1. Define Operating System and explain the various types of Operating Systems?

An operating system is a set of programs that enables a user to operate and interact with a computer. Examples of operating systems are Linux distributions, windows, mac os, FreeBSD, etc.

### Batch Operating Systems

A batch operating system grabs all programs and data in the batch form and then processes them. The main aim of using a batch processing system is to decrease the setup time while submitting similar jobs to the CPU. Batch processing techniques were implemented in the hard disk and card readers as well. In this case, all jobs are saved on the hard disk for making the pool of jobs for their execution as a batch form.

### Time-sharing Operating Systems

Time-sharing is a logical extension of multiprogramming. The CPU executes multiple jobs by switching, among them, but the switches occur so frequently that the users can interact with each program while it is running. An interactive computer provides direct communication between the user and the system. The user gives instructions to the OS or a program directly, using hardware, and waits for results.

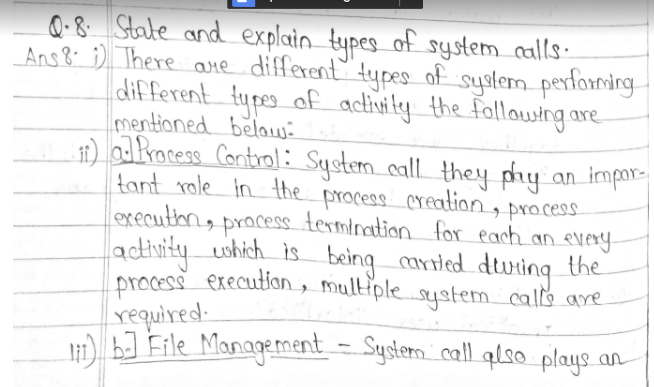
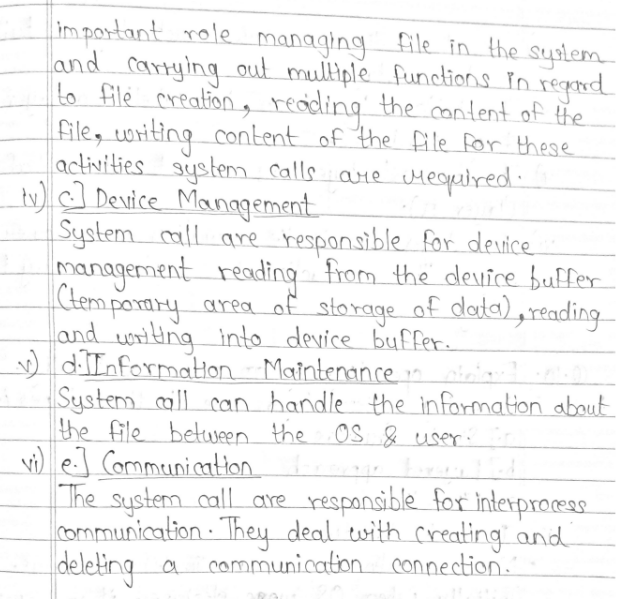
### Desktop Operating System

The control program which operates in the machine of a user is referred to as a desktop system. It is also called a Client Operating System. The client can be said as a computer in a network where the user performs some task or activity over the network. Such operating systems do not have complete control over the resources but use the network to access them.

Network Operating System is an operating system that has special functions for connecting computers and devices into a local-area network or Inter-network. Some popular network operating systems are Windows NT/2000, Novell Netware, Linux, UNIX, Sun Solaris, and IBM OS/2. The network operating system which was first developed is Novell Netware, in 1983.

Etc..

2 Explain System Calls with respect to following: definition, types and execution.

3.What is file management? Write activities of operating system in regard to file management.

Repeated

4.What are advantages and disadvantages of threads?

Repeated

5.Discuss Layered Approach in comparison with Kernel based Approach.

Repeated

6.Explain the term cooperating processes. What are the advantages of the same?

Cooperating processes are those that can affect or are affected by other processes running on the system. Cooperating processes may share data with each other.

**Modularity**

**Information Sharing**

**Convenience**

**Computation Speedup**

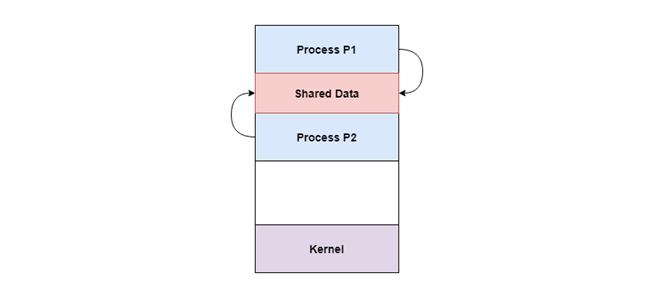
## **Methods of Cooperation**

Cooperating processes can coordinate with each other using shared data or messages. Details about these are given as follows −

* **Cooperation by Sharing**

The cooperating processes can cooperate with each other using shared data such as memory, variables, files, databases etc. Critical section is used to provide data integrity and writing is mutually exclusive to prevent inconsistent data.

A diagram that demonstrates cooperation by sharing is given as follows −

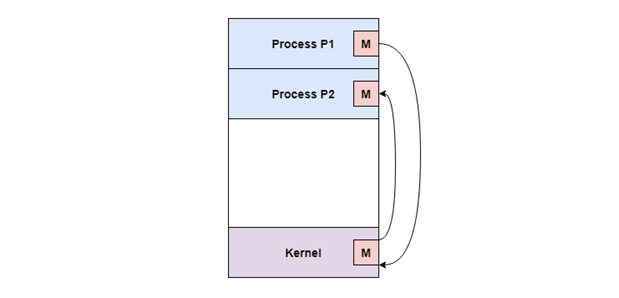


In the above diagram, Process P1 and P2 can cooperate with each other using shared data such as memory, variables, files, databases etc.

* **Cooperation by Communication**

The cooperating processes can cooperate with each other using messages. This may lead to deadlock if each process is waiting for a message from the other to perform a operation. Starvation is also possible if a process never receives a message.

A diagram that demonstrates cooperation by communication is given as follows −



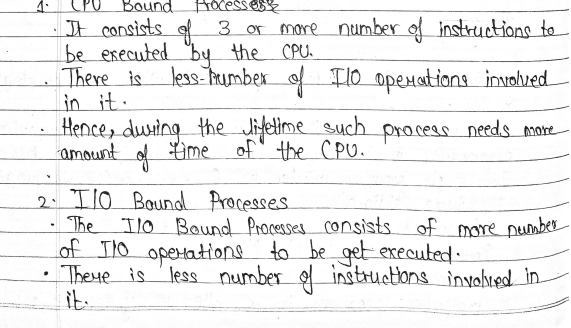
7.What are the advantages of peer-to-peer systems over client-server systems?

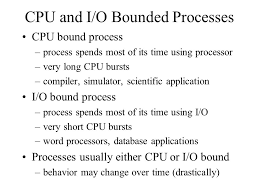
### **Advantages of Peer-to-Peer Network**

The following are the advantages of peer-to-peer networks:

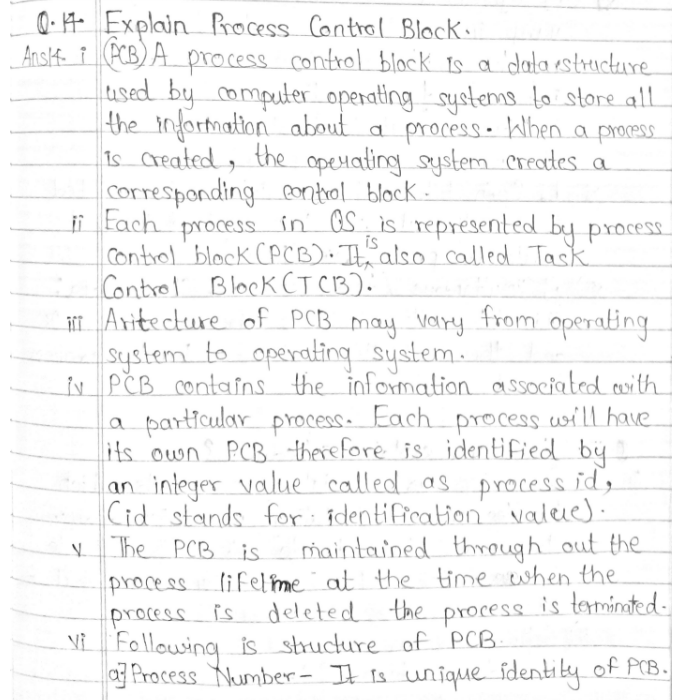
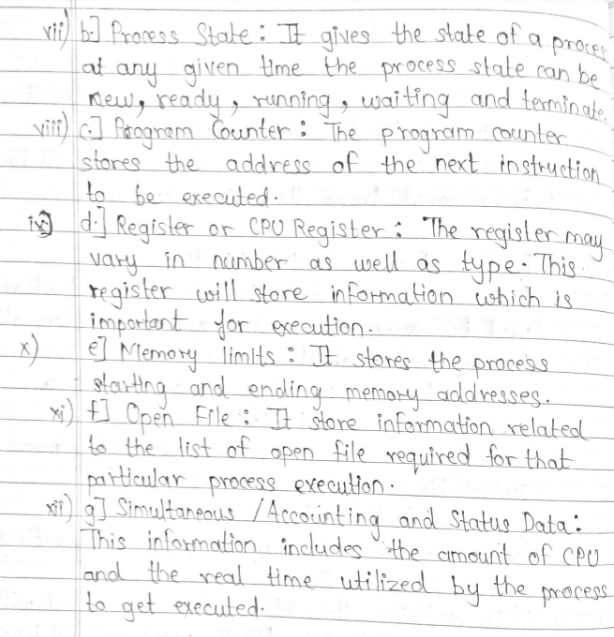
* Each device linked to the peer-to-peer network exchanges resources with other network nodes.
* The setup of a peer-to-peer network is easily established with the help of specialized software.
* Between several devices, resources are exchanged without any issues.
* Peer-to-peer networks are very reliable because other systems continue to function even when a server fails.
* Being a part of a peer-to-peer network makes it simple for nodes to share resources like a

8. Distinguish between CPU bounded, I/O bounded processes





9.What is a process? Explain Process Control Block.

10.State and explain various multithreading models.

Multithreading allows the execution of multiple parts of a program at the same time. These parts are known as threads and are lightweight processes available within the process. Therefore, multithreading leads to maximum utilization of the CPU by multitasking.

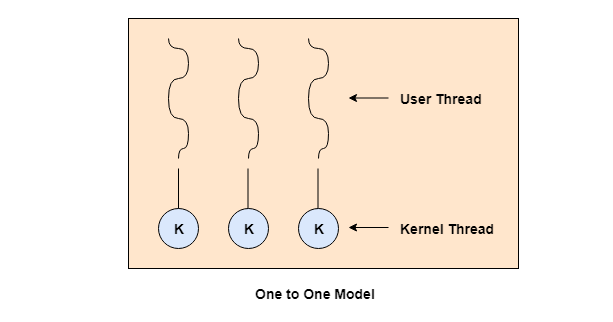
The main models for multithreading are one to one model, many to one model and many to many model. Details about these are given as follows −

## **One to One Model**

The one to one model maps each of the user threads to a kernel thread. This means that many threads can run in parallel on multiprocessors and other threads can run when one thread makes a blocking system call.

A disadvantage of the one to one model is that the creation of a user thread requires a corresponding kernel thread. Since a lot of kernel threads burden the system, there is restriction on the number of threads in the system.

