Operating System:- An operating system is a large collection of software which manages resources of the computer system, such as memory, processor, file system and input/output devices. It keep track of the status of each resource sand decides who will have control over computer resources, for how long and when. Operating system acts as interface between users and hardware of a computer system . It directly controls computer hardware resources. Other programs rely on facilities provided by the operating system to gain access to computer resources.

There are two ways on can interact with operating system.

1. By means of operating system call in a program.
2. Directly by means of operating system commands.

System call:- System calls provides the interface to a running program and the operating system. User program receives operating system services through the set of system calls.

For example → let us consider a simple program to copy data from one file to another. In an interactive system the following calls will be generate by the operating system.

- ➔Prompt messages for inputting two file names and reading it from terminal
- ➔Open source and destination file.
- ➔Prompt error message in case the source file cannot be open because it is protected against access or destination file cannot be created because there already a file with this name.
- ➔Read the source file
- ➔Write into the destination file
- ➔Display status information regarding various read/write error conditions
- ➔Close both files after the entire file is copied.

Operating system commands:- Apart from system calls user may interact with operating system directly be means of operating system commands.

For example :- If we want to list files or subdirectories in MS-DOS, we invoke "DIR" commands.

Types of Operating system:-

1. Batch Operating System:- In batch operating system, processing requires the program, data and appropriate system commands to be submit ted

together in the form of a job. Batch operating system usually allow little or no interaction between users and executing programs. Batch processing has a greater potential for resource utilization than simple serial processing in computer systems serving multiple users.

In batch operating jobs are typically processed in the order of submission that is in the first come first served basis.

Since there is only one program in the execution at a time, there is no competition for I?O devices, there for deallocation and de-allocation for I?O devices is very trivial.

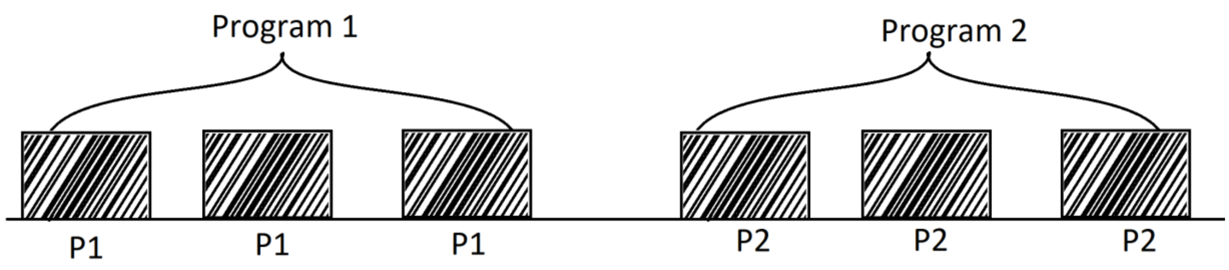
Batch operating system has two major disadvantages

i) Non-interactive environment → Batch operating system allow little or no interaction between users and executing programs. The turnaround time taken between job submission and job completion in batch operating system is very high. Users have no control over intermediate results of a program.

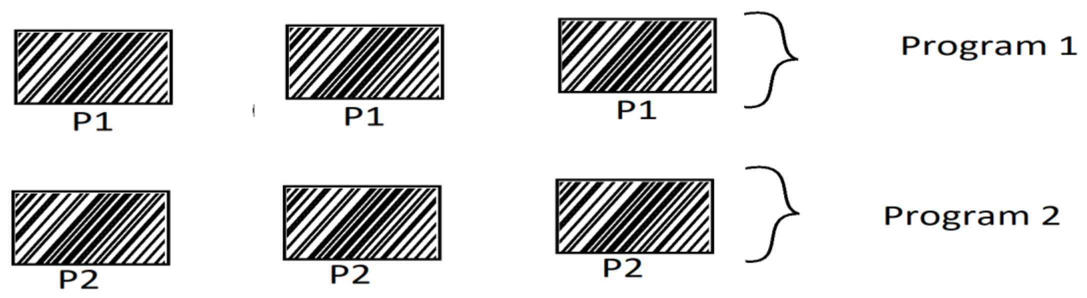
ii) Off-line debugging→ The second disadvantage with this approach is that programs must be debugged which means a programmer cannot correct bugs the moment it occur,

2. Multi programming operating system → The term multi programming denotes an operating system that, in addition to supporting multitasking, provides sophisticated forms of memory protection and enforces concurrency control when process access shared I/O devices and files. Multiprogramming operating system usually support multiple users in which case they are also called multiuser systems. In general multiprogramming implies multitasking, but multitasking doesn't imply multiprogramming. Different forms of multiprogramming operating system are multitasking, multi process and multiuser operating .

