Unit 1:

Introduction to operating System:

1.1)Introduction to Operating System:

An **Operating System (OS)** is a software that acts as an interface between computer hardware components and the user. Every computer system must have at least one operating system to run other programs. Applications like Browsers, MS Office, Notepad Games, etc. need some environment to run and perform its tasks. The OS helps you to communicate with the computer without knowing how to speak the computer's language. It is not possible for the user to use any computer or mobile device without having an operating system.

1.2)Types of Operating System (OS)

Following are the popular types of OS (Operating System):

- Batch Operating System
- Multitasking/Time Sharing OS
- Multiprocessing OS
- Real Time OS
- Distributed OS
- Network OS
- Mobile OS

i)Batch Operating System

Some computer processes are very lengthy and time-consuming. To speed the same process, a job with a similar type of needs are batched together and run as a group.

The user of a batch operating system never directly interacts with the computer. In this type of OS, every user prepares his or her job on an offline device like a punch card and submit it to the computer operator.

ii)Multi-Tasking/Time-sharing Operating systems

Time-sharing operating system enables people located at a different terminal(shell) to use a single computer system at the same time. The processor time (CPU) which is shared among multiple users is termed as time sharing.

iii)Real time OS

A real time operating system time interval to process and respond to inputs is very small. Examples: Military Software Systems, Space Software Systems are the Real time OS example.

iv)Distributed Operating System

Distributed systems use many processors located in different machines to provide very fast computation to its users.

v)Network Operating System

Network Operating System runs on a server. It provides the capability to serve to manage data, user, groups, security, application, and other networking functions.

vi)Mobile OS

Mobile operating systems are those OS which is especially that are designed to power smartphones, tablets, and wearables devices.

Some most famous mobile operating systems are Android and iOS, but others include BlackBerry, Web, and watchOS.

1.3)Functions of Operating System:

1. **Process management**:- Process management helps OS to create and delete processes. It also provides mechanisms for synchronization and communication among processes.

2. **Memory management:-** Memory management module performs the task of allocation and de-allocation of memory space to programs in need of this resources.

3. **File management**:- It manages all the file-related activities such as organization storage, retrieval, naming, sharing, and protection of files.

4. **Device Management**: Device management keeps tracks of all devices. This module also responsible for this task is known as the I/O controller. It also performs the task of allocation and de-allocation of the devices.

5. **I/O System Management:** One of the main objects of any OS is to hide the peculiarities of that hardware devices from the user.

6. **Secondary-Storage Management**: Systems have several levels of storage which includes primary storage, secondary storage, and cache storage. Instructions and data must be stored in primary storage or cache so that a running program can reference it.

7. **Security**:- Security module protects the data and information of a computer system against malware threat and authorized access.

8. **Command interpretation**: This module is interpreting commands given by the and acting system resources to process that commands.

9. **Networking:** A distributed system is a group of processors which do not share memory, hardware devices, or a clock. The processors communicate with one another through the network.

10. **Job accounting**: Keeping track of time & resource used by various job and users.

11. **Communication management**: Coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.

1.4)Features of Operating System (OS)

- Protected and supervisor mode
- Allows disk access and file systems Device drivers Networking Security
- Program Execution
- Memory management Virtual Memory Multitasking
- Handling I/O operations
- Manipulation of the file system
- Error Detection and handling
- Resource allocation
- Information and Resource Protection

Advantage of using Operating System

- Allows you to hide details of hardware by creating an abstraction
- Easy to use with a GUI
- Offers an environment in which a user may execute programs/applications
- The operating system must make sure that the computer system convenient to use

- Operating System acts as an intermediary among applications and the hardware components
- It provides the computer system resources with easy to use format
- Acts as an intermediator between all hardware's and software's of the system

Disadvantages of using Operating System

- If any issue occurs in OS, you may lose all the contents which have been stored in your system
- Operating system's software is quite expensive for small size organization which adds burden on them. Example Windows
- It is never entirely secure as a threat can occur at any time

1.5)OS Based on User interface(Command user interface and graphical user interface):

The OS provides a user interface (UI), an environment for the user to interact with the machine. The UI is either graphical or text-based. A **User interface (UI)** facilitates communication between an application and its user by acting as an intermediary between them. Each application including the operating system is provided with a specific UI for effective communication. The two basic function of a user interface of an application is to take the inputs from the user and to provide the output to the users. However, the types of inputs taken by the UI and the types of output provided by the UI may vary from one application to another.

A user interface of any operating system can be classified into one of the following types:

1. Graphical user interface (GUI)
2. Command line user interface (CLI)

a) Command line interface (CLI)

An OS also provides a method of interaction that is non-graphical, called the command line interface (CLI). This is a text-only service with feedback from the OS appearing in text. Using a CLI requires knowledge of the commands available on a particular machine.

Command line interface is a type of UI that enables the users to interact with the operating system by issuing some specific commands. In order to perform a task in this interface, the user needs to type a command at the command line. When the user enters the key, the command line interpreter received a command. The software program that is responsible for receiving and processing the commands issued by the user. After processing the command are called command line interpreter, the command line interpreter displays the command prompt again along with the output of the previous command issued by the user. The disadvantages of the CLI is that the user needs to remember a lot to interact with the operating system. Therefore these types of interface are not considered very friendly from the users perspective.

- MS DOS Commands
- Linux Terminal Commands

Advantages of using the command line include:

- a **faster** way to get tasks done
- it is **more flexible** than a GUI
- it uses **less** memory

Some games, such as Minecraft, also make use of a command line tool which allows the user to bypass the main interface and alter the game's mechanics or environment.

b) Graphical user interface (GUI)

The OS on most computers and smartphones provides an environment with tiles, icons and/or menus. This type of interface is called the graphical user interface (GUI) because the user interacts with images through a mouse, keyboard or touchscreen.

The graphical user interface is a type of GUI that enables the users to interact with the operating system by means of point-and-click operations. GUI contains several icons representing pictorial representation of the variables such as a file, directory, and device. The graphical icon provided in the UI can be manipulated by the users using a suitable pointing device such as a mouse, trackball, touch screen and light pen. The other input devices like keyboard can also be used to manipulate these graphical icons. GUIs are considered to be very user- friendly interface because each object is represented with a corresponding icon. Unlike the other UIs the users need not provide text command for executing tasks.

Some advantages of GUI based operating system

- The GUI interface is easy to understand and even the new users can operate on them on their own.
- The GUI interface visually acknowledges and confirms each type of activities performed by the users. For example when the user deletes a file in the Windows operating system, then the operating system asks for the confirmation before deleting it.
- The GUI interface enables the users to perform a number of tasks at the same time. This features of the operating system are also known as multitasking.

1.6)Describe OS based on Mode of User(Single user and Multiuser):

a)**Single User operating system**:This is the type of computer system which is mostly used in desktop and laptop.It generally provides a simple computer system,which facilates the running of a variety of software package as well as it allows users to develop and execute programs of their own.

b)**Multi-user operating system**:The multi-user operating system allows concurrent access by multiple users of a computer.It allows many different users to take advantage of the computer resources simultaneously.It is often used in business and offices where different users need to access the same resources,but these resources cannot be installed on every system.

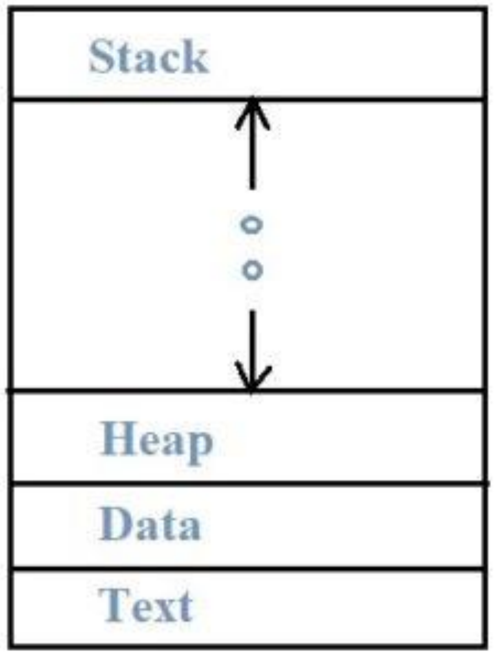
Unit 2: Process and process scheduling

1) Introduce Process, Program and process life cycle:

Process

A process is an instance of a program in execution. The execution of a process must progress in a sequential fashion.

To put it simply, we write our computer program in a text file and when we execute this program, it becomes a process that meets all the tasks mentioned in the program.
When a program is loaded into memory and it becomes a process, it can be divided into four segments- **stack, heap, text and data**. The following image shows a simplified layout of a process inside the main memory –



SR. NO.	COMPONENTS	DESCRIPTION
1.	Stack	This contains the temporary data such as method/function parameters, local variables and return address.
2.	Heap	This is dynamically allocated memory to a process during its run time.
3.	Data	This includes the current activity represented by the value of Program Counter and the contents of the processor's registers.
4.	Text	This contains the global and static variables.

Program: A **Program** is an executable file which contains a certain set of instructions written to complete the specific job or operation on your computer. For example, Google browser chrome.exe is an executable file which stores a set of instructions written in it which allow you to open the browser and explore web pages.

Process State

Process state is those states which tell the status of the process. Status of the process includes whether the Process has Executed or whether the process is Waiting for Some input and output from the user and whether the Process is Waiting for the CPU to Run the Program after the Completion of the Process.

A Process has a five States:-

1. New State
2. Ready State
3. Waiting State
4. Running State
5. Terminated State

Process Life Cycle

When a process is executed, it passes through different states. These steps may vary in different operating systems, and the names of these states are not standardized.

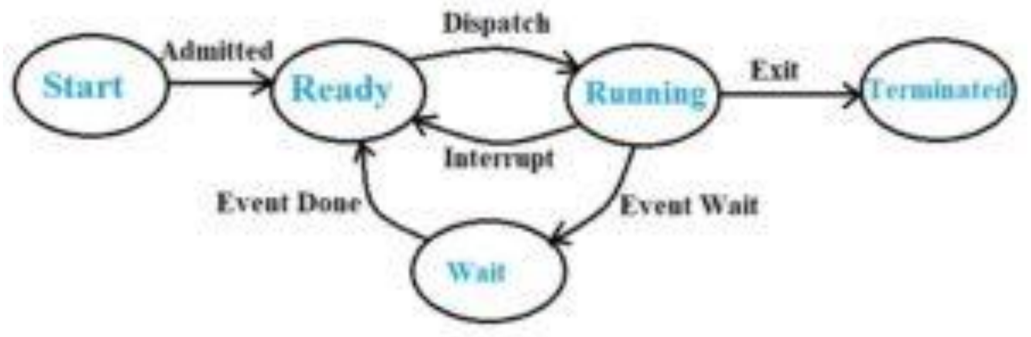


Fig)- Process Life Cycle

In general, a process can have one of the following five states at a time.

SR. NO.	PROCESS STATES	DESCRIPTION
1.	Start	This is the initial state when a process is first started/created.
2.	Ready	When the Process is Ready to Execute but he is waiting for the CPU to Execute then this is called as the Ready State. After the Completion of the Input and outputs the Process will be on Ready State means the Process will Wait for the Processor to Execute.
3.	Running	When the Process is Running under the CPU, or When the Program is Executed by the CPU , then this is called as the Running process and when a process is Running then this will also provides us Some Outputs on the Screen.
4.	Waiting	Process moves into the waiting state if it needs to wait for a resource, such as waiting for user input, or waiting for a file to become available.

5.	Terminated	Once the process finishes its execution, or it is terminated by the operating system, it is moved to the terminated state where it waits to be removed from main memory.

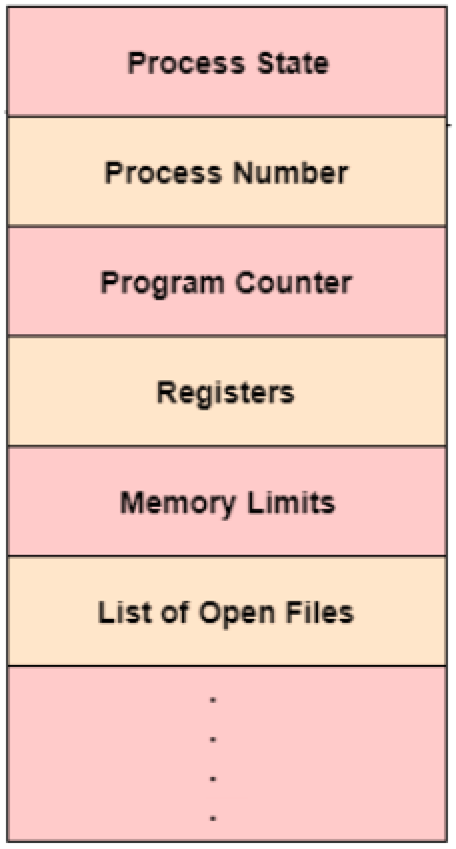
2) Describe Process Control block:

Process Control Block is a data structure that contains information of the process related to it. The process control block is also known as a task control block, entry of the process table, etc.

It is very important for process management as the data structuring for processes is done in terms of the PCB. It also defines the current state of the operating system.

Structure of the Process Control Block

The process control stores many data items that are needed for efficient process management. Some of these data items are explained with the help of the given diagram −



Process Control Block (PCB)

The following are the data items −

Process State

This specifies the process state i.e. new, ready, running, waiting or terminated.

Process Number

This shows the number of the particular process.

Program Counter

This contains the address of the next instruction that needs to be executed in the process.

Registers

This specifies the registers that are used by the process. They may include accumulators, index registers, stack pointers, general purpose registers etc.

List of Open Files

These are the different files that are associated with the process

CPU Scheduling Information

The process priority, pointers to scheduling queues etc. is the CPU scheduling information that is contained in the PCB.

This may also include any other scheduling parameters.

Memory Management Information

The memory management information includes the page tables or the segment tables depending on the memory system used.

It also contains the value of the base registers, limit registers etc.

I/O Status Information

This information includes the list of I/O devices used by the process, the list of files etc.

Accounting information

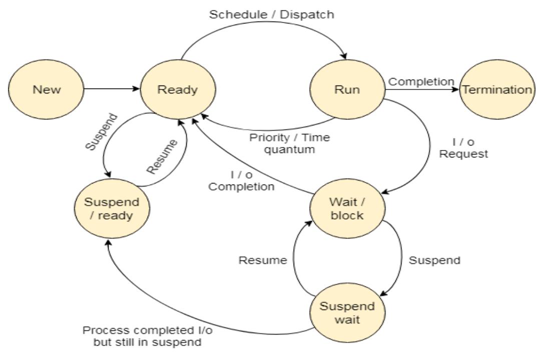
The time limits, account numbers, amount of CPU used, process numbers etc. are all a part of the PCB accounting information.

Location of the Process Control Block

The process control block is kept in a memory area that is protected from the normal user access. This is done because it contains important process information. Some of the operating systems place the PCB at the beginning of the kernel stack for the process as it is a safe location.

3) Explain Process state:

Process States
State Diagram



The process, from its creation to completion, passes through various states. The minimum number of states is five.

The names of the states are not standardized although the process may be in one of the following states during execution.

1. New

A program which is going to be picked up by the OS into the main memory is called a new process.

2. Ready

Whenever a process is created, it directly enters in the ready state, in which, it waits for the CPU to be assigned. The OS picks the new processes from the secondary memory and put all of them in the main memory.

The processes which are ready for the execution and reside in the main memory are called ready state processes. There can be many processes present in the ready state.

3. Running

One of the processes from the ready state will be chosen by the OS depending upon the scheduling algorithm. Hence, if we have only one CPU in our system, the number of running processes for a particular time will always be one. If we have n processors in the system then we can have n processes running simultaneously.

4. Block or wait

From the Running state, a process can make the transition to the block or wait state depending upon the scheduling algorithm or the intrinsic behavior of the process.

When a process waits for a certain resource to be assigned or for the input from the user then the OS move this process to the block or wait state and assigns the CPU to the other processes.

5. Completion or termination

When a process finishes its execution, it comes in the termination state. All the context of the process (Process Control Block) will also be deleted the process will be terminated by the Operating system.

6. Suspend ready

A process in the ready state, which is moved to secondary memory from the main memory due to lack of the resources (mainly primary memory) is called in the suspend ready state.

If the main memory is full and a higher priority process comes for the execution then the OS have to make the room for the process in the main memory by throwing the lower priority process out into the secondary memory. The suspend ready processes remain in the secondary memory until the main memory gets available.

7. Suspend wait

Instead of removing the process from the ready queue, it's better to remove the blocked process which is waiting for some resources in the main memory. Since it is already waiting for some resource to get available hence it is better if it waits in the secondary memory and make room for the higher priority process. These processes complete their execution once the main memory gets available and their wait is finished.

4) Introduce 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.

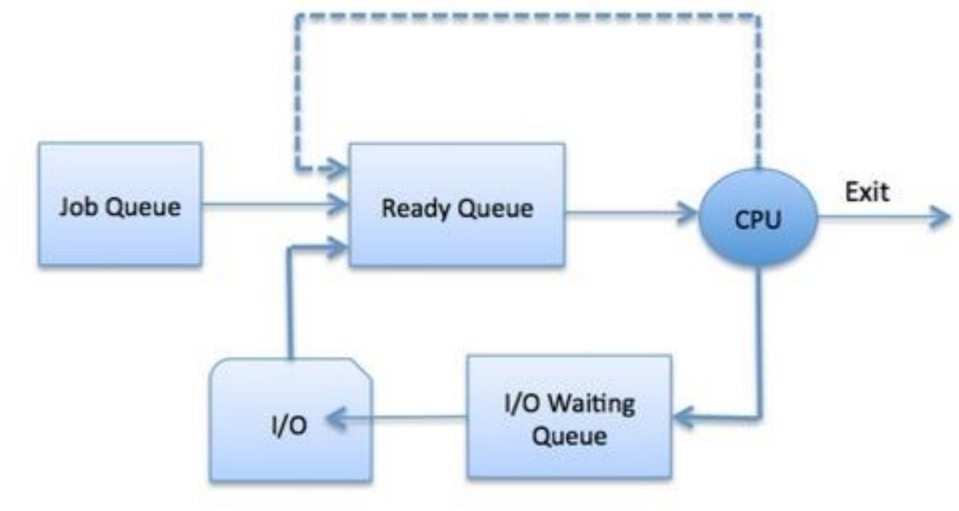
Process Scheduling Queues

The OS maintains all PCBs in Process Scheduling Queues. The OS maintains a separate queue for each of the process states and PCBs of all processes in the same execution state are placed in the same queue. When the state of a process is changed, its PCB is unlinked from its current queue and moved to its new state queue.

The Operating System maintains the following important process scheduling queues −

- **Job queue** − This queue keeps all the processes in the system.

- **Ready queue** − This queue keeps a set of all processes residing in main memory, ready and waiting to execute. A new process is always put in this queue.
- **Device queues** − The processes which are blocked due to unavailability of an I/O device constitute this queue.



The OS can use different policies to manage each queue (FIFO, Round Robin, Priority, etc.). The OS scheduler determines how to move processes between the ready and run queues which can only have one entry per processor core on the system; in the above diagram, it has been merged with the CPU.

Two-State Process Model

Two-state process model refers to running and non-running states which are described below −

S.N.	State & Description
1	**Running** When a new process is created, it enters into the system as in the running state.
2	**Not Running** Processes that are not running are kept in queue, waiting for their turn to execute. Each entry in the queue is a pointer to a particular process. Queue is implemented by using linked list. Use of dispatcher is as follows. When a process is interrupted, that process is transferred in the waiting queue. If the process has completed or aborted, the process is discarded. In either case, the dispatcher then selects a process from the queue to execute.

5)Explain Process scheduling queues and types of Process schedulers (short term scheduler, Medium term scheduler and Long term schedule.

Schedulers

Schedulers are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types −

- Long-Term Scheduler
- Short-Term Scheduler
- Medium-Term Scheduler

Long Term Scheduler

It is also called a **job scheduler**. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling.

The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. If the degree of multiprogramming is stable, then the average rate of process creation must be equal to the average departure rate of processes leaving the system.

On some systems, the long-term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When a process changes the state from new to ready, then there is use of long-term scheduler.

Short Term Scheduler

It is also called as **CPU scheduler**. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.

Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

Medium Term Scheduler

Medium-term scheduling is a part of **swapping**. It removes the processes from the memory. It reduces the degree of multiprogramming. The medium-term scheduler is in-charge of handling the swapped out-processes.

A running process may become suspended if it makes an I/O request. A suspended processes cannot make any progress towards completion. In this condition, to remove the process from memory and make space for other processes, the suspended process is moved to the secondary storage. This process is called **swapping**, and the process is said to be swapped out or rolled out. Swapping may be necessary to improve the process mix.