A diagram that demonstrates the one to one model is given as follows −

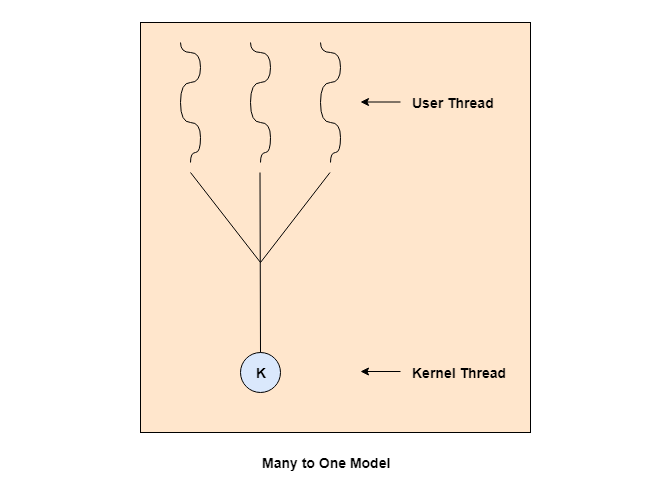


## **Many to One Model**

The many to one model maps many of the user threads to a single kernel thread. This model is quite efficient as the user space manages the thread management.

A disadvantage of the many to one model is that a thread blocking system call blocks the entire process. Also, multiple threads cannot run in parallel as only one thread can access the kernel at a time.

A diagram that demonstrates the many to one model is given as follows −



## **Many to Many Model**

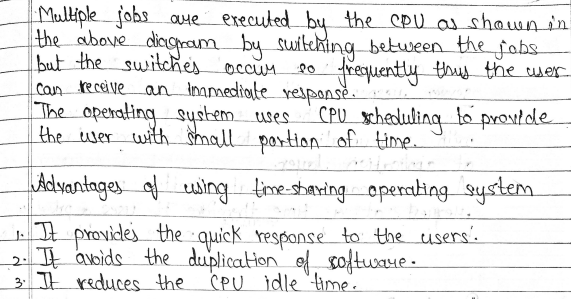
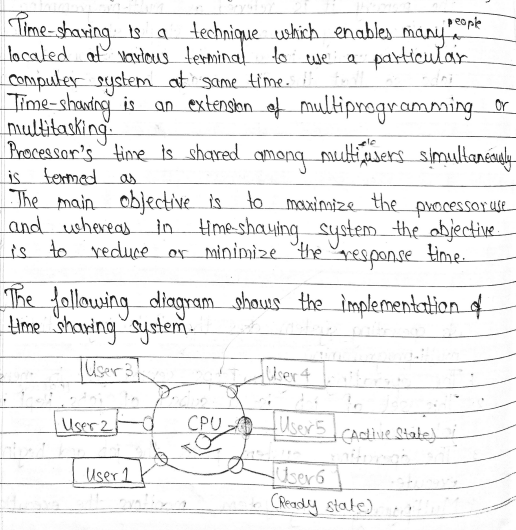
The many to many model maps many of the user threads to a equal number or lesser kernel threads. The number of kernel threads depends on the application or machine.

The many to many does not have the disadvantages of the one to one model or the many to one model. There can be as many user threads as required and their corresponding kernel threads can run in parallel on a multiprocessor.

A diagram that demonstrates the many to many model is given as follows −



11. Write a note on: Time sharing operating system.



OR

Time-sharing enables many people, located at various terminals, to use a particular computer system at the same time. Multitasking or Time-Sharing Systems is a logical extension of multiprogramming. Processor’s time is shared among multiple users simultaneously is termed as time-sharing.

An operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time. Computer systems which were designed primarily as batch systems have been modified to time-sharing systems.

Advantages of Timesharing operating systems are −

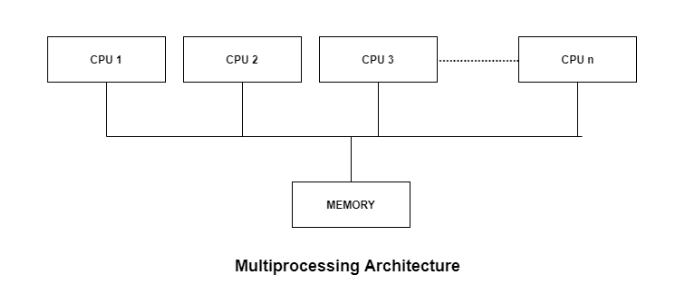
* It provides the advantage of quick response.
* This type of operating system avoids duplication of software.
* It reduces CPU idle time.

Disadvantages of Time-sharing operating systems are −

* Time sharing has problem of reliability.
* Question of security and integrity of user programs and data can be raised.
* Problem of data communication occurs.

12. Define single and multiprocessor systems. Write the advantages of multiprocessor systems?

These systems have multiple processors working in parallel that share the computer clock, memory, bus, peripheral devices etc. An image demonstrating the multiprocessor architecture is



## Advantages of Multiprocessor Systems

There are multiple advantages to multiprocessor systems. Some of these are −

**More reliable Systems**

In a multiprocessor system, even if one processor fails, the system will not halt. This ability to continue working despite hardware failure is known as graceful degradation. For example: If there are 5 processors in a multiprocessor system and one of them fails, then also 4 processors are still working. So the system only becomes slower and does not ground to a halt.

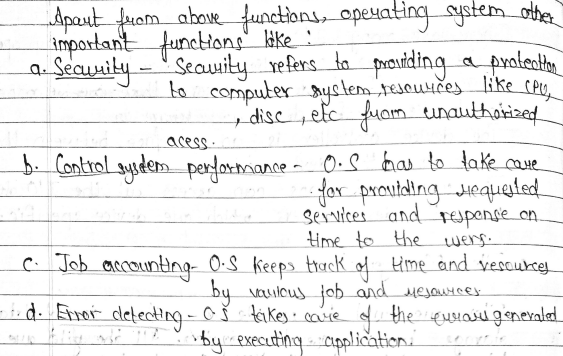
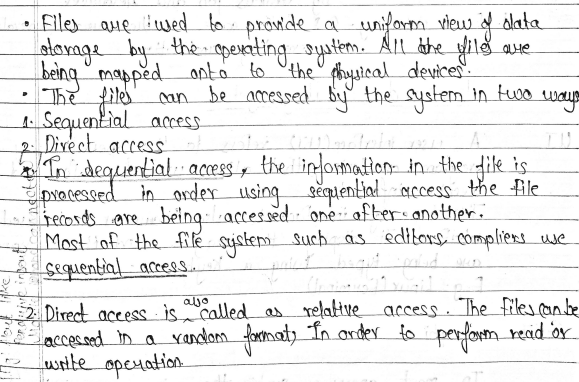
**Enhanced Throughput**

If multiple processors are working in tandem, then the throughput of the system increases i.e. number of processes getting executed per unit of time increase. If there are N processors then the throughput increases by an amount just under N.

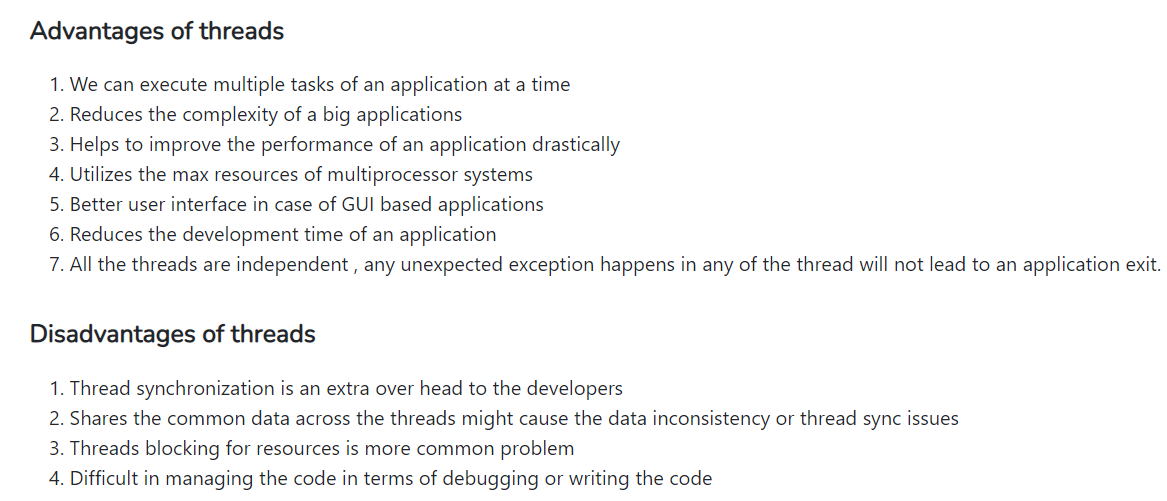
**More Economic Systems**

Multiprocessor systems are cheaper than single processor systems in the long run because they share the data storage, peripheral devices, power supplies etc. If there are multiple processes that share data, it is better to schedule them on multiprocessor systems with shared data than have different computer systems with multiple copies of the data.

13. What is file management? Write activities of operating system in regard to file management.



14. What are advantages and disadvantages of threads?



15. Discuss Layered Approach in comparison with Kernel based Approach.

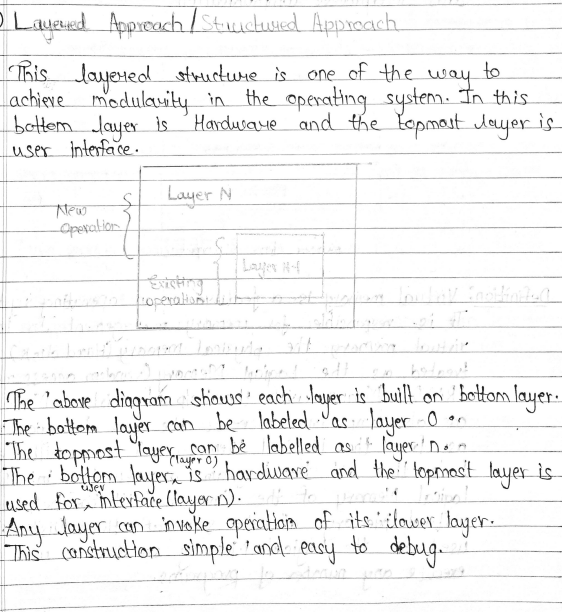
Layered Structure is a type of system structure in which the different services of the [operating system](https://www.geeksforgeeks.org/operating-systems/) are split into various layers, where each layer has a specific well-defined task to perform.

**Example –** The Windows NT operating system uses this layered approach as a part of it.

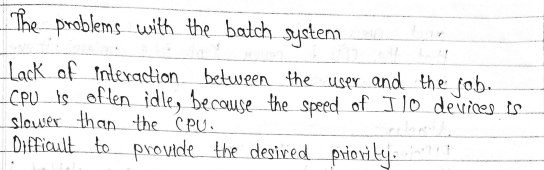
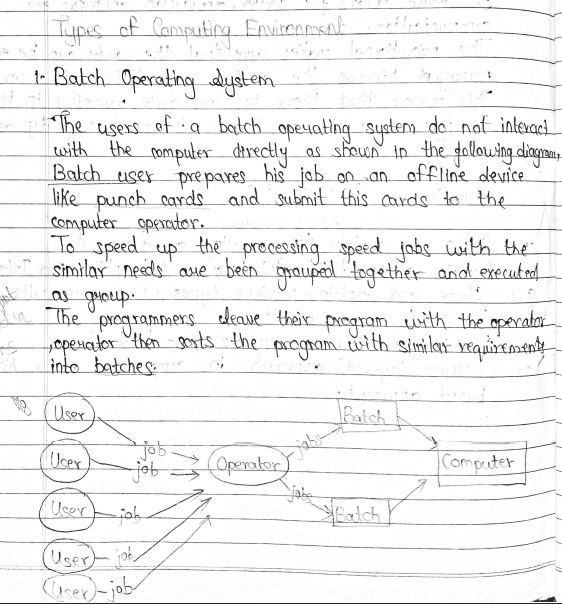
**Design Analysis :**  
The whole Operating System is separated into several layers ( from 0 to n ) as the diagram shows. Each of the layers must have its own specific function to perform. There are some rules in the implementation of the layers as follows.