   Multiprogramming operating system compared to batch operating system are fairly sophisticated. Multiprogramming has a significant and resources utilization with very minor difference.



Sequential Execution



Multiprogramming Execution

Network Operating System: :→ A network operating system is a collection of software and associated protocols that allows a set of autonomous computers which are interconnected by a computer network to be used together in a convenient and cost effective manner. In a network operating systems the users are aware of existence of multiple computers and can log into remote machine and copy files from one machine to another machine.

Some characteristics of network operating system are:-

1. Each computer has its own private operating system instead of running part of a global system wide operating system.
2. Each user normally works on his/her own system. Use of different system requires some kind of remote login.
3. Users are typically aware of where each of their files are kept and must move files from one system to another with explicit file transfer commands instead of having file placement managed by the operating system.

Network operating system offers many capabilities including

- ➔Allowing users to access the various resources of the network hosts.
- ➔Controlling access so that only user in the proper authorization are allowed to access particular resources.
- ➔Making the use of remote resources appear to be identical to the use of local resources.

Distributed operating system:- A distributed operating system is one that looks to its user like an ordinary centralized operating system but runs on multiple independent CPU's. In other words the use of multiple processors should be invisible to the user. In distributed operating system users views the system as virtual uniprocessor but not as collection of distinct machines. In a true, distributed system users are not aware of where their programs are being run or where their files are residing, they should all be handled automatically and efficiently by the operating system.

Distributed operating system often allow programs to run on several processors at the same time, thus requiring more complex processor. Scheduling algorithm in order to achieve maximum utilization of CPU's time. Distributed operating systems ae considered to be more reliable than uniprocessor based system.

Advantages of Distributed Operating System:

1. Major breakthrough in microprocessor technology:- Micro-processors have become very powerful and cheap, compared with mainframes and minicomputers so it become attractive to think about designing large systems consisting of small processors. These distributed systems clearly have price/performance advantages over more traditional systems.
2. Incremental growth :- The second advantage is that if there is a need of 10 percent more computing power, one should just add 10 percent more processors.
3. Reliability :- Reliability and availability can also be a big advantage. A few parts of the system can be down without disturbing people using another parts.

Fil System in Distributed O/S :- Distributed operating system supports a single global file system visible from all machines.

Protection :- In a distributed operating system there is a unique UID for every user, and that UID should be valid on all machines.

Structure of Operating System:

1. **Layered Structure Approach:-** The operating system architecture based on layered approach consists of number of layers(levels), each built on to of lower layers. The bottom layer is the hardware, the highest layer is the user interface. The first system constructed in this way was "THE" built by E. W. Dijkestra (1968) and his students. THE system was system was a simple batch operating system.

5	**User Programs**
4	**Buffering for I/O devices**
3	**Device driver**
2	**Memory manager**

1	**CPU Scheduling**
0	**Hardware**

The layered structure of THE operating system

The main advantage of the layered approach is modularity which helps in debugging and verification of the system easily. The layers are designed in such a way that it uses operation and services only if a lower below it. A higher layer need not know how these operations are implemented only what these operation do. Hence each layer hides implementation details from higher level layers. Any layer can be debugged without any concern about the rest of the layer.

The major difficulty with the layered approach is definition of new level i.e. how to differentiate one level from another.

2. **Kernel Approach:-** Kernel is that part of operating system which directly makes interface with hardware system. Its main functions are:

➔To provide a mechanism for creation and deletion of processes.

