Chapter 6

What is a process?

In computing, a process is the instance of a computer program that is being executed by one or many threads. It contains the program code and its activity. Depending on the operating system (OS), a process may be made up of multiple threads of execution that execute instructions concurrently.

Objectives of Process Scheduling Algorithm:

Utilization of CPU at maximum level. Keep the CPU as busy as possible.

Allocation of CPU should be fair.

Throughput should be Maximum. i.e. Number of processes that complete their execution per time unit should be maximized.

Minimum turnaround time, i.e. time taken by a process to finish execution should be the least.

There should be a minimum waiting time and the process should not starve in the ready queue.

Minimum response time. It means that the time when a process produces the first response should be as less as possible.

What are the different terminologies to take care of in any CPU Scheduling algorithm?

Arrival Time: Time at which the process arrives in the ready queue.

Completion Time: Time at which process completes its execution.

Burst Time: Time required by a process for CPU execution.

Turn Around Time: Time Difference between completion time and arrival time.

Turn Around Time = Completion Time – Arrival Time

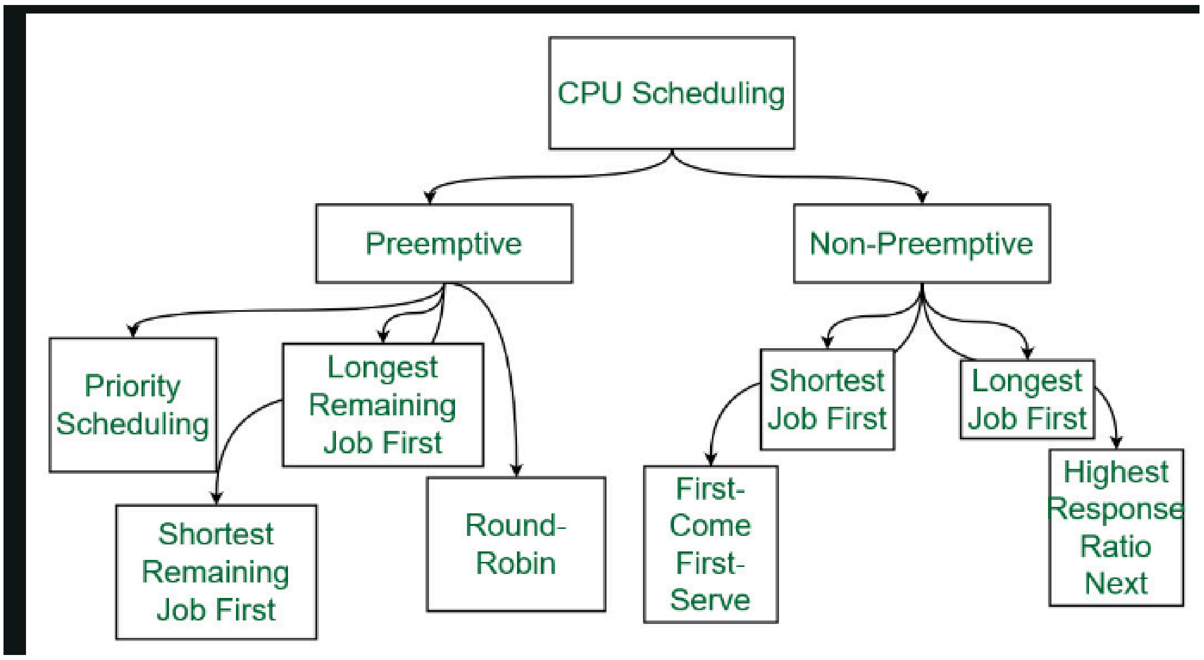
Waiting Time(W.T): Time Difference between turn around time and burst time.

Waiting Time = Turn Around Time – Burst Time

What are the different types of CPU Scheduling Algorithms?

There are mainly two types of scheduling methods:

- Preemptive Scheduling: Preemptive scheduling is used when a process switches from running state to ready state or from the waiting state to the ready state.
- Non-Preemptive Scheduling: Non-Preemptive scheduling is used when a process terminates , or when a process switches from running state to waiting state



1. First Come First Serve:
FCFS considered to be the simplest of all operating system scheduling algorithms. First come first serve scheduling algorithm states that the process that requests the CPU first is allocated the CPU first and is implemented by using FIFO queue.

Characteristics of FCFS:

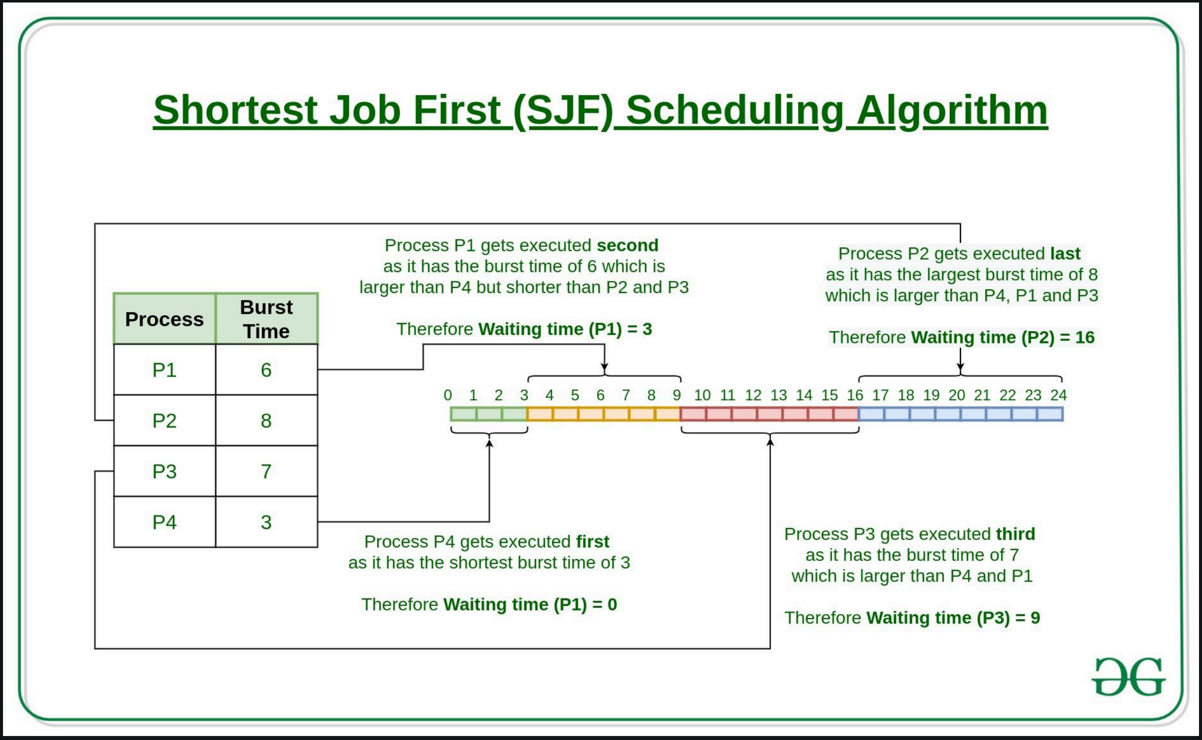
FCFS supports non-preemptive and preemptive CPU scheduling algorithms.
Tasks are always executed on a First-come, First-serve concept.
FCFS is easy to implement and use.
This algorithm is not much efficient in performance, and the wait time is quite high.
Advantages of FCFS:

