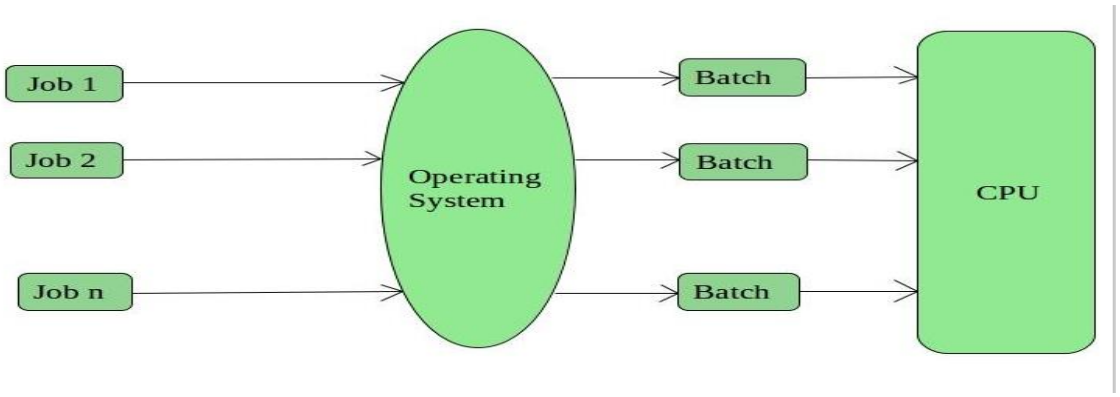
Types of Operating Systems

An Operating System performs all the basic tasks like managing files, processes, and memory. Thus operating system acts as the manager of all the resources, i.e. **resource manager**. Thus, the operating system becomes an interface between user and machine.

Types of Operating Systems: Some widely used operating systems are as follows-

1. **Batch Operating System**

   This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having the same requirement and group them into batches. It is the responsibility of the operator to sort jobs with similar needs.



Advantages of Batch Operating System:

- It is very difficult to guess or know the time required for any job to complete. Processors of the batch systems know how long the job would be when it is in queue
- Multiple users can share the batch systems
- The idle time for the batch system is very less
- It is easy to manage large work repeatedly in batch systems

Disadvantages of Batch Operating System:

- The computer operators should be well known with batch systems
- Batch systems are hard to debug
- It is sometimes costly
- The other jobs will have to wait for an unknown time if any job fails

Examples of Batch based Operating System: Payroll System, Bank Statements, etc.

2. **Time-Sharing Operating Systems**

Each task is given some time to execute so that all the tasks work smoothly. Each user gets the time of CPU as they use a single system. These systems are also known as Multitasking Systems. The task can be from a single user or different users also. The time that each task gets to execute is called quantum. After this time interval is over OS switches over to the next task.

Advantages of Time-Sharing OS:

- Each task gets an equal opportunity
- Fewer chances of duplication of software
- CPU idle time can be reduced

Disadvantages of Time-Sharing OS:

- Reliability problem
- One must have to take care of the security and integrity of user programs and data
- Data communication problem

Examples of Time-Sharing OSs are: Multics, Unix, etc.

3. **Distributed Operating System**

These types of the operating system is a recent advancement in the world of computer technology and are being widely accepted all over the world and, that

too, with a great pace. Various autonomous interconnected computers communicate with each other using a shared communication network. Independent systems possess their own memory unit and CPU. These are referred to as **loosely coupled systems** or distributed systems. These system's processors differ in size and function. The major benefit of working with these types of the operating system is that it is always possible that one user can access the files or software which are not actually present on his system but some other system connected within this network i.e., remote access is enabled within the devices connected in that-network.

Advantages of Distributed Operating System:

- Failure of one will not affect the other network communication, as all systems are independent from each other
- Electronic mail increases the data exchange speed
- Since resources are being shared, computation is highly fast and durable
- Load on host computer reduces
- These systems are easily scalable as many systems can be easily added to the network
- Delay in data processing reduces

Disadvantages of Distributed Operating System:

- Failure of the main network will stop the entire communication
- To establish distributed systems the language which is used are not well defined yet
- These types of systems are not readily available as they are very expensive. Not only that the underlying software is highly complex and not understood well yet

Examples of Distributed Operating System are- LOCUS, etc.

5. Real-Time Operating System

These types of OSs serve real-time systems. The time interval required to process and respond to inputs is very small. This time interval is called **response time**. **Real-time systems** are used when there are time requirements that are very strict like missile systems, air traffic control systems, robots, etc.

Two types of Real-Time Operating System which are as follows:

- **Hard Real-Time Systems:**
  These OSs are meant for applications where time constraints are very strict and even the shortest possible delay is not acceptable. These systems are built for saving life like automatic parachutes or airbags which are required to be readily available in case of any accident. Virtual memory is rarely found in these systems.
- **Soft Real-Time Systems:**
  These OSs are for applications where for time-constraint is less strict.