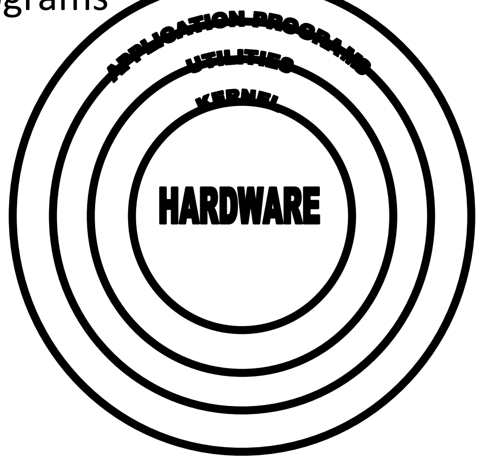
➔To provide processor scheduling, memory management and I/O management.

➔To provide mechanism for synchronization of processes so that processes synchronize their actions.

➔To provide mechanism for inter process communication.

The UNIX operating system is based on kernel approach. It consists of t wo seperatable parts

1. Kernel
2. System programs



As shown in above figure kernel is between system programs and hardware. The kernel supports the file system, processor scheduling, memory management and other operating system function through system calls.

Time sharing operating system:- Time sharing is a popular representative of multi programmed, multiuser systems. In addition to general program development environments, many large computer aided design (CAD) and text processing systems belongs to this category.

One of the primary objectives of multiuser systems in general and time-sharing in particular, is good terminal response time. Giving the illusion to each user of having a machine to oneself, time-sharing systems often attempt to provide equitable sharing of common resources.

Most time sharing systems use time-slicing (round-robin) scheduling. In this approach programs are executed with rotating priority that increases during and drops after the service is granted.

Memory management in time sharing systems provides for isolation and protection of co-resident programs.

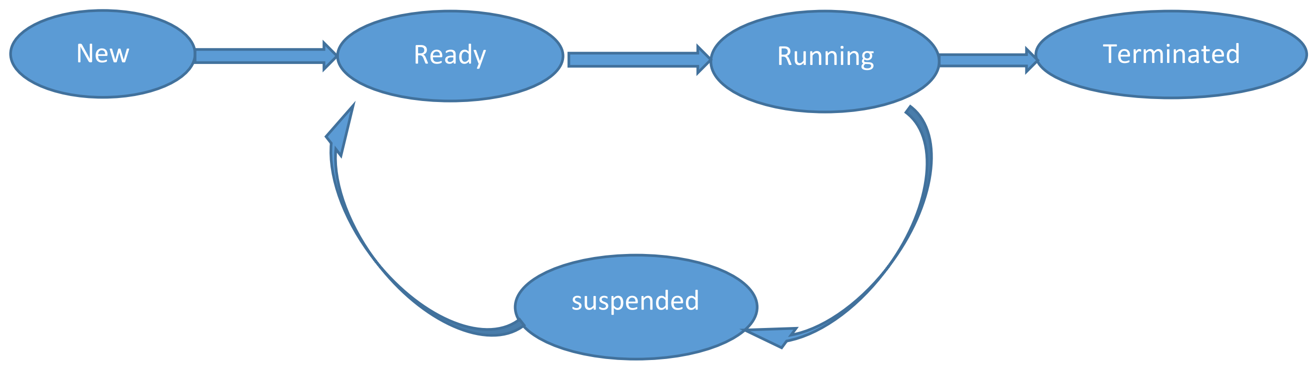
I/O management in time sharing systems must be sophisticated enough to cope with multiple user and divices.

UNIT – 2 [Process Management]

Process :- A process is an instance of a program in execution. It is basically a program while it is being executed. It is a running program with some specific task to do. It is the smallest unit of work individually schedulable by an operating system. Each multiprogramming OS keeps track of all active processes and allocated system resources to them according to policies devised to meet design performance objectives.

Process States:- The life time of a process can be divided into several stages as states, each with certain characteristics that describe the process. It means that as a process starts executing, it goes through one state to another state. Each process may be in one of the following states.

- ⇨ **New :-** process has been created
- ⇨ **Ready:-** The process is waiting to be allocated to a processor. Processor comes to this state immediately after it is created.
- ⇨ **Running:-** Instructions are being executed when a process gets control from CPU plus other resources, it starts executing. The running process may require some I/O during execution.
- ⇨ **Suspended:-** A suspended process lacks some resource other than the CPU, such processes are normally not considered for execution until the related suspending conditions is fulfilled. The running process become suspended by invoking I/O routines whose result it needs in order to proceed.
- ⇨ **Terminated:-** When the process finally stops. A process terminates when it finishes executing its last statement.



