An **Operating System** can be defined as an **interface between user and hardware**. It is responsible for the execution of all the processes, Resource Allocation, [CPU](https://www.javatpoint.com/cpu-full-form) management, File Management and many other tasks.

The purpose of an operating **system is to provide an environment in which a user can execute programs in convenient and efficient manner.**

What does an Operating system do?

1. Process Management
2. Process Synchronization
3. Memory Management
4. CPU Scheduling
5. File Management
6. Security

## Batch Operating System

In the 1970s, Batch processing was very popular. In this technique, similar types of jobs were batched together and executed in time. People were used to having a single computer which was called a mainframe.

In Batch operating system, access is given to more than one person; they submit their respective jobs to the system for the execution.

The purpose of this operating system was mainly to transfer control from one job to another as soon as the job was completed. It contained a small set of programs called the resident monitor that always resided in one part of the main memory. The remaining part is used for servicing jobs.

### **Advantages of Batch OS**

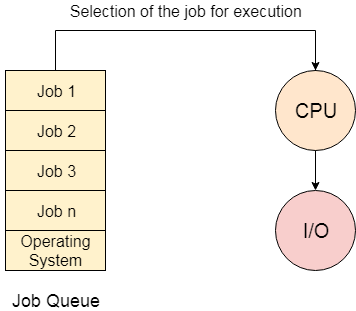
* The use of a resident monitor improves computer efficiency as it eliminates CPU time between two jobs.

### **Disadvantages of Batch OS**

**1. Starvation**

Batch processing suffers from starvation.

**For Example:**



There are five jobs J1, J2, J3, J4, and J5, present in the batch. If the execution time of J1 is very high, then the other four jobs will never be executed, or they will have to wait for a very long time. Hence the other processes get starved.

**2. Not Interactive**

Batch Processing is not suitable for jobs that are dependent on the user's input. If a job requires the input of two numbers from the console, then it will never get it in the batch processing scenario since the user is not present at the time of execution.

## Multiprogramming Operating System

Multiprogramming **is an extension to batch processing where the CPU is always kept busy**. Each process needs two types of system time: CPU time and IO time.

In **a multiprogramming environment, when a process does its I/O, The CPU can start the execution of other processes. Therefore, multiprogramming improves the efficiency of the system.**

### **Advantages of Multiprogramming OS**

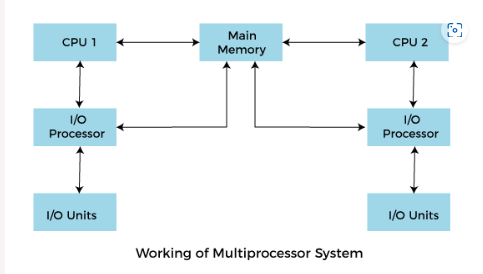
* Throughout the system, it increased as the CPU always had one program to execute.
* Response time can also be reduced.

### **Disadvantages of Multiprogramming OS**

* Multiprogramming systems provide an environment in which various systems resources are used efficiently, but they do not provide any user interaction with the computer system

## Multiprocessing Operating System

In Multiprocessing, Parallel computing is achieved. **There are more than one processors present in the system which can execute more than one process at the same time. This will increase the throughput of the system.**

****

Advantages of Multiprocessing operating system:

* **Increased reliability:** Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails, the task can be given to another processor for completion.
* **Increased throughout:** As several processors increase, more work can be done in less time.

Disadvantages of Multiprocessing operating System

* Multiprocessing operating system is more complex and sophisticated as it takes care of multiple CPUs simultaneously.

### **Multitasking Operating System**

The multitasking operating system **is a logical extension of a multiprogramming system** **that enables** **multiple** **programs simultaneously.** It allows a user to perform more than one computer task at the same time.

### **Advantages of Multitasking operating system**

* This operating system is more suited to supporting multiple users simultaneously.
* The multitasking operating systems have **well-defined memory management.**

### **Disadvantages of Multitasking operating system**

* The multiple processors are busier at the same time to complete any task in a multitasking environment, so the CPU generates more heat.

## Real Time Operating System

In Real-Time Systems, **each job carries a certain deadline within which the job is supposed to be completed**, otherwise, the **huge loss will be there, or even if the result is produced, it will be completely useless.**

### **Advantages of Real-time operating system:**

* Easy to layout, develop and execute real-time applications under the real-time operating system.
* In a Real-time operating system, the maximum utilization of devices and systems.

