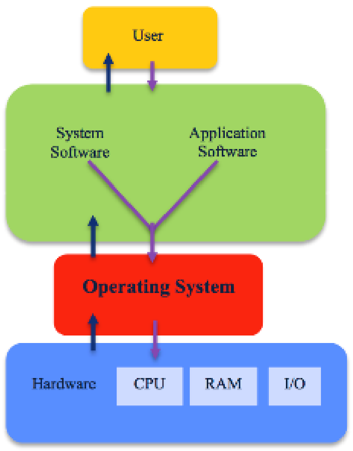
UNIT-I

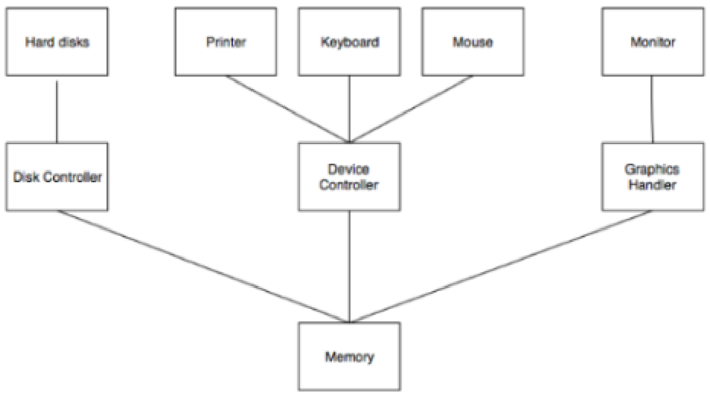
Introduction: What Operating Systems do, Computer system organization, Computer system architecture, Operating system structure. System Structure: Operating system services, User operating system interface, System Calls, Types of System Calls, Overview of UNIX Operating System, Basic features of Unix operating System.

What Operating Systems do:

The operating system (OS) manages all of the software and hardware on the computer. It performs basic tasks such as file, memory and process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

Most of the time, there are several different computer programs running at the same time, and they all need to access your





computer's central processing unit (CPU), memory and storage. The operating system coordinates all of this to make sure each program gets what it needs.

Computer system organization:

The computer system is a combination of many parts such as peripheral devices, secondary memory, CPU etc. This can be explained more clearly using a diagram.

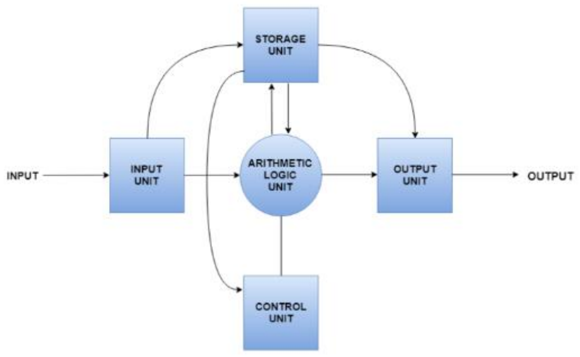
The salient points about the above figure displaying Computer System Organization is −

- The I/O devices and the CPU both execute concurrently. Some of the processes are scheduled for the CPU and at the same time, some are undergoing input/output operations.
- There are multiple device controllers, each in charge of a particular device such as keyboard, mouse, printer etc.
- There is buffer available for each of the devices. The input and output data can be stored in these buffers.
- The data is moved from memory to the respective device buffers by the CPU for I/O operations and then this data is moved back from the buffers to memory.
- The device controllers use an interrupt to inform the CPU that I/O operation is completed.

Computer system architecture:

A computer system is basically a machine that simplifies complicated tasks. It should maximize performance and reduce costs as well as power consumption. The different components in the Computer System Architecture are Input Unit, Output Unit, Storage Unit, Arithmetic Logic Unit, Control Unit etc.

A diagram that shows the flow of data between these units is as follows −



The input data travels from input unit to ALU. Similarly, the computed data travels from ALU to output unit. The data constantly moves from storage unit to ALU and back again. This is because stored data is computed on before being stored again. The control unit controls all the other units as well as their data.

Details about all the computer units are −

- **Input Unit**

  The input unit provides data to the computer system from the outside. So, basically it links the external environment with the computer. It takes data from the input devices, converts it into machine language and then loads it into the computer system. Keyboard, mouse etc. are the most commonly used input devices.