Process-State Transaction Diagram

Process Control Block[PCB]:- The operating system groups all information that it needs about a particular process into a data structure called Process Control Block [PCB]. It is also called process descriptor. It simply serves as the storage for any information for process. When a process is created, the operating system creates a corresponding PCB and when it terminates, its PCB is released to the pool of free memory locations from which new PCBs are drawn. A process is eligible to complete for system resources only when it has active PCB associated with it. A PCB is implemented as a record containing many pieces of information associated with a specific process including.

- Process Number:- Each process is identified by its process number, called process ID.
- Priority
- Process states:- Each process may be in any of these states: New, Ready, Running, Suspended and Terminated.
- Program Counter:- It indicates the address of the next instruction to be executed for this process.
- Register:- When a process switches over from one process to another process, information about current states of the old process is saved in the register along with the program counter so that the process be allowed to continue correctly afterwards.
- Accounting information:- I includes actual CPU time used in executing a process.
- I/O status:- It includes outstanding I/O requests list of open files, information about allocation of peripheral devices to processes.
- Processor scheduling details:- It includes a priority of a process, address to scheduling queues and any other.

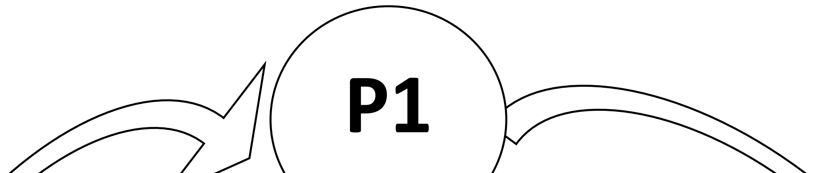
Mutual Exclusion:- Process that are working together often share some common storage that one can read and write. The shared storage may be in main memory or it may be a shared file. Each process has segment of code, called a critical section, which access shred memory or files. The key issue involving shared memory or shared files is to find way to prohibit more than one process from reading and writing the shared data at the same time. What we need is mutual exclusion. Some way of making sure that if one process is executing in its critical section, the other process will be excluded from doing the same thing.

Critical Section:- The critical section is a sequence of instructions with a clearly marked beginning and end. It usually safeguards updating of one or more shared variable. When a process enters a critical section, it must complete all instructions therein before any other process is allowed to enter the same critical section. Only the process executing the critical section is allowed access to the shared variable, all other process should be prevented from doing so until the completion of the critical section.

Semaphores:- The Dutch mathematician Dekker is believed to be the first to solve the mutual exclusion problem. But its original algorithm works for two processes only and it can be extended beyond that number. To over come this problem a synchronization tool called semaphore was proposed by Dijkestra which gained wide acceptance and implemented in several commercial operating system through system calls and built-in-functions. A semaphore is a variable which accepts non-negative integer values and except for initialization may be accessed and manipulated through two primitive operations WAIT and SIGNAL (originally defined as 'P' and 'V' respectively). These names come from the Dutch words Problem(to test) and Verogen (to increment). The two primitives take only argument as the semaphore variable, and may be difined as follows:

a) WAIT(S):
   While s<=0 do {keep testing} s:=S-1;
   Wait operation decrements the value of semaphore variable as soon as it would become non-negative.
b) SIGNAL(S) S:=S+1
   Signal operation increments the value of semaphore variable.

Deadlock:- A deadlock is a situation where a group of processes is permanently blocked as a result of each process having acquired a set of resources needed for its completion and having to wait for release of the remaining resources held by others thus making it impossible for any of the deadlocked processes to proceed. For example, take a system with one tape drive and one plotter. Process P1 requests tape drive and P2 requests the plotter. Both requests are granted. Now P1 requests the plotter (without giving up the tape drive) and P2 request the tape drive (without giving up the plotter). Neither request can be granted so both process enter a deadlock situation.



Deadlock Situation

Necessary Condition for Deadlock: