CPU Scheduling in Operating Systems

Last Updated : 24 Jun, 2024

Scheduling of processes/work is done to finish the work on time. **CPU Scheduling** is a process that allows one process to use the CPU while another process is delayed (in standby) due to unavailability of any resources such as I / O etc, thus making full use of the CPU. The purpose of CPU Scheduling is to make the system more efficient, faster, and fairer.

CPU scheduling is a key part of how an operating system works. It decides which task (or process) the CPU should work on at any given time. This is important because a CPU can only handle one task at a time, but there are usually many tasks that need to be processed. In this article, we are going to discuss CPU scheduling in detail.

Tutorial on CPU Scheduling Algorithms in Operating System

Whenever the CPU becomes idle, the operating system must select one of the processes in the line ready for launch. The selection process is done by a temporary (CPU) scheduler. The Scheduler selects between memory processes ready to launch and assigns the CPU to one of them.

Table of Content

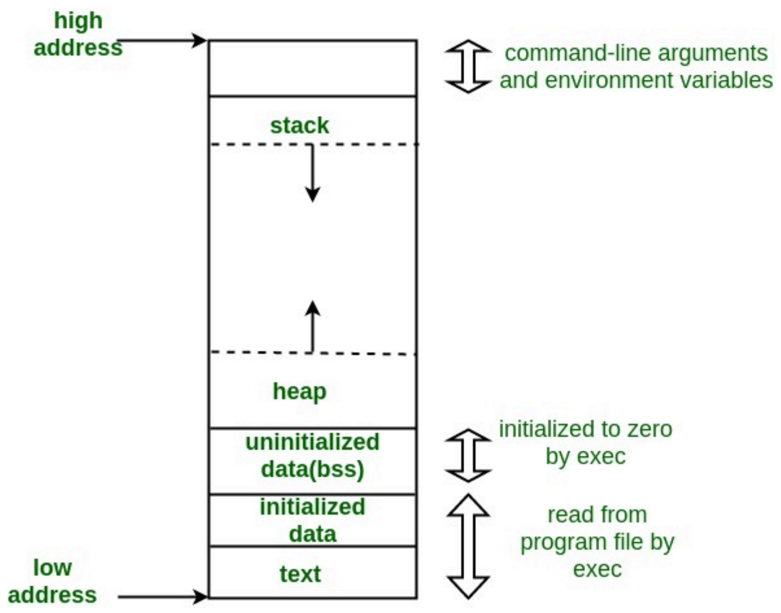
What is a Process?

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.

How is Process Memory Used For Efficient Operation?

The process memory is divided into four sections for efficient operation:

- The **text category** is composed of integrated program code, which is read from fixed storage when the program is launched.
- The **data class** is made up of global and static variables, distributed and executed before the main action.
- Heap is used for flexible, or dynamic memory allocation and is managed by calls to new, delete, malloc, free, etc.
- The stack is used for local variables. The space in the stack is reserved for local variables when it is announced.


high address
command-line arguments and environment variables
stack
heap
uninitialized data(bss)
initialized to zero by exec
initialized data
read from program file by exec
text
low address

To know further, you can refer to our detailed article on **States of a Process in Operating system**.

What is Process Scheduling?

Process Scheduling is the process of the process manager handling the removal of an active process from the CPU and selecting another process based on a specific strategy.

Process Scheduling is an integral part of Multi-programming applications. Such operating systems allow more than one process to be loaded into usable memory at a time and the loaded shared CPU process uses repetition time.

There are three types of process schedulers:

- Long term or Job Scheduler
- Short term or CPU Scheduler
- Medium-term Scheduler

Why do We Need to Schedule Processes?

- **Scheduling** is important in many different computer environments. One of the most important areas is scheduling which programs will work on the CPU. This task is handled by the Operating System (OS) of the computer and there are many different ways in which we can choose to configure programs.

- **Process Scheduling** allows the OS to allocate CPU time for each process. Another important reason to use a process scheduling system is that it keeps the CPU busy at all times. This allows you to get less response time for programs.
- Considering that there may be hundreds of programs that need to work, the OS must launch the program, stop it, switch to another program, etc. The way the OS configures the system to run another in the CPU is called "context switching". If the OS keeps context-switching programs in and out of the provided CPUs, it can give the user a tricky idea that he or she can run any programs he or she wants to run, all at once.
- So now that we know we can run 1 program at a given CPU, and we know we can change the operating system and remove another one using the context switch, how do we choose which programs we need. run, and with what program?
- That's where **scheduling** comes in! First, you determine the metrics, saying something like "the amount of time until the end". We will define this metric as "the time interval between which a function enters the system until it is completed". Second, you decide on a metrics that reduces metrics. We want our tasks to end as soon as possible.