- **Output Unit**

  The output unit provides the results of computer process to the users i.e it links the computer with the external environment. Most of the output data is the form of audio or video. The different output devices are monitors, printers, speakers, headphones etc.

- **Storage Unit**

  Storage unit contains many computer components that are used to store data. It is traditionally divided into primary storage and secondary storage. Primary storage is also known as the main memory and is the memory directly accessible by the CPU. Secondary or external storage is not directly accessible by the CPU. The data from secondary storage needs to be brought into the primary storage before the CPU can use it. Secondary storage contains a large amount of data permanently.

- **Arithmetic Logic Unit**

  All the calculations related to the computer system are performed by the arithmetic logic unit. It can perform operations like addition, subtraction, multiplication, division etc. The control unit transfers data from storage unit to arithmetic logic unit when calculations need to be performed. The arithmetic logic unit and the control unit together form the central processing unit.

- **Control Unit**

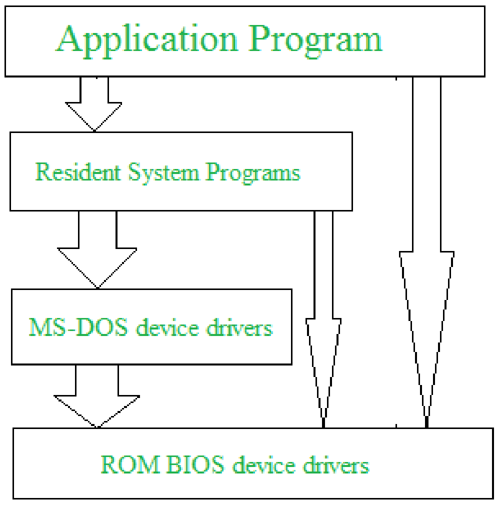
  This unit controls all the other units of the computer system and so is known as its central nervous system. It transfers data throughout the computer as required including from storage unit to central processing unit and vice versa. The control unit also dictates how the memory, input output devices, arithmetic logic unit etc. should behave.

OPERATING SYSTEM STRUCTURE:

Operating system can be implemented with the help of various structures. The structure of the OS depends mainly on how the various common components of the operating system are interconnected and melded into the kernel. Depending on this we have following structures of the operating system:

Simple_structure:

Such operating systems do not have well defined structure and are small, simple and limited systems. The interfaces and levels of functionality are not well separated. MS-DOS is an example of such operating system. In MS-DOS application programs are able to access the basic I/O routines. These types of operating system cause the entire system to crash if one of the user programs fails. Diagram of the structure of MS-DOS is shown below.



Advantages of Simple structure:

- It delivers better application performance because of the few interfaces between the application program and the hardware.
- Easy for kernel developers to develop such an operating system.

Disadvantages of Simple structure:

- The structure is very complicated as no clear boundaries exists between modules.
- It does not enforce data hiding in the operating system.

OPERATING SYSTEM SERVICES :

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system −

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

Program execution :

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use). Following are the major activities of an operating system with respect to program management –

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

I/O Operation :

An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

An Operating System manages the communication between user and device drivers.

- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

File system manipulation :

A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Following are the major activities of an operating system with respect to file management −

- Program needs to read a file or write a file.
- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

Communication:

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication −

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

Error handling :

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling −

- The OS constantly checks for possible errors.
- The OS takes an appropriate action to ensure correct and consistent computing.

Resource Management :

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management −

- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

Protection:

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.

Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection −

- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.

USER OPERATING SYSTEM INTERFACE :

The user and operating system are connected with each other with the help of interface, so interface is used to connect the user and OS.

In computers there are different types of interface that can be used for connection with computers to users and their connection is responsible for data transfer.

Also, in computers there are different interfaces. These interfaces are not necessarily used but can be used in computers whenever it is needed. So, different types of tasks can be performed by the help of different interfaces.

Command line interface

The command-line interface is an interface whenever the user needs to have different commands regarding the input and output and then a task is performed so this is called the command-line argument and it is used to execute the output and create, delete, print, copy, paste, etc.

All these operations are performed with the help of the command-line interface.

The interface is always connected to the OS so that the command given by the user directly works by the OS and a number of operations can be performed with the help of the command line interface because multiple commands can be interrupted at same time and execute only one.