### **Disadvantages of Real-time operating system:**

* Real-time operating systems are very costly to develop.
* Real-time operating systems are very complex and can consume critical CPU cycles.

## Time-Sharing Operating System

In the Time Sharing operating system, computer resources are allocated in a time-dependent fashion to several programs simultaneously. Thus it helps to provide a large number of user's direct access to the main computer.

**A time-sharing operating system allows many users to be served simultaneously**, so **sophisticated CPU scheduling schemes and Input/output management are required.**

Time-sharing operating systems are very difficult and expensive to build.

# **What is Kernel?**

In computer science, **Kernel is a computer program that is a core or heart of an operating system.**

* it has full control over everything in the system. **Each operation of hardware and software is managed and administrated by the kernel.**
* It **acts as a bridge between applications and data processing done at the hardware level**. It is the central component of an OS.
* It is the part of the OS **that always resides in computer memory and enables the communication between software and hardware components.**

Kernel mode, also known as supervisor mode or privileged mode, is a hardware-assisted execution mode in a computer's central processing unit (CPU) that provides the operating system with unrestricted access to the hardware resources and system-wide control over the computer's functions. In this mode, the operating system's kernel (the core part of the operating system) executes with full control over the hardware and system operations.

## Functions of a Kernel

A kernel of an OS is responsible for performing various functions and has control over the system.

**Device Management**

**Memory Management**

**Resource Management**

**Accessing Computer Resources**

### **1. Monolithic Kernels**

In a monolithic kernel, the **same memory space is used to implement user services and kernel services.**

It means, in this type of kernel**, there is no different memory used for user services and kernel services.**

**Same memory use for user service and kernel services.**

As it uses the same memory space, the size of the kernel increases, increasing the overall size of the OS.

**Advantages:**

* **The execution of processes is also faster as there is no separate user space and kernel space and less software involved**.
* As it is **a single piece of software hence, it's both sources and compiled forms are smaller.**

**Disadvantages:**

* If **any service generates any error, it may crash down the whole system**.
* **These kernels are not portable, which means for each new architecture, they must be rewritten.**
* **Large in size and hence become difficult to manage**.
* **To add a new service, the complete operating system needs to be modified**.

### **2. Microkernel**

A microkernel is also referred to as **μK**, and it is different from a traditional kernel or Monolithic Kernel. In this, **user services and kernel services are implemented into two different address spaces: user space and kernel space**. Since it uses different spaces for both the services, so, the size of the microkernel is decreased, and which also reduces the size of the OS.

Microkernels are **easier to manage and maintain as compared to monolithic kernels**. Still, if there will be a greater number of system calls and context switching, then it might reduce the performance of the system by making it slow.

**Advantages**

* **Microkernels can be managed easily.**
* A **new service can be easily added without modifying the whole OS.**
* In a microkernel**, if a kernel process crashes, it is still possible to prevent the whole system from crashing.**

**Disadvantages**

### **3. Nanokernel**

As the name suggests, **in Nanokernel, the complete code of the kernel is very small, which means the code executing in the privileged mode of the hardware is very small**. Here the term nano defines a kernel that supports a nanosecond clock resolution.

Examples of Nanokernel **are EROS etc.**

**Advantages**

* It ***provides hardware abstractions even with a very small size.***

**Disadvantages**

* Nanokernel lacks system services.

### **4. Exokernel**

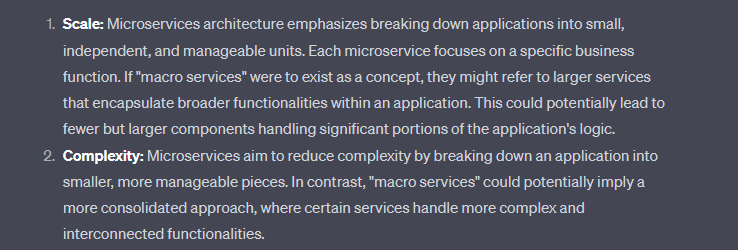
This type of kernel is different from other kernels as in this; resource protection is kept separated from management, which allows us to perform application-specific customization.