Comparison among Scheduler

S.N.	Long-Term Scheduler	Short-Term Scheduler	Medium-Term Scheduler
1	It is a job scheduler	It is a CPU scheduler	It is a process swapping scheduler.
2	Speed is lesser than short term scheduler	Speed is fastest among other two	Speed is in between both short and long term scheduler.
3	It controls the degree of multiprogramming	It provides lesser control over degree of multiprogramming	It reduces the degree of multiprogramming.
4	It is almost absent or minimal in time sharing system	It is also minimal in time sharing system	It is a part of Time sharing systems.
5	It selects processes from pool and loads them into memory for execution	It selects those processes which are ready to execute	It can re-introduce the process into memory and execution can be continued.

6) Illustrate the concept of Preemptive and Non-Preemptive Scheduling:

Preemptive Scheduling is a CPU scheduling technique that works by dividing time slots of CPU to a given process. The time slot given might be able to complete the whole process or might not be able to it. When the burst time of the process is greater than CPU cycle, it is placed back into the ready queue and will execute in the next chance. This scheduling is used when the process switch to ready state.

Algorithms that are backed by preemptive Scheduling are round-robin (RR), priority, SRTF (shortest remaining time first).

Non-preemptive Scheduling is a CPU scheduling technique the process takes the resource (CPU time) and holds it till the process gets terminated or is pushed to the waiting state. No process is interrupted until it is completed, and after that processor switches to another process.

Algorithms that are based on non-preemptive Scheduling are non-preemptive priority, and shortest Job first.

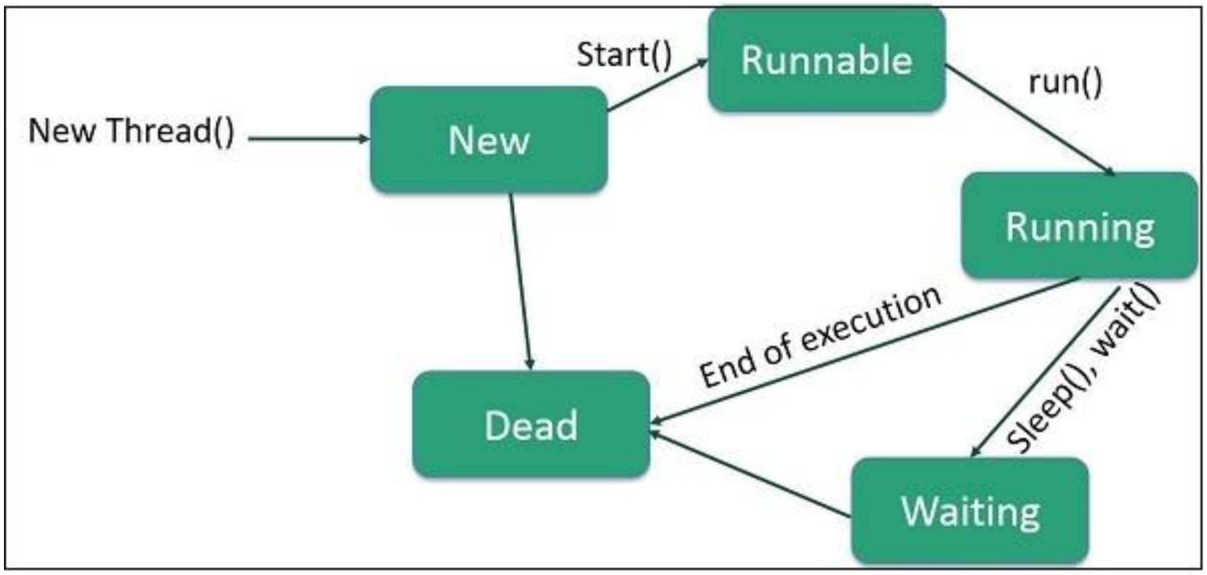
Preemptive Vs Non-Preemptive Scheduling

Preemptive Scheduling	Non-Preemptive Scheduling
Resources are allocated according to the cycles for a limited time.	Resources are used and then held by the process until it gets terminated.
The process can be interrupted, even before the completion.	The process is not interrupted until its life cycle is complete.
Starvation may be caused, due to the insertion of priority process in the queue.	Starvation can occur when a process with large burst time occupies the system.

Preemptive Scheduling	Non-Preemptive Scheduling
Maintaining queue and time needs storage overhead.	No such overheads are required.

7) Illustrate the concept of thread and its life cycle:

A thread goes through various stages in its lifecycle. For example, a thread is born, started, runs, and then dies. The following diagram shows the complete life cycle of a thread.



Following are the stages of the life cycle −

- **New** − A new thread begins its life cycle in the new state. It remains in this state until the program starts the thread. It is also referred to as a born thread.
- **Runnable** − After a newly born thread is started, the thread becomes runnable. A thread in this state is considered to be executing its task.
- **Waiting** − Sometimes, a thread transitions to the waiting state while the thread waits for another thread to perform a task. Thread transitions back to the runnable state only when another thread signals the waiting thread to continue executing.
- **Timed Waiting** − A runnable thread can enter the timed waiting state for a specified interval of time. A thread in this state transitions back to the runnable state when that time interval expires or when the event it is waiting for occurs.
- **Terminated (Dead)** − A runnable thread enters the terminated state when it completes its task or otherwise terminates.

8) Describe Algorithm: FCFS/SJF/SRT:

First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.
As the process enters the ready queue, its PCB (Process Control Block) is linked with the tail of the queue and, when the CPU becomes free, it should be assigned to the process at the beginning of the queue.

Characteristics of FCFS method

- It supports non-preemptive and pre-emptive scheduling algorithm.
- Jobs are always executed on a first-come, first-serve basis.
- It is easy to implement and use.
- This method is poor in performance, and the general wait time is quite high.

Example of FCFS scheduling

A real-life example of the FCFS method is buying a movie ticket on the ticket counter. In this scheduling algorithm, a person is served according to the queue manner. The person who arrives first in the queue first buys the ticket and then the next one. This will continue until the last person in the queue purchases the ticket. Using this algorithm, the CPU process works in a similar manner.

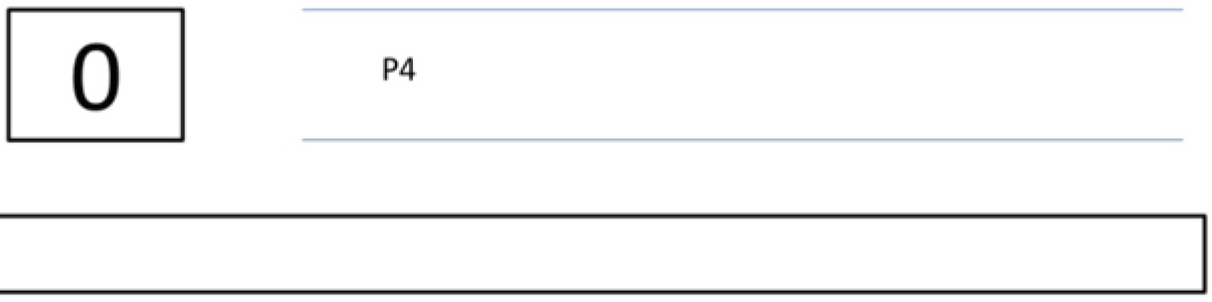
How FCFS Works? Calculating Average Waiting Time

Here is an example of five processes arriving at different times. Each process has a different burst time.

Process	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

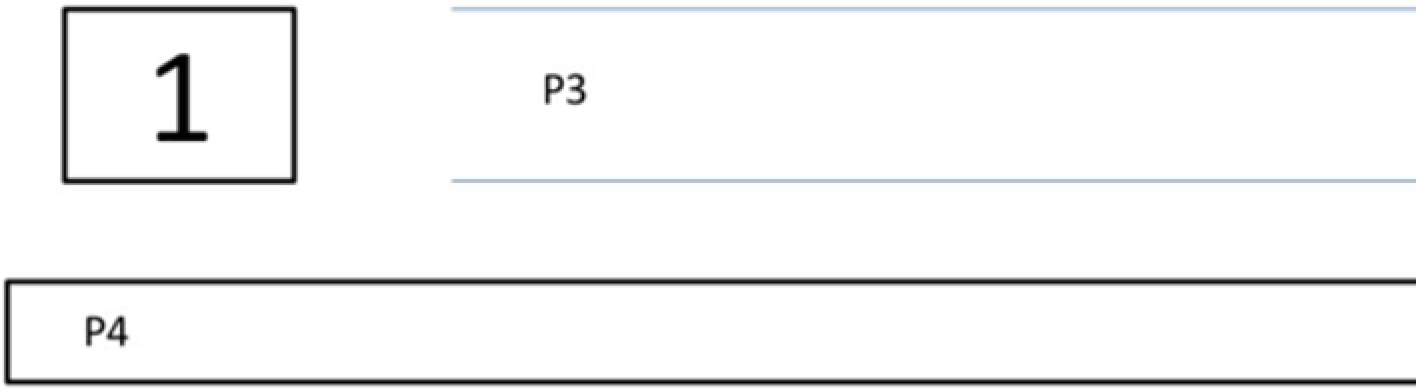
Using the FCFS scheduling algorithm, these processes are handled as follows.

Step 0) The process begins with P4 which has arrival time 0



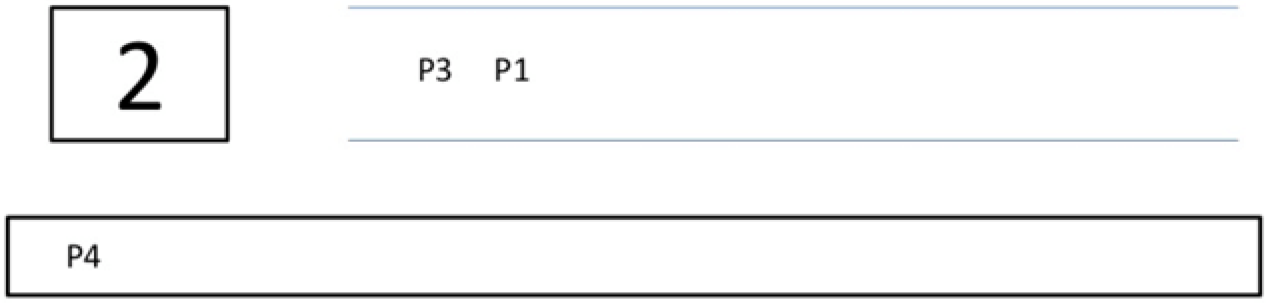
Step 1) At time=1, P3 arrives. P4 is still executing. Hence, P3 is kept in a queue.

Process	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

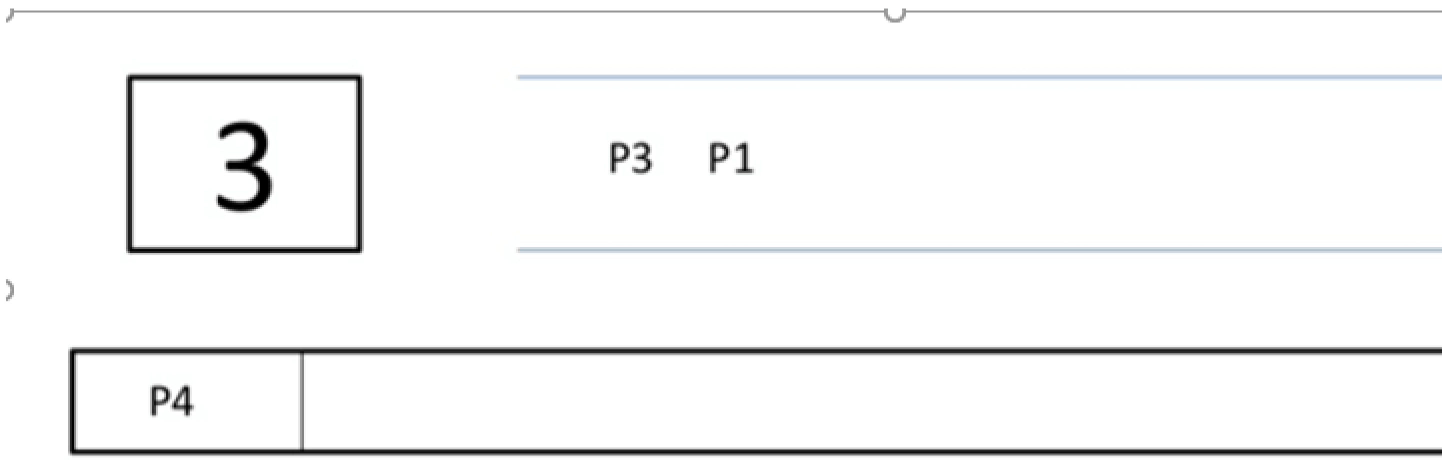


Step 2) At time= 2, P1 arrives which is kept in the queue.

Process	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

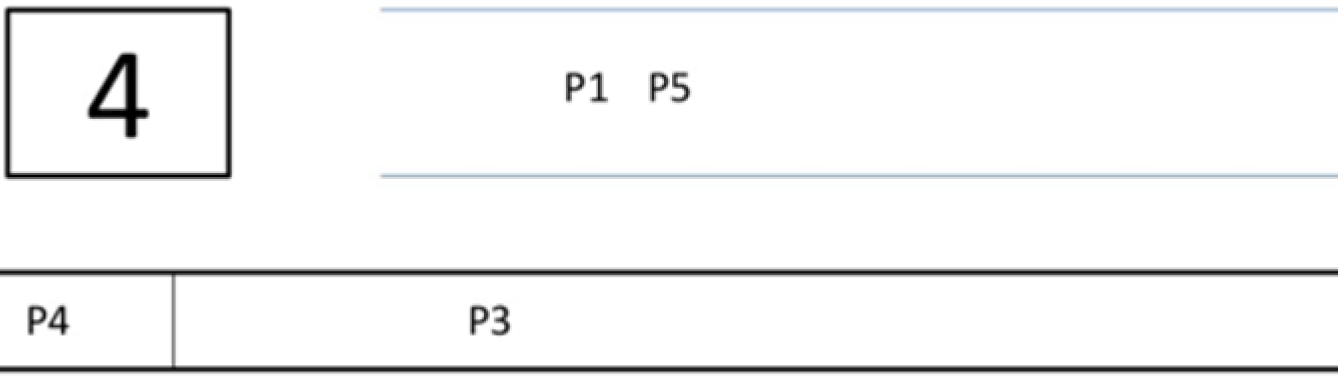


Step 3) At time=3, P4 process completes its execution.



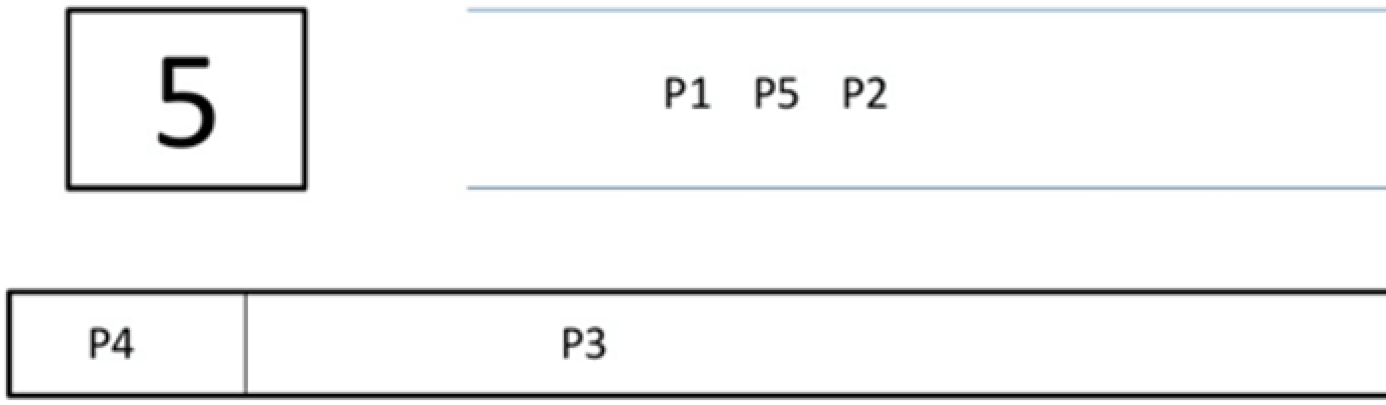
Step 4) At time=4, P3, which is first in the queue, starts execution.

Process	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

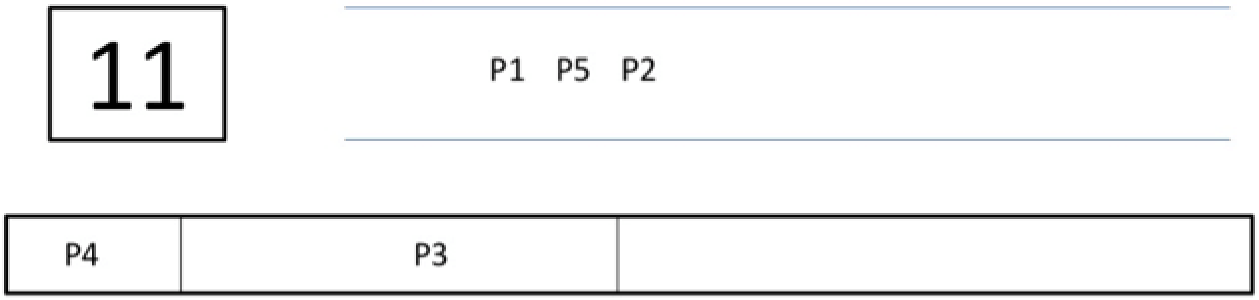


Step 5) At time =5, P2 arrives, and it is kept in a queue.

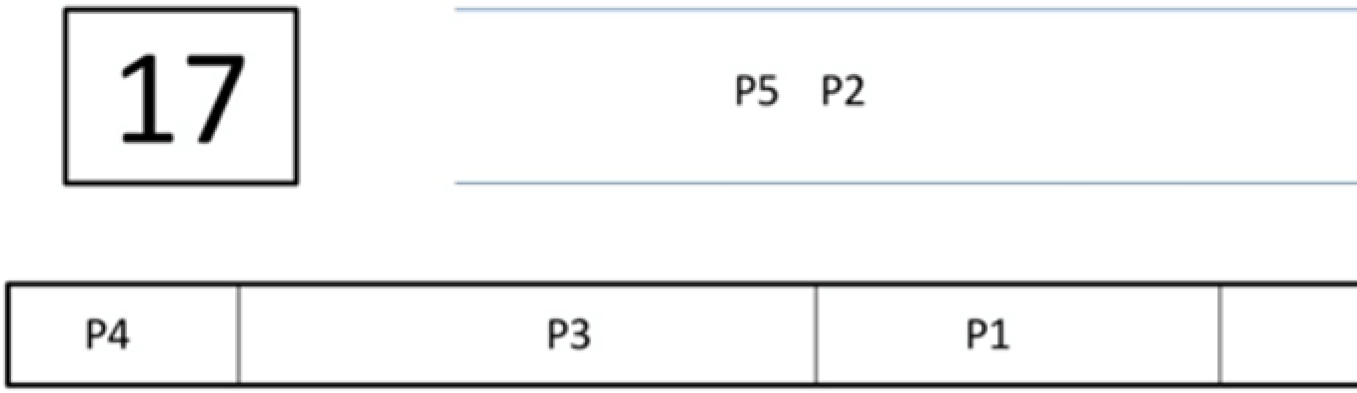
Process	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4



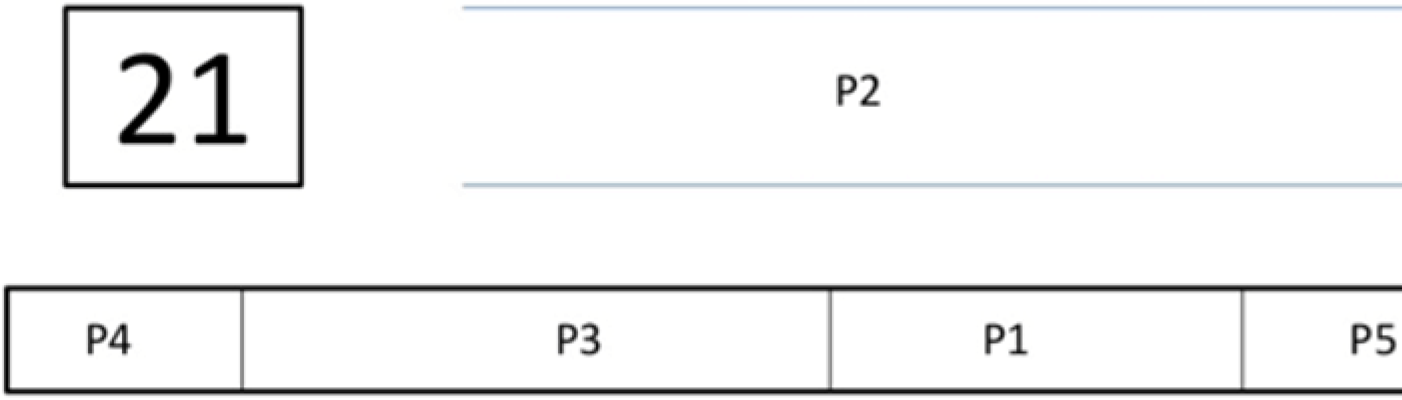
Step 6) At time 11, P3 completes its execution.



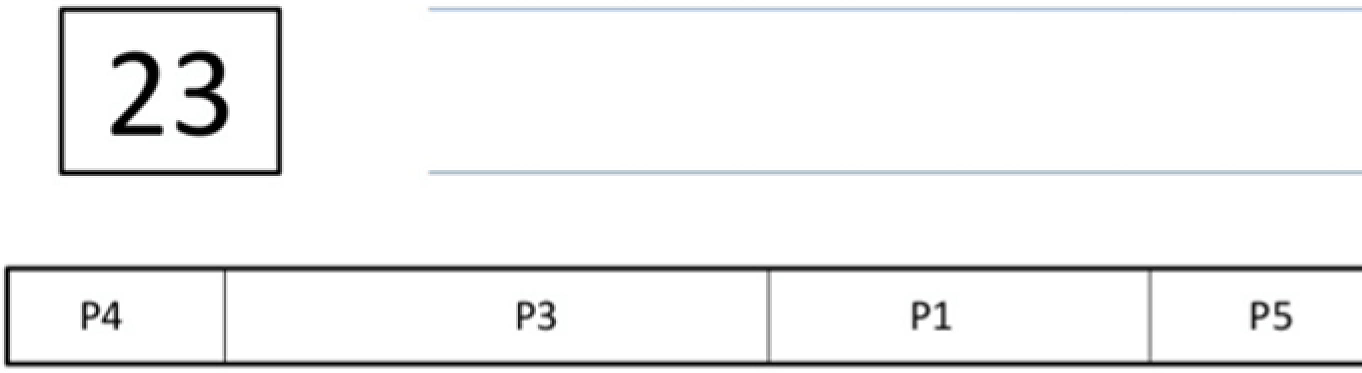
Step 7) At time=11, P1 starts execution. It has a burst time of 6. It completes execution at time interval 17



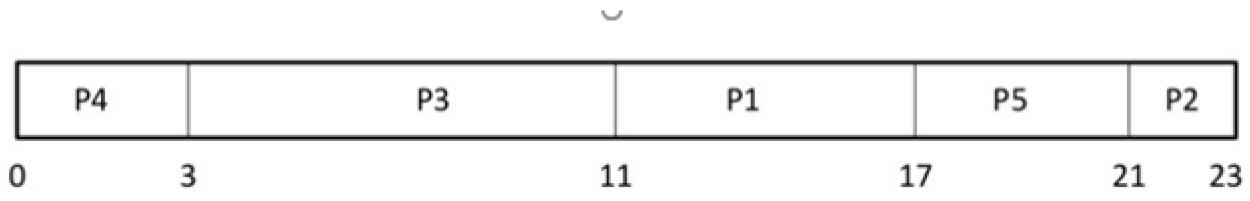
Step 8) At time=17, P5 starts execution. It has a burst time of 4. It completes execution at time=21



Step 9) At time=21, P2 starts execution. It has a burst time of 2. It completes execution at time interval 23



Step 10) Let's calculate the average waiting time for above example.



Waiting time = Start time - Arrival time
P4 = 0-0 = 0

P3 = 3-1 = 2

PI = 11-2 = 9

P5= 17-4 = 13

P2= 21-5= 16

Average Waiting Time

$$\frac{0+2+9+13+16}{5}$$

= 40/5= 8

Advantages of FCFS

Here, are pros/benefits of using FCFS scheduling algorithm:

- The simplest form of a CPU scheduling algorithm
- Easy to program
- First come first served

Disadvantages of FCFS

Here, are cons/ drawbacks of using FCFS scheduling algorithm:

- It is a Non-Preemptive CPU scheduling algorithm, so after the process has been allocated to the CPU, it will never release the CPU until it finishes executing.
- The Average Waiting Time is high.
- Short processes that are at the back of the queue have to wait for the long process at the front to finish.
- Not an ideal technique for time-sharing systems.
- Because of its simplicity, FCFS is not very efficient.

Shortest Job First (SJF) is an algorithm in which the process having the smallest execution time is chosen for the next execution. This scheduling method can be preemptive or non-preemptive. It significantly reduces the average waiting time for other processes awaiting execution. The full form of SJF is Shortest Job First.
There are basically two types of SJF methods:

- Non-Preemptive SJF
- Preemptive SJF

Characteristics of SJF Scheduling

- It is associated with each job as a unit of time to complete.
- This algorithm method is helpful for batch-type processing, where waiting for jobs to complete is not critical.
- It can improve process throughput by making sure that shorter jobs are executed first, hence possibly have a short turnaround time.
- It improves job output by offering shorter jobs, which should be executed first, which mostly have a shorter turnaround time.

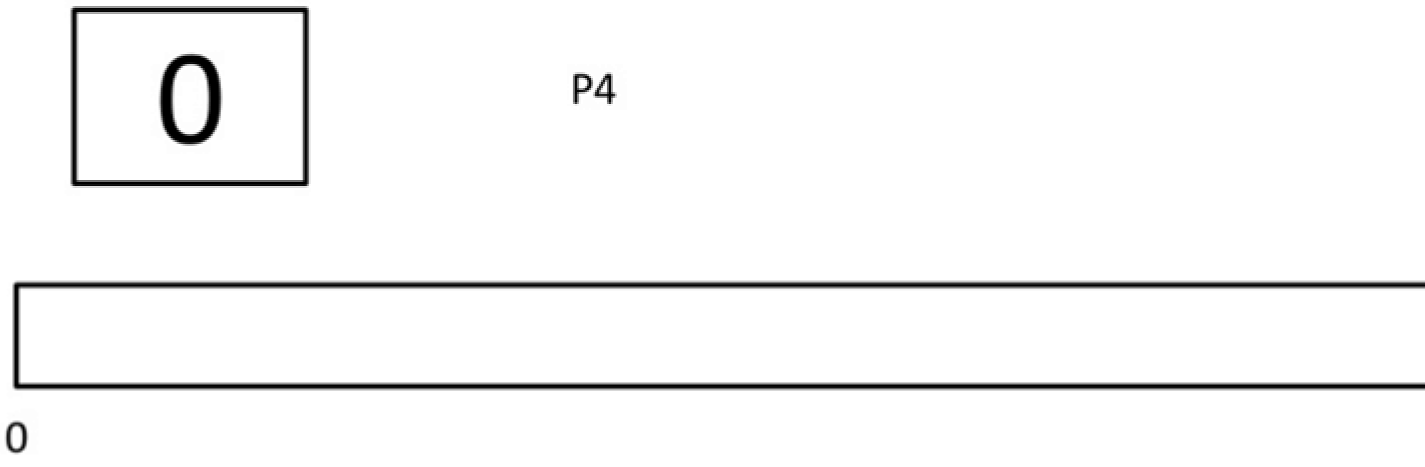
Non-Preemptive SJF

In non-preemptive scheduling, once the CPU cycle is allocated to process, the process holds it till it reaches a waiting state or terminated.

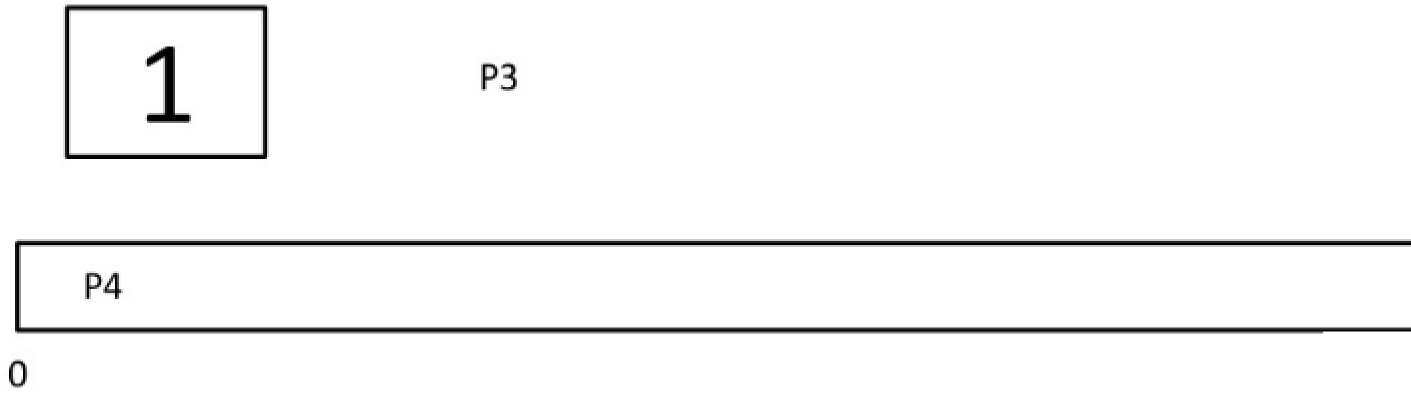
Consider the following five processes each having its own unique burst time and arrival time.

Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

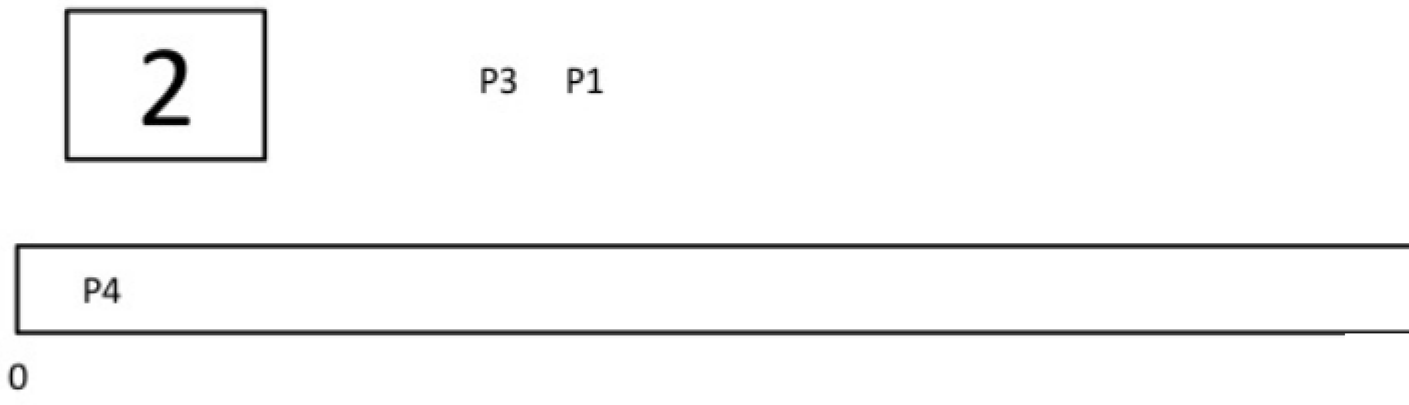
Step 0) At time=0, P4 arrives and starts execution.



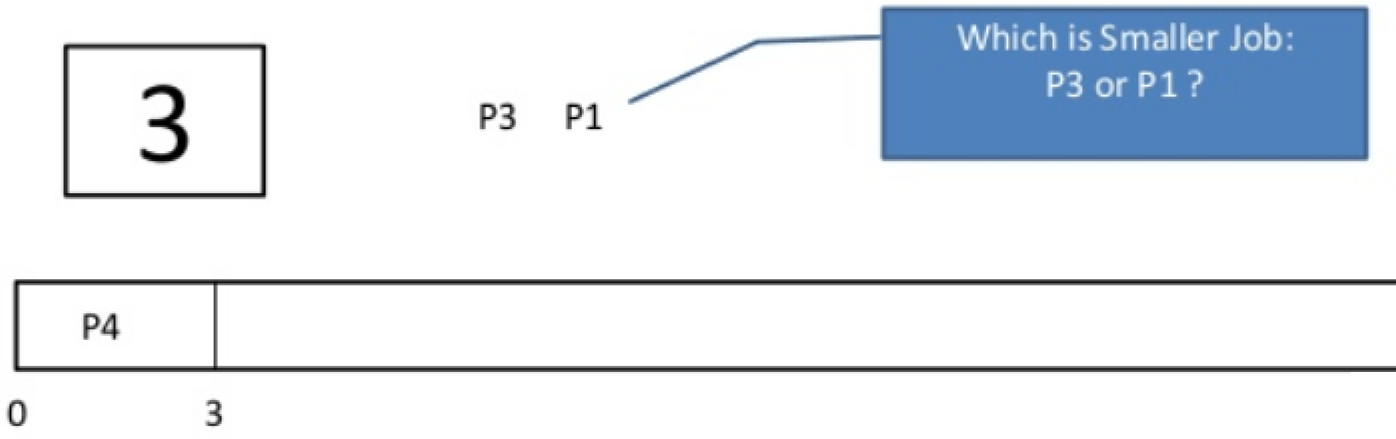
Step 1) At time= 1, Process P3 arrives. But, P4 still needs 2 execution units to complete. It will continue execution.



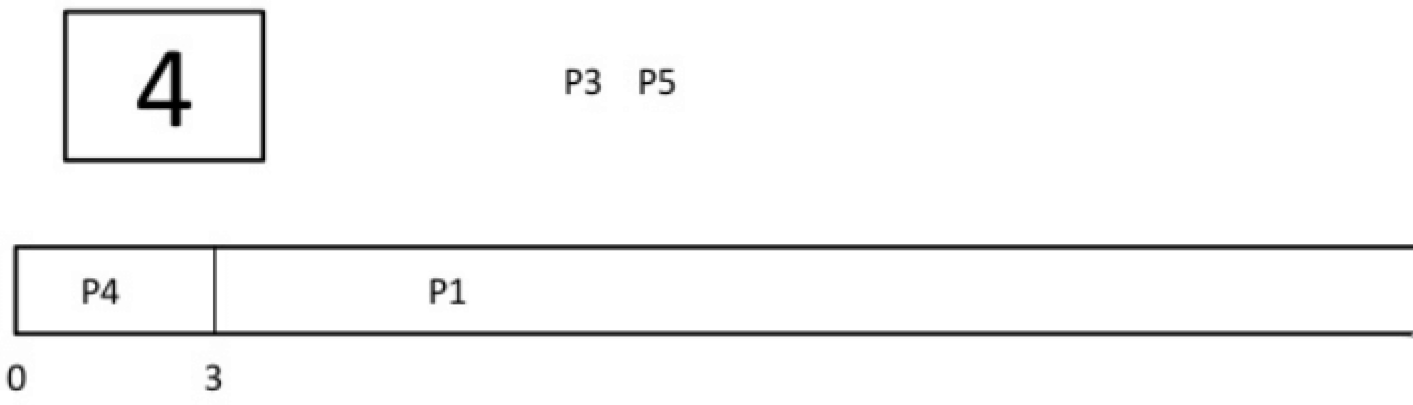
Step 2) At time =2, process P1 arrives and is added to the waiting queue. P4 will continue execution.



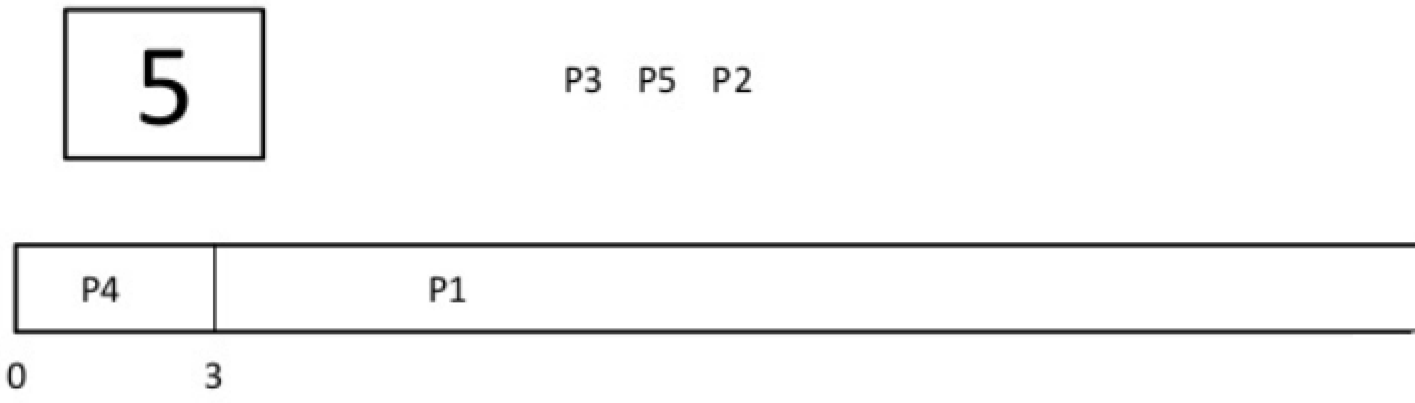
Step 3) At time = 3, process P4 will finish its execution. The burst time of P3 and P1 is compared. Process P1 is executed because its burst time is less compared to P3.



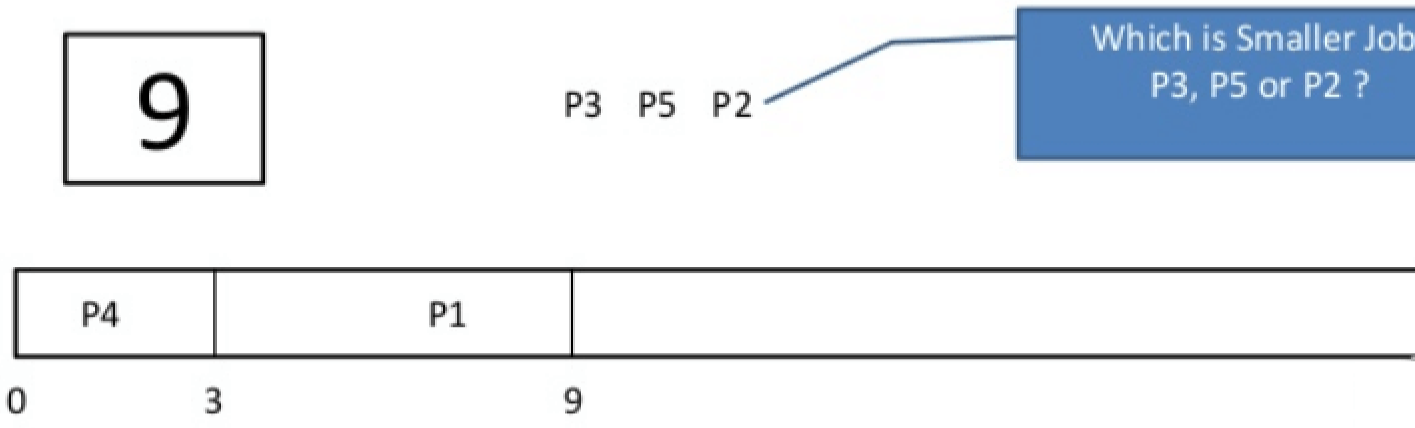
Step 4) At time = 4, process P5 arrives and is added to the waiting queue. P1 will continue execution.



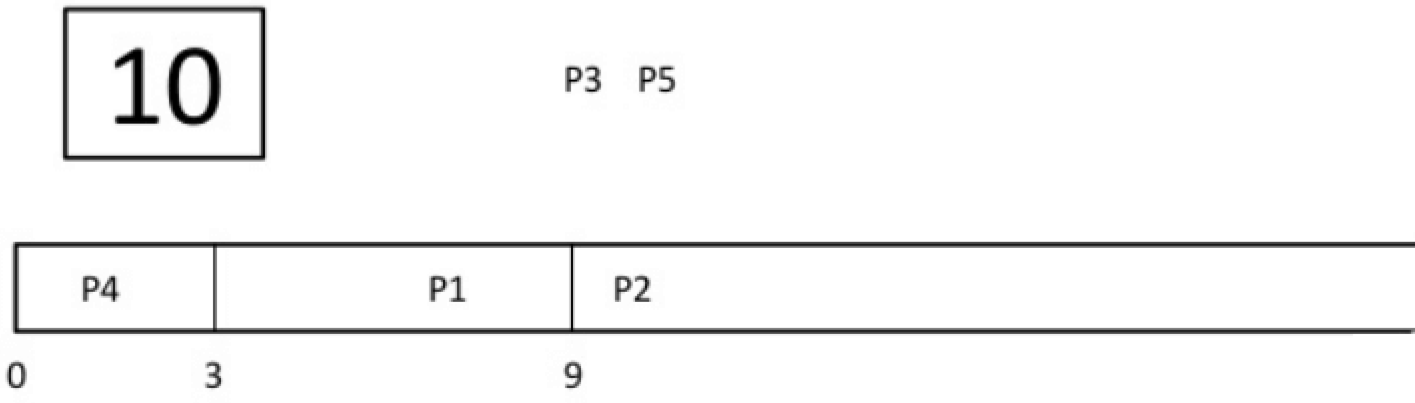
Step 5) At time = 5, process P2 arrives and is added to the waiting queue. P1 will continue execution.



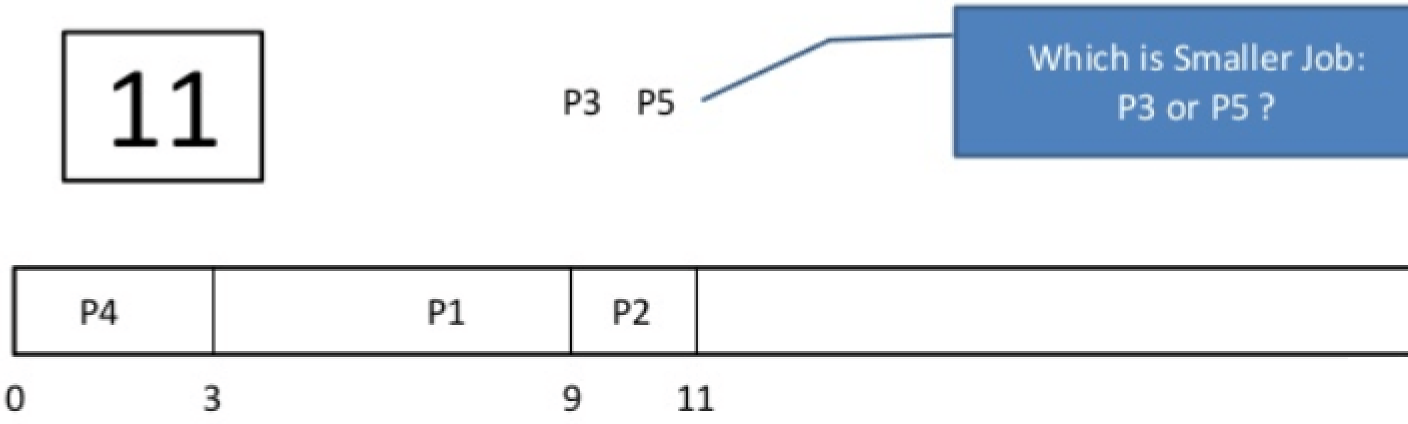
Step 6) At time = 9, process P1 will finish its execution. The burst time of P3, P5, and P2 is compared. Process P2 is executed because its burst time is the lowest.



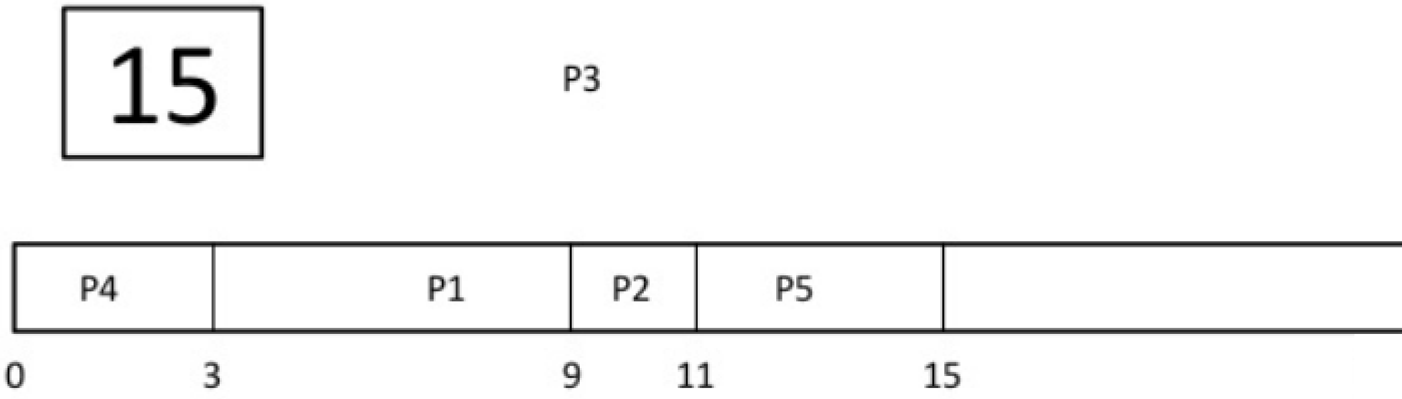
Step 7) At time=10, P2 is executing and P3 and P5 are in the waiting queue.



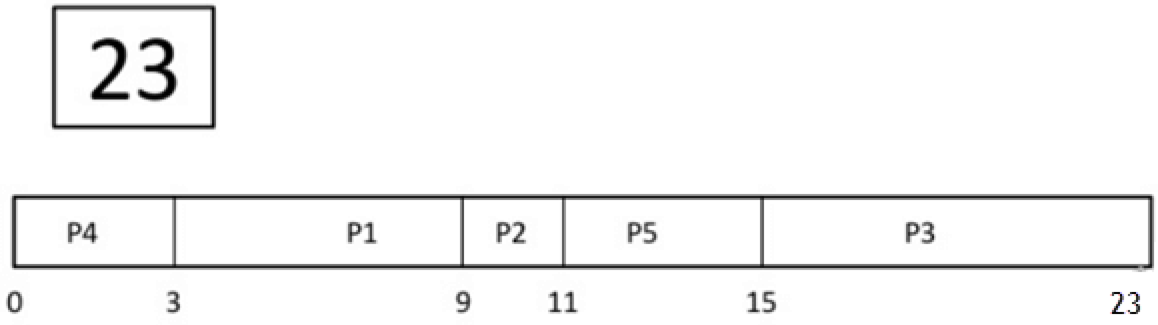
Step 8) At time = 11, process P2 will finish its execution. The burst time of P3 and P5 is compared. Process P5 is executed because its burst time is lower.



Step 9) At time = 15, process P5 will finish its execution.



Step 10) At time = 23, process P3 will finish its execution.



Step 11) Let's calculate the average waiting time for above example.

Wait time
P4= 0-0=0
P1= 3-2=1

P2= 9-5=4
P5= 11-4=7
P3= 15-1=14
Average Waiting Time= 0+1+4+7+14/5 = 26/5 = 5.2

Preemptive SJF

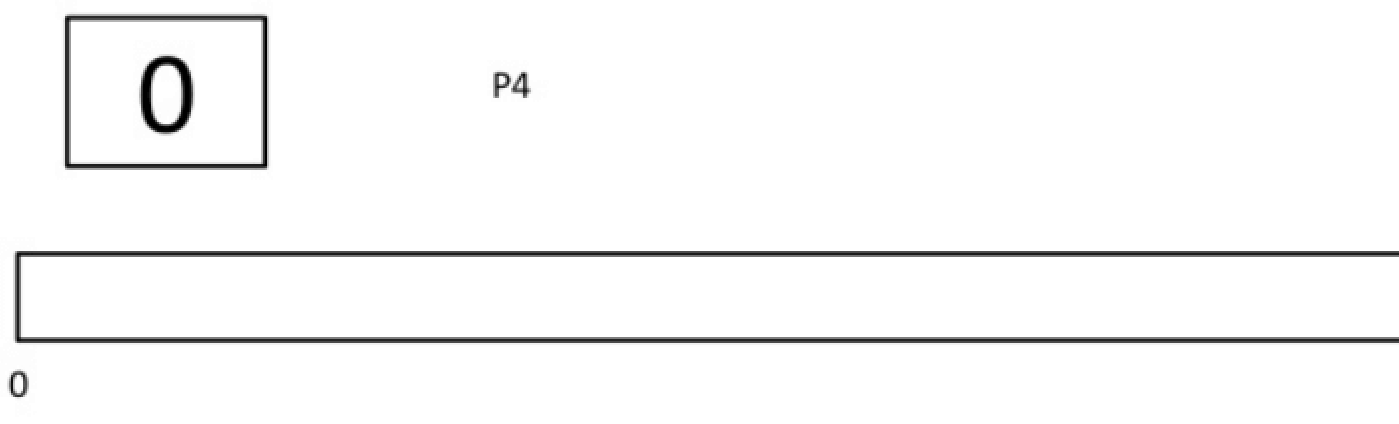
In Preemptive SJF Scheduling, jobs are put into the ready queue as they come. A process with shortest burst time begins execution. If a process with even a shorter burst time arrives, the current process is removed or preempted from execution, and the shorter job is allocated CPU cycle.

Consider the following five process:

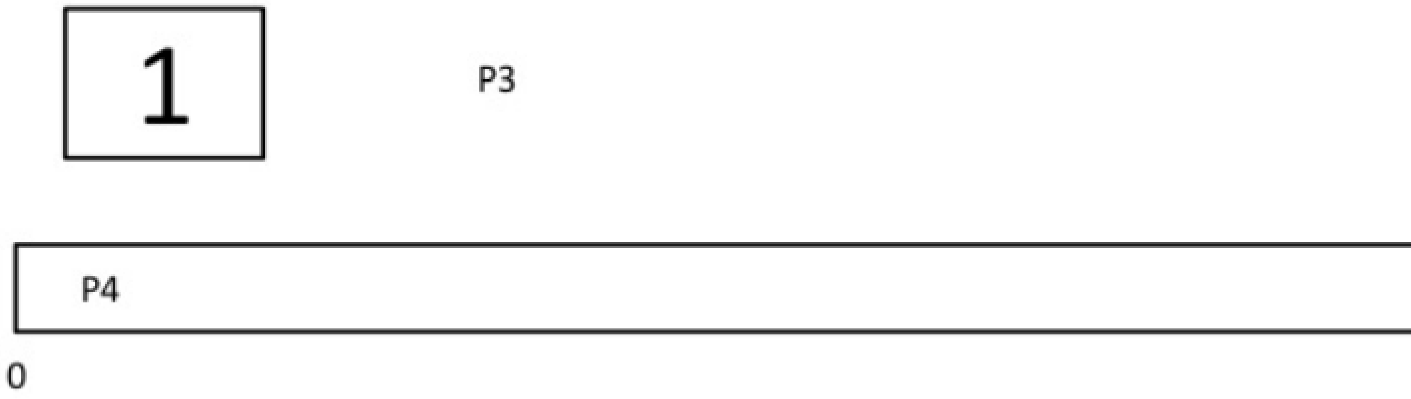
Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

Step 0) At time=0, P4 arrives and starts execution.

