the Reload/Refresh button doesn't quite work. In that case hit Reload/Refresh while holding down the SHIFT key.
- The BODY tag has following attributes:
  1) **BGCOLOR:** It can be used for changing the background color of the page. By default the background color is white.
  2) **BACKGROUND:** It is used for specifying the image to be displayed in the background of the page.
  3) **LINK:** It indicates the color of the hyperlinks, which have not been visited or clicked on.
  4) **ALINK:** It indicates the color of the active hyperlink. An active link is the one on which the mouse button is pressed.
  5) **VLINK:** It indicates the color of the hyperlinks after the mouse is clicked on it.
  6) **TEXT:** It is used for specifying the color of the text displayed on the page.
- Consider the following example:

```
<HTML>
<TITLE> TCA </TITLE>
<BODY BGCOLOR="#123456" TEXT = "#FF0000"> Welcome to TCA
</BODY>
</HTML>
```



A Web Page with a Background Color

- The values specified for BGCOLOR and TEXT tags indicate the color of the background of the page and that of the text respectively. These are specified in hexadecimal format.
- The range of allowable values in this format is from "#000000" to "#FFFFFF". The"#" symbol has to precede the value of the color so as to indicate to the browser that has to be interpreted as a hexadecimal value.
- In this six-digit value, the first two digits specify the concentration of the color red, the next two digits specify the concentration of the color green and the last two digits specify the concentration of the color blue.
- So the value is a combination of the primary colors red, green and blue and that is why it is called RGB color. If we specify the value "#FF0000", the color appears to be red." #000000" gives black and "#FFFFFF" gives the color white. You also have the option of specifying the color by giving its name, like:

  <BODY TEXT = "WHITE">.

- You can also specify a background image instead. (Note that the image should be in the same folder as your HTML file. More on this below).

```
<HTML>
<BODY BACKGROUND="tca.jpg">
Welcome to INDIA
</BODY>
</HTML>
```

2.3. HEADING TAGS

- Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>. While displaying any heading, browser adds one line before and one line after that heading.
- Example:

```
<HTML>
<body>
        <h1>TEACHER'S CARE ACADEMY</h1>
        <h2> TEACHER'S CARE ACADEMY </h2>
        <h3> TEACHER'S CARE ACADEMY </h3>
        <h4> TEACHER'S CARE ACADEMY </h4>
        <h5> TEACHER'S CARE ACADEMY </h5>
        <h6> TEACHER'S CARE ACADEMY </h6>
</body>
</HTML>
```

2.4. PARAGRAPH TAG

- The tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening and a closing tag.
- Example

```
<html>
  <head>
    <title>
        Paragraph Example
    </title>
  </head>
  <body>
    <p>Here is a first paragraph of text.</p>
    <p>Here is a second paragraph of text.</p>
    <p>Here is a third paragraph of text.</p>
  </body>
</html>
```

- This will produce the following result

Here is a first paragraph of text.

Here is a second paragraph of text.

Here is a third paragraph of text.

2.5. LINE BREAK TAG

- Whenever you use the
 element, anything following it starts from the next line. This tag is an example of an empty element, where you do not need opening and closing tags, as there is nothing to go in between them.
- The
 tag has a space between the characters br and the forward slash. If you omit this space, older browsers will have trouble rendering the line break, while if you miss the forward slash character and just use
 it is not valid in XHTML.
- Example

```
<html>
  <head>
    <title>Line Break Example</title>
  </head>
  <body>
    <p>Hello<br />
      You delivered your assignment ontime.<br />
      Thanks<br />
      Mahnaz</p>
  </body>
</html>
```


- This will produce the following result

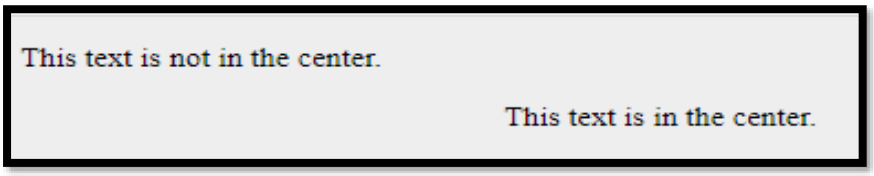
Hello
You delivered your assignment on time.
Thanks
Mahnaz

2.6. CENTER TAG

- You can use tag to put any content in the center of the page or any table cell.
- Example

```
<html>
  <head>
    <title> Centring Content Example</title>
  </head>
  <body>
    <p>This text is not in the center.</p>
    <center>
      <p>This text is in the center.</p>
    </center>
  </body>
</html>
```

❖ This will produce following result

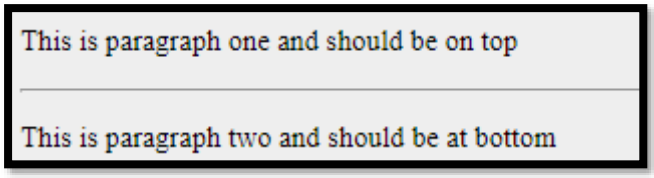


2.7. HORIZONAL RULE TAG

- Horizontal lines are used to visually break-up sections of a document. The <hr> tag creates a line from the current position in the document to the right margin and breaks the line accordingly.
- For example, you may want to give a line between two paragraphs as in the given example below:

```
<html>
  <head>
    <title>Horizontal Line Example</title>
  </head>
  <body>
    <p>This is paragraph one and should be on top</p>
    <hr />
    <p>This is paragraph two and should be at bottom</p>
  </body>
</html>
```

- This will produce the following result



- Again <hr /> tag is an example of the empty element, where you do not need opening and closing tags, as there is nothing to go in between them.
- The <hr /> element has a space between the characters hr and the forward slash. If you omit this space, older browsers will have trouble rendering the horizontal line, while if you miss the forward slash character and just use <hr> it is not valid in XHTML.

2.8. PRESERVE FORMATTING TAG

- Sometimes, you want your text to follow the exact format of how it is written in the HTML document. In these cases, you can use the preformatted tag <pre>.
- Any text between the opening <pre> tag and the closing </pre> tag will preserve the formatting of the source document.
- Example

```
<html>
  <head>
    <title>Preserve Formatting Example</title>
  </head>
  <body>
    <pre>
      function testFunction( strText )
      {
        alert (strText)
      }
    </pre>
  </body>
</html>
```


- This will produce the following result

```
function testFunction( strText ){
   alert (strText)
}
```

2.9.NON-BREAKING SPACES

- Suppose you want to use the phrase "12 Angry Men." Here, you would not want a browser to split the "12, Angry" and "Men" across two lines.
- An example of this technique appears in the movie "12 Angry Men."
- In cases, where you do not want the client browser to break text, you should use a nonbreaking space entity   instead of a normal space. For example, when coding the "12 Angry Men" in a paragraph, you should use something similar to the following code:

```
<html>
  <head>
    <title>
        Nonbreaking Spaces Example
    </title>
  </head>
  <body>
    <p>
        An example of this technique appears in the movie "12 Angry Men."
    </p>
  </body>
</html>
```


- This will produce the following result

An example of this technique appears in the movie "12 Angry Men."

2.10. COMMENT LINES TAG

- Comments are some text or code written in your code to give an explanation about the code, and not visible to the user. Comments which are used for HTML file are known as HTML comments. Anything written between these tags will be ignored by the browser, so comments will not be visible on the webpage.
- Comments of any code make code easy to understand and increase readability of code. Comments are also part of the code, which gives an explanation of the code.
- You can add comments in your HTML file using <! -- ... --> tag. So if you will write anything between theses comment tag that will be treated as comment and browser will not read it.
- **Syntax**

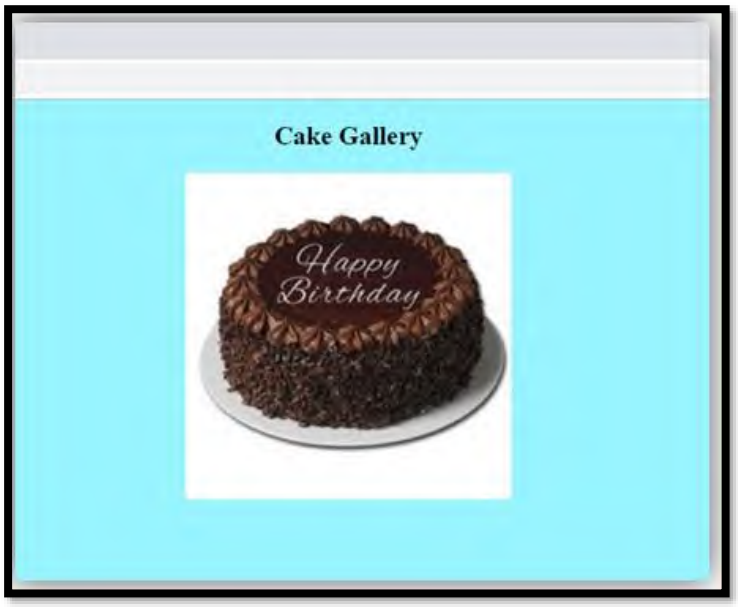
```
<! -- Write commented text here -->
```