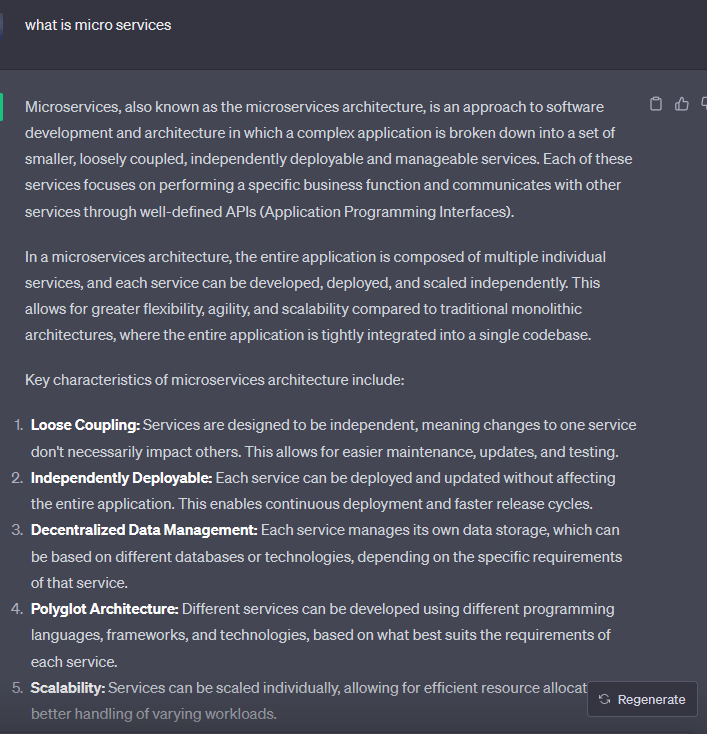
Rather than cloning the actual machine it is done with virtual machine

Its give every user a sub set of resources.

It is program that run at bottom level running in kernel mode.

Micro Services.





### **1. What is a process and process table?**

A process is an instance of a program in execution. For example, a Web Browser is a process, and a shell (or command prompt) is a process. The operating system is responsible for managing all the processes that are running on a computer and allocates each process a certain amount of time to use the processor. In addition, the operating system also allocates various other resources that processes will need, such as computer memory or disks. **To keep track of the state of all the processes**, the operating system maintains a table known as the process table. **Inside this table, every process is listed along with the resources the process is using and the current state of the process.**

### **2. What are the different states of the process?**

Processes can be in one of **three states: running, ready, or waiting**. The *running state means that the process has all the resources it needs for execution and it has been given permission by the operating system to use the processor. Only one process can be in the running state at any given time*. The *remaining processes are either in a waiting state* (i.e., waiting for some external event to occur such as user input or disk access) or a ready state (i.e., waiting for permission to use the processor or no CPU available at that time). In a real operating system, the waiting and ready states are implemented as queues that hold the processes in these states.

### **What is Semaphore?**

Semaphore is simply a variable that is non-negative and shared between threads. A semaphore is a signaling mechanism, and another thread can signal a thread that is waiting on a semaphore.

It is used to stop multiple process to enter into critical region.

### **Types of Semaphore**

Semaphore is distinguished by the operating system in two categories **Counting semaphore** and **Binary semaphore**.

**1. Counting Semaphore:** The semaphore S value is initialized to the **number of resources** present in the system. Whenever a process wants to access the resource, it performs **the wait()**operation on the semaphore and **decrements** the semaphore value by one. When it releases the resource, it performs **the signal()** operation on the semaphore and **increments** the semaphore value by one.

When the semaphore count goes to 0, it means the processes occupy all resources. A process needs to use a resource when the semaphore count is 0. It executes the **wait()** operation and gets **blocked** until the semaphore value becomes greater than 0.

**2. Binary semaphore:** The value of a semaphore ranges between **0**and **1**. It is similar to mutex lock, but mutex is a locking mechanism, whereas the semaphore is a signaling mechanism. In binary semaphore, if a process wants to access the resource, it performs **the wait()** operation on the semaphore and decrements the value of the semaphore from 1 to 0. When it releases the resource, it performs a **signal()** operation on the semaphore and increments its value to 1. Suppose the value of the semaphore is 0 and a process wants to access the resource. In that case, it performs **wait()** operation and block itself till the current process utilizing the resources releases the resource.