Advantages of RTOS:

- **Maximum Consumption:** Maximum utilization of devices and system, thus more output from all the resources
- **Task Shifting:** The time assigned for shifting tasks in these systems are very less. For example, in older systems, it takes about 10 microseconds in shifting one task to another, and in the latest systems, it takes 3 microseconds.
- **Focus on Application:** Focus on running applications and less importance to applications which are in the queue.
- Real-time **operating system in** the **embedded system:** Since the size of programs are small, RTOS can also be used in embedded systems like in transport and others.
- **Error Free:** These types of systems are error-free.

- **Memory Allocation:** Memory allocation is best managed in these types of systems.

Disadvantages of RTOS:

- **Limited Tasks:** Very few tasks run at the same time and their concentration is very less on few applications to avoid errors.
- **Use heavy system resources:** Sometimes the system resources are not so good and they are expensive as well.
- **Complex Algorithms:** The algorithms are very complex and difficult for the designer to write on.
- **Device driver and interrupt signals:** It needs specific device drivers and interrupts signals to respond earliest to interrupts.
- **Thread Priority:** It is not good to set thread priority as these systems are very less prone to switching tasks.

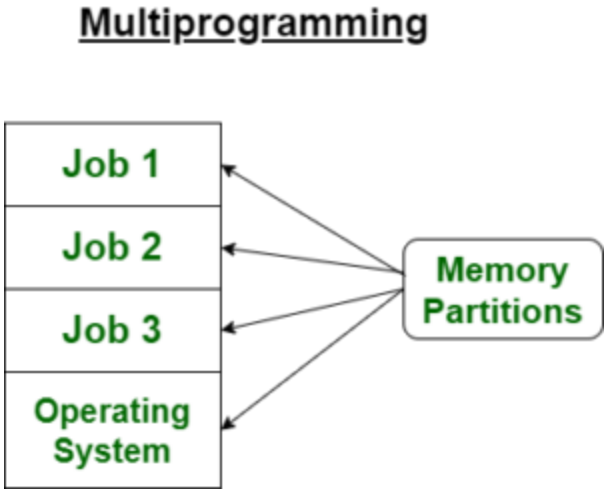
Examples of Real-Time Operating Systems are: Scientific experiments, medical imaging systems, industrial control systems, weapon systems, robots, air traffic control systems, etc.

Multi-programming and Multi-tasking

CPU is a super fast device and keeping it occupied for a single task is never a good idea. Considering the huge differences between CPU speed and IO speed, many concepts like multiprogramming, multitasking, multithreading, etc have been introduced to make better CPU utilisation.

Multi programming:-

Multi-programming increases CPU utilisation by organising jobs (code and data) so that the CPU always has one to execute. The idea is to keep multiple jobs in main memory. If one job gets occupied with IO, CPU can be assigned to other job.

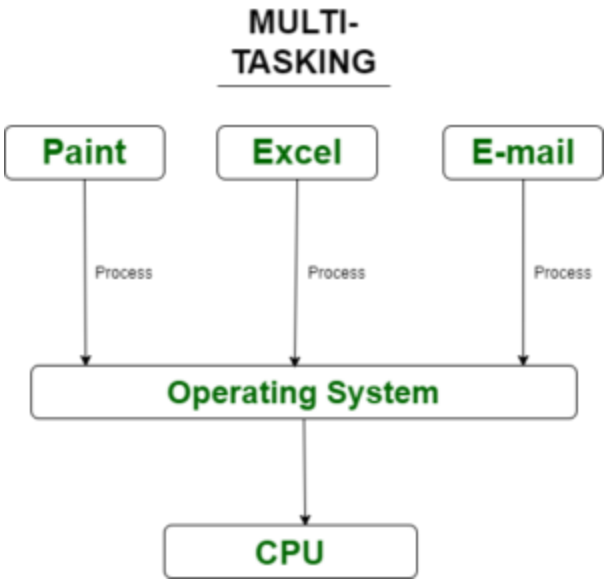


Multi-tasking:-

Multi-tasking is a logical extension of multiprogramming. Multitasking is the ability of an OS to execute more than one **task** simultaneously on *a CPU machine.* These multiple tasks share common resources (like CPU and memory). In multi-tasking systems, the CPU executes multiple jobs by switching among them typically using a small time quantum, and the switches occur so quickly that the users feel like interact with each executing task at the same time.

Note: 1. A task in a multitasking system is not whole application program but it can refers to a "thread of execution" when one process is divided into sub-tasks.

2. Multi programming and multitasking OS are time sharing systems.



Difference between Multiprogramming and Multi-tasking

Sr.no	Multiprogramming	Multi-tasking
1.	Both of these concepts are for single CPU.	Both of these concepts are for single CPU.
2.	Concept of Context Switching is used.	Concept of Context Switching and Time Sharing is used.
3.	In multiprogrammed system, the operating system simply switches to, and executes, another job when current job needs to wait.	The processor is typically used in time sharing mode. Switching happens when either allowed time expires or where there other reason for current process needs to wait (example process needs to do IO).
4.	Multi-programming increases CPU utilization by organising jobs .	In multi-tasking also increases CPU utilization, it also increases responsiveness.
5.	The idea is to reduce the CPU idle time for as long as possible.	The idea is to further extend the CPU Utilization concept by increasing responsiveness Time Sharing.