The command line interface is necessary because all the basic operations in the computer are performed with the help of the OS and it is responsible for memory management. By using this we can divide the memory and we can use the memory.

Command Line Interface advantages −

- Controls OS or application
- faster management
- ability to store scripts which helps in automating regular tasks.
- Troubleshoot network connection issues.

Command Line Interface disadvantages −

- The steeper learning curve is associated with memorizing commands and a complex syntax.
- Different commands are used in different shells.

Graphical user interface

The graphical user interface is used for playing games, watching videos, etc. these are done with the help of GUI because all these applications require graphics.

The GUI is one of the necessary interfaces because only by using the user can clearly see the picture, play videos.

So we need GUI for computers and this can be done only with the help of an operating system.

When a task is performed in the computer then the OS checks the task and defines the interface which is necessary for the task. So, we need GUI in the OS.

The basic components of GUIs are −

- Start menu with program groups
- Taskbar which showing running programs
- Desktop screen
- Different icons and shortcuts.

Choice of interface

The interface that is used with the help of OS for a particular task and that task can be performed with minimum possible time and the output is shown on the screen in that case we use the choice of interface.

The choice of interface means the OS checks the task and finds out which interface can be suitable for a particular task. So that type of interface is called the choice of interface and this can be done with the help of an OS.

System Calls:

A system call is a way for a user program to interface with the operating system. The program requests several services, and the OS responds by invoking a series of system calls to satisfy the request. A system call can be written in assembly language or a high-level language like **C** or **Pascal**. System calls are predefined functions that the operating system may directly invoke if a high-level language is used.

In this article, you will learn about the system calls in the operating system and discuss their types and many other things.

What is a System Call?

A system call is a method for a computer program to request a service from the kernel of the operating system on which it is running. A system call is a method of interacting with the operating system via programs. A system call is a request from computer software to an operating system's kernel.

The **Application Program Interface (API)** connects the operating system's functions to user programs. It acts as a link between the operating system and a process, allowing user-level programs to request operating system services. The kernel system can only be accessed using system calls. System calls are required for any programs that use resources.

How are system calls made?

When a computer software needs to access the operating system's kernel, it makes a system call. The system call uses an API to expose the operating system's services to user programs. It is the only method to access the kernel system. All programs or processes that require resources for execution must use system calls, as they serve as an interface between the operating system and user programs.

Below are some examples of how a system call varies from a user function.

1. A system call function may create and use kernel processes to execute the asynchronous processing.
2. A system call has greater authority than a standard subroutine. A system call with kernel-mode privilege executes in the kernel protection domain.
3. System calls are not permitted to use shared libraries or any symbols that are not present in the kernel protection domain.
4. The code and data for system calls are stored in global kernel memory.

Why do you need system calls in Operating System?

There are various situations where you must require system calls in the operating system. Following of the situations are as follows:

1. It is must require when a file system wants to create or delete a file.
2. Network connections require the system calls to sending and receiving data packets.
3. If you want to read or write a file, you need to system calls.
4. If you want to access hardware devices, including a printer, scanner, you need a system call.
5. System calls are used to create and manage new processes.

How System Calls Work:

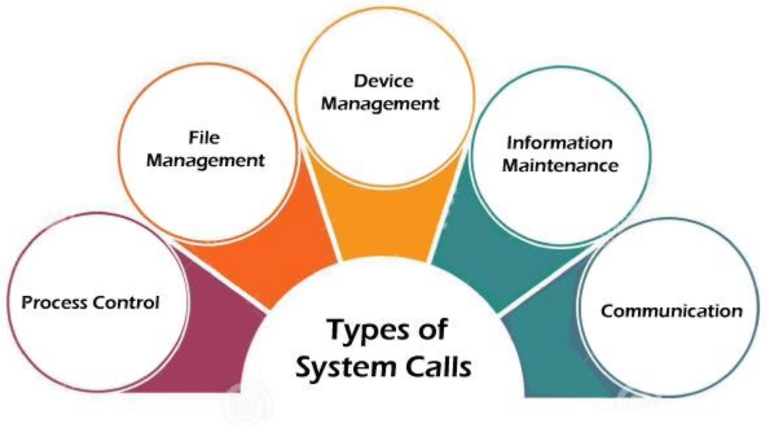
The Applications run in an area of memory known as user space. A system call connects to the operating system's kernel, which executes in kernel space. When an application creates a system call, it must first obtain permission from the kernel. It achieves this using an interrupt request, which pauses the current process and transfers control to the kernel.