- **Example:**

```
<h2>Cake Gallery</h2>
```

```
<!-- This is image for a yummy cake
you can see it on your web-page
of your favorite browser -->
<img src="https://static.javatpoint.com/htmlpages/images/cake.png" alt="cake image" height=
"300px"
 width="300px">
```

Output:



EXERCISE 2:

1) **What is the correct syntax for an HTML tag?**
   a) <tag>
   b) [tag]
   c) (tag)
   d) <tag>
2) **Which of the following is a pair tag?**
   a) <img>
   b)
   c) <p>
   d) <hr>
3) **What is the purpose of the <head> tag?**
   a) To define the header section of a web page
   b) To contain metadata and links to styles/scripts
   c) To define content visible on the web page
   d) To display the page title

4) **Which of the following is NOT part of the <head> section?**
   a) <title>
   b) <meta>
   c) <link>
   d) <body>

5) Which of the following is TRUE about the <body> tag?

a) It must be included in every HTML document.
b) It contains all visible content on the web page.
c) Both a and b
d) None of the above

6) Which tag represents the smallest heading?

a) <h6>
b) <h1>
c) <header>
d) <h6>

7) **How many heading levels does HTML support?**

a) 4
b) 5
c) 6
d) Unlimited

8) **Which tag is used to define a paragraph in HTML?**

a) <pg>
b) <para>
c) <p>
d) <paragraph>

9) **Which tag is used to insert a line break?**

a) <lb>
b) <brk>
c)
d) <line>

10) The correct sequence of HTML tags for starting a webpage is -

a) Head, Title, HTML, body
b) HTML, Body, Title, Head
c) HTML, Head, Title, Body
d) HTML, Head, Title, Body

3.FORMATTING AND FONTS

3.1.HTML FORMATTING

- HTML Formatting is a process of formatting text for better look and feel. HTML provides us ability to format text without using CSS. There are many formatting tags in HTML. These tags are used to make text bold, italicized, or underlined. There are almost 14 options available that how text appears in HTML and XHTML.
- In HTML the formatting tags are divided into two categories:
  1) **Physical tag:** These tags are used to provide the visual appearance to the text.
  2) **Logical tag:** These tags are used to add some logical or semantic value to the text.

S.NO	ELEMENT	DESCRIPTION
1	<b>	This is a physical tag, which is used to bold the text written between it.
2	<i>	This is a physical tag which is used to make text italic.
3	<u>	This tag is used to underline text written between it

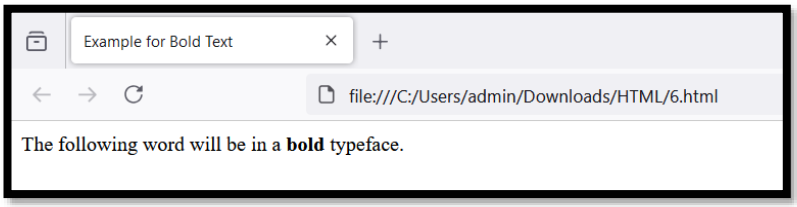
4		This is a logical tag, which tells the browser that the text is important.
5		This is a logical tag which is used to display content in italic.
6	<mark>	This tag is used to highlight text.
7	<tt>	This tag is used to appear a text in teletype. (not supported in HTML5)
8	<strike>	This tag is used to draw a strikethrough on a section of text. (Not supported in HTML5)
9	<sup>	It displays the content slightly above the normal line.
10	<sub>	It displays the content slightly below the normal line.
11	<del>	This tag is used to display the deleted content.
12	<ins>	This tag displays the content which is added
13	<big>	This tag is used to increase the font size by one conventional unit.
14	<small>	This tag is used to decrease t

3.2. BOLD TEXT

- Any content that is placed in between ... element will be displayed as bold text.
- Here is an example.

```
<html>
<head>
<title>Example for Bold Text</title>
</head>
 <body> <p>The following word will be in a <b>bold</b> typeface. </p>
 </body>
</html>
```

- The above code will produce the following output:



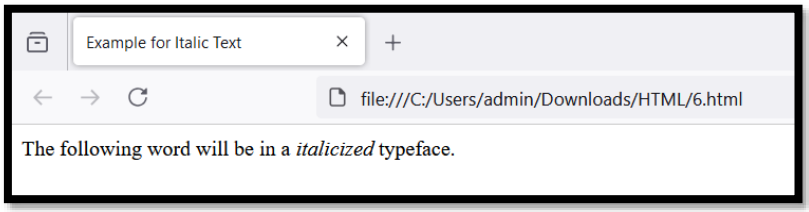
3.3. ITALIC TEXT

- Any content that is placed in between ... element will be displayed as italicized text.
- Here is an example.

```
<html>
<head> <title>Example for Italic Text</title>
```

```
</head>
<body> <p>The following word will be in a <i>italicized</i> typeface.</p>
</body>
</html>
```

- The above code will produce the following output:

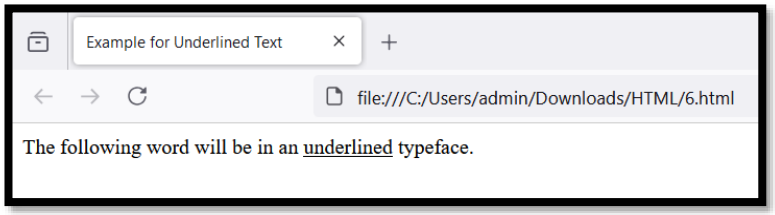


3.3. UNDERLINED TEXT

- Any content that is placed in between <u>...</u> element will be displayed as underlined text.
- Here is an example.

```
<html>
<head>
<title>Example for Underlined Text</title>
</head>
<body> <p>The following word will be in an <u>underlined</u> typeface. </p>
</body>
</html>
```


- The above code will produce the following output:



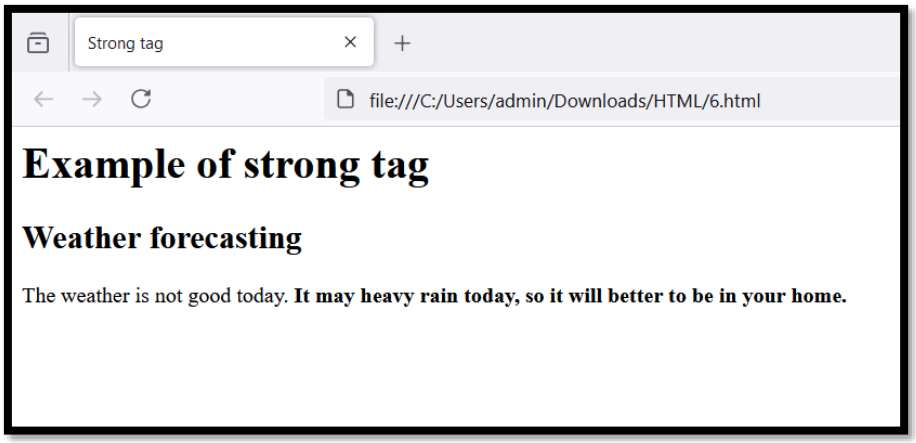
3.4. STRONG TEXT:

- HTML tag is a phrase tag which is used to represent the important text of a document on the browser. The text within text has semantic importance for the search engines and emphasize the text with special intonation.
- The text within tag renders in bold font on the browser by default; however, it can be changed using CSS.
- Here is an example.

```
<html>
<head>
<title>Strong tag</title>
```

```
</head>
<body>
<h1>Example of strong tag</h1>
<h2>Weather forecasting</h2>
<p>The weather is not good today.
<strong>It may heavy rain today, so it will better to be in your home.</strong>
</p>
</body>
</html>
```

- The above code will produce the following output:



3.5. EMPHASIZING TEXT

- The HTML ** tag** marks text that has stress emphasis which traditionally means that the text is displayed in italics by the browser. This tag is also commonly referred to as the element.
- Syntax

  In HTML, the syntax for the ** tag** is:
- Example

```
<body>
  <p><em>Stress emphasized text goes here</em> but not here</p>
</body>
```

Output

3.6. MARK TEXT

- The <mark> tag in HTML is used to define the marked text. It is used to highlight the part of the text in a paragraph. The <mark> tag is new in HTML 5.
- Syntax:

<mark> Contents... </mark>

- Example

```
<html>
<body>
<h1 style="color: green;">TEACHER'S CARE ACADEMY</h1>
<h2>HTML mark Tag </h2>
<p>
<mark>TCA:</mark> It is a <mark>computer science</mark> portal
</p>
</body>
</html>
```