What is The Need For CPU Scheduling Algorithm?

CPU scheduling is the process of deciding which process will own the CPU to use while another process is suspended. The main function of the CPU scheduling is to ensure that whenever the CPU remains idle, the OS has at least selected one of the processes available in the ready-to-use line.

In Multiprogramming, if the long-term scheduler selects multiple I / O binding processes then most of the time, the CPU remains an idle. The function of an effective program is to improve resource utilization.

If most operating systems change their status from performance to waiting then there may always be a chance of failure in the system. So in order to minimize this excess, the OS needs to schedule tasks in order to make full use of the CPU and avoid the possibility of deadlock.

Objectives of Process Scheduling Algorithm

- Utilization of CPU at maximum level. **Keep 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.

Terminologies Used in CPU Scheduling

- **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

Things to Take Care While Designing a CPU Scheduling Algorithm

Different **CPU Scheduling algorithms** have different structures and the choice of a particular algorithm depends on a variety of factors. Many conditions have been raised to compare CPU scheduling algorithms.

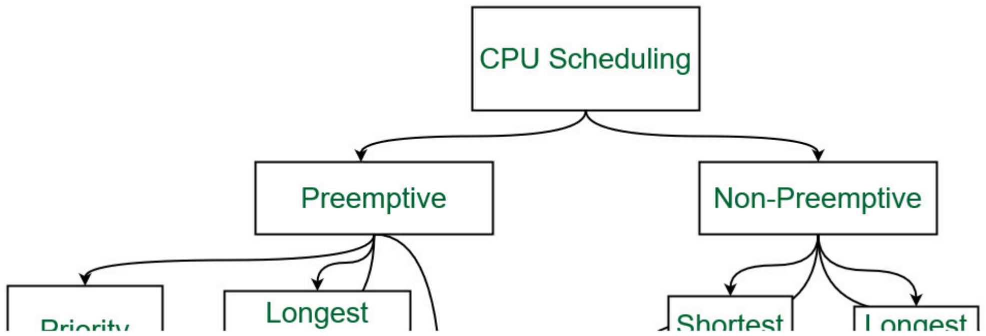
The criteria include the following:

- **CPU Utilization:** The main purpose of any CPU algorithm is to keep the CPU as busy as possible. Theoretically, CPU usage can range from 0 to 100 but in a real-time system, it varies from 40 to 90 percent depending on the system load.
- **Throughput:** The average CPU performance is the number of processes performed and completed during each unit. This is called throughput. The output may vary depending on the length or duration of the processes.
- **Turn Round Time:** For a particular process, the important conditions are how long it takes to perform that process. The time elapsed from the time of process delivery to the time of completion is known as the conversion time. Conversion time is the amount of time spent waiting for memory access, waiting in line, using CPU, and waiting for I / O.
- **Waiting Time:** The Scheduling algorithm does not affect the time required to complete the process once it has started performing. It only affects the waiting time of the process i.e. the time spent in the waiting process in the ready queue.
- **Response Time:** In a collaborative system, turn around time is not the best option. The process may produce something early and continue to computing the new results while the previous results are released to the user. Therefore another method is the time taken in the submission of the application process until the first response is issued. This measure is called response 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.



Different types of CPU Scheduling Algorithms

Let us now learn about these CPU scheduling algorithms in operating systems one by one:

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.

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 SJF

- 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

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Shortest Job First.

3. Longest Job First(LJF)

Longest Job First(LJF) scheduling process is just opposite of shortest job first (SJF), as the name suggests this algorithm is based upon the fact that the process with the largest burst time is processed first. Longest Job First is non-preemptive in nature.

Characteristics of LJF

- Among all the processes waiting in a waiting queue, CPU is always assigned to the process having largest burst time.
- If two processes have the same burst time then the tie is broken using FCFS i.e. the process that arrived first is processed first.
- LJF CPU Scheduling can be of both preemptive and non-preemptive types.