If the request is permitted, the kernel performs the requested action, like creating or deleting a file. As input, the application receives the kernel's output. The application resumes the procedure after the input is received. When the operation is finished, the kernel returns the results to the application and then moves data from kernel space to user space in memory.

A simple system call may take few nanoseconds to provide the result, like retrieving the system date and time. A more complicated system call, such as connecting to a network device, may take a few seconds. Most operating systems launch a distinct kernel thread for each system call to avoid bottlenecks. Modern operating systems are multi-threaded, which means they can handle various system calls at the same time.

Types of System Calls :

There are commonly five types of system calls. These are as follows:



1. **Process Control**
2. **File Management**
3. **Device Management**
4. **Information Maintenance**
5. **Communication**

Now, you will learn about all the different types of system calls one-by-one.

Process Control

Process control is the system call that is used to direct the processes. Some process control examples include creating, load, abort, end, execute, process, terminate the process, etc.

File Management

File management is a system call that is used to handle the files. Some file management examples include creating files, delete files, open, close, read, write, etc.

Device Management

Device management is a system call that is used to deal with devices. Some examples of device management include read, device, write, get device attributes, release device, etc.

Information Maintenance

Information maintenance is a system call that is used to maintain information. There are some examples of information maintenance, including getting system data, set time or date, get time or date, set system data, etc.

Communication

Communication is a system call that is used for communication. There are some examples of communication, including create, delete communication connections, send, receive messages, etc.

Examples of Windows and Unix system calls

There are various examples of Windows and Unix system calls. These are as listed below in the table:

Process	Windows	Unix

Process Control	CreateProcess()
	ExitProcess()
	WaitForSingleObject()
		Fork()
		Exit()
		Wait()

File Manipulation	CreateFile()
	ReadFile()
	WriteFile()
	CloseHandle()
		Open()
		Read()
		Write()
		Close()

Device Management	SetConsoleMode()
	ReadConsole()
	WriteConsole()
		Ioctl()
		Read()
		Write()

Information Maintenance	GetCurrentProcessID()
	SetTimer()
	Sleep()
		Getpid()
		Alarm()
		Sleep()

Communication	CreatePipe()
	CreateFileMapping()
	MapViewOfFile()
		Pipe()
		Shmget()
		Mmap()

Protection	SetFileSecurity()
	InitializeSecurityDescriptor()
	SetSecurityDescriptorgroup()
		Chmod()
		Umask()
		Chown()

Here, you will learn about some methods briefly:

open()

The **open()** system call allows you to access a file on a file system. It allocates resources to the file and provides a handle that the process may refer to. Many processes can open a file at once or by a single process only. It's all based on the file system and structure.

read()

It is used to obtain data from a file on the file system. It accepts three arguments in general:

- A file descriptor.
- A buffer to store read data.
- The number of bytes to read from the file.

The file descriptor of the file to be read could be used to identify it and open it using **open()** before reading.

wait()

In some systems, a process may have to wait for another process to complete its execution before proceeding. When a parent process makes a child process, the parent process execution is suspended until the child process is finished. The **wait()** system call is used to suspend the parent process. Once the child process has completed its execution, control is returned to the parent process.

write()

It is used to write data from a user buffer to a device like a file. This system call is one way for a program to generate data. It takes three arguments in general:

- A file descriptor.
- A pointer to the buffer in which data is saved.
- The number of bytes to be written from the buffer.

fork()

Processes generate clones of themselves using the **fork()** system call. It is one of the most common ways to create processes in operating systems. When a parent process spawns a child process, execution of the parent process is interrupted until the child process completes. Once the child process has completed its execution, control is returned to the parent process.

close()

It is used to end file system access. When this system call is invoked, it signifies that the program no longer requires the file, and the buffers are flushed, the file information is altered, and the file resources are de-allocated as a result.

exec()

When an executable file replaces an earlier executable file in an already executing process, this system function is invoked. As a new process is not built, the old process identification stays, but the new process replaces data, stack, data, head, etc.

exit()

The **exit()** is a system call that is used to end program execution. This call indicates that the thread execution is complete, which is especially useful in multi-threaded environments. The operating system reclaims resources spent by the process following the use of the **exit()** system function.

Overview of UNIX Operating System:

- UNIX is a computer operating system.
- An operating system is the program that controls all the other parts of a computer system, both the hardware and the software. It allocates the computer's resources and schedules tasks. It allows you to make use of the facilities provided by the system. Every computer requires an operating system.
- UNIX is a multi-user, multi-tasking operating system. Multiple users may have multiple tasks running simultaneously. This is very different from PC operating systems such as MS-DOS or MS-Windows (which allows multiple tasks to be carried out simultaneously but not multiple users).
- UNIX is a machine independent operating system. Not specific to just one type of computer hardware. Designed from the beginning to be independent of the computer hardware.
- UNIX is a software development environment. Was born in and designed to function within this type of environment.
- The "UNIX" trademark, previously owned by AT&T and then deeded to UNIX Systems Laboratories (USL), an AT&T subsidiary, passed to Novell when it acquired USL. After a brief period of negotiations with rival Unix vendors, namely, Sun Microsystems, Santa Cruz Operation, International Business Machines, and Hewlett-Packard, Novell granted exclusive licensing rights of the UNIX trademark to X/Open Co. Ltd., an Open Systems industry standards branding agent based in the United Kingdom.

History of UNIX

- 1969: Developed at AT&T Bell Labs in Murray Hill, New Jersey, one of the largest research facilities in the world. Created in an environment when most computer jobs were fed into a batch system.
- Developed by researchers who needed a set of computing tools to help them with their projects and their collaborators. Allowed a group of people working together on a project to share selected data and programs.
- 1975: AT&T makes UNIX widely available - offered to educational institutions at minimal cost. Becomes popular with university computer science programs. AT&T distributes standard versions in source form: Version 6 (1975), Version 7 (1978), System III (1981).
- 1984 to date: University of California, Berkeley adds major enhancements, creates Berkeley Standard Distribution (BSD)
- 1984 to date: Many Berkeley features incorporated into new AT&T version: System V
- UNIX has become the operating system of choice for engineering and scientific workstations.
- Two variations maintain popularity today, AT&T System V based and the Berkeley Standard Distribution.
- Current versions (1/95)are System V release 4.2 .and 4.4 BSD
- Work is in progress to develop a Portable Operating System specification based on UNIX (IEEE POSIX committee).

UNIX Philosophy

- Make each program do one thing well. Reusable software tools: 1 tool = 1 function
- Expect the output of every program to become the input of another, yet unknown, program to combine simple tools to perform complex tasks
- Prototyping: get something small working as soon as possible and modify it incrementally until it is finished
- Use terse commands and messages: reduces typing and screen output

---

Why UNIX?

- Hardware independence
  - operating system code is written in C language rather than a specific assembly language
  - operating system software can be easily moved from one hardware system to another
  - UNIX applications can be easily moved to other UNIX machines. Porting is usually as simple as transfer of the source and a recompile
- Productive environment for software development
  - rich set of tools
  - versatile command language
- Distributed processing and multi-tasking