3.7. MONOSPACED TEXT

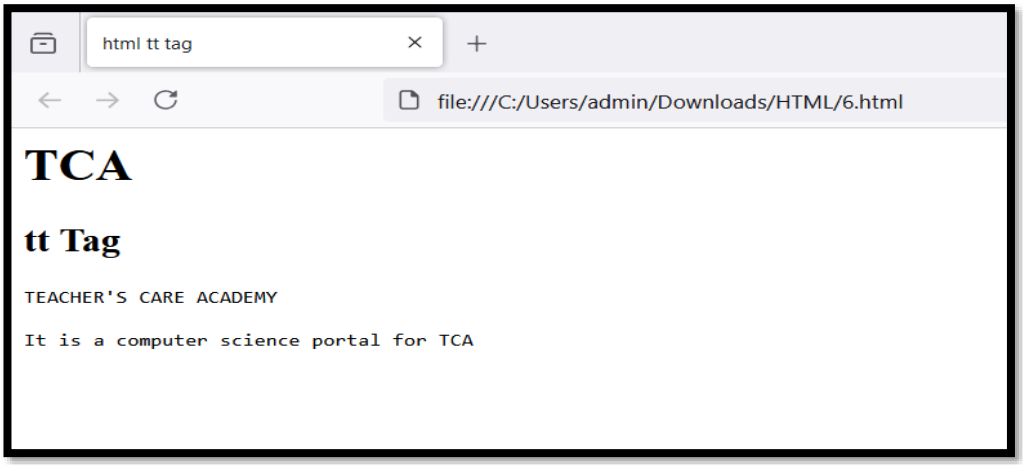
- The <tt> tag in HTML stands for teletype text. It was used in HTML 4 to define text in a monospaced, teletype-style font. However, the tag has been deprecated in HTML5 because it only served a presentational purpose without adding any semantic meaning to the content.
- Syntax

<tt> Content... </tt>

- Example

```
<html>
<head>
<title>html tt tag</title>
</head>
<body>
<h1>TCA</h1>
<h2>tt Tag</h2>
<!-- HTML tt Tag is used here-->
<tt>
```

```
TEACHER'S CARE ACADEMY
</tt>
<p>
<tt> It is a computer science portal for TCA
</tt>
</p>
</body>
</html>
```



3.7. STRIKE TEXT

- The <strike> tag defines a strike or line through Text. This tag creates a cut line in the text. This tag is depreciated from HTML 5. Now, the <del> tag is used instead of this tag.
- Syntax:

```
<strike> Contents </strike>
```

- Example

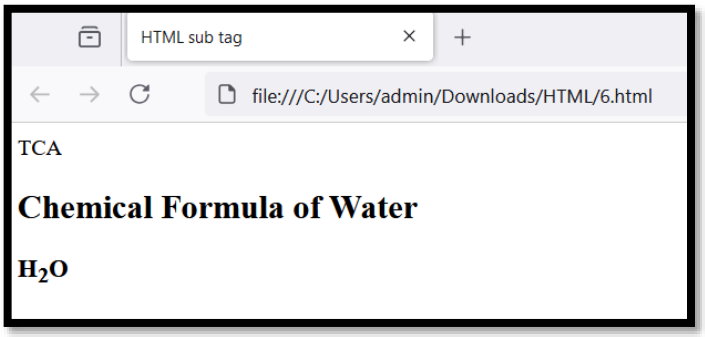
```
<html>
<body>
<!-- Strike Tag -->
<h2> Welcome To Teacher Care Academy </h2>
<strike> TCA </strike>
</body>
</html>
```

- The above code will produce the following output:

3.8. SUBSCRIPT TEXT

- The <sub> tag in HTML is used to define subscript text, which appears smaller and slightly below the normal text baseline.
- It is commonly used for mathematical expressions, chemical formulas, and footnotes, like in H_2O (water) or $E=mc^2$ (Einstein's equation).
- Syntax

  _{Contents. . .}



- Example

```
<html>
<head>
<title>HTML sub tag</title>
</head>
<body>
</h1> TCA </h1>
<h2>Chemical Formula of Water</h2>
<h3>H<sub>2</sub>O</h3>
</body>
</html>
```

3.9. SUPERSCRIPT TEXT

- The <sup> tag is used to add a superscript text to the HTML document. The <sup> tag defines the superscript text. Superscript text appears half a character above the normal line and is sometimes rendered in a smaller font. Superscript text can be used for footnotes.
- Syntax

  ^{Contents. . .}

- Example

```
<html>
<head>
<titlt> TCA </title>
</head>
<body>
<p>A sup element is displayed like this:</p>
<p>This text contains <sup>superscript text</sup></p>
<p>Change the default CSS settings to see the effect. </p>
</body>
</html>
```

TEACHER'S CARE PUBLICATION
38/23, Vaigundaperumal Koil Street,Kancheepuram- 631502
Mobile : 95665 35080, 9786269980 Land Line : 044-2723 5080

2025-2026

PG TRB

G.O No. 219 New Syllabus

COMPUTER SCIENCE

UNIT-10
Cyber Security

Your Success is Our Goal...

- Cyber Crime
- Cyber Security Techniques
- Password Management
- Cryptography
- Attacks

tcaexamguide
Teacher's Care Academy
www.tcaexamguide.com
tcaexamguide@gmail.com

TEACHER'S CARE ACADEMY, KANCHIPURAM

TNPSC-TRB- COMPUTER SCIENCE -TET COACHING CENTER

HEAD OFFICE: NO. 38/23, VAIGUNDA PERUMAL KOIL, SANNATHI STREET, KANCHIPURAM – 1. CELL: 9566535080

B.Off 2: 65C, Thillai Ngr(West), 4th Cross St, Trichy – 620018

B.Off 3: Vijiyaraghavachariar Memorial Hall(Opp to Sundar Lodge), Salem

Trichy : 76399 67359 **Salem : 93602 68118**

UNIT-10

CYBER SECURITY SYLLABUS

1. **CYBER SECURITY**: Introduction to Cyber Crime, Malware type, Kinds of Cyber Crime.
2. **Cyber Security Techniques:** Authentication, Encryption, Digital Signatures, Antivirus, Firewall, Steganography.
3. **Password Management:** Guidelines for Secure Password, Two Step verification, Generating Secure password, Using Password Manager, Enabling Two-step verification, Securing Computer using antivirus.
4. **Cryptography:** Symmetric cipher model, cryptographic system, substitution techniques, Caesar cipher, mono alphabetic ciphers, Hill ciphers, Transposition techniques, steganography, Data encryption standard, The strength of DES.
5. **Attacks:** Investigating DoS Attacks, Types of DoS Attacks, Classification of DoS Attacks, Techniques to Detect, DoS Attacks.

Refference Books:

1. **"Cyber Security Essentials" by James Graham Richard Howard Ryan Olson, Taylor & Francis Group**
2. **"Introduction to Cyber Security", Chwan-Hwa(john) Wu,J. David Irwin, CRC Press T&FGroup.**
3. **"Cyber Security Understanding Cyber Crimes,Computer Forensics and Legal Perspectives",Wiley by Nina Godbole and SunitBelpure**
4. **"Introduction to Cyber Security" by Jeetendra Pande, Uttarakhand Open University, Haldwani.**
5. **"Everyday Cybersecurity" by Christopher K. COX**

TCA

CHAPTER 1 : CYBER SECURITY

1.1. INTRODUCTION TO CYBER CRIME

- The internet was born around 1960"s where its access was limited to few scientist, researchers and the defence only.
- Internet user base have evolved expontinanlty. Initially the computer crime was only confined to making a physical damage to the computer and related infrastructure.
- Around 1980"s the trend changed from causing the physical damaging to computers to making a computer malfunction using a malicious code called virus.
- Till then the effect was not so widespread beacouse internet was only comfined to defence setups, large international companies and research communities.
- In 1996, when internet was launched for the public, it immeditly became populer among the masses and they slowly became dependent on it to an extent that it have changed their lifestyle.
- The GUIs were written so well that the user don"t have to bother how the internet was functioning. They have to simply make few click over the hyber links or type the desired information at the desired place without bothering where this data is stored and how it is sent over the internet or wether the data can accessed by another person who is conneted to the internet or wether the data packet sent over the internet can be snoofed and tempered.
- The focus of the computer crime shifted from marely damaging the computer or destroying or manipulating data for personal benefit to financial crime.
- These computer attacks are incresing at a rapid pase.

1.1.1. CLASSIFICATION OF CYBER CRIMES

- Cybercrimes can be classified based on various factors, including the target, the motive, and the techniques used. Here are some common classifications:

i.) Internal & External attacks

ii.) Structured & Unstructured attacks

iii.) Active & Passive attacks

iv.) Hardware & Software attacks:

i.) INTERNAL & EXTERNAL ATTACKS

- The cyber criminal could be internal or external to the organization facing the cyber attack. Based on this cyber crime could be categorized into two types:

a.) Internal Cybercrime

- Internal cybercrime involves threats or attacks initiated by individuals within the organization, such as employees, contractors, or other insiders with access to sensitive systems or data.