Advantages of LJF

- No other task can schedule until the longest job or process executes completely.
- All the jobs or processes finish at the same time approximately.

Disadvantages of LJF

- Generally, the LJF algorithm gives a very high average waiting time and average turn-around time for a given set of processes.
- This may lead to convoy effect.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on the Longest job first scheduling.

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.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Priority Preemptive Scheduling algorithm.

5. Round Robin

Round Robin is a CPU scheduling algorithm where each process is cyclically assigned a fixed time slot. It is the preemptive version of First come First Serve CPU Scheduling algorithm. Round Robin CPU Algorithm generally focuses on Time Sharing technique.

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.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on the Round robin Scheduling algorithm.

6. Shortest Remaining Time First(SRTF)

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 SRTF

- 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.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on the shortest remaining time first.

7. Longest Remaining Time First(LRTF)

The longest remaining time first is a preemptive version of the longest job first scheduling algorithm. This scheduling algorithm is used by the operating system to program incoming processes for use in a systematic way. This algorithm schedules those processes first which have the longest processing time remaining for completion.

Characteristics of LRTF

- Among all the processes waiting in a waiting queue, the CPU is always assigned to the process having the largest burst time.
- If two processes have the same burst time then the tie is broken using FCFS i.e. the process that arrived first is processed first.
- LRTF CPU Scheduling can be of both preemptive and non-preemptive.

- No other process can execute until the longest task executes completely.
- All the jobs or processes finish at the same time approximately.

Advantages of LRTF

- Maximizes Throughput for Long Processes.
- Reduces Context Switching.
- Simplicity in Implementation.

Disadvantages of LRTF

- This algorithm gives a very high average waiting time and average turn-around time for a given set of processes.
- This may lead to a convoy effect.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on the longest remaining time first.

8. Highest Response Ratio Next(HRRN)

Highest Response Ratio Next is a non-preemptive CPU Scheduling algorithm and it is considered as one of the most optimal scheduling algorithms. The name itself states that we need to find the response ratio of all available processes and select the one with the highest Response Ratio. A process once selected will run till completion.

Characteristics of HRRN

- The **criteria** for HRRN is **Response Ratio,** and the **mode** is **Non-Preemptive.**
- HRRN is considered as the modification of Shortest Job First to reduce the problem of starvation.
- In comparison with SJF, during the HRRN scheduling algorithm, the CPU is allotted to the next process which has the **highest response ratio** and not to the process having less burst time.

Response Ratio = (W + S)/S

Here, ***W*** *is the waiting time of the process so far and* ***S*** *is the Burst time of the process.*

Advantages of HRRN

- HRRN Scheduling algorithm generally gives better performance than the shortest job first Scheduling.
- There is a reduction in waiting time for longer jobs and also it encourages shorter jobs.

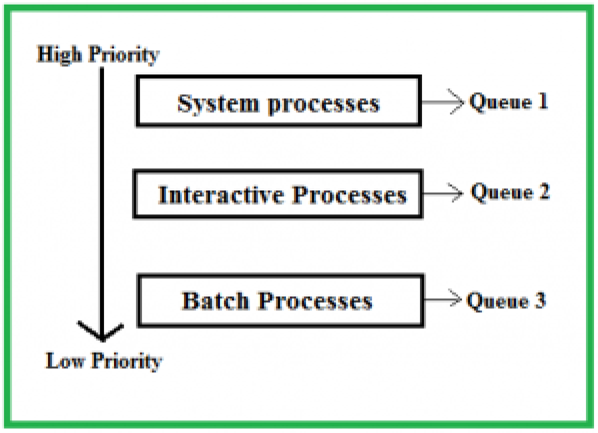
Disadvantages of HRRN

- The implementation of HRRN scheduling is not possible as it is not possible to know the burst time of every job in advance.
- In this scheduling, there may occur an overload on the CPU.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Highest Response Ratio Next.

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

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Multilevel Queue Scheduling.

10. Multilevel Feedback Queue Scheduling

Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling is like **Multilevel Queue Scheduling** but in this process can move between the queues. And thus, much more efficient than multilevel queue scheduling.

Characteristics of Multilevel Feedback Queue Scheduling

- In a multilevel queue-scheduling algorithm, processes are permanently assigned to a queue on entry to the system, and processes are not allowed to move between queues.
- As the processes are permanently assigned to the queue, this setup has the advantage of low scheduling overhead,

- But on the other hand disadvantage of being inflexible.