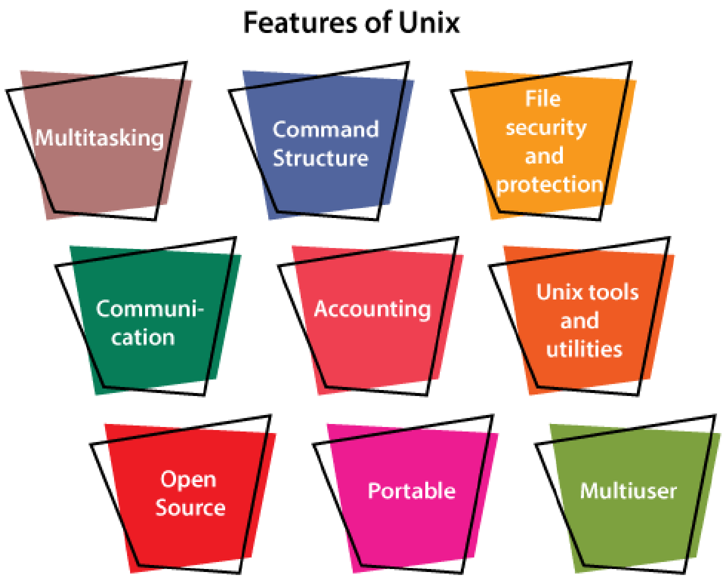
UNIX Components

- **Kernel**
  - The core of the UNIX system. Loaded at system start up (boot). Memory-resident control program.
  - Manages the entire resources of the system, presenting them to you and every other user as a coherent system. Provides service to user applications such as device management, process scheduling, etc.
  - Example functions performed by the kernel are:
    - managing the machine's memory and allocating it to each process.
    - scheduling the work done by the CPU so that the work of each user is carried out as efficiently as is possible.
    - accomplishing the transfer of data from one part of the machine to another
    - interpreting and executing instructions from the shell
    - enforcing file access permissions
  - You do not need to know anything about the kernel in order to use a UNIX system. These details are provided for your information only.
- **Shell**
  - Whenever you login to a Unix system you are placed in a shell program. The shell's prompt is usually visible at the cursor's position on your screen. To get your work done, you enter commands at this prompt.
  - The shell is a command interpreter; it takes each command and passes it to the operating system kernel to be acted upon. It then displays the results of this operation on your screen.
  - Several shells are usually available on any UNIX system, each with its own strengths and weaknesses.
  - Different users may use different shells. Initially, your system administrator will supply a default shell, which can be overridden or changed. The most commonly available shells are:
    - Bourne shell (sh)
    - C shell (csh)
    - Korn shell (ksh)
    - TC Shell (tcsh)
    - Bourne Again Shell (bash)
  - Each shell also includes its own programming language. Command files, called "shell scripts" are used to accomplish a series of tasks.

- **Utilities**
  - UNIX provides several hundred utility programs, often referred to as commands.
  - Accomplish universal functions
    - editing
    - file maintenance
    - printing
    - sorting
    - programming support
    - online info
    - etc.
  - Modular: single functions can be grouped to perform more complex tasks

Basic features of Unix operating System:

Let's discuss the features of UNIX OS one by one in detail.



Multitasking: A UNIX operating system is a multitasking operating system that allows you to initiate more than one task from the same terminal so that one task is performed as a foreground and the other task as a background process.

Multi-user: UNIX operating system supports more than one user to access computer resources like main memory, hard disk, tape drives, etc. Multiple users can log on to the system from different terminals and run different jobs that share the resources of a command terminal. It deals with the principle of time-sharing. Time-sharing is done by a scheduler that divides the CPU time into several segments also called a time slice, and each segment is assigned to each user on a scheduled basis. This time slice is tiny. When this time is expired, it passes control to the following user on the system. Each user executes their set of instructions within their time slice.

Portability: This feature makes the UNIX work on different machines and platforms with the easy transfer of code to any computer system. Since a significant portion of UNIX is written in C language, and only a tiny portion is coded in assembly language for specific hardware.

File Security and Protection: Being a multi-user system, UNIX makes special consideration for file and system security. UNIX has different levels of security using assigning username and password to individual users ensuring the authentication, at the level providing file access permission viz. read, write and execute and lastly file encryption to change the file into an unreadable format.

Command Structure: UNIX commands are easy to understand and simple to use. Example: "cp", mv etc. While working in the UNIX environment, the UNIX commands are case-sensitive and are entered in lower case.

Communication: In UNIX, communication is an excellent feature that enables the user to communicate worldwide. It supports various communication facilities provided using the write command, mail command, talk command, etc.

Open Source: UNIX operating system is open source it means it is freely available to all and is a community-based development project.

Accounting: UNIX keeps an account of jobs created by the user. This feature enhances the system performance in terms of CPU monitoring and disk space checking. It allows you to keep an account of disk space used by each user, and the disk space can be limited by each other. You can assign every user a different disk quota. The root user can perform these accounting tasks using various commands such as quota, df, du, etc.

UNIX Tools and Utilities: UNIX system provides various types of tools and utilities facilities such as UNIX grep, sed and awk, etc. Some of the general-purpose tools are compilers, interpreters, network applications, etc. It also includes various server programs which provide remote and administration services.