Characteristics:

- Access to Internal Resources: Insiders already have access to systems, making it easier to execute attacks.
- Motives: Can include financial gain, revenge, espionage, or negligence.
- Examples of Perpetrators: Employees, disgruntled staff, former employees, or contractors.

Examples:

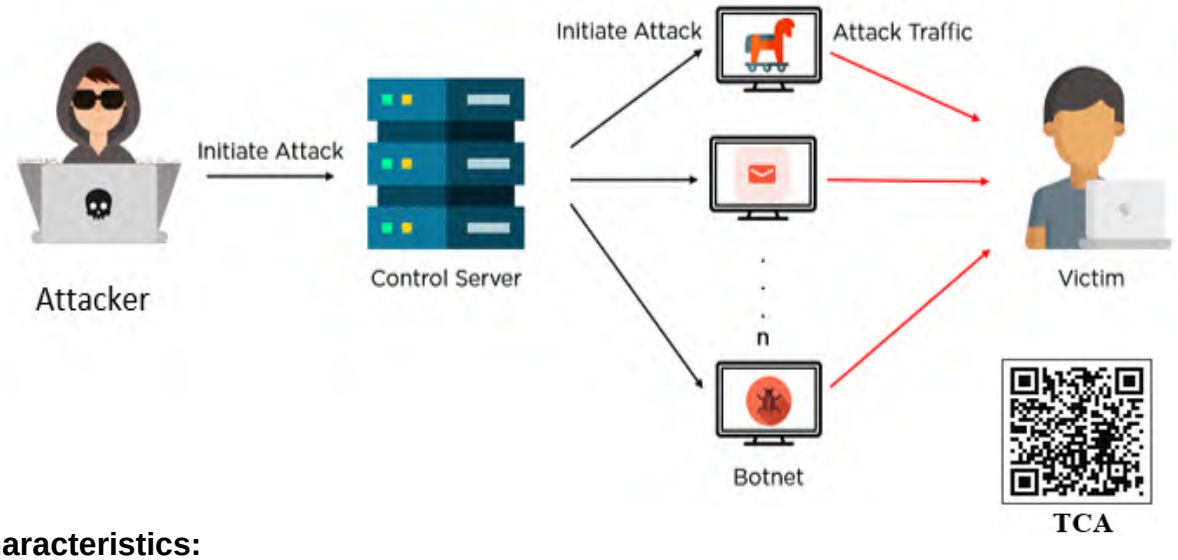
1. Data Theft: Stealing sensitive information like trade secrets, customer databases, or financial records.
2. Sabotage: Deliberately damaging systems, deleting data, or disrupting operations.
3. Privileged Account Misuse: Abusing access rights to manipulate or steal data.
4. Negligence: Carelessness, such as clicking on phishing links or failing to follow security protocols.

Prevention:

- Implement strict access controls and privilege management.
- Monitor insider activities using security tools (e.g., User Behavior Analytics).
- Regularly audit sensitive data and systems.
- Establish clear policies for cybersecurity and employee conduct.

b.) External Cybercrime

➢ External cybercrime involves attacks initiated by individuals or groups outside the organization. These attackers often use sophisticated techniques to bypass security systems.



Characteristics:

- Motives: Include financial gain, espionage, ideological causes, or sabotage.
- Techniques Used: Phishing, hacking, malware, ransomware, etc.
- Examples of Perpetrators: Hackers, cybercriminal groups, hacktivists, or nation-state actors.

Examples:

1. Phishing Attacks: Fraudulent attempts to steal sensitive data through emails or fake websites.
2. Hacking: Unauthorized access to systems or networks.
3. Distributed Denial of Service (DDoS): Flooding servers to disrupt services.
4. Ransomware: Encrypting data and demanding payment for its release.
5. Cyber Espionage: Stealing confidential data or intellectual property.

Prevention:

- Use firewalls, intrusion detection systems, and endpoint protection.
- Regularly update and patch systems to fix vulnerabilities.
- Educate employees about cybersecurity best practices.
- Deploy multi-factor authentication (MFA) and encryption.

C.) Comparison Between Internal and External Cybercrime

Aspect	Internal Cybercrime	External Cybercrime
Perpetrators	Insiders (employees, contractors, etc.)	Outsiders (hackers, groups, etc.)
Access	Direct access to systems and data	Must breach security to gain access
Motives	Revenge, negligence, or financial gain	Financial gain, espionage, or sabotage
Techniques	Misuse of privileges, sabotage	Malware, phishing, hacking, etc.
Detection	Harder to detect due to trusted access	Detected through security systems

ii.) STRUCTURED & UNSTRUCTURED ATTACKS

- The cyber attacks can also be classified as structure attacks and unstructured attacks based on the level of maturity of the attacker. Differentiated by their level of sophistication, planning, and execution.

a.) Structured Attacks

- Structured attacks are well-organized, methodical, and often executed by skilled individuals or groups with a clear purpose. These attacks are typically premeditated and involve advanced techniques.
- Cyber crimes have turned out to be a low-investment, low-risk business with huge returns.
- Now-a-days these structured crimes are performed are highly organized.
- There is a perfect hierarchical organizational setup like formal organizations and some of them have reached a level in technical capabilities.

Characteristics:

- **Planning**: Highly organized and planned in detail.
- **Sophistication**: Uses advanced tools and technologies.
- **Perpetrators**: Often carried out by professionals, such as hackers, cybercriminal organizations, or nation-state actors.
- **Target**: Often aimed at specific individuals, organizations, or governments to achieve a particular objective.
- **Duration**: Can last for weeks, months, or even years (e.g., Advanced Persistent Threats, APTs).

Examples:

1. **Advanced Persistent Threats (APTs)**: Long-term, stealthy attacks designed to infiltrate and extract data from high-value targets.
2. **Ransomware Campaigns**: Carefully targeted attacks on organizations with the intent to extort large sums of money.
3. **Corporate Espionage**: Stealing intellectual property or trade secrets using sophisticated methods.
4. **Critical Infrastructure Attacks**: Targeting power grids, water supply systems, or communication networks for geopolitical motives.

Prevention:

- Employ advanced cybersecurity tools, such as Endpoint Detection and Response (EDR) and Intrusion Detection Systems (IDS).
- Use multi-factor authentication and robust encryption.
- Continuously monitor networks for anomalies.
- Implement incident response plans and conduct regular threat simulations.

b.) Unstructured Attacks

- Unstructured attacks are less sophisticated and often carried out by individuals with limited technical expertise. These attacks are typically opportunistic and rely on readily available tools.

Characteristics:

- **Planning**: Minimal or no planning involved.
- **Sophistication**: Relies on basic methods and tools, often freely available online.
- **Perpetrators**: Usually amateurs or novice attackers, sometimes referred to as "script kiddies."
- **Target**: Generally random, without a specific victim in mind.
- **Duration**: Short-lived and often easily detectable.

Examples:

1. **Brute Force Attacks**: Attempting to guess passwords using trial-and-error methods.
2. **Phishing Scams**: Sending fraudulent emails to a broad audience hoping to steal credentials.
3. **Malware Distribution**: Spreading basic viruses or Trojans through infected attachments or software.
4. **Website Defacement**: Hacktivists or pranksters altering websites for fun or minor political messages.

Prevention:

- Educate users on basic cybersecurity practices (e.g., recognizing phishing attempts).
- Use strong passwords and change them regularly.
- Keep software and systems up to date.
- Install antivirus and antimalware software.

c.) Comparison Between Structured and Unstructured Attacks

Aspect	Structured Attacks	Unstructured Attacks
Skill Level	High (professionals, organizations)	Low (amateurs, script kiddies)
Planning	Extensive planning and preparation	Minimal or no planning
Tools Used	Custom-built tools, advanced malware	Freely available tools and scripts
Targets	Specific and high-value targets	Random or opportunistic targets
Impact	Potentially catastrophic	Usually limited in scale
Detection	Harder to detect due to stealth	Easier to detect

iii.) ACTIVE & PASSIVE ATTACKS

- Cyber attacks can be broadly categorized into two main types: active and passive.

a.) Active attacks:

- Active attacks involve direct interaction with the target system, aiming to alter, damage, or disrupt data or operations.
- The attacker actively interferes with systems.
- These attacks can compromise the integrity and availability of data.

Types of Active attacks:

a) **Masquerade:** in this attack, the intruder pretends to be a particular user of a system to gain access or to gain greater privileges than they are authorized for. A masquerade may be attempted through the use of stolen login IDs and passwords, through finding security gaps in programs or through bypassing the authentication mechanism.

b) **Session replay**: In this type of attack, a hacker steals an authorized user's log in information by stealing the session ID. The intruder gains access and the ability to do anything the authorized user can do on the website.

c) **Message modification**: In this attack, an intruder alters packet header addresses to direct a message to a different destination or modify the data on a target machine.

d) **Denial of service (DoS)** attack, users are deprived of access to a network or web resource. This is generally accomplished by overwhelming the target with more traffic than it can handle.

e) **Distributed denial-of-service (DDoS)** exploit, large numbers of compromised systems (sometimes called a botnet or zombie army) attack a single target.

b.) Passive attack:

- Passive attacks involve monitoring network traffic without interfering with it.
- These attacks aim to gather information about a system or network to identify vulnerabilities and plan future attacks.
- The attacker remains undetected to gather sensitive information.

Types of Passive attacks:

a) **Eavesdropping (tapping)**: the attacker simply listens to messages exchanged by two entities. For the attack to be useful, the traffic must not be encrypted. Any unencrypted information, such as a password sent in response to an HTTP request, may be retrieved by the attacker.

b) **Traffic analysis:** the attacker looks at the metadata transmitted in traffic in order to deduce information relating to the exchange and the participating entities, e.g. the form of the exchanged traffic (rate, duration, etc.). In the cases where encrypted data are used, traffic analysis can also lead to attacks by cryptanalysis, whereby the attacker may obtain information or succeed in unencrypting the traffic.

c) **Software Attacks:** Malicious code (sometimes called malware) is a type of software designed to take over or damage a computer user's operating system, without the user's knowledge or approval. It can be very difficult to remove and very damaging.

c.) Comparison Between Active & Passive Attacks

Feature	Active Attacks	Passive Attacks
Interaction	Direct interaction with the system or network	Monitoring network traffic without interaction
Impact	Can compromise integrity and availability	Primarily threatens confidentiality
Detection	Easier to detect due to their disruptive nature	Harder to detect as they operate silently
Mitigation	Requires strong security measures, such as firewalls, intrusion detection systems, and encryption	Focuses on data protection, encryption, and access controls

iv.) HARDWARE & SOFTWARE ATTACKS:

- Cyberattacks can target both hardware and software components of a system.

a.) Hardware Attacks

- Hardware attacks target physical components of a system.

Types of Hardware attacks:

- **Physical Tampering:** This involves physically accessing a device to modify or steal components.
- **Firmware Attacks**: Injecting malicious code into firmware (e.g., BIOS).
- **Hardware Trojans:** Malicious hardware components, often microscopic, that are inserted into legitimate devices to compromise their security.
- **Side-Channel Attacks:** Exploiting physical characteristics of a device, like timing variations or power consumption, to extract sensitive information.
- **Supply Chain Attacks:** Targeting the supply chain to introduce malicious hardware components into legitimate devices.

b.) Software Attacks

- Software attacks target vulnerabilities in software applications and operating systems.

Types of software attacks:

- **Malware:** Malicious software, such as viruses, worms, and ransomware, designed to harm systems or steal data.
- **Phishing Attacks:** Deceiving users into revealing sensitive information through fraudulent emails or websites.
- **SQL Injection:** Exploiting vulnerabilities in web applications to execute malicious SQL queries.
- **Cross-Site Scripting (XSS):** Injecting malicious code into web pages to steal user data or hijack sessions.
- **Buffer Overflow / Exploits**: Exploiting vulnerabilities in software that allow attackers to execute arbitrary code.
- **Zero-Day Exploits:** Exploiting vulnerabilities that are unknown to the software vendor, allowing attackers to exploit them before a patch is available.
- **Trojan Horses**: Disguising malicious software as legitimate programs.

1.1.2. REASONS FOR COMMISSION OF CYBER CRIMES

- Cybercrime occurs due to a variety of reasons, often influenced by the motives, capabilities, and opportunities of the perpetrators. Here are the primary reasons behind cybercrime:

1. Financial Gain

- **Theft**: Cybercriminals steal money, sensitive data, or intellectual property for personal profit.
  - Examples: Credit card fraud, ransomware attacks, and online banking fraud.
- **Scams**: Use phishing, fake investment schemes, or lottery fraud to exploit victims financially.

2. Revenge or Grudges

- **Disgruntled Insiders**: Employees or associates who sabotage systems or steal data as an act of retaliation.
- **Personal Revenge**: Hacking into personal accounts or sharing private information to harm someone's reputation or privacy.

3. Espionage and Information Theft

- **Corporate Espionage**: Stealing trade secrets or intellectual property to gain a competitive advantage.
- **Government Espionage**: Cyberattacks between nations to obtain confidential political, military, or economic information.

4. Ideological or Political Motivations

- **Hacktivism**: Cyberattacks carried out to promote a political or social agenda, often targeting institutions or governments.
- **Terrorism**: Cyberterrorists attack critical infrastructure to create fear and disrupt public order.
- **Disinformation Campaigns**: Spreading fake news to manipulate public opinion or influence elections.

5. Curiosity or Challenge

- **Amateur Hackers**: Individuals who hack systems to satisfy curiosity or test their technical skills.
- **Thrill-Seekers**: Attackers who enjoy the excitement of breaking into secure systems.

6. Lack of Awareness and Security Measures

- **Exploitation of Weaknesses**: Many attacks occur because organizations or individuals fail to implement proper security measures.
    - Examples: Using weak passwords, unpatched software, or unsecured networks.
- **Human Error**: Clicking on phishing links or downloading malicious attachments inadvertently aids cybercriminals.

7. Opportunity and Accessibility

- **Availability of Tools**: The internet provides access to hacking tools, malware kits, and tutorials, enabling even inexperienced individuals to engage in cybercrime.
- **Global Connectivity**: The widespread use of digital platforms creates numerous entry points for cybercriminals.

8. Anonymity in Cyberspace

- **Difficult to Trace**: The internet provides a degree of anonymity, making it harder for authorities to identify perpetrators.
- **Use of Dark Web**: Cybercriminals operate on the dark web to trade illegal goods, services, and information while hiding their identities.

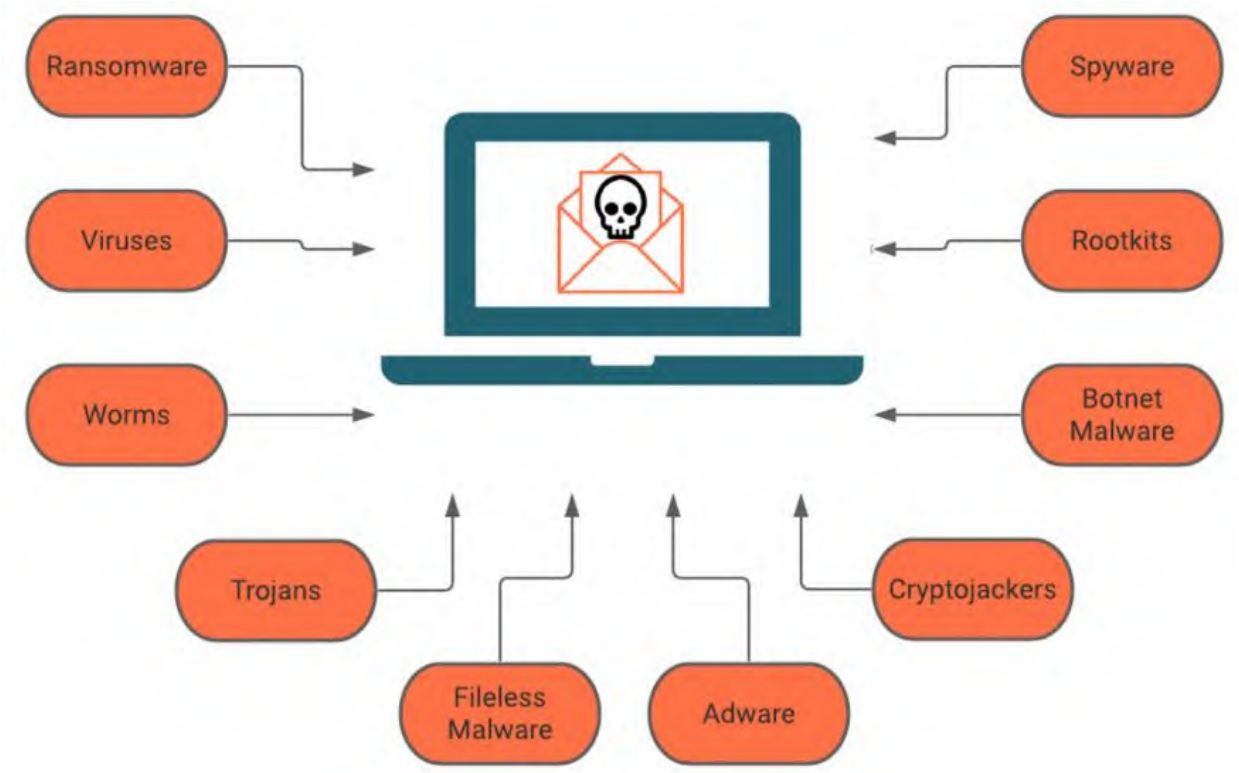
9. Weak Legal Frameworks

- **Inadequate Laws**: Some countries lack comprehensive cybercrime laws, making it easier for criminals to operate without fear of prosecution.
- **Jurisdictional Challenges**: Cybercrimes often cross national borders, complicating legal enforcement.

10. Psychological and Sociological Factors

- **Greed**: Desire for quick and easy money drives many individuals into cybercrime.
- **Peer Influence**: Association with criminal networks or communities encourages illegal activities.
- **Lack of Ethical Values**: Some individuals engage in cybercrime due to moral indifference or lack of awareness about its consequences.

1.2. MALWARE AND IT'S TYPES



- Malware stands for **"Malicious Software"** and it refers to any software designed to infiltrate, damage, or exploit computers and networks without the user's consent.
- Malware is normally installed and runs on your computer, smartphone, or other device without your knowledge and/or consent. It can get onto your system by a variety of methods, including email, web pages and links, and infected USB drives.
- They perform unwanted tasks in the host computer for the benefit of a third party.
- Malwares which can seriously degrade the performance of the host machine.
- There is a full range of malwares which are simply written to distract/annoy the user, to the complex ones which captures the sensitive data from the host machine and send it to remote servers.
- Steal sensitive personal, business, proprietary, or financial information.
- Disrupt normal system operations.
- Spy.

- Lock files on the victim's system and hold them for ransom.
- Perform attacks on other systems, including distributed-denialof-service (DDoS) attacks.
- Gain unauthorized access to a system.
- Send spam and other emails.
- While malware can perform more than one purpose, most malware is categorized by its main function or how it attacks the system. Some of the popular ones are:

1.2.1. VIRUSES

- Programs that **attach themselves** to legitimate files or software and spread when the infected file is executed.
- A virus is a malicious code written to damage/harm the host computer by deleting or appending a file, occupy memory space of the computer by replicating the copy of the code, slow down the performance of the computer, format the host machine, etc.
- It can be spread via email attachment, pen drives, digital images, e-greeting, audio or video clips, etc.
- A virus may be present in a computer but it **cannot activate itself without the human intervention**.
- Until and unless the executable file(.exe) is execute, a virus cannot be activated in the host machine.
- **Purpose**: Corrupt, delete, or steal data; disrupt systems.
- **Example**: File-infecting viruses that damage executable files.
- **Prevention**: Use antivirus software and avoid running unknown files.

Type of Viruses

1. **File Infector Viruses** : Infect executable files such as .exe or .com. When the infected program is run, the virus activates. **Effect**: Corrupts files, slows down system performance, or spreads further. **Example**: Cascade Virus.
2. **Boot Sector Viruses :** Target the boot sector of a hard drive or removable media (e.g., USB drives). They execute during the system boot process. **Effect**: Prevent the system from booting, corrupt critical startup files. **Example**: Michelangelo Virus.
3. **Macro Viruses :**Written in macro languages used by software like Microsoft Word or Excel. These viruses activate when infected documents are opened. **Effect**: Execute malicious macros, corrupt documents, or steal data. **Example**: Melissa Virus.

4. **Polymorphic Viruses** : Alter their code each time they infect a new file, making them harder to detect by traditional antivirus software. **Effect**: Evades detection, spreading widely. **Example**: Storm Worm.
5. **Metamorphic Viruses** : Rewrite their own code completely while maintaining their functionality, making detection more challenging than polymorphic viruses. **Effect**: Avoid signature-based antivirus tools. **Example**: Simile Virus.
6. **Resident Viruses :** Reside in a computer's memory and activate whenever the operating system performs specific functions, such as file access. **Effect**: Continuously infect files or programs. **Example**: Randex.
7. **Non-Resident Viruses** : Do not reside in memory. They operate by infecting files directly and do not stay active after execution. **Effect**: Spread less persistently than resident viruses. **Example**: Vienna Virus.
8. **Multipartite Viruses** : Combine multiple infection methods, targeting both files and the boot sector. **Effect**: Harder to remove as they re-infect the system through different mechanisms. **Example**: GhostBall Virus.
9. **Web Scripting Viruses** : Exploit vulnerabilities in web browsers or scripts on websites. **Effect**: Redirect users to malicious websites or execute harmful scripts on the user's device. **Example**: JS.Fortnight.
10. **Armored Viruses :** Use obfuscation techniques to make themselves more difficult to analyze or detect. **Effect**: Complicates reverse engineering by cybersecurity experts. **Example**: Whale Virus.
11. **Overwrite Viruses :** Overwrite file content with malicious code, rendering the original file useless. **Effect**: Permanent loss of file data. **Example**: Trivial.88.D.
12. **Spacefiller (Cavity) Viruses** : Insert themselves into unused sections of a file to avoid detection and maintain file functionality. **Effect**: Difficult to detect without corrupting the host file. **Example**: Lehigh Virus.

1.2.2. WORMS

- They are a class of virus which can replicate themselves.
- Standalone programs that self-replicate and spread across networks without user intervention.
- They are different from the virus by the fact that they does not require human intervention to travel over the network and spread from the infected machine to the whole network.
- Worms can spread either through network, using the loopholes of the Operating System or via email.

- The replication and spreading of the worm over the network consumes the network resources like space and bandwidth and force the network to choke.
- **Purpose**: Consume system resources, cause network congestion, and spread malicious payloads.
- **Example**: Conficker worm, which exploited vulnerabilities in Windows systems.
- **Prevention**: Keep systems updated and use firewalls.

Type of Worms

1. **Internet Worms :** Spread through the internet by exploiting vulnerabilities in software or protocols. **Effect**: Overwhelm networks, slow down systems, or deliver malicious payloads. **Example**: SQL Slammer Worm.
2. **Email Worms** : Spread via email attachments or links. They often disguise themselves as legitimate files or messages. **Effect**: Infect email systems, steal data, or distribute spam. **Example**: Mydoom Worm.
3. **File-Sharing Worms** : Distribute through peer-to-peer (P2P) networks or file-sharing platforms. **Effect**: Infect files shared on these platforms, spreading to users who download them. **Example**: Kazaa Worm.
4. **Instant Messaging (IM) Worms** : Spread through instant messaging platforms by sending malicious links or files to contacts. **Effect**: Compromise user accounts, spread quickly to other contacts. **Example**: Bropia Worm.
5. **Network Worms** : Propagate through networks by exploiting unpatched vulnerabilities in devices or protocols. **Effect**: Cause network congestion, steal data, or deploy secondary payloads like ransomware. **Example**: Conficker Worm.
6. **USB Worms** : Spread through removable storage devices like USB drives by creating hidden malicious files. **Effect**: Infect systems when the device is plugged in, spreading further through connected devices. **Example**: Stuxnet.
7. **IRC Worms (Internet Relay Chat Worms)** : Spread through IRC channels by sending malicious commands or links. **Effect**: Infect users connected to the same IRC network or channel. **Example**: Agobot.
8. **Mobile Worms** : Target mobile devices, spreading through Bluetooth, SMS, or malicious apps. **Effect**: Steal sensitive information, infect other devices, or cause system malfunctions. **Example**: Cabir Worm.
9. **Cryptoworms** : Encrypt data or systems as they spread, effectively acting like a ransomware worm. **Effect**: Lock systems or files, demanding payment for decryption keys. **Example**: WannaCry Worm.

10. **Botnet Worms** : Turn infected devices into bots within a larger botnet, often used for cyberattacks like DDoS. **Effect**: Gain control of large networks of compromised devices for malicious purposes. **Example**: Sasser Worm.
11. **Social Worms** : Spread through social media platforms by sending malicious links or posts that appear legitimate. **Effect**: Infect accounts, steal credentials, or propagate further through users' networks. **Example**: Koobface Worm.
12. **Multi-Vector Worms** : Use multiple propagation methods, such as email, networks, and file sharing, to maximize spread. **Effect**: Versatile and more difficult to contain due to their varied attack vectors. **Example**: Nimda Worm.

1.2.3. TROJANS

- **Trojan horse** is a malicious code that is installed in the host machine by **pretending to be useful software**. **(NET 2014)**
- The user clicks on the link or download the file which pretends to be a useful file or software from legitimate source.
- It not only damages the host computer by manip ulating the data but also it creates a backdoor in the host computer so that it could be controlled by a remote computer.
- It can become a part of **botnet (robot-network)**, a network of computers which are infected by malicious code and controlled by central controller.
- The computers of this network which are infected by malicious code are known as **zombies**.
- Trojens neither infect the other computers in the network nor do they replicate.

- **Purpose**: Provide unauthorized access, steal data, or install additional malware.
- **Example**: Remote Access Trojans (RATs) allow attackers to control a device remotely.
- **Prevention**: Download software only from trusted sources.

Types of Trojans

1. **Backdoor Trojans :** Create a backdoor in the victim's system, allowing unauthorized access for attackers. **Effect**: Enable remote control of the system, installation of other malware, or data theft. **Example**: Poison Ivy.

2. **Banking Trojans** : Designed to steal financial information, such as banking credentials, payment details, or personal identification. **Effect**: Conduct unauthorized transactions or steal funds directly. **Example**: Zeus Trojan.
3. **Remote Access Trojans (RATs)** : Provide attackers with full control over the victim's system remotely. **Effect**: Monitor user activity, steal data, activate webcams or microphones, or manipulate files. **Example**: DarkComet RAT.
4. **Downloader Trojans** : Download and install additional malicious software onto the infected system. **Effect**: Serve as an entry point for other malware, such as ransomware or spyware. **Example**: Emotet Trojan.
5. **Data-Sending Trojans** : Harvest sensitive information like usernames, passwords, and other personal data, then send it to attackers. **Effect**: Compromise accounts, enable identity theft, or sell data on the dark web. **Example**: Agent Tesla.

6. **Keylogger Trojans** : Record keystrokes made on the victim's device to capture sensitive information, such as login credentials. **Effect**: Compromise accounts and steal personal or financial data. **Example**: HawkEye.
7. **Rootkit Trojans** : Install rootkits that modify the operating system to conceal their presence and other malicious activity. **Effect**: Avoid detection, enabling persistent access for attackers. **Example**: ZeroAccess Trojan.
8. **Trojan-Banker** : Specifically designed to steal banking and credit card information. **Effect**: Fraudulent financial transactions, emptying accounts, or selling data. **Example**: Dyre Trojan.
9. **Trojan-Ransom** : Encrypts or locks the victim's files, demanding a ransom for their release. **Effect**: Extort victims for payment to regain access to their data. **Example**: Cryptolocker.
10. **Trojan-Spy** : Spies on user activity, capturing screenshots, tracking browsing history, or recording sensitive data. **Effect**: Collect personal, financial, or business-critical information. **Example**: SpyEye.
11. **Fake AV Trojans** : Pose as legitimate antivirus software to trick users into purchasing fake services or downloading more malware. **Effect**: Steal money or install additional malicious software. **Example**: Win32/FakeRean.
12. **Game-Thief Trojans** : Target online gamers, stealing login credentials or virtual assets. **Effect**: Hijack gaming accounts or sell virtual goods for profit. **Example**: OnlineGames Trojan.

13. **Mailfinder Trojans** : Harvest email addresses from the victim's system. **Effect**: Send spam or phishing campaigns using stolen email data. **Example**: Trojan-Mailfinder.
14. **DDoS Trojans** : Compromise devices to form botnets that launch Distributed Denial of Service (DDoS) attacks. **Effect**: Overload target systems, making them inaccessible. **Example**: Mirai Trojan.
15. **Exploit Trojans** : Exploit known vulnerabilities in software or systems to install other malware or gain access. **Effect**: Compromise systems with outdated or unpatched software. **Example**: Blackhole Exploit Kit.
16. **SMS Trojans** : Target mobile devices to send unauthorized SMS messages to premium numbers or steal data. **Effect**: Financial loss or unauthorized access to mobile communications. **Example**: FakePlayer Trojan.

1.2.4. RANSOMWARE

- Encrypts the victim's files or locks them out of their system and demands a ransom for access.
- It holds the host computer hostage until the ransom is paid. The malicious code can neither be uninstalled nor can the computer be used till the ransom is paid.
- **Purpose**: Extort money from individuals or organizations.
- **Example**: WannaCry ransomware attack in 2017. **Prevention**: Use regular backups, avoid suspicious emails, and maintain updated security software.

Types of Ransomware

1. **Crypto Ransomware** : Encrypts the victim's files, rendering them inaccessible until a ransom is paid. **Effect**: Loss of access to personal or business-critical data. **Example**: WannaCry, Cryptolocker.
2. **Locker Ransomware** : Locks users out of their devices or systems entirely, preventing access to anything. **Effect**: Denial of system use until a ransom is paid. **Example**: WinLock.
3. **Scareware** : Uses fake warnings or alerts claiming the system is infected with viruses or has issues, demanding payment to fix the "problem." **Effect**: Tricks users into paying, though the files may not actually be encrypted. **Example**: Rogue security software.
4. **RaaS (Ransomware-as-a-Service)** : A subscription-based model where ransomware developers sell or lease their tools to other attackers. **Effect**:

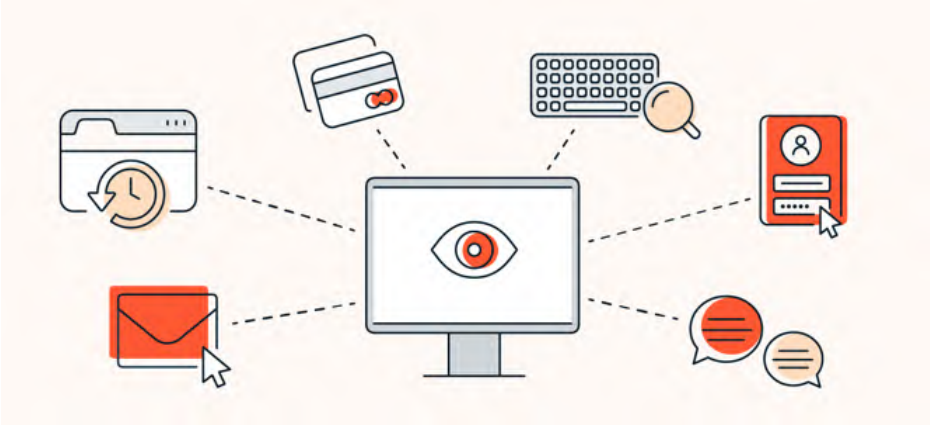
Expands ransomware attacks through less-skilled cybercriminals. **Example**: REvil, DarkSide.

5. **Mobile Ransomware** : Targets mobile devices, locking them or encrypting files, often spread via malicious apps or links. **Effect**: Denial of access to the phone or its data. **Example**: Simplocker.
6. **Double Extortion Ransomware** : Encrypts data and simultaneously exfiltrates sensitive information, threatening to release it unless the ransom is paid. **Effect**: Increases pressure on victims to pay, even if backups are available. **Example**: Maze, Clop.
7. **Triple Extortion Ransomware** : Adds a third layer of extortion, such as launching Distributed Denial of Service (DDoS) attacks if the ransom is not paid. **Effect**: Further disrupts the victim's operations and tarnishes their reputation. **Example**: Avaddon.
8. **Wiper Ransomware** : Instead of providing a decryption key upon payment, it wipes data, causing permanent loss. **Effect**: Data destruction regardless of ransom payment. **Example**: NotPetya.
9. **Social Engineering Ransomware** : Delivered via phishing emails, malicious links, or fake downloads to trick users into activating the ransomware. **Effect**: Exploits human error to spread malware. **Example**: Ryuk.
10. **Fileless Ransomware** : Operates without leaving traditional file traces, running entirely in memory to evade detection by antivirus tools. **Effect**: Harder to detect and prevent. **Example**: Sodinokibi (REvil).
11. **IoT Ransomware** : Targets Internet of Things (IoT) devices, locking them or threatening to disable critical operations. **Effect**: Poses significant risks to smart homes, healthcare devices, and industrial systems. **Example**: R4IoT.
12. **Disk Encryptors** : Encrypt entire disks or drives, preventing the victim from booting the system. **Effect**: Total denial of access to the system until payment is made. **Example**: Petya.
13. **Localized Ransomware** : Designed to target specific regions or languages, often using culturally relevant messaging. **Effect**: Aims at specific demographic or business targets for higher success rates. **Example**: "Chinese Ransomware" targeting Chinese-speaking users.
14. **Hybrid Ransomware** : Combines elements of different ransomware types, such as crypto and locker functionalities. **Effect**: Increases the attack's effectiveness and victim's likelihood to pay. **Example**: Cerber.

15. **Human-Operated Ransomware** : Attacks are manually conducted by cybercriminals who infiltrate systems to ensure maximum damage before deploying ransomware. **Effect**: Tailored, high-impact attacks targeting large organizations. **Example**: Conti.

1.2.5. SPYWARE

- Software that secretly monitors user activity and collects information without consent.