Easy to implement
First come, first serve method
Disadvantages of FCFS:

FCFS suffers from Convoy effect.
The average waiting time is much higher than the other algorithms.
FCFS is very simple and easy to implement and hence not much efficient.
To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on First come, First serve Scheduling.

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.


Shortest Job First (SJF) Scheduling Algorithm
Process | Burst Time
P1 6
P2 8
P3 7
P4 3
Process P1 gets executed second as it has the burst time of 6 which is larger than P4 but shorter than P2 and P3
Therefore Waiting time (P1) = 3
Process P2 gets executed last as it has the largest burst time of 8 which is larger than P4, P1 and P3
Therefore Waiting time (P2) = 16
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
Process P4 gets executed first as it has the shortest burst time of 3
Therefore Waiting time (P1) = 0
Process P3 gets executed third as it has the burst time of 7 which is larger than P4 and P1
Therefore Waiting time (P3) = 9

Characteristics of SJF:

- Shortest Job first has the advantage of having a minimum average waiting time among all operating system scheduling algorithms.
- 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

What is Starvation?

Starvation or indefinite blocking is a phenomenon associated with the Priority scheduling algorithms, in which a process ready for the CPU (resources) can wait to run indefinitely because of low priority. In a heavily loaded computer system, a steady stream of higher-priority processes can prevent a low-priority process from ever getting the CPU. There have been rumors that in 1967 Priority Scheduling was used in IBM 7094 at MIT, and they found a low-priority process that had not been submitted till 1973.

4. Priority Scheduling:

Preemptive Priority CPU Scheduling Algorithm is a pre-emptive method of CPU scheduling algorithm that works based on the priority of a process. In this algorithm, the editor sets the functions to be as important, meaning that the most important process must be done first. In the case of any conflict, that is, where there is more than one process with equal value, then the most important CPU planning algorithm works on the basis of the FCFS (First Come First Serve) algorithm.

Characteristics of Priority Scheduling:

Schedules tasks based on priority.
When the higher priority work arrives and a task with less priority is executing, the higher priority proess will takes the place of the less priority proess and
The later is suspended until the execution is complete.
Lower is the number assigned, higher is the priority level of a process.
Advantages of Priority Scheduling:

The average waiting time is less than FCFS
Less complex
Disadvantages of Priority Scheduling:

One of the most common demerits of the Preemptive priority CPU scheduling algorithm is the Starvation Problem. This is the problem in which a process has to wait for a longer amount of time to get scheduled into the CPU. This condition is called the starvation problem.
How does Preemptive Priority CPU Scheduling Algorithm work?

Step-1: Select the first process whose arrival time will be 0, we need to select that process because that process is only executing at time t=0.
Step-2: Check the priority of the next available process. Here we need to check for 3 conditions.
if priority(current_process) > priority(prior_process) :- then execute the current process.

if priority(current_process) < priority(prior_process) :- then execute the prior process.
if priority(current_process) = priority(prior_process) :- then execute the process which arrives first i.e., arrival time should be first.
Step-3: Repeat Step-2 until it reaches the final process.
Step-4: When it reaches the final process, choose the process which is having the highest priority & execute it. Repeat the same step until all processes complete their execution.

Characteristics of Round robin:

- It's simple, easy to use, and starvation-free as all processes get the balanced CPU allocation.
- One of the most widely used methods in CPU scheduling as a core.
- It is considered preemptive as the processes are given to the CPU for a very limited time.

Advantages of Round robin:

- Round robin seems to be fair as every process gets an equal share of CPU.
- The newly created process is added to the end of the ready queue.

Disadvantages of Round Robin CPU Scheduling Algorithm

- There is Larger waiting time and Response time.
- There is Low throughput.
- There is Context Switches.
- Gantt chart seems to come too big (if quantum time is less for scheduling. For Example:1 ms for big scheduling.)
- Time consuming scheduling for small quantum.

6. Shortest Remaining Time First:

Shortest remaining time first is the preemptive version of the Shortest job first which we have discussed earlier where the processor is allocated to the job closest to completion. In SRTF the process with the smallest amount of time remaining until completion is selected to execute.

Characteristics of Shortest remaining time first:

- SRTF algorithm makes the processing of the jobs faster than SJF algorithm, given it's overhead charges are not counted.
- The context switch is done a lot more times in SRTF than in SJF and consumes the CPU's valuable time for processing. This adds up to its processing time and diminishes its advantage of fast processing.

Advantages of SRTF:

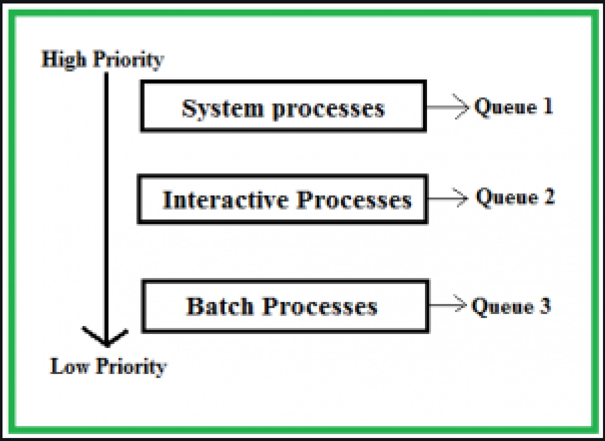
- In SRTF the short processes are handled very fast.
- The system also requires very little overhead since it only makes a decision when a process completes or a new process is added.