1. The outermost layer must be the User Interface layer.
2. The innermost layer must be the Hardware layer.
3. A particular layer can access all the layers present below it but it cannot access the layers present above it. That is layer n-1 can access all the layers from n-2 to 0 but it cannot access the nth layer.

OR



16. What are Batch systems?

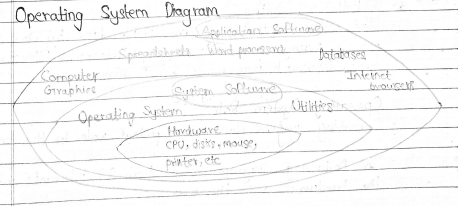


17. What is an Operating system

An operating system (OS) is **the program that, after being initially loaded into the computer by a boot program, manages all of the other application programs in a computer**. The application programs make use of the operating system by making requests for services through a defined application program interface (API).

An **Operating System** (OS) is an interface between a computer user and computer hardware. An operating system is a software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

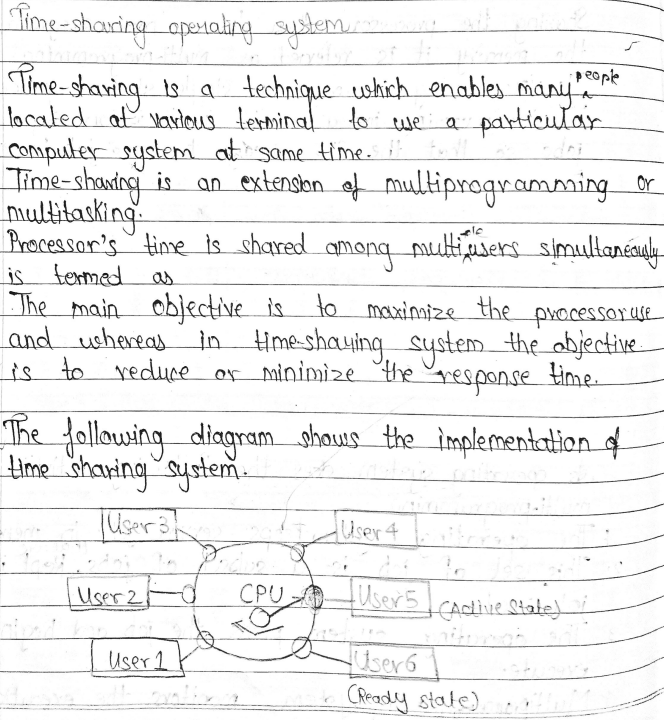
An operating system is software that enables applications to interact with a computer's hardware. The software that contains the core components of the operating system is called the **kernel**.

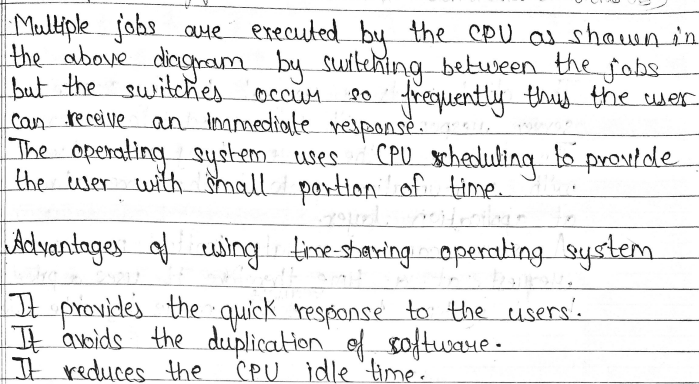


The primary purposes of an **Operating System** are to enable applications (software) to interact with a computer's hardware and to manage a system's hardware and software resources.

Some popular Operating Systems include Linux Operating System, Windows Operating System, VMS, OS/400, AIX, z/OS, etc. Today, Operating systems is found almost in every device like mobile phones, personal computers, mainframe computers, automobiles, TV, Toys etc.

18. What is meant by Time-sharing Systems





20. Write a short note on: Process Scheduling.

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

Process scheduling is an essential part of a Multiprogramming operating systems. Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

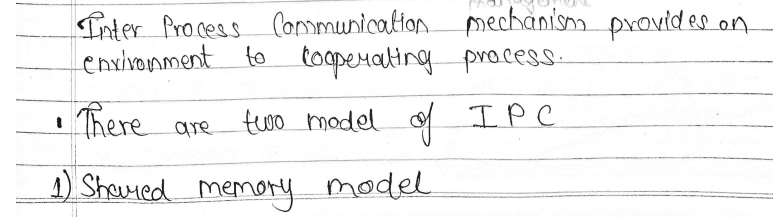
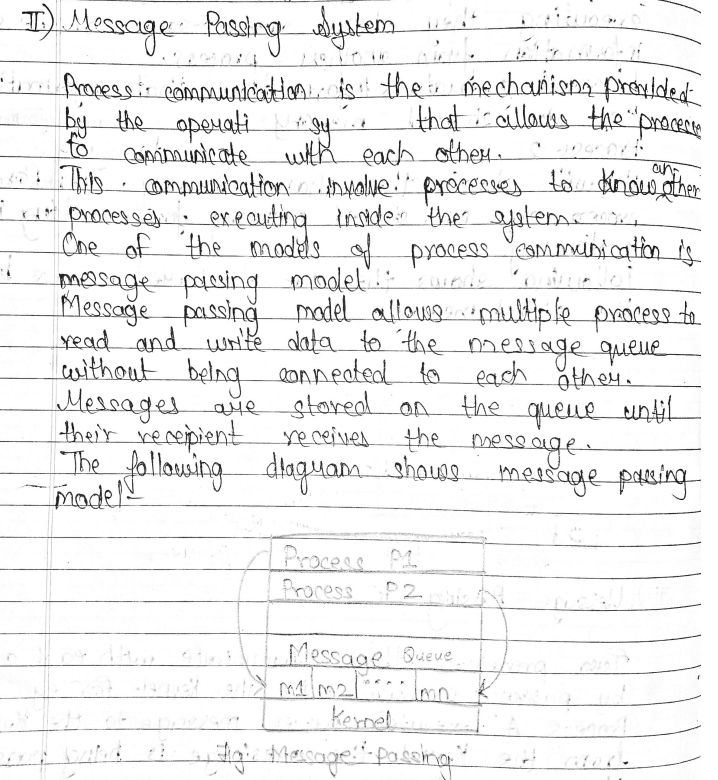
## **Categories of Scheduling**

There are two categories of scheduling:

1. **Non-preemptive:** Here the resource can’t be taken from a process until the process completes execution. The switching of resources occurs when the running process terminates and moves to a waiting state.
2. **Preemptive:** Here the OS allocates the resources to a process for a fixed amount of time. During resource allocation, the process switches from running state to ready state or from waiting state to ready state. This switching occurs as the CPU may give priority to other processes and replace the process with higher priority with the running process.

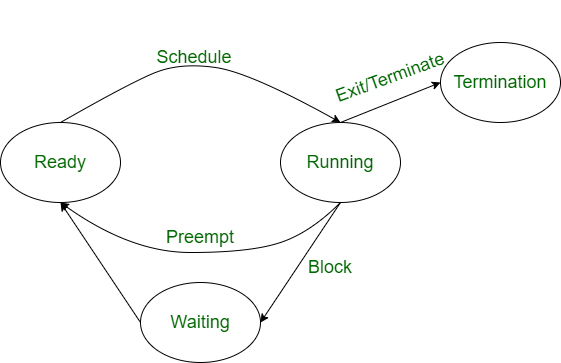
There are three types of process scheduler. 

1. **Long Term or job scheduler :**   
   It brings the new process to the ‘Ready State’. It controls ***Degree of Multi-programming***, i.e., number of process present in ready state at any point of time. It is important that the long-term scheduler make a careful selection of both I/O and CPU-bound processes. I/O bound tasks are which use much of their time in input and output operations while CPU bound processes are which spend their time on CPU. The job scheduler increases efficiency by maintaining a balance between the two.
2. **Short term or CPU scheduler :**   
   It is responsible for selecting one process from ready state for scheduling it on the running state. Note: Short-term scheduler only selects the process to schedule it doesn’t load the process on running.  Here is when all the scheduling algorithms are used. The CPU scheduler is responsible for ensuring there is no starvation owing to high burst time processes.  
   ***Dispatcher*** is responsible for loading the process selected by Short-term scheduler on the CPU (Ready to Running State) Context switching is done by dispatcher only. A dispatcher does the following:
   1. Switching context.
   2. Switching to user mode.
   3. Jumping to the proper location in the newly loaded program.
3. **Medium-term scheduler :**   
   It is responsible for suspending and resuming the process. It mainly does swapping (moving processes from main memory to disk and vice versa). Swapping may be necessary to improve the process mix or because a change in memory requirements has overcommitted available memory, requiring memory to be freed up. It is helpful in maintaining a perfect balance between the I/O bound and the CPU bound. It reduces the degree of multiprogramming.

21. What do you mean by inter process communication?  

22. Write a note on: operations on processes.

**Operation on a Process:**  
The execution of a process is a complex activity. It involves various operations. Following are the operations that are performed while execution of a process:



**1. Creation:** This the initial step of process execution activity. Process creation means the construction of a new process for the execution. This might be performed by system, user or old process itself. There are several events that leads to the process creation. Some of the such events are following:

* When we start the computer, system creates several background processes.
* A user may request to create a new process.
* A process can create a new process itself while executing.
* Batch system takes initiation of a batch job.

**2. Scheduling/Dispatching:** The event or activity in which the state of the process is changed from ready to running. It means the operating system puts the process from ready state into the running state. Dispatching is done by operating system when the resources are free or the process has higher priority than the ongoing process. There are various other cases in which the process in running state is preempted and process in ready state is dispatched by the operating system.

**3. Blocking:** When a process invokes an input-output system call that blocks the process and operating system put in block mode. Block mode is basically a mode where process waits for input-output. Hence on the demand of process itself, operating system blocks the process and dispatches another process to the processor. Hence, in process blocking operation, the operating system puts the process in ‘waiting’ state.

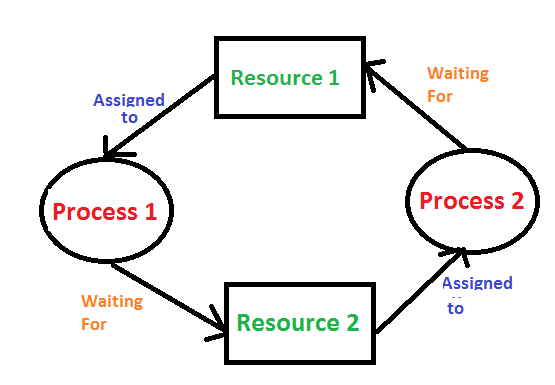
**4. Preemption:** When a timeout occurs that means the process hadn’t been terminated in the allotted time interval and next process is ready to execute, then the operating system preempts the process. This operation is only valid where CPU scheduling supports preemption. Basically this happens in priority scheduling where on the incoming of high priority process the ongoing process is preempted. Hence, in process preemption operation, the operating system puts the process in ‘ready’ state.

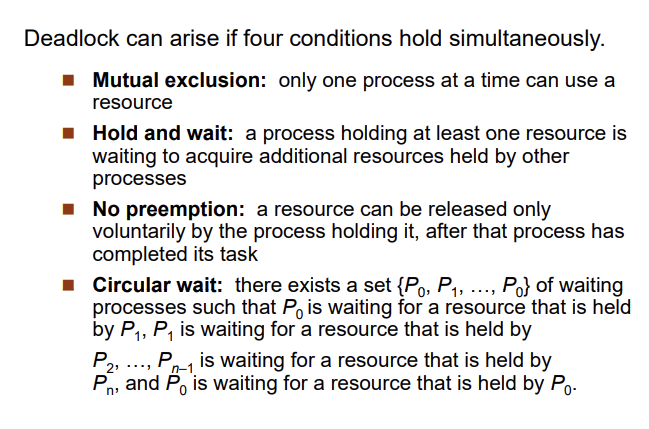
**5. Termination:** Process termination is the activity of ending the process. In other words, process termination is the relaxation of computer resources taken by the process for the execution. Like creation, in termination also there may be several events that may lead to the process termination. Some of them are:

* Process completes its execution fully and it indicates to the OS that it has finished.
* Operating system itself terminates the process due to service errors.
* There may be problem in hardware that terminates the process.
* One process can be terminated by another process.

1.What is deadlock? State necessary and sufficient conditions for the same.

***Deadlock***is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.   
Consider an example when two trains are coming toward each other on the same track and there is only one track, none of the trains can move once they are in front of each other. A similar situation occurs in operating systems when there are two or more processes that hold some resources and wait for resources held by other(s). For example, in the below diagram, Process 1 is holding Resource 1 and waiting for resource 2 which is acquired by process 2, and process 2 is waiting for resource 1.





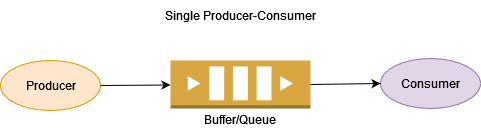
Or

**Deadlock can arise if**the **following four conditions hold simultaneously (Necessary Conditions)**  
***Mutual Exclusion:*** Two or more resources are non-shareable (Only one process can use at a time)   
***Hold and Wait:***A process is holding at least one resource and waiting for resources.   
***No Preemption:*** A resource cannot be taken from a process unless the process releases the resource.   
***Circular Wait:*** A set of processes are waiting for each other in circular form.

2.Explain producer consumer problem with example?

Producer and Consumer are two separate processes. Both processes share a common buffer or queue. The producer continuously produces certain data and pushes it onto the buffer, whereas the consumer consumes those data from the buffer.

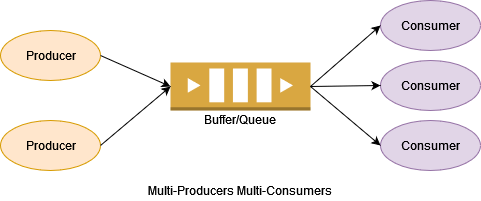
Let's review a diagram showing this simple scenario:

[](https://www.baeldung.com/wp-content/uploads/2022/02/Producer-Consumer1.png)

**Inherently, this problem has certain complexities to deal with**:

* Both producer and consumer may try to update the queue at the same time. This could lead to data loss or inconsistencies.
* Producers might be slower than consumers. In such cases, the consumer would process elements fast and wait.
* In some cases, the consumer can be slower than a producer. This situation leads to a queue overflow issue.
* In real scenarios, we may have multiple producers, multiple consumers, or both. This may cause the same message to be processed by different consumers.

The diagram below depicts a case with multiple producers and multiple consumers:

[](https://www.baeldung.com/wp-content/uploads/2022/02/Multi-Producers-Multi-Consumers.png)

We need to handle resource sharing and synchronization to solve a few complexities:

* Synchronization on queue while adding and removing data
* On queue empty, the consumer has to wait until the producer adds new data to the queue
* When the queue is full, the producer has to wait until the consumer consumes data and the queue has some empty buffer

The producer consumer problem is a synchronization problem. There is a fixed size buffer and the producer produces items and enters them into the buffer. The consumer removes

Following points shows problems that occur in Producer-Consumer:

* The producer should produce data only when the buffer is not full. In case it is found that the buffer is full, the producer is not allowed to store any data into the memory buffer.
* Data can only be consumed by the consumer if and only if the memory buffer is not empty. In case it is found that the buffer is empty, the consumer is not allowed to use any data from the memory buffer.
* Accessing memory buffer should not be allowed to producer and consumer at the same time.

3.Explain Dead lock detection (Banker’s Algorithm) with Example?

If a system does not employ either a deadlock prevention or deadlock avoidance algorithm then a deadlock situation may occur. In this case-

Apply an algorithm to examine state of system to determine whether deadlock has occurred or not.

Apply an algorithm to recover from the deadlock. For more refer- Deadlock Recovery

Deadlock Avoidance Algorithm/ Bankers Algorithm:

The algorithm employs several times varying data structures:

Available –

A vector of length m indicates the number of available resources of each type.

Allocation –

An n\*m matrix defines the number of resources of each type currently allocated to a process. Column represents resource and rows represent process.

Request –

An n\*m matrix indicates the current request of each process. If request[i][j] equals k then process Pi is requesting k more instances of resource type Rj.

This algorithm has already been discussed here

Now, Bankers algorithm includes a Safety Algorithm / Deadlock Detection Algorithm

The algorithm for finding out whether a system is in a safe state can be described as follows:

Steps of Algorithm:

Let Work and Finish be vectors of length m and n respectively. Initialize Work= Available. For i=0, 1, …., n-1, if Requesti = 0, then Finish[i] = true; otherwise, Finish[i]= false.

Find an index i such that both

a) Finish[i] == false

b) Requesti <= Work

If no such i exists go to step 4.

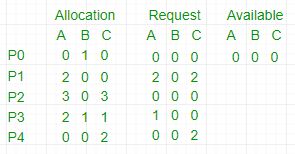
Work= Work+ Allocationi

Finish[i]= true

Go to Step 2.

If Finish[i]== false for some i, 0<=i<n, then the system is in a deadlocked state. Moreover, if Finish[i]==false the process Pi is deadlocked.

For example,



allocation, request matrix

In this, Work = [0, 0, 0] &

Finish = [false, false, false, false, false]

i=0 is selected as both Finish[0] = false and [0, 0, 0]<=[0, 0, 0].

Work =[0, 0, 0]+[0, 1, 0] =>[0, 1, 0] &

Finish = [true, false, false, false, false].

i=2 is selected as both Finish[2] = false and [0, 0, 0]<=[0, 1, 0].

Work =[0, 1, 0]+[3, 0, 3] =>[3, 1, 3] &

Finish = [true, false, true, false, false].

i=1 is selected as both Finish[1] = false and [2, 0, 2]<=[3, 1, 3].

Work =[3, 1, 3]+[2, 0, 0] =>[5, 1, 3] &

Finish = [true, true, true, false, false].

i=3 is selected as both Finish[3] = false and [1, 0, 0]<=[5, 1, 3].

Work =[5, 1, 3]+[2, 1, 1] =>[7, 2, 4] &

Finish = [true, true, true, true, false].

i=4 is selected as both Finish[4] = false and [0, 0, 2]<=[7, 2, 4].

Work =[7, 2, 4]+[0, 0, 2] =>[7, 2, 6] &

Finish = [true, true, true, true, true].

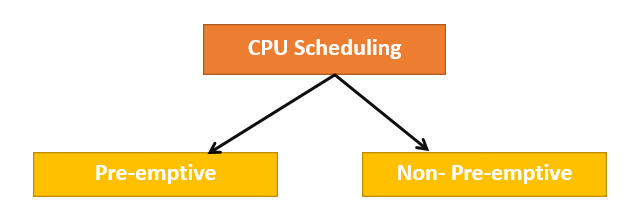
Since Finish is a vector of all true it means there is no deadlock in this example.

4.Explain CPU Scheduling Algorithms with examples?

**CPU Scheduling** is a process of determining which process will own CPU for execution while another process is on hold. The main task of CPU scheduling is to make sure that whenever the CPU remains idle, the OS at least select one of the processes available in the ready queue for execution. The selection process will be carried out by the CPU scheduler. It selects one of the processes in memory that are ready for execution.

## Types of CPU Scheduling

Here are two kinds of Scheduling methods:



### Preemptive Scheduling

In Preemptive Scheduling, the tasks are mostly assigned with their priorities. Sometimes it is important to run a task with a higher priority before another lower priority task, even if the lower priority task is still running. The lower priority task holds for some time and resumes when the higher priority task finishes its execution.

### Non-Preemptive Scheduling

In this type of scheduling method, the CPU has been allocated to a specific process. The process that keeps the CPU busy will release the CPU either by switching context or terminating. It is the only method that can be used for various hardware platforms. That’s because it doesn’t need special hardware (for example, a timer) like preemptive scheduling.

### When scheduling is Preemptive or Non-Preemptive?

To determine if scheduling is preemptive or non-preemptive, consider these four parameters:

1. A process switches from the running to the waiting state.
2. Specific process switches from the running state to the ready state.
3. Specific process switches from the waiting state to the ready state.
4. Process finished its execution and terminated.

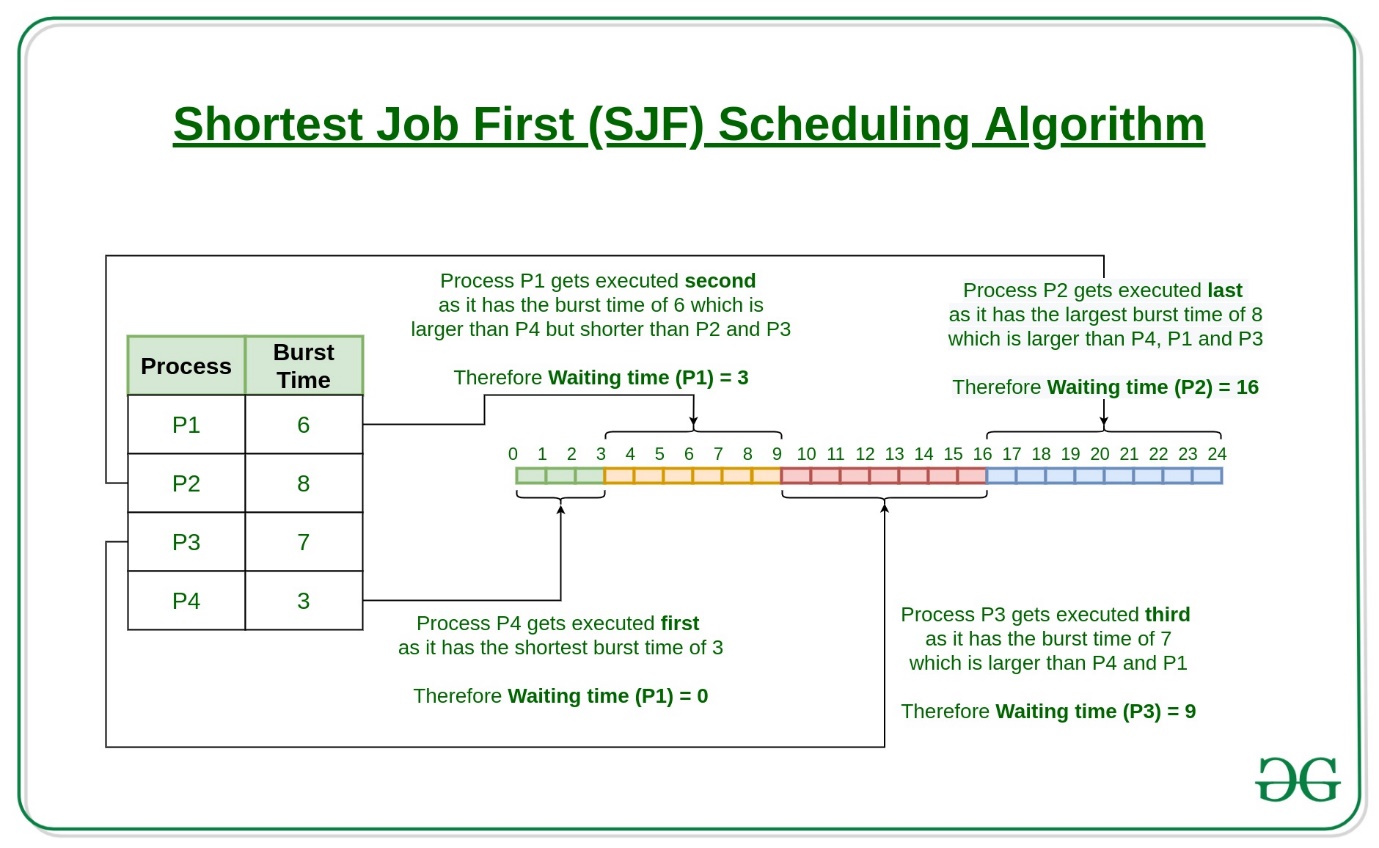
**Only conditions 1 and 4 apply, the scheduling is called non- preemptive.**

**All other scheduling are preemptive.**

### Example:

### 2. Shortest Job First(SJF):

**Shortest job first (SJF)** is a scheduling process that selects the waiting process with the smallest execution time to execute next. This scheduling method may or may not be preemptive. Significantly reduces the average waiting time for other processes waiting to be executed. The full form of SJF is Shortest Job First.



**Characteristics of SJF:**

* Shortest Job first has the advantage of having a minimum average waiting time among all [operating system scheduling algorithms.](https://www.geeksforgeeks.org/cpu-scheduling-in-operating-systems/)
* It is associated with each task as a unit of time to complete.
* It may cause starvation if shorter processes keep coming. This problem can be solved using the concept of ageing.

**Advantages of Shortest Job first:**

* As SJF reduces the average waiting time thus, it is better than the first come first serve scheduling algorithm.
* SJF is generally used for long term scheduling

**Disadvantages of SJF:**

* One of the demerit SJF has is starvation.
* Many times it becomes complicated to predict the length of the upcoming CPU request

## Types of CPU scheduling Algorithm

There are mainly six types of process scheduling algorithms

1. First Come First Serve (FCFS)
2. Shortest-Job-First (SJF) Scheduling
3. Shortest Remaining Time
4. Priority Scheduling
5. Round Robin Scheduling
6. Multilevel Queue Scheduling

5.What are the methods for handling deadlock.

Methods of handling deadlocks : There are three approaches to deal with deadlocks.

1. Deadlock Prevention

2. Deadlock avoidance

3. Deadlock detection

These are explained as following below.

1**. Deadlock Prevention** : The strategy of deadlock prevention is to design the system in such a way that the possibility of deadlock is excluded. Indirect method prevent the occurrence of one of three necessary condition of deadlock i.e., mutual exclusion, no pre-emption and hold and wait. Direct method prevent the occurrence of circular wait. Prevention techniques – Mutual exclusion – is supported by the OS. Hold and Wait – condition can be prevented by requiring that a process requests all its required resources at one time and blocking the process until all of its requests can be granted at a same time simultaneously. But this prevention does not yield good result because :

long waiting time required

in efficient use of allocated resource

A process may not know all the required resources in advance

No pre-emption – techniques for ‘no pre-emption are’

If a process that is holding some resource, requests another resource that can not be immediately allocated to it, the all resource currently being held are released and if necessary, request them again together with the additional resource.

If a process requests a resource that is currently held by another process, the OS may pre-empt the second process and require it to release its resources. This works only if both the processes do not have same priority.

Circular wait One way to ensure that this condition never hold is to impose a total ordering of all resource types and to require that each process requests resource in an increasing order of enumeration, i.e., if a process has been allocated resources of type R, then it may subsequently request only those resources of types following R in ordering.

2. **Deadlock Avoidance** : This approach allows the three necessary conditions of deadlock but makes judicious choices to assure that deadlock point is never reached. It allows more concurrency than avoidance detection A decision is made dynamically whether the current resource allocation request will, if granted, potentially lead to deadlock. It requires the knowledge of future process requests. Two techniques to avoid deadlock :

Process initiation denial

Resource allocation denial

Advantages of deadlock avoidance techniques :

Not necessary to pre-empt and rollback processes

Less restrictive than deadlock prevention

Disadvantages :

Future resource requirements must be known in advance

Processes can be blocked for long periods

Exists fixed number of resources for allocation

3**. Deadlock Detection** : Deadlock detection is used by employing an algorithm that tracks the circular waiting and killing one or more processes so that deadlock is removed. The system state is examined periodically to determine if a set of processes is deadlocked. A deadlock is resolved by aborting and restarting a process, relinquishing all the resources that the process held.

This technique does not limit resources access or restrict process action.

Requested resources are granted to processes whenever possible.

It never delays the process initiation and facilitates online handling.

The disadvantage is the inherent pre-emption losses.

**Or**

**1. Deadlock Ignorance**

Deadlock Ignorance is the most widely used approach among all the mechanism. This is being used by many operating systems mainly for end user uses. In this approach, the Operating system assumes that deadlock never occurs. It simply ignores deadlock. This approach is best suitable for a single end user system where User uses the system only for browsing and all other normal stuff.

There is always a tradeoff between Correctness and performance. The operating systems like Windows and Linux mainly focus upon performance. However, the performance of the system decreases if it uses deadlock handling mechanism all the time if deadlock happens 1 out of 100 times then it is completely unnecessary to use the deadlock handling mechanism all the time.

In these types of systems, the user has to simply restart the computer in the case of deadlock. Windows and Linux are mainly using this approach.

**2. Deadlock prevention**

Deadlock happens only when Mutual Exclusion, hold and wait, No preemption and circular wait holds simultaneously. If it is possible to violate one of the four conditions at any time then the deadlock can never occur in the system.

The idea behind the approach is very simple that we have to fail one of the four conditions but there can be a big argument on its physical implementation in the system.

**3. Deadlock avoidance**

In deadlock avoidance, the operating system checks whether the system is in safe state or in unsafe state at every step which the operating system performs. The process continues until the system is in safe state. Once the system moves to unsafe state, the OS has to backtrack one step.

In simple words, The OS reviews each allocation so that the allocation doesn't cause the deadlock in the system.

**4. Deadlock detection and recovery**

This approach let the processes fall in deadlock and then periodically check whether deadlock occur in the system or not. If it occurs then it applies some of the recovery methods to the system to get rid of deadlock.

6. Write a program to give a solution to the producer–consumer problem using shared memory.

import java.util.concurrent.Semaphore;

class Q{

// an item

int item;

// semCon initialized with 0 permits

// to ensure put() executes first

static Semaphore semCon = new Semaphore(0);

static Semaphore semprod = new Semaphore(1);

// to get an item from buffer

void get(){

try{

// Before consumer can consume an item

// it must aquire a permit from semCon

semCon.acquire();

}

catch(InterruptedException e){

System.out.println("InterruptedException caught");

}

// consumer consuming an item

System.out.println("\n Consumer consumed item : "+item);

// After consumer consumes the item

// It releases semProd to notify producer

semprod.release();

}

// to put an item in buffer

void put(int item){

try{

// Before producer can produce an item

// it must acquire a permit from semprod

semprod.acquire();

}

catch(InterruptedException e){

System.out.println("InterruptedException caught");

}

// producer producing an item

this.item = item;

System.out.println("\n Producer produced item : "+item);

// After producer produces the item

// it releases semcon to notify consumer

semCon.release();

}

}

// Producer class

class producer implements Runnable{

Q q;

producer(Q q){

this.q = q;

new Thread(this,"producer").start();

}

@Override

public void run(){

for(int i=0; i<5; i++)

// Producer put items

q.put(i);

}

}

// consumer class

class consumer implements Runnable{

Q q;

consumer(Q q){

this.q = q;

new Thread(this,"consumer").start();

}

@Override

public void run(){

for(int i=0; i<5; i++)

// Consumer put items

q.get();

}

}

// Driver class

public class PT {

public static void main(String[] args){

// creating buffer queue

Q q = new Q();

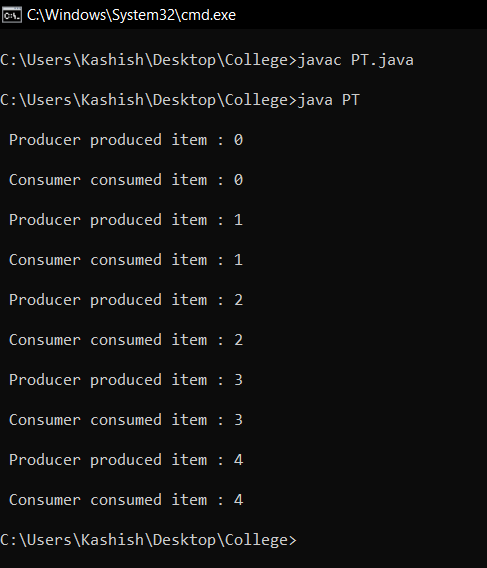
// Starting consumer thread

new consumer(q);

// Starting producer thread

new producer(q);

}

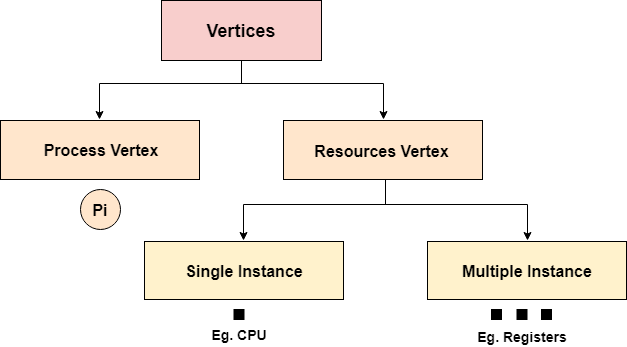


7. What is resource-allocation graph?

The resource allocation graph is the pictorial representation of the state of a system. As its name suggests, the resource allocation graph is the complete information about all the processes which are holding some resources or waiting for some resources.

It also contains the information about all the instances of all the resources whether they are available or being used by the processes.

In Resource allocation graph, the process is represented by a Circle while the Resource is represented by a rectangle. Let's see the types of vertices and edges in detail.



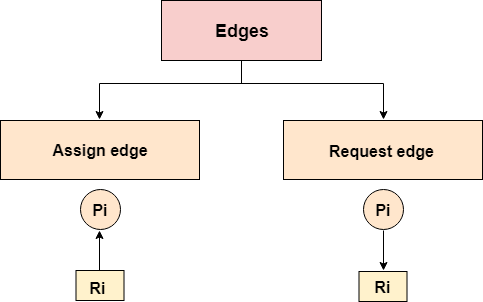
Vertices are mainly of two types, Resource and process. Each of them will be represented by a different shape. Circle represents process while rectangle represents resource.

65.9M

1.1K

Hello Java Program for Beginners

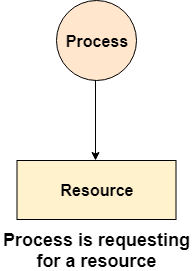
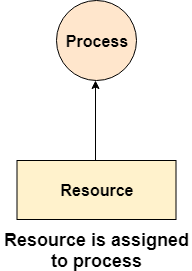
A resource can have more than one instance. Each instance will be represented by a dot inside the rectangle.



Edges in RAG are also of two types, one represents assignment and other represents the wait of a process for a resource. The above image shows each of them.

A resource is shown as assigned to a process if the tail of the arrow is attached to an instance to the resource and the head is attached to a process.

A process is shown as waiting for a resource if the tail of an arrow is attached to the process while the head is pointing towards the resource.

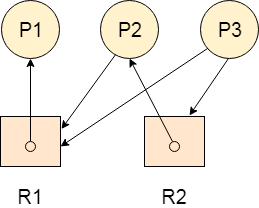
 

Example

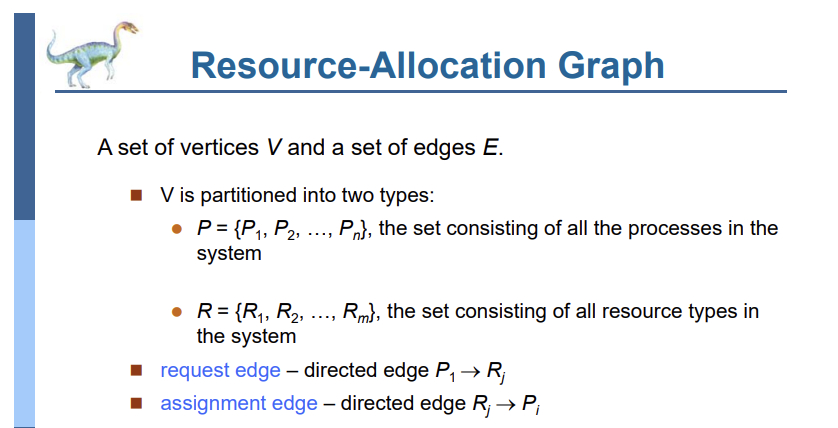
Let'sconsider 3 processes P1, P2 and P3, and two types of resources R1 and R2. The resources are having 1 instance each.

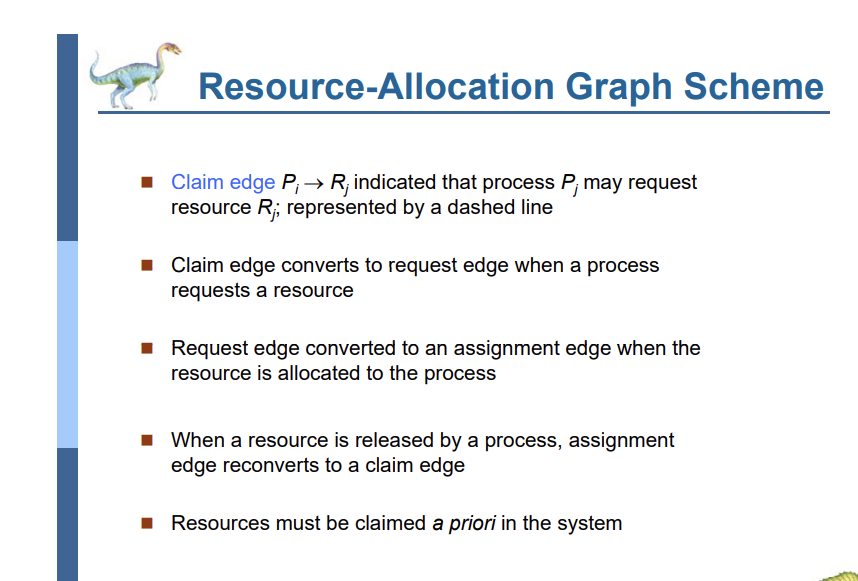
According to the graph, R1 is being used by P1, P2 is holding R2 and waiting for R1, P3 is waiting for R1 as well as R2.

The graph is deadlock free since no cycle is being formed in the graph.



Or





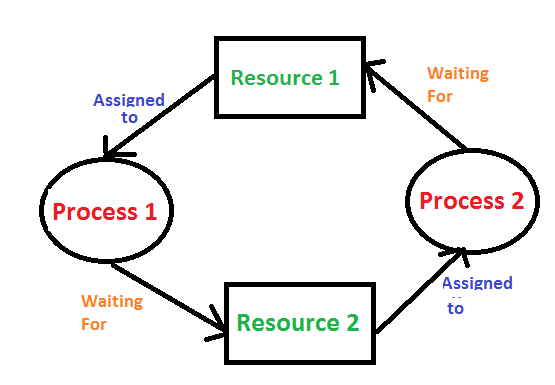
8.CPU scheduling examples FCFS,SJF ,priority (Examples done in class)

(Given in OS.pdf)

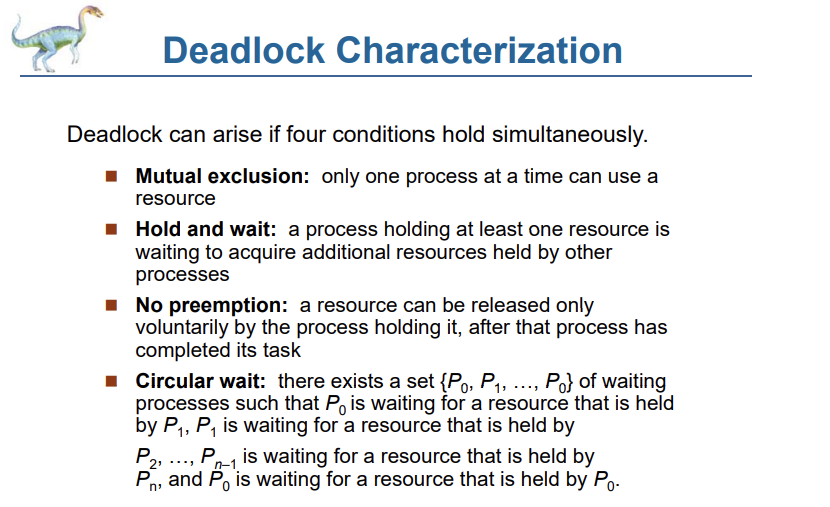
9.Define deadlock?

Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.

Consider an example when two trains are coming toward each other on the same track and there is only one track, none of the trains can move once they are in front of each other. A similar situation occurs in operating systems when there are two or more processes that hold some resources and wait for resources held by other(s). For example, in the below diagram, Process 1 is holding Resource 1 and waiting for resource 2 which is acquired by process 2, and process 2 is waiting for resource 1.

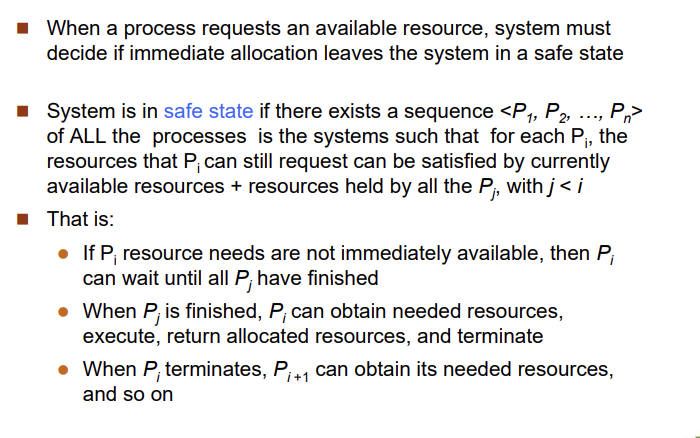


10.Give the condition necessary for a deadlock situation to arise?



11.Define ‘Safe State”?

If the system can allocate resources to the [process](https://t4tutorials.com/program-and-process-in-operating-systems/) in such a way that it can avoid deadlock. Then the system is in a safe state.



12.Explain about Deadlock Avoidance?

In deadlock avoidance, the request for any resource will be granted if the resulting state of the system doesn't cause deadlock in the system. The state of the system will continuously be checked for safe and unsafe states.

In order to avoid deadlocks, the process must tell OS, the maximum number of resources a process can request to complete its execution.

The simplest and most useful approach states that the process should declare the maximum number of resources of each type it may ever need. The Deadlock avoidance algorithm examines the resource allocations so that there can never be a circular wait condition.

The resource allocation state of a system can be defined by the instances of available and allocated resources, and the maximum instance of the resources demanded by the processes.

A state of a system recorded at some random time is shown below.

#### **Resources Assigned**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Process** | **Type 1** | **Type 2** | **Type 3** | **Type 4** |
| A | 3 | 0 | 2 | 2 |
| B | 0 | 0 | 1 | 1 |
| C | 1 | 1 | 1 | 0 |
| D | 2 | 1 | 4 | 0 |

#### **Resources still needed**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Process** | **Type 1** | **Type 2** | **Type 3** | **Type 4** |
| A | 1 | 1 | 0 | 0 |
| B | 0 | 1 | 1 | 2 |
| C | 1 | 2 | 1 | 0 |
| D | 2 | 1 | 1 | 2 |

1. E = (7 6 8 4)
2. P = (6 2 8 3)
3. A = (1 4 0 1)

Above tables and vector E, P and A describes the resource allocation state of a system. There are 4 processes and 4 types of the resources in a system. Table 1 shows the instances of each resource assigned to each process.

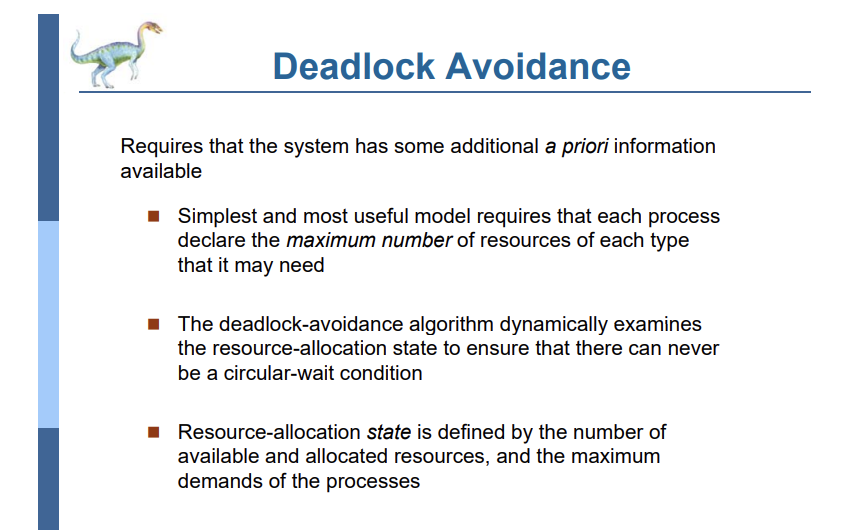
Table 2 shows the instances of the resources, each process still needs. Vector E is the representation of total instances of each resource in the system.

Vector P represents the instances of resources that have been assigned to processes. Vector A represents the number of resources that are not in use.

A state of the system is called safe if the system can allocate all the resources requested by all the processes without entering into deadlock.

If the system cannot fulfill the request of all processes then the state of the system is called unsafe.

The key of Deadlock avoidance approach is when the request is made for resources then the request must only be approved in the case if the resulting state is also a safe state.

OR

13.Explain recovery from deadlock?

Prerequisite – Deadlock Detection And Recovery

When a Deadlock Detection Algorithm determines that a deadlock has occurred in the system, the system must recover from that deadlock. There are two approaches of breaking a Deadlock:

1. Process Termination:

To eliminate the deadlock, we can simply kill one or more processes. For this, we use two methods:

(a). Abort all the Deadlocked Processes:

Aborting all the processes will certainly break the deadlock, but with a great expense. The deadlocked processes may have computed for a long time and the result of those partial computations must be discarded and there is a probability to recalculate them later.

(b). Abort one process at a time until deadlock is eliminated:

Abort one deadlocked process at a time, until deadlock cycle is eliminated from the system. Due to this method, there may be considerable overhead, because after aborting each process, we have to run deadlock detection algorithm to check whether any processes are still deadlocked.

2. Resource Preemption:

To eliminate deadlocks using resource preemption, we preempt some resources from processes and give those resources to other processes. This method will raise three issues –

(a). Selecting a victim:

We must determine which resources and which processes are to be preempted and also the order to minimize the cost.

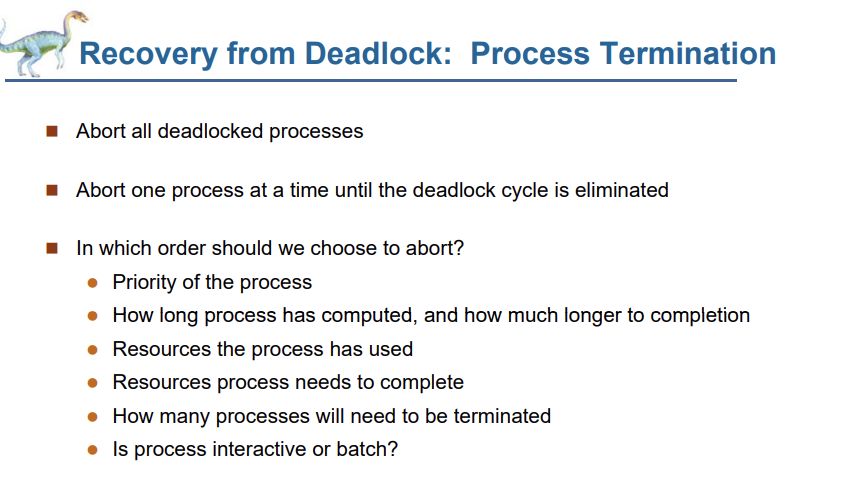
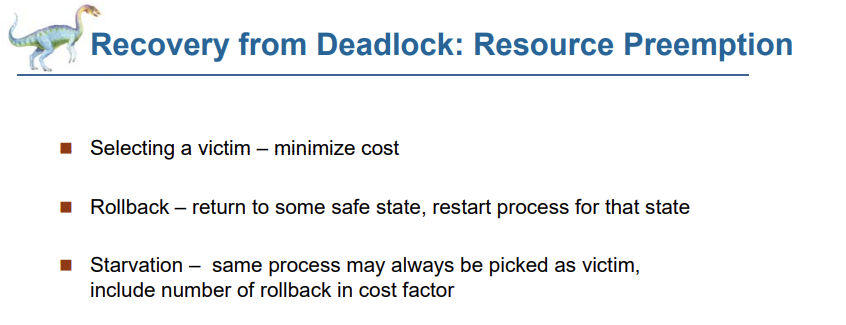
(b). Rollback:

We must determine what should be done with the process from which resources are preempted. One simple idea is total rollback. That means abort the process and restart it.

(c). Starvation:

In a system, it may happen that same process is always picked as a victim. As a result, that process will never complete its designated task. This situation is called Starvation and must be avoided. One solution is that a process must be picked as a victim only a finite number of times.

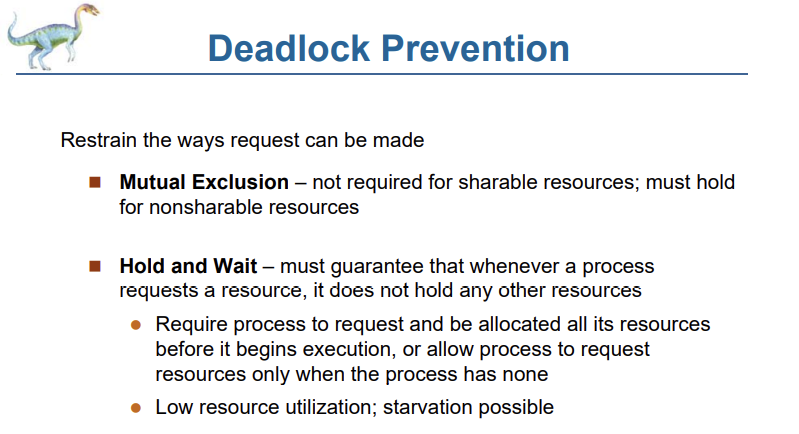
**OR**

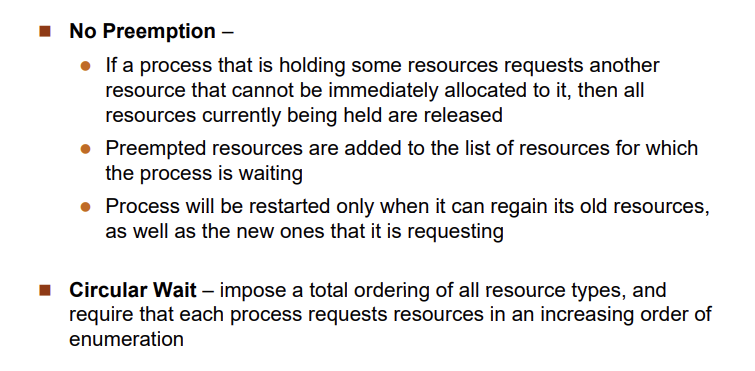
 

14.Explain Dead lock detection (Banker’s Algorithm) with Example?

Repeated

15. Write about Deadlock Prevention Methods





**OR**

<https://www.scaler.com/topics/operating-system/deadlock-prevention-in-operating-system/>

**Unit 3**

Q.1] What is Segmentation? Explain with Example.

# Segmentation

In Operating Systems, Segmentation is a memory management technique in which the memory is divided into the variable size parts. Each part is known as a segment which can be allocated to a process.

The details about each segment are stored in a table called a segment table. Segment table is stored in one (or many) of the segments.

Segment table contains mainly two information about segment:

1. Base: It is the base address of the segment
2. Limit: It is the length of the segment.

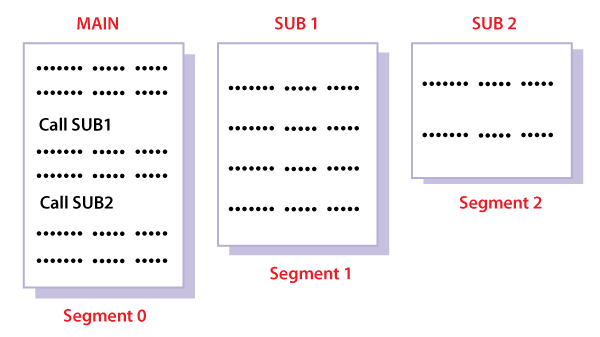
## **Why Segmentation is required?**

Till now, we were using Paging as our main memory management technique. Paging is more close to the Operating system rather than the User. It divides all the processes into the form of pages regardless of the fact that a process can have some relative parts of functions which need to be loaded in the same page.

49.of India (1947-2020)

Operating system doesn't care about the User's view of the process. It may divide the same function into different pages and those pages may or may not be loaded at the same time into the memory. It decreases the efficiency of the system.

It is better to have segmentation which divides the process into the segments. Each segment contains the same type of functions such as the main function can be included in one segment and the library functions can be included in the other segment.



## **Translation of Logical address into physical address by segment table**

CPU generates a logical address which contains two parts:

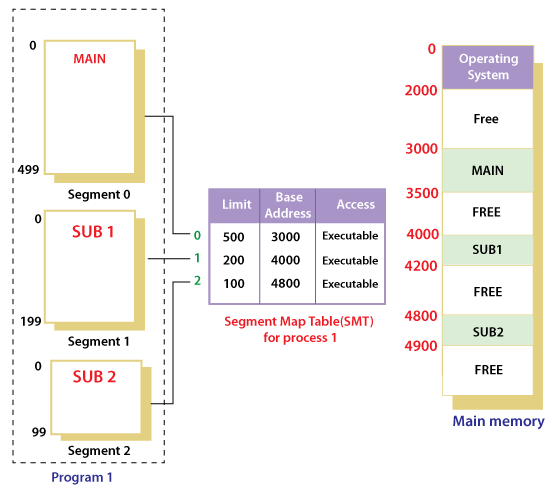
1. Segment Number
2. Offset

**For Example:**

Suppose a 16 bit address is used with 4 bits for the segment number and 12 bits for the segment offset so the maximum segment size is 4096 and the maximum number of segments that can be refereed is 16.

When a program is loaded into memory, the segmentation system tries to locate space that is large enough to hold the first segment of the process, space information is obtained from the free list maintained by memory manager. Then it tries to locate space for other segments. Once adequate space is located for all the segments, it loads them into their respective areas.

The operating system also generates a segment map table for each program.



With the help of segment map tables and hardware assistance, the operating system can easily translate a logical address into physical address on execution of a program.

The **Segment number** is mapped to the segment table. The limit of the respective segment is compared with the offset. If the offset is less than the limit then the address is valid otherwise it throws an error as the address is invalid.

In the case of valid addresses, the base address of the segment is added to the offset to get the physical address of the actual word in the main memory.

The above figure shows how address translation is done in case of segmentation.

## **Advantages of Segmentation**

1. No internal fragmentation
2. Average Segment Size is larger than the actual page size.
3. Less overhead
4. It is easier to relocate segments than entire address space.
5. The segment table is of lesser size as compared to the page table in paging.

## **Disadvantages**

1. It can have external fragmentation.
2. it is difficult to allocate contiguous memory to variable sized partition.
3. Costly memory management algorithms.

Q.2] Important FIFO and Optimal page replacement algorithm to be studied from the link already provided.

Q.3] What is virtual memory? Discuss the benefits of virtual memory techniques.

(Repeated)

Q.4] Suppose disk is having 200 cylinders numbered from 0 to 199 .The disk is currently servicing at cylinder 59 and the previous request was at cylinder 60.The queue of pending request in FIFO order is 98,183,37,122,14,124,65,67.Calculate the total distance the read/write head will traverse. Disk Scheduling example to be studied from the link already provided.

Q.5] Explain the concept of file with Example.

## **File**

A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the files creator and user.

## **File Structure**

A File Structure should be according to a required format that the operating system can understand.

* A file has a certain defined structure according to its type.
* A text file is a sequence of characters organized into lines.
* A source file is a sequence of procedures and functions.
* An object file is a sequence of bytes organized into blocks that are understandable by the machine.
* When operating system defines different file structures, it also contains the code to support these file structure. Unix, MS-DOS support minimum number of file structure.

## **File Type**

File type refers to the ability of the operating system to distinguish different types of file such as text files source files and binary files etc. Many operating systems support many types of files. Operating system like MS-DOS and UNIX have the following types of files −

### **Ordinary files**

* These are the files that contain user information.
* These may have text, databases or executable program.
* The user can apply various operations on such files like add, modify, delete or even remove the entire file.

### **Directory files**

* These files contain list of file names and other information related to these files.

### **Special files**

* These files are also known as device files.
* These files represent physical device like disks, terminals, printers, networks, tape drive etc.

These files are of two types −

* **Character special files** − data is handled character by character as in case of terminals or printers.
* **Block special files** − data is handled in blocks as in the case of disks and tapes.

## **File Access Mechanisms**

File access mechanism refers to the manner in which the records of a file may be accessed. There are several ways to access files −

* Sequential access
* Direct/Random access
* Indexed sequential access

### **Sequential access**

A sequential access is that in which the records are accessed in some sequence, i.e., the information in the file is processed in order, one record after the other. This access method is the most primitive one. Example: Compilers usually access files in this fashion.

### **Direct/Random access**

* Random access file organization provides, accessing the records directly.
* Each record has its own address on the file with by the help of which it can be directly accessed for reading or writing.
* The records need not be in any sequence within the file and they need not be in adjacent locations on the storage medium.

### **Indexed sequential access**

* This mechanism is built up on base of sequential access.
* An index is created for each file which contains pointers to various blocks.
* Index is searched sequentially and its pointer is used to access the file directly.

## **Space Allocation**

Files are allocated disk spaces by operating system. Operating systems deploy following three main ways to allocate disk space to files.

* Contiguous Allocation
* Linked Allocation
* Indexed Allocation

### **Contiguous Allocation**

* Each file occupies a contiguous address space on disk.
* Assigned disk address is in linear order.
* Easy to implement.
* External fragmentation is a major issue with this type of allocation technique.

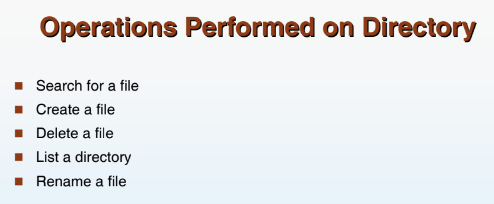
### **Linked Allocation**

* Each file carries a list of links to disk blocks.
* Directory contains link / pointer to first block of a file.
* No external fragmentation
* Effectively used in sequential access file.
* Inefficient in case of direct access file.

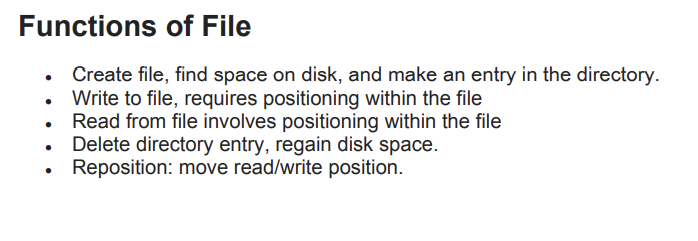
### **Indexed Allocation**

* Provides solutions to problems of contiguous and linked allocation.
* A index block is created having all pointers to files.
* Each file has its own index block which stores the addresses of disk space occupied by the file.
* Directory contains the addresses of index blocks of files.

Q.6] Write about different types of operation performed on file. All types of file operations are to be studied.



**OR**



**OR**

### **File Operations**

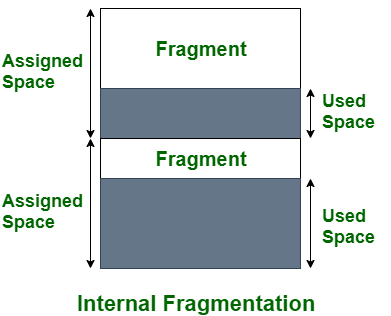
* A file is an abstract data type. To define a file properly, we need to consider the operations that can be performed on files.
* Six basic file operations. The OS can provide system calls to create, write, read, reposition, delete, and truncate files.
  + **Creating a file**. Two steps are necessary to create a file.
    1. Space in the file system must be found for the file.
    2. An entry for the new file must be made in the directory.
  + **Writing a file**. To write a file, we make a system call specifying both the name of the file and the information to be written to the file. The system must keep a write pointer to the location in the file where the next write is to take place. The write pointer must be updated whenever a write occurs.
  + **Reading a file**. To read from a file, we use a system call that specifies the name of the file and where (in memory) the next block of the file should be put. The system needs to keep a read pointer to the location in the file where the next read is to take place.
    1. Because a process is usually either reading from or writing to a file, the current operation location can be kept as a per-process current-file-position pointer.
    2. Both the read and write operations use this same pointer, saving space and reducing system complexity.
  + **Repositioning within a file**. The directory is searched for the appropriate entry, and the current-file-position pointer is repositioned to a given value. Repositioning within a file need not involve any actual I/O. This file operation is also known as a file seek.
  + **Deleting a file**. To delete a file, we search the directory for the named file. Having found the associated directory entry, we release all file space, so that it can be reused by other files, and erase the directory entry.
  + **Truncating a file**. The user may want to erase the contents of a file but keep its attributes. Rather than forcing the user to delete the file and then recreate it, this function allows all attributes to remain unchanged (except for file length) but lets the file be reset to length zero and its file space released.

Q.7] What are Pages and Frames? Explain External and Internal fragmentation?

A **page**, **memory page**, or **virtual page** is a fixed-length contiguous block of [virtual memory](https://en.wikipedia.org/wiki/Virtual_memory), described by a single entry in the [page table](https://en.wikipedia.org/wiki/Page_table).( A **page table** is the [data structure](https://en.wikipedia.org/wiki/Data_structure) used by a [virtual memory](https://en.wikipedia.org/wiki/Virtual_memory) system in a [computer](https://en.wikipedia.org/wiki/Computer) [operating system](https://en.wikipedia.org/wiki/Operating_system) to store the mapping between [virtual addresses](https://en.wikipedia.org/wiki/Virtual_address) and [physical addresses](https://en.wikipedia.org/wiki/Physical_address). ) It is the smallest unit of data for memory management in a virtual memory [operating system](https://en.wikipedia.org/wiki/Operating_system).

Similarly, a **page frame** is the smallest fixed-length contiguous block of [physical memory](https://en.wikipedia.org/wiki/Physical_memory) into which memory pages are mapped by the operating system.

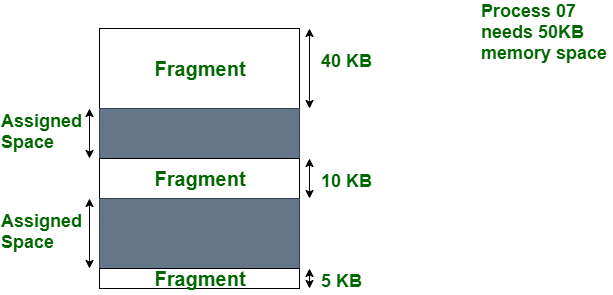
**1. Internal Fragmentation:**   
Internal fragmentation happens when the memory is split into mounted-sized blocks. Whenever a method is requested for the memory, the mounted-sized block is allotted to the method. In the case where the memory allotted to the method is somewhat larger than the memory requested, then the difference between allotted and requested memory is called internal fragmentation.



***Internal Fragmentation***

The above diagram clearly shows the internal fragmentation because the difference between memory allocated and required space or memory is called Internal fragmentation.

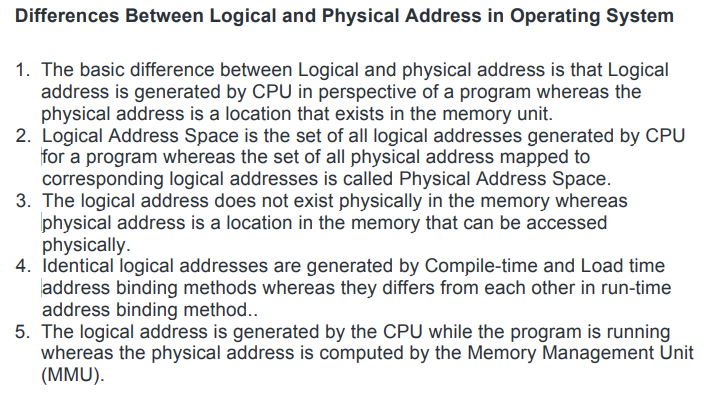
**2. External Fragmentation:**   
External fragmentation happens when there’s a sufficient quantity of area within the memory to satisfy the memory request of a method. However, the process’s memory request cannot be fulfilled because the memory offered is in a non-contiguous manner. Whether you apply a first-fit or best-fit memory allocation strategy it’ll cause external fragmentation.

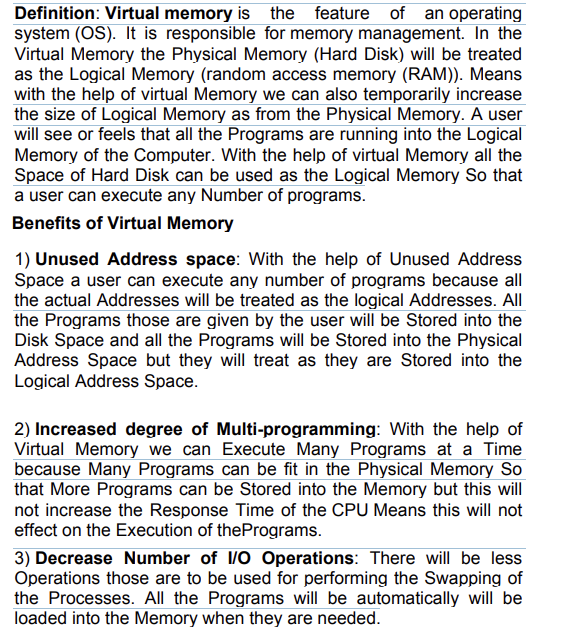


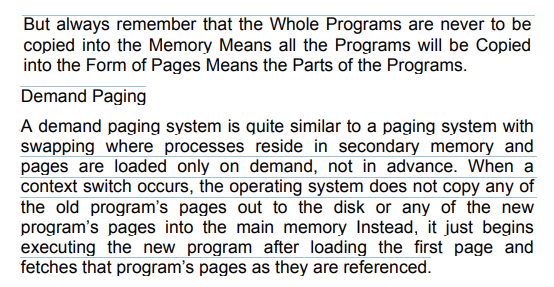
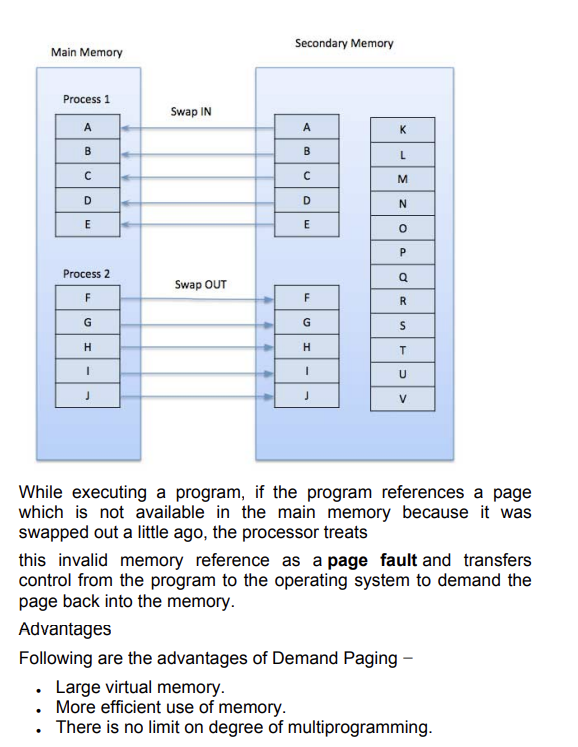
***External Fragmentation***

In the above diagram, we can see that, there is enough space (55 KB) to run a process-07 (required 50 KB) but the memory (fragment) is not contiguous. Here, we use compaction, paging, or segmentation to use the free space to run a process.

Q.8] List two differences between logical and physical addresses.



Q.9] What is virtual memory? Discuss the benefits of virtual memory techniques.

**OR**

[**https://www.techtarget.com/searchstorage/definition/virtual-memory#:~:text=The%20advantages%20to%20using%20virtual,when%20RAM%20space%20runs%20out**](https://www.techtarget.com/searchstorage/definition/virtual-memory#:~:text=The%20advantages%20to%20using%20virtual,when%20RAM%20space%20runs%20out)**.**