- It is a special type software of which is installed in the target computer with or without the user permission and is designed to steal sensitive information from the target machine.
- Mostly it gathers the browsing habits of the user a nd send it to the remote server without the knowledge of the owner of the computer.
- Most of the time they are downloaded in to the host computer while downloading freeware i.e. free application programmes from the internet.
- Spywares may be of various types; It can keeps track of the cookies of the host computer, it can act as a keyloggers to sniff the banking passwords and sensitive information, etc.
- **Purpose**: Steal sensitive information such as passwords, credit card details, or browsing habits.
- **Example**: Keyloggers record keystrokes to capture login credentials.
- **Prevention**: Use anti-spyware tools and regularly check for unusual system behavior.

Types of Spyware

1. **Adware** : Tracks users' online activities to display targeted advertisements. **Effect**: Slows down devices, disrupts user experience with intrusive ads, and compromises privacy. **Example**: CoolWebSearch.

இன்றைய *TRB* பயிற்சியாளரே நாளைய அரசு பள்ளி ஆசிரியரே!

Teacher's Care Academy **கடந்த *14* ஆண்டுகளாக *TRB* தேர்வுகளுக்கான சிறப்பு பயிற்சியை வழங்கி வருகிறது. இதுவரை *10,000*-க்கும் மேற்பட்ட ஆசிரியர்களை அரசு வேலைகளில் வெற்றிகரமாக நியமிக்க உதவியதில் நாங்கள் பெருமிதம் கொள்கிறோம். எங்கள் நிறுவனத்தில் அனைத்து *TRB* தேர்வுகளுக்கும் விரிவான பயிற்சிகள் உள்ளன, அவை:**

- ***PGTRB***
- ***UGTRB***
- ***SGT***
- ***POLYTECHNIC TRB***
- ***BEO***
- ***TET Paper I & II***
- ***College TRB***
- ***Special Teachers***

கூடுதலாக, தமிழ்நாடு அரசு இப்போது அனைத்து அரசு பணிக்கான தேர்வாணையங்களுக்கு *(TRB, TNPSC, MRB, TNUSRB)* தமிழ் மொழி கட்டாய தகுதி தேர்வு *(Tamil Compulsory Exam)* முதற்கட்ட தேர்வாக அறிவித்துள்ளது இதற்காக தமிழ் மொழி கட்டாய தகுதி தேர்வு என்ற புத்தகத்தை பிரத்தியேகமாக உங்கள் *Teacher's Care Academy* வெளியிட்டுள்ளது. இந்த புத்தகம் அமேசானிலும் கிடைக்கிறது ஆனால் எங்களை நேரடியாக தொடர்பு கொண்டு வாங்கும் போது உங்களுக்கு கூடுதல் தள்ளுபடி கிடைக்கும்

PGTRB

PGTRB தேர்விற்கு நாங்கள் அனைத்து மொழி பாடத்திற்கும் பயிற்சிகளை வழங்கி வருகிறோம் அதாவது

- ***Tamil***
- ***English***
- ***Mathematics***
- ***Physics***
- ***Chemistry***
- ***Botany***
- ***Zoology***
- ***Economics***
- ***Commerce***
- ***Computer Science***
- ***History***

மேற்கண்ட அனைத்து படப்பிரிவுகளுக்கான ***Study Material***-களுடன் ***Psychology, Tamil Eligibility Book, Question Bank*** மற்றும் ***General Knowldge Material***-களும் வழங்கப்படும்

TET (Teachers Eligibility Test)

TET தேர்விற்கு நம் ***Teachers Care Academy***-யில் ***Paper I*** மற்றும் ***Paper II*** என இரண்டு தாள்களுக்கும் பிரத்தியேகமாக பயிற்சிகளை வழங்குகிறோம்

இதற்கு தமிழ்நாடு அரசால் வழங்கப்பட்டுள்ள பள்ளி பாட புத்தகத்தில் இருந்து குறிப்புகளை எடுத்து ***Study Material***-களாக வழங்குகிறோம்

மேலும் ***Psychology***-க்கு ***TRB***-ஆல் வழங்கப்பட்டுள்ள பாடத்திட்டத்தை பின்பற்றி பல்வேறு ***Reference Book***-லிருந்து குறிப்புகளை எடுத்து ***Study Material***-களாக வழங்குகிறோம்

UGTRB

TET தேர்வில் வெற்றி பெற்ற ஆசிரியர்களுக்கு நடத்தப்படும் ***UGTRB*** போட்டி தேர்வுக்காக அனைத்து மொழி பாடத்திற்கும் பயிற்சிகளை வழங்கி வருகிறோம் அதாவது

- ***Tamil***
- ***English***
- ***Mathematics***
- ***Physics***
- ***Chemistry***
- ***Botany***
- ***Zoology***
- ***History***
- ***Geography***

SGTRB

TET தேர்வில் வெற்றி பெற்ற ஆசிரியர்களுக்கு நடத்தப்படும் ***SGTRB*** போட்டி தேர்வுக்காக தமிழ்நாடு அரசால் வழங்கப்பட்டுள்ள பள்ளி பாட புத்தகத்தில் இருந்து குறிப்புகளை எடுத்து ***Study Material***-களாக வழங்குகிறோம்

BEO

BEO தேர்வுக்காக ***TRB***-ஆல் பாடத்திட்டம் வெளியிடப்பட்டுள்ளது அந்த பாடத்திட்டத்தின் அடிப்படையில் அனைத்து பாடத்திற்கும் உங்கள் ***Teachers Care Academy*** அலகு ***(Unit-Wise)*** வாரியாக ***Study Material***-களை வழங்குகிறது.

POLYTECHNIC TRB

Polytechnic தேர்விற்காக உங்கள் ***Teachers Care Academy*** பின்வரும் மொழி பாடத்திற்கு பயிற்சிகளை வழங்கி வருகிறது. அதாவது,

- ***Civil***
- ***EEE***
- ***ECE***
- ***CSE***
- ***Mechanical***
- ***English***
- ***Mathematics***
- ***Physics***
- ***Chemistry***

College TRB

தமிழ்நாட்டில் அரசு கல்லூரிகளில் காலியாக உள்ள உதவி பேராசிரியர் பணிக்கு ***TRB*** வெகு விரைவில் போட்டித் தேர்வை நடத்த இருக்கிறது

அந்த தேர்வுக்காக நம் ***Teachers Care Academy***-யில் பின்வரும் மொழி பாடத் திட்டத்திற்கும் பயிற்சிகளை வழங்கி வருகிறது

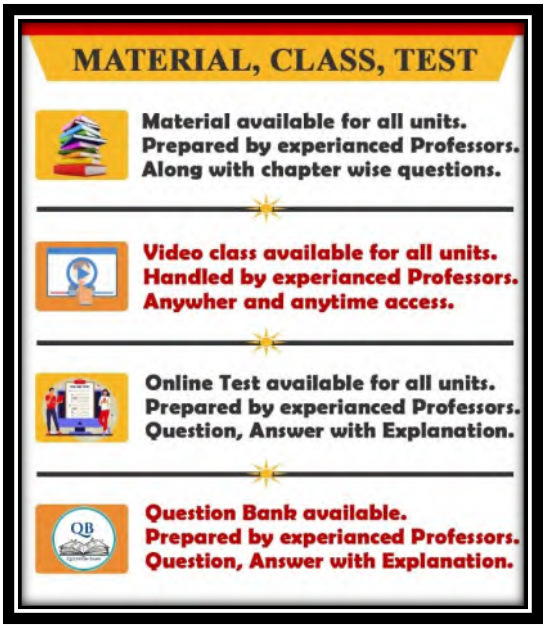
- ***Tamil***
- ***English***
- ***Mathematics***
- ***Physics***
- ***Chemistry***
- ***Botany***
- ***Zoology***
- ***Economics***
- ***Commerce***
- ***Computer Science***
- ***History***
- ***Geography***

Special Teachers

TRB-ஆல் நடத்தப்படும் சிறப்பாசிரியர் தேர்வுக்காக நம் Teachers Care Academy-யில் பின்வரும் பாடத்திட்டத்திற்கு பிரத்தியேகமாக பயிற்சிகள் வழங்கப்பட்டு வருகிறது. அதாவது,

- ***Sewing***
- ***Drawing***
- ***Music***
- ***PET***

மேற்கண்ட அனைத்து தேர்வுகளுக்கும் உங்கள் Teachers Care Academy பலவிதமான பயிற்சிகளை வழங்குகிறது, அவை



இந்த ஆண்டு (2024) TNPSC Batch-யும் அறிமுகம் செய்திருப்பதில் நாங்கள் பெருமிதம் கொள்கிறோம். எங்கள் வழிகாட்டுதல் வரவிருக்கும் தேர்வுகளில் நீங்கள் வெற்றி பெற உதவும் என நாங்கள் உறுதியாக நம்புகிறோம்.

உங்கள் அரசு ஆசிரியர் பணி கனவு நிறைவேற வாழ்த்துக்கள்!

அன்புடன்,

Teacher's Care Academy