Disadvantages of SRTF:

- Like the shortest job first, it also has the potential for process starvation.
- Long processes may be held off indefinitely if short processes are continually added.

9. Multiple Queue Scheduling:

Processes in the ready queue can be divided into different classes where each class has its own scheduling needs. For example, a common division is a **foreground (interactive)** process and a **background (batch)** process. These two classes have different scheduling needs. For this kind of situation **Multilevel Queue Scheduling** is used.



The description of the processes in the above diagram is as follows:

- **System Processes:** The CPU itself has its process to run, generally termed as System Process.
- **Interactive Processes:** An Interactive Process is a type of process in which there should be the same type of interaction.
- **Batch Processes:** Batch processing is generally a technique in the Operating system that collects the programs and data together in the form of a **batch** before the **processing** starts.

Advantages of multilevel queue scheduling:

- The main merit of the multilevel queue is that it has a low scheduling overhead.

Disadvantages of multilevel queue scheduling:

- Starvation problem
- It is inflexible in nature

Mid-1 18-19

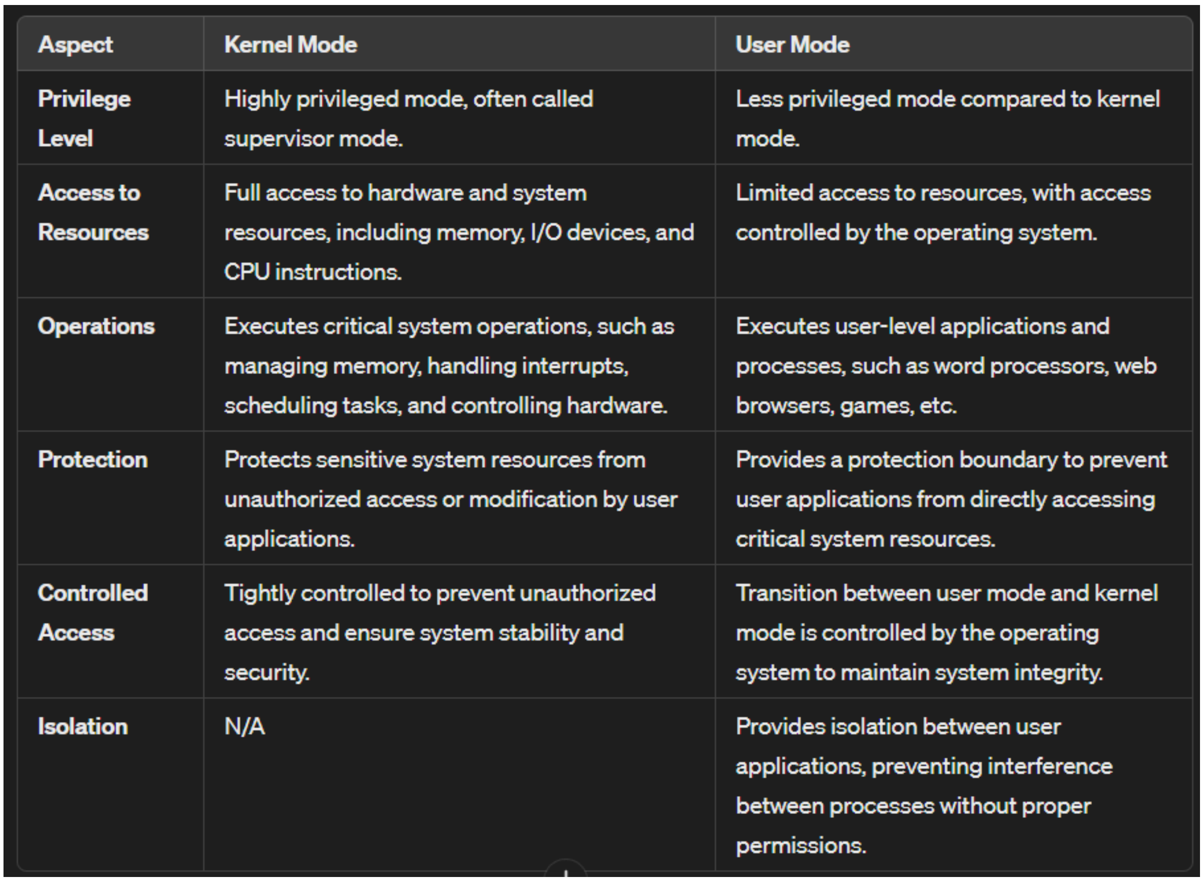
Q1.What are the 3 main purpose of an operating system

1. Resource Management: Operating systems ensure that your computer's resources (like the processor, memory, and storage) are used efficiently, so everything runs smoothly.

2. Abstraction: Operating systems hide the complicated inner workings of the computer, making it easier for software developers to create programs without worrying about hardware details.

3. User Interface: Operating systems provide ways for you to interact with your computer, whether through clicking icons on a screen or typing commands, making it user-friendly and accessible for various tasks.

Q2.How does the distinction between kernel mode and user mode function as a rudimentary form of protection (security) system



Aspect	Kernel Mode	User Mode
Privilege Level	Highly privileged mode, often called supervisor mode.	Less privileged mode compared to kernel mode.
Access to Resources	Full access to hardware and system resources, including memory, I/O devices, and CPU instructions.	Limited access to resources, with access controlled by the operating system.
Operations	Executes critical system operations, such as managing memory, handling interrupts, scheduling tasks, and controlling hardware.	Executes user-level applications and processes, such as word processors, web browsers, games, etc.
Protection	Protects sensitive system resources from unauthorized access or modification by user applications.	Provides a protection boundary to prevent user applications from directly accessing critical system resources.
Controlled Access	Tightly controlled to prevent unauthorized access and ensure system stability and security.	Transition between user mode and kernel mode is controlled by the operating system to maintain system integrity.
Isolation	N/A	Provides isolation between user applications, preventing interference between processes without proper permissions.

Q3.List five services provided by an operating system and explain how each creates convenience for users. In which cases it would be impossible for user level programs to provide these services?

Answer:

1. Program execution - the operating system will schedule on behalf of the user. This service could not be handled by the user because you need access to the hardware.

2. I/O operations - This makes it easy for users to access I/O streams. This means the user does not need to know the physical access of data in the machine. If there were not interface provided the user could not do this on their own.

3. File-system manipulation - This means the user does not need to worry about accessing and updating the file system table. Such access is best handled by the operating system because of this complexity.

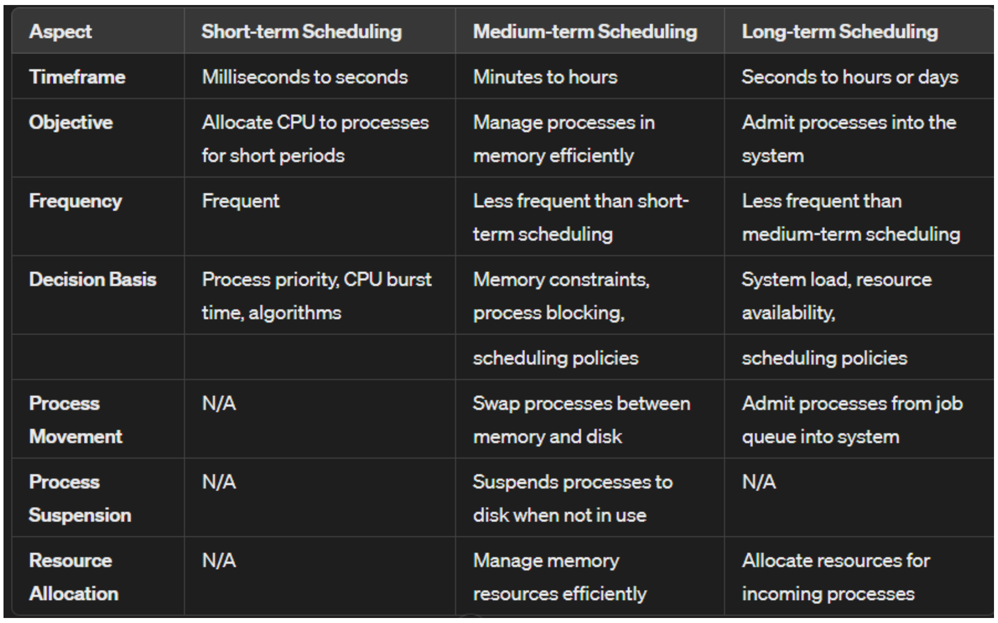
4. Communications - in the case of memory mapping this is extremely beneficial for the OS to handle access and control to memory regions. The user could not in this case access such a system to share the map.

5. Error detection - If there is some error on one of the lower levels the user is notified so that they can take action. If there is no memory left on the heap for instance. The user could not do this because it is simply too much work for the user.

Q4.Why do some systems store the operating system in firmware while others store it on disc?(2)

A:For certain devices, such as handheld PDAs and cellular telephones, a disk with a file system may not be available for the device. In this situation, the operating system must be stored in firmware.

Q5:Describe the difference between Short term,medium term and long term scheduling.



Aspect	Short-term Scheduling	Medium-term Scheduling	Long-term Scheduling
Timeframe	Milliseconds to seconds	Minutes to hours	Seconds to hours or days
Objective	Allocate CPU to processes for short periods	Manage processes in memory efficiently	Admit processes into the system
Frequency	Frequent	Less frequent than short-term scheduling	Less frequent than medium-term scheduling
Decision Basis	Process priority, CPU burst time, algorithms	Memory constraints, process blocking,	System load, resource availability,
		scheduling policies	scheduling policies
Process Movement	N/A	Swap processes between memory and disk	Admit processes from job queue into system
Process Suspension	N/A	Suspends processes to disk when not in use	N/A
Resource Allocation	N/A	Manage memory resources efficiently	Allocate resources for incoming processes

Difference between Short-Term, Medium term, and Long-Term Schedulers:

Basis	**Short-Term Scheduler**	**Medium-term Scheduler**	**Long-Term Scheduler**
1. Alternate Name	It is also called a CPU scheduler.	It is also called a process swapping scheduler.	It is also called a job scheduler.

2. Degree in programming	It provides lesser control over the degree of multiprogramming.	It reduces the control over the degree of multiprogramming.	It controls the degree of multiprogramming.
3. Speed	The speed of the short-term scheduler is very fast.	Speed of medium scheduler between the short-term and long-term scheduler	The speed of a long-term term scheduler is more than medium-term scheduler.
4. Usage in time- sharing system sharing system	It is minimal in the time-sharing system.	It is a part of the time-sharing system.	It is almost absent or minimal in a sharing system.
5. Purpose	It selects the processes from among the process that is ready to execute.	It can reintroduce the from among the process into memory that executes and its execution can be continued.	It selects processes from the pool and loads them into memory for execution.
6. Process state	Process state is ready to running	Process state is not present	Process state is new to ready.

7. Selection of process	Select a new process for a CPU quite frequently.	Select that process, which is currently not need to load fully on RAM, so it swap it into swap partition.	Select a good process , mix of I/O bound and CPU

Q6:What is a process?What are the two essential parts of a process ?How is a process different from a program?
Process:
The term process (Job) refers to program code that has been loaded into a computer's memory so that it can be executed by the central processing unit (CPU). A process can be described as an instance of a program running on a computer or as an entity that can be assigned to and executed on a processor. A program becomes a process when loaded into memory and thus is an active entity.

The two essential elements of a process are inputs and outputs.

Difference Between Program and Process

Program	**Process**
Program contains a set of instructions designed to complete a specific task.	Process is an instance of an executing program.
Program is a passive entity as it resides in the secondary memory.	Process is a active entity as it is created during execution and loaded into the main memory.

Program exists at a single place and continues to exist until it is deleted.	Process exists for a limited span of time as it gets terminated after the completion of task.
Program is a static entity.	Process is a dynamic entity.
Program does not have any resource requirement, it only requires memory space for storing the instructions.	Process has a high resource requirement, it needs resources like CPU, memory address, I/O during its lifetime.
Program does not have any control block.	Process has its own control block called Process Control Block.
Program has two logical components: code and data.	In addition to program data, a process also requires additional information required for the management and execution.
Program does not change itself.	Many processes may execute a single program. There program code may be the same but program data may be different. these are never same.
Program contains instructions	Process is a sequence of instruction execution.

Q7:Explain the importance of operating system in your own words and is it possible to communicate with computer without operating system?

It manages the computer's memory and processes, as well as all of its software and hardware. It also allows you to communicate with the computer without knowing how to speak the computer's language. Without an operating system, a computer is useless.
No, it's not possible to communicate with a computer without an operating system (OS).

MID 2(18-19)

Q1:Which of the following scheduling algorithms could result in starvation?justify your answer i)First come first serve ii)Shortest Job First iii)Round Robin iv)Priority

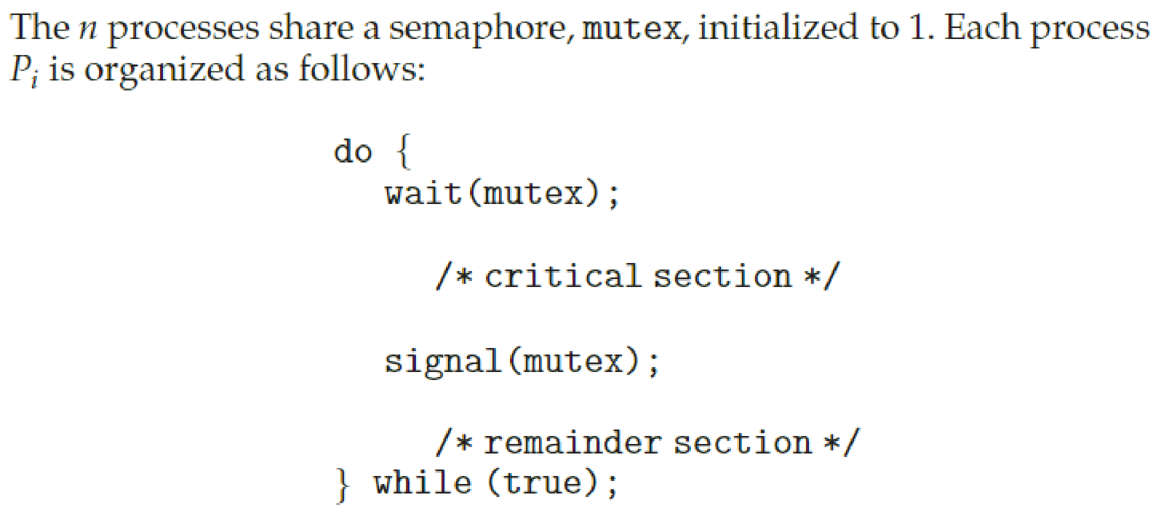
1.In First Come First Serve(FCFS) if a process with a very large Burst Time comes before other processes, the other process will have to wait for a long time but it is clear that other process will definitely get their chance to execute, so it will not suffer from starvation.
2.In Round Robin there is a fixed time quant and every process will get their chance to be executed, so no starvation is here.
3.In Priority based scheduling if higher priority process keep on coming then low priority process will suffer from starvation.
4.In Shortest Job First(SJF) if process with short process time keep on coming continuously then process with higher burst time will do wait and suffer from starvation.
Additional
5.In Shortest remaining time first(SRTF) process with shortest burst time will execute first because this process with high burst time may suffer from starvation.
Q3:What are the two differences between user level threads and kernel level threads?Under what circumstance is one type better than the others?

Sr.No	**User level thread**	**Kernel level thread**
1	User-level threads are faster to create and manage.	Kernel-level threads are slower to create and manage.

2	Implementation is by a thread library at the user level.	Operating system supports creation ofKernel threads.
3	User-level thread is generic and can run on any operating system.	Kernel-level thread is specific to the operating system.
4	Multi-threaded applications cannot take advantage of multiprocessing	Kernel routines themselves can be multithreaded.

In a multiprocessor environment, the kernel-level threads are better than user-level threads, because kernel-level threads can run on different processors simultaneously while user-level threads of a process will run on one processor only even if multiple processors are available.

Q4:Illustrate how a binary semaphore can be used to implement mutual exclusion among a processes?



The *n* processes share a semaphore, `mutex`, initialized to 1. Each process P_i is organized as follows:

```
do {
   wait(mutex);

      /* critical section */

   signal(mutex);

      /* remainder section */
} while (true);
```

Q5:Mention the chapter No and Title of each from Mid Syllabus?
CSE-2207 Mid-01 Syllabus
1. ch1_Introduction
2. ch2_Operating-System Structures
3. ch3_Processes
4. ch6_CPU Scheduling

Q6:Note regarding the preparation you have taken before attending the midterm exam.

1. Review the introduction, operating system structures, processes, and CPU scheduling.
2. Understand the purpose, evolution, and types of operating systems.
3. Study operating system components, process states, and scheduling algorithms.
4. Solve practice problems and create summary notes.
5. Form study groups and utilize online resources for clarification.
6. Ensure a thorough understanding of key concepts and algorithms.

Final Exam (2019-20)

1(a)What do you mean by an operating system.What are the 3 main purpose of an operating system?write short note on"Operating system is a resource allocator.

Operating system is a program that acts as an interface between users and hardware.

It is a resource allocator and provides a platform for the application program.

3 main purpose:Mid 1 18-19 Q1.

Operating system is an resource allocator: Operating system is a system software which acts as an interface between hardware and application programs. It interacts with various devices, memory (ram) , secondary memory (disk) and allows various programs to access these resources. Since operating system decides over which program will be using RAM / disk or any other device at a particular time. That's why operating system is called as resource allocator.

(b)List five services provided by an operating system and explain how each creates convenience for users?