Advantages of Multilevel feedback Queue Scheduling

- It is more flexible
- It allows different processes to move between different queues

Disadvantages of Multilevel Feedback Queue Scheduling

- It also produces CPU overheads
- It is the most complex algorithm.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Multilevel Feedback Queue Scheduling.

Comparison Between Various CPU Scheduling Algorithms

Here is a brief comparison between different CPU scheduling algorithms:

Algorithm	Allocation is	Complexity	Average waiting time (AWT)	Preemption
FCFS	According to the arrival time of the processes, the CPU is allocated.	Simple and easy to implement	Large.	No
SJF	Based on the lowest	More complex than FCFS	Smaller than FCFS	No

Algorithm	Allocation is	Complexity	Average waiting time (AWT)	Preemption
	CPU burst time (BT).			
LJFS	Based on the highest CPU burst time (BT)	More complex than FCFS	Depending on some measures e.g., arrival time, process size, etc.	No
LRTF	Same as LJFS the allocation of the CPU is based on the highest CPU burst time (BT). But it is preemptive	More complex than FCFS	Depending on some measures e.g., arrival time, process size, etc.	Yes
SRTF	Same as SJF the allocation of the CPU is based on the lowest CPU burst time (BT).	More complex than FCFS	Depending on some measures e.g., arrival time, process size, etc	Yes

Algorithm	Allocation is	Complexity	Average waiting time (AWT)	Preemption
	But it is preemptive.			
RR	According to the order of the process arrives with fixed time quantum (TQ)	The complexity depends on Time Quantum size	Large as compared to SJF and Priority scheduling.	Yes
Priority Pre-emptive	According to the priority. The bigger priority task executes first	This type is less complex	Smaller than FCFS	Yes
Priority non-preemptive	According to the priority with monitoring the new incoming higher priority jobs	This type is less complex than Priority preemptive	Preemptive Smaller than FCFS	No

Algorithm	Allocation is	Complexity	Average waiting time (AWT)	Preemption
MLQ	According to the process that resides in the bigger queue priority	More complex than the priority scheduling algorithms	Smaller than FCFS	No
MFLQ	According to the process of a bigger priority queue.	It is the most Complex but its complexity rate depends on the TQ size	Smaller than all scheduling types in many cases	No

Exercise:

- Consider a system which requires 40-time units of burst time. The Multilevel feedback queue scheduling is used and time quantum is 2 unit for the top queue and is incremented by 5 unit at each level, then in what queue the process will terminate the execution?
- Which of the following is false about SJF? S1: It causes minimum average waiting time S2: It can cause starvation (A) Only S1 (B) Only S2 (C) Both S1 and S2 (D) Neither S1 nor S2 Answer (D) S1 is true SJF will always give minimum average waiting time. S2 is true SJF can cause starvation.
- Consider the following table of arrival time and burst time for three processes P0, P1 and P2. (GATE-CS-2011)

Process	Arrival time	Burst Time
P0	0 ms	9 ms
P1	1 ms	4 ms
P2	2 ms	9 ms

- The pre-emptive shortest job first scheduling algorithm is used. Scheduling is carried out only at arrival or completion of processes. What is the average waiting time for the three processes? (A) 5.0 ms (B) 4.33 ms (C) 6.33 (D) 7.33 Solution : Answer: – (A) Process P0 is allocated processor at 0 ms as there is no other process in the ready queue. P0 is preempted after 1 ms as P1 arrives at 1 ms and burst time for P1 is less than remaining time of P0. P1 runs for 4ms. P2 arrived at 2 ms but P1 continued as burst time of P2 is longer than P1. After P1 completes, P0 is scheduled again as the remaining time for P0 is less than the burst time of P2. P0 waits for 4 ms, P1 waits for 0 ms and P2 waits for 11 ms. So average waiting time is (0+4+11)/3 = 5.

- Consider the following set of processes, with the arrival times and the CPU-burst times given in milliseconds (GATE-CS-2004)

Process	Arrival time	Burst Time
P1	0 ms	5 ms
P2	1 ms	3 ms

P3	2 ms	3 ms
P4	4 ms	1 ms

- What is the average turnaround time for these processes with the preemptive shortest remaining processing time first (SRPT) algorithm ? (A) 5.50 (B) 5.75 (C) 6.00 (D) 6.25 Answer (A) Solution: The following is Gantt Chart of execution

P1	P2	P4	P3	P1
1	4	5	8	12

- Turn Around Time = Completion Time – Arrival Time Avg Turn Around Time = (12 + 3 + 6+ 1)/4 = 5.50
- An operating system uses the Shortest Remaining Time First (SRTF) process scheduling algorithm. Consider the arrival times and execution times for the following processes:

Process	Arrival time	Burst Time
P1	20 ms	0 ms
P2	25 ms	15 ms
P3	10 ms	30 ms
P4	15 ms	45 ms

- What is the total waiting time for process P2? (A) 5 (B) 15 (C) 40 (D) 55 Answer (B) At time 0, P1 is the only process, P1 runs for 15 time units. At time 15, P2 arrives, but P1 has the shortest remaining time. So P1 continues for 5 more time units. At time 20, P2 is the only process. So it runs for 10 time units At time 30, P3 is the shortest remaining time process. So it runs for 10 time units At time 40, P2 runs as it is the only process. P2 runs for 5 time units. At time 45, P3 arrives, but P2 has the shortest remaining time. So P2 continues for 10 more time units. P2 completes its execution at time 55

  Total waiting time for P2
  = Completion time – (Arrival time + Execution time)
  = 55 – (15 + 25)
  = 15

Conclusion

In conclusion, CPU scheduling is a fundamental component of operating systems, playing a crucial role in managing how processes are allocated CPU time. Effective CPU scheduling ensures that the system runs efficiently, maintains fairness among processes, and meets various performance criteria. Different scheduling algorithms, such as First-Come, First-Served (FCFS), Shortest Job Next (SJN), Priority Scheduling, and Round Robin (RR), each have their own strengths and are suited to different types of workloads and system requirements.

Frequently Asked Questions on CPU Sheduling – FAQs

Why is CPU scheduling important?

It's important because it helps the CPU work efficiently, ensures all tasks get a fair chance to run, and improves overall system *performance.*

How does Shortest Job Next (SJN) work?

SJN schedules the task with the shortest execution time first, aiming to reduce the average waiting time for all tasks.

How does Round Robin (RR) scheduling work?

In RR scheduling, each task gets a small, fixed amount of CPU time called a time slice. The CPU cycles through all tasks, giving each one a turn.

What is context switching?

Context switching is the process of saving the state of a currently running task and loading the state of the next task. This allows the CPU to switch between tasks

GeeksforGeeks

Previous Article

Preemptive and Non-Preemptive Scheduling

Next Article

CPU Scheduling Criteria

Similar Reads

Operating Systems | CPU Scheduling | Question 1

Consider three processes (process id 0, 1, 2 respectively) with compute time bursts 2, 4 and 8 time units. All processes arrive at time zero. Consid...

1 min read

Operating Systems | CPU Scheduling | Question 2

Consider three processes, all arriving at time zero, with total execution time of 10, 20 and 30 units, respectively. Each process spends the first 20% of...

2 min read

Operating Systems | CPU Scheduling | Question 3

Consider three CPU-intensive processes, which require 10, 20 and 30 time units and arrive at times 0, 2 and 6, respectively. How many context...

1 min read

Operating Systems | CPU Scheduling | Question 4

Which of the following process scheduling algorithm may lead to starvation (A) FIFO (B) Round Robin (C) Shortest Job Next (D) None of the...

1 min read

Operating Systems | CPU Scheduling | Question 5

If the quantum time of round robin algorithm is very large, then it is equivalent to: (A) First in first out (B) Shortest Job Next (C) Lottery...

1 min read

View More Articles

Article Tags : Android Operating Systems Algorithms-Graph Traversals C++-Misc C++

+14 More

Practice Tags : Functions

DSA
Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Data Science & ML
Data Science With Python
Data Science For Beginner
Machine Learning Tutorial
ML Maths
Data Visualisation Tutorial
Pandas Tutorial
NumPy Tutorial
NLP Tutorial
Deep Learning Tutorial

Web Technologies
HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tutorial
Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science
Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths
Software Development
Software Testing

DevOps
Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design
High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

Inteview Preparation
Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

School Subjects
Mathematics
Physics
Chemistry

GeeksforGeeks Videos
DSA
Python
Java