Read any five from here

Services of Operating System

1.Program execution

2.Input Output Operations

3.Communication between Process

4.File Management

5.Memory Management

6.Process Management

7.Security and Privacy

8.Resource Management

9.User Interface

10.Networking

11.Error handling

12.Time Management

Program Execution

It is the Operating System that manages how a program is going to be executed. It loads the program into the memory after which it is executed. The order in which they are executed depends on the CPU Scheduling Algorithms. A few are FCFS, SJF, etc. When the program is in execution, the Operating System also handles deadlock i.e. no two processes come for execution at the same time. The Operating System is responsible for the smooth execution of both user and system programs. The Operating System utilizes various resources available for the efficient running of all types of functionalities.

Input Output Operations

Operating System manages the input-output operations and establishes communication between the user and device drivers. Device drivers are software that is associated with hardware that is being managed by the OS so that the sync between the devices works properly. It also provides access to input-output devices to a program when needed.

Communication between Processes

The Operating system manages the communication between processes. Communication between processes includes data transfer among them. If the processes are not on the same computer but connected through a computer network, then also their communication is managed by the Operating System itself.

File Management

The operating system helps in managing files also. If a program needs access to a file, it is the operating system that grants access. These permissions include read-only, read-write, etc. It also provides a platform for the user to create, and delete files. The Operating System is responsible for making decisions regarding the storage of all types of data or files, i.e, floppy disk/hard disk/pen drive, etc. The Operating System decides how the data should be manipulated and stored.

Memory Management

Let’s understand memory management by OS in simple way. Imagine a cricket team with limited number of player . The team manager (OS) decide whether the upcoming player will be in playing 11 ,playing 15 or will not be included in team , based on his performance . In the same way, OS first check whether the upcoming program fulfil all requirement to get memory space or not ,if all things good, it checks how much memory space will be sufficient for program and then load the program into memory at certain location. And thus , it prevents program from using unnecessary memory.

Process Management

Let’s understand the process management in unique way. Imagine, our kitchen stove as the (CPU) where all cooking(execution) is really happen and chef as the (OS) who uses kitchen-stove(CPU) to cook different dishes(program). The chef(OS) has to cook different dishes(programs) so he ensure that any particular dish(program) does not take long time(unnecessary time) and all dishes(programs) gets a chance to cooked(execution) .The chef(OS) basically scheduled time for all dishes(programs) to run kitchen(all the system) smoothly and thus cooked(execute) all the different dishes(programs) efficiently.

Security and Privacy

Security : OS keep our computer safe from an unauthorized user by adding security layer to it. Basically, Security is nothing but just a layer of protection which protect computer from bad guys like viruses and hackers. OS provide us defenses like firewalls and anti-virus software and ensure good safety of computer and personal information.

Privacy : OS give us facility to keep our essential information hidden like having a lock on our door, where only you can enter and other are not allowed . Basically , it respect our secrets and provide us facility to keep it safe.

Resource Management

System resources are shared between various processes. It is the Operating system that manages resource sharing. It also manages the CPU time among processes using CPU Scheduling Algorithms. It also helps in the memory management of the system. It also controls input-output devices. The OS also ensures the proper use of all the resources available by deciding which resource to be used by whom.

User Interface

User interface is essential and all operating systems provide it. Users either interface with the operating system through the command-line interface or graphical user interface or GUI. The command interpreter executes the next user-specified command.

A GUI offers the user a mouse-based window and menu system as an interface.

Networking

This service enables communication between devices on a network, such as connecting to the internet, sending and receiving data packets, and managing network connections.

Error Handling

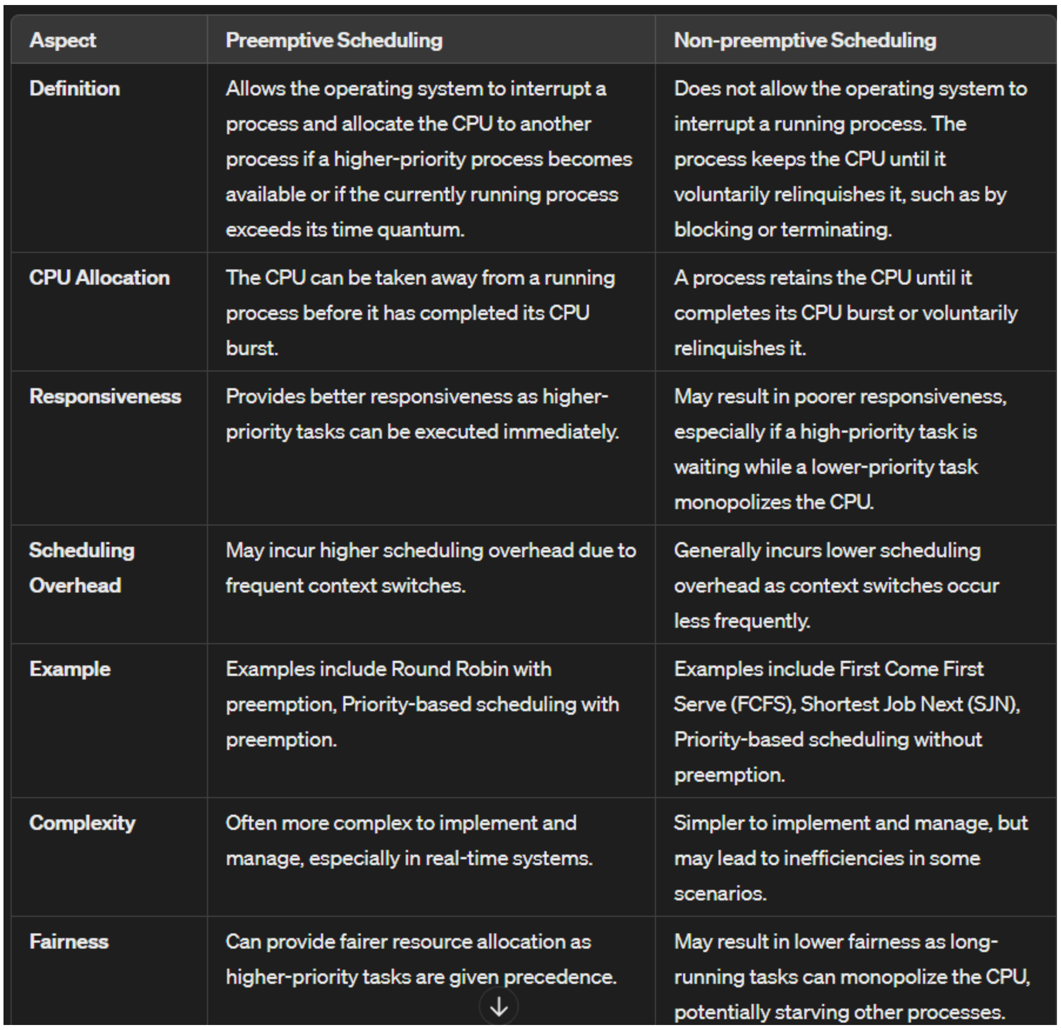
The Operating System also handles the error occurring in the CPU, in Input-Output devices, etc. It also ensures that an error does not occur frequently and fixes the errors. It also prevents the process from coming to a deadlock. It also looks for any type of error or bugs that can occur while any task. The well-secured OS sometimes also acts as a countermeasure for preventing any sort of breach of the Computer System from any external source and probably handling them.

Time Management

Imagine traffic light as (OS), which indicates all the cars(programs) whether it should be stop(red)=>(simple queue) , start(yellow)=>(ready queue),move(green)=>(under execution) and this light (control) changes after a certain interval of time at each side of the road(computer system) so that the cars(program) from all side of road move smoothly without traffic.

3(a)Mid 2 (2018-19)

c) Explain the difference between preemptive and non preemptive scheduling.



Aspect	Preemptive Scheduling	Non-preemptive Scheduling
Definition	Allows the operating system to interrupt a process and allocate the CPU to another process if a higher-priority process becomes available or if the currently running process exceeds its time quantum.	Does not allow the operating system to interrupt a running process. The process keeps the CPU until it voluntarily relinquishes it, such as by blocking or terminating.
CPU Allocation	The CPU can be taken away from a running process before it has completed its CPU burst.	A process retains the CPU until it completes its CPU burst or voluntarily relinquishes it.
Responsiveness	Provides better responsiveness as higher-priority tasks can be executed immediately.	May result in poorer responsiveness, especially if a high-priority task is waiting while a lower-priority task monopolizes the CPU.
Scheduling Overhead	May incur higher scheduling overhead due to frequent context switches.	Generally incurs lower scheduling overhead as context switches occur less frequently.
Example	Examples include Round Robin with preemption, Priority-based scheduling with preemption.	Examples include First Come First Serve (FCFS), Shortest Job Next (SJN), Priority-based scheduling without preemption.
Complexity	Often more complex to implement and manage, especially in real-time systems.	Simpler to implement and manage, but may lead to inefficiencies in some scenarios.
Fairness	Can provide fairer resource allocation as higher-priority tasks are given precedence.	May result in lower fairness as long-running tasks can monopolize the CPU, potentially starving other processes.

Comparison Chart

Parameter	PREEMPTIVE SCHEDULING	NON-PREEMPTIVE SCHEDULING
Basic	In this resources(CPU Cycle) are allocated to a process for a limited time.	Once resources(CPU Cycle) are allocated to a process, the process holds it till it completes its burst time or switches to waiting state.

Interrupt	Process can be interrupted in between.	Process can not be interrupted until it terminates itself or its time is up.
Starvation	If a process having high priority frequently arrives in the ready queue, a low priority process may starve.	If a process with a long burst time is running CPU, then later coming process with less CPU burst time may starve.
Overhead	It has overheads of scheduling the processes.	It does not have overheads.
Flexibility	flexible	rigid
Cost	cost associated	no cost associated
CPU Utilization	In preemptive scheduling, CPU utilization is high.	It is low in non preemptive scheduling.
Waiting Time	Preemptive scheduling waiting time is less.	Non-preemptive scheduling waiting time is high.

Response Time	Preemptive scheduling response time is less.	Non-preemptive scheduling response time is high.
Decision making	Decisions are made by the scheduler and are based on priority and time slice allocation	Decisions are made by the process itself and the OS just follows the process's instructions
Process control	The OS has greater control over the scheduling of processes	The OS has less control over the scheduling of processes
Overhead	Higher overhead due to frequent context switching	Lower overhead since context switching is less frequent
Examples	Examples of preemptive scheduling are Round Robin and Shortest Remaining Time First.	Examples of non-preemptive scheduling are First Come First Serve and Shortest Job First.

Mid 1 19-20

1. **Hand Written**
2.Banker's algorithm (porayni)

Final exam 18-19

**1. a).We have stressed the need for an operating system to make efficient use of the computing
hardware. When is it appropriate for the operating system to forsake this principle and to
"waste" resources? Why is such a system not really wasteful?**
b) Why do some systems store the operating system in firmware, while others store it on disk?
**c)How does the distinction between kernel mode and user mode function as a rudimentary form
of protection (security) system?**
c) List five services provided by an operating system and explain how each creates convenience for users.
ANS:Solved in 19-20 Final

2. a) Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU's execution load.
i) How does the CPU interface with the device to coordinate the transfer?
**ii) The CPU is allowed to execute other programs while the DMA controller is transferring
data. Does this process interfere with the execution of the user programs? If so, describe what forms Of interference are caused.**
ANS:
i) CPU Interface with DMA for Data Transfer Coordination

1. Initialization: CPU sets up DMA controller with transfer details.
2. Permission Granted: CPU grants DMA controller access to system bus.
3. Initiation of Transfer: CPU commands DMA controller to start transfer.
4. Data Transfer: DMA controller moves data between device and memory independently.
5. Notification: DMA controller interrupts CPU upon transfer completion.
6. Completion Handling: CPU processes transferred data as needed.

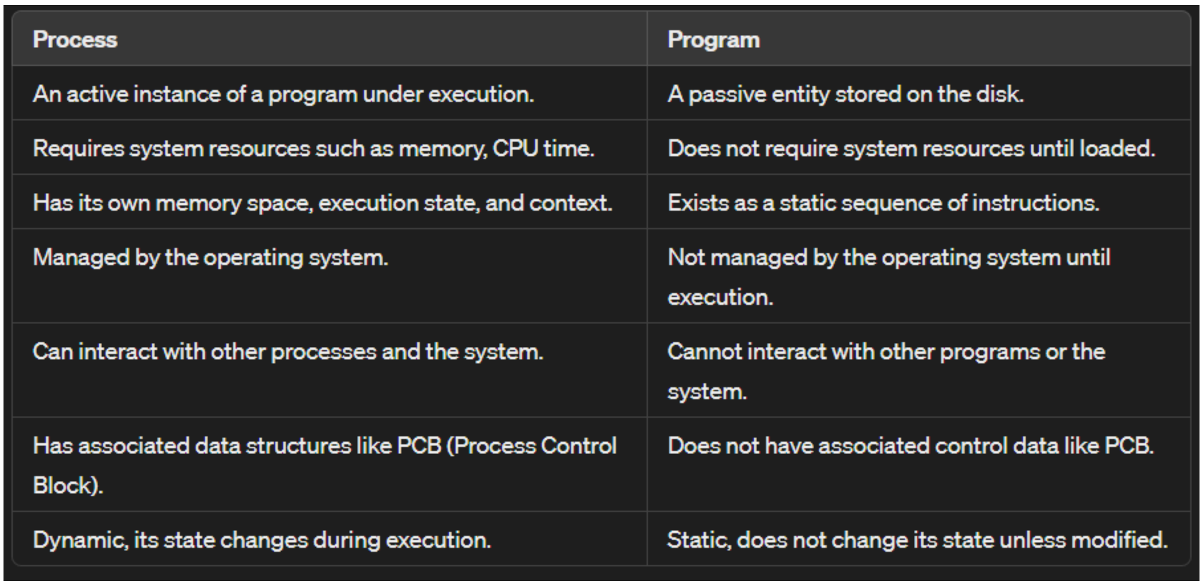
ii) Interference with User Programs

- Bus Contention: DMA monopolises system bus, causing delays for CPU and other devices.
- Memory Bandwidth: DMA consumes memory bandwidth, slowing down user program access.
- Interrupt Overhead: CPU handles interrupts from DMA, adding processing overhead.
- Priority Handling: DMA may delay user programs to prioritise transfers, affecting responsiveness.

b) What are the two essential parts of a process? How is a process different from a program?
Two essential parts of a process are:

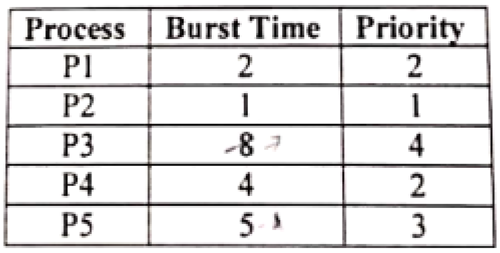
Program: The program is the set of instructions written by the programmer to perform a specific task. It resides on the disk as a passive entity and does not carry out any functions. It's a static part of the process.

Process Control Block (PCB): The PCB is a data structure maintained by the operating system for each process. It contains various pieces of information about the process, including its current state, program counter, CPU registers, memory allocation, process ID, priority, etc. The PCB represents the dynamic part of the process, as it changes with the state of the process during its execution.



Process	Program
An active instance of a program under execution.	A passive entity stored on the disk.
Requires system resources such as memory, CPU time.	Does not require system resources until loaded.
Has its own memory space, execution state, and context.	Exists as a static sequence of instructions.
Managed by the operating system.	Not managed by the operating system until execution.
Can interact with other processes and the system.	Cannot interact with other programs or the system.
Has associated data structures like PCB (Process Control Block).	Does not have associated control data like PCB.
Dynamic, its state changes during execution.	Static, does not change its state unless modified.

c) Consider the following set of processes. with the length of the CPU burst given in milliseconds:



Process	Burst Time	Priority
P1	2	2
P2	1	1
P3	8	4
P4	4	2
P5	5	3

1

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5 all at time O.
i) Draw four Gantt charts that illustrate the execution of these processes using the
following scheduling algorithms: FCFS, SJF, non preemptive priority (a large priority
number implies a higher priority), and RR (quantum = 2)

ii) What is the turnaround time of each process for each of the scheduling algorithms
iii) What is the waiting time of each process for each of these scheduling algorithms?
iv) Which of the algorithms results in the minimum average waiting time (over all
processes)?

i) Gantt Charts
1. FCFS (First Come First Serve):

```
| P1 | P2 | P3 | P4 | P5 |
0    2    3    11   15   20
```

2. SJF (Shortest Job First):

```
| P2 | P1 | P4 | P5 | P3 |
0    1    3    7    12   17
```

3. Non-preemptive Priority (higher priority first):

```
| P1 | P2 | P4 | P5 | P3 |
0    2    3    7    12   20
```

4. RR (Round Robin with Quantum = 2):

```
| P1 | P2 | P3 | P4 | P5 | P1 | P3 | P5 | P3 |
0    2    3    5    7    9    11   13   15   17
```

ii) Turnaround Time
The turnaround time for each process is the time taken from the arrival of the process to its completion.

- FCFS: P1 = 20, P2 = 1, P3 = 20, P4 = 16, P5 = 19
- SJF: P1 = 18, P2 = 1, P3 = 19, P4 = 15, P5 = 14
- Non-preemptive Priority: P1 = 20, P2 = 1, P3 = 20, P4 = 16, P5 = 19
- RR: P1 = 18, P2 = 7, P3 = 17, P4 = 13, P5 = 17

iii) Waiting Time
The waiting time for each process is the total time it spends waiting in the ready queue.

- FCFS: P1 = 0, P2 = 0, P3 = 6, P4 = 3, P5 = 7
- SJF: P1 = 0, P2 = 0, P3 = 1, P4 = 2, P5 = 7
- Non-preemptive Priority: P1 = 0, P2 = 0, P3 = 6, P4 = 3, P5 = 7
- RR: P1 = 7, P2 = 0, P3 = 9, P4 = 5, P5 = 7

iv) Minimum Average Waiting Time

To find the algorithm with the minimum average waiting time, we can calculate the average waiting time for each algorithm and then compare.

- FCFS: (0+0+6+3+7)/5 = 3.2
- SJF: (0+0+1+2+7)/5 = 2
- Non-preemptive Priority: (0+0+6+3+7)/5 = 3.2
- RR: (7+0+9+5+7)/5 = 5.6

So, SJF has the minimum average waiting time over all processes.