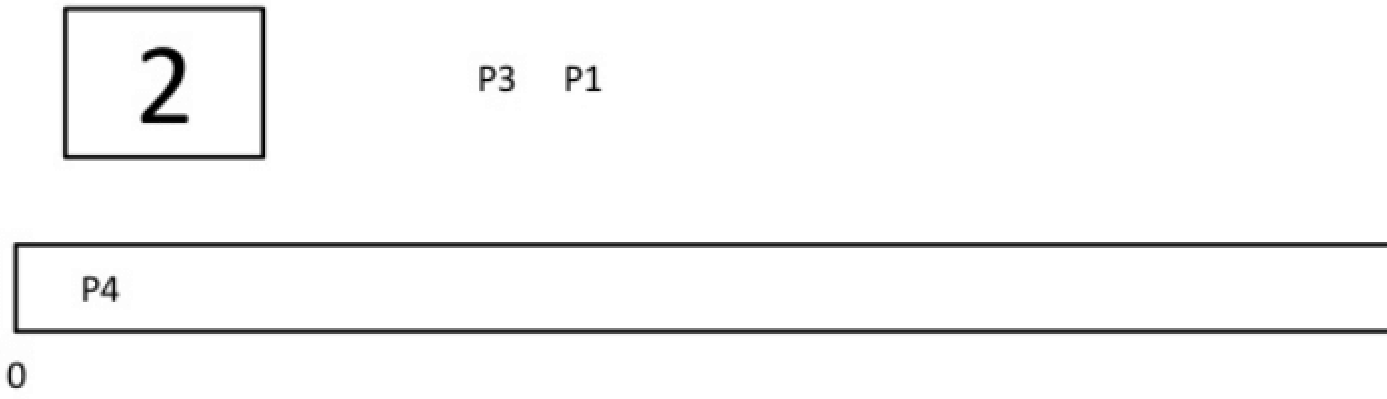
Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4



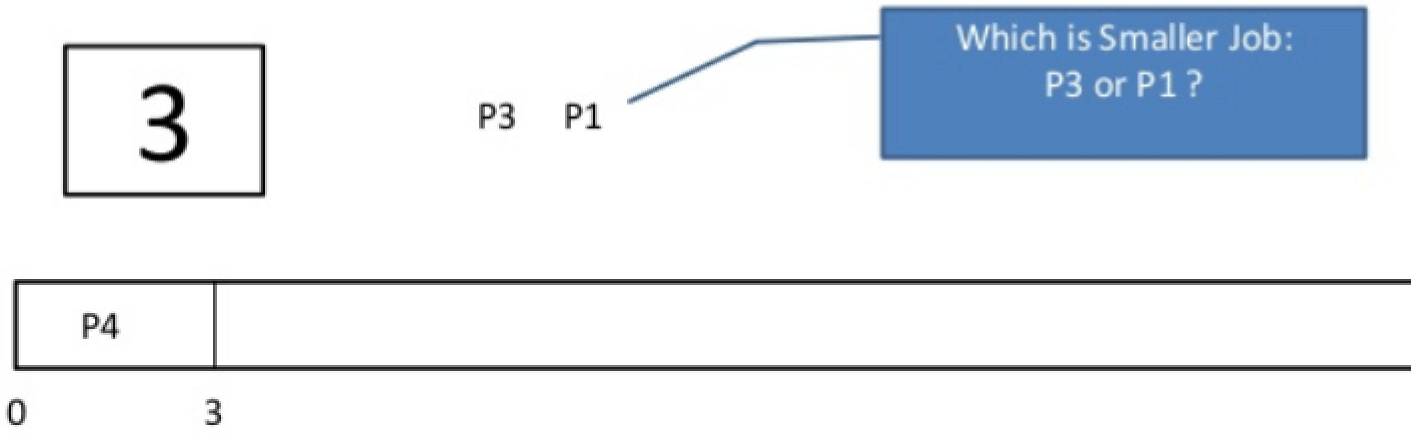
Step 1) At time= 1, Process P3 arrives. But, P4 has a shorter burst time. It will continue execution.



Step 2) At time = 2, process P1 arrives with burst time = 6. The burst time is more than that of P4. Hence, P4 will continue execution.

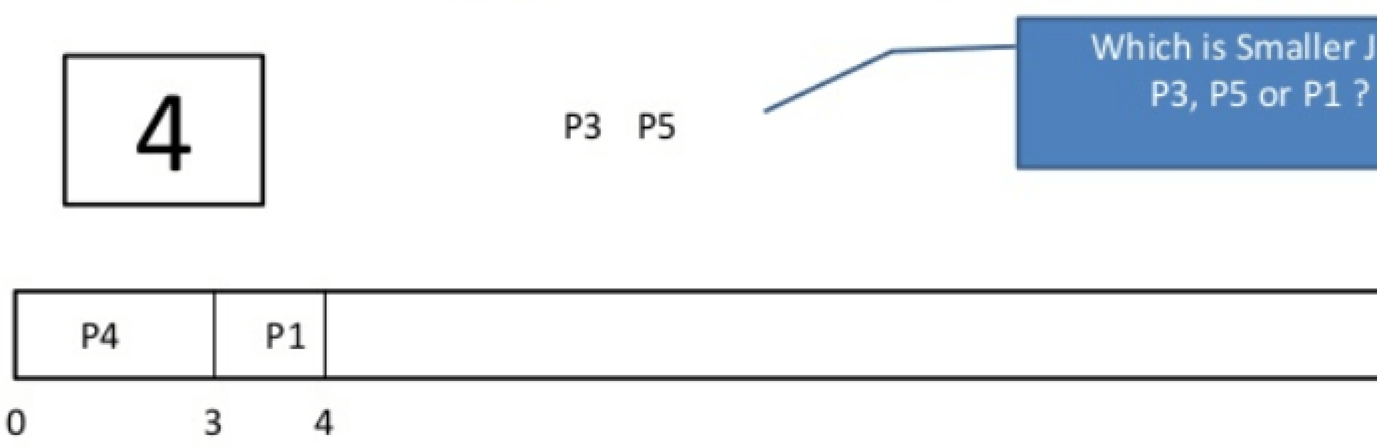


Step 3) At time = 3, process P4 will finish its execution. The burst time of P3 and P1 is compared. Process P1 is executed because its burst time is lower.



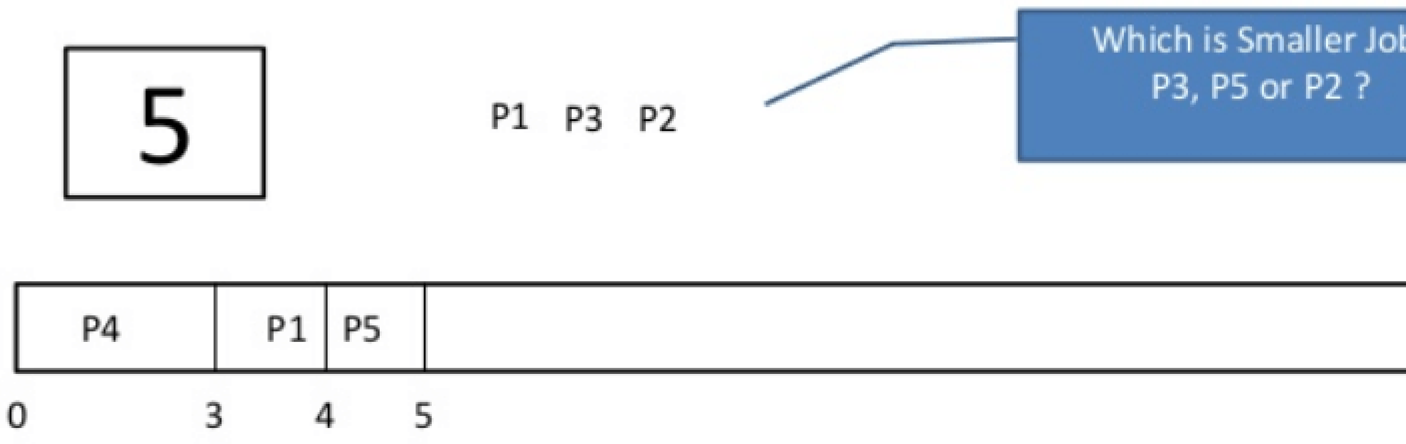
Step 4) At time = 4, process P5 will arrive. The burst time of P3, P5, and P1 is compared. Process P5 is executed because its burst time is lowest. Process P1 is preempted.

Process Queue	Burst time	Arrival time
P1	5 out of 6 is remaining	2
P2	2	5
P3	8	1
P4	3	0
P5	4	4

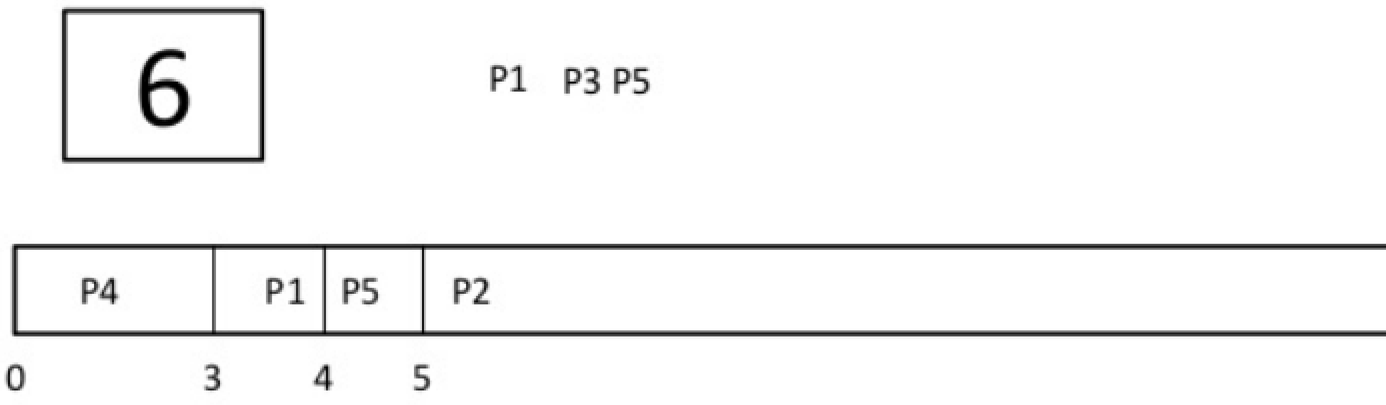


Step 5) At time = 5, process P2 will arrive. The burst time of P1, P2, P3, and P5 is compared. Process P2 is executed because its burst time is least. Process P5 is preempted.

Process Queue	Burst time	Arrival time
P1	5 out of 6 is remaining	2
P2	2	5
P3	8	1
P4	3	0
P5	3 out of 4 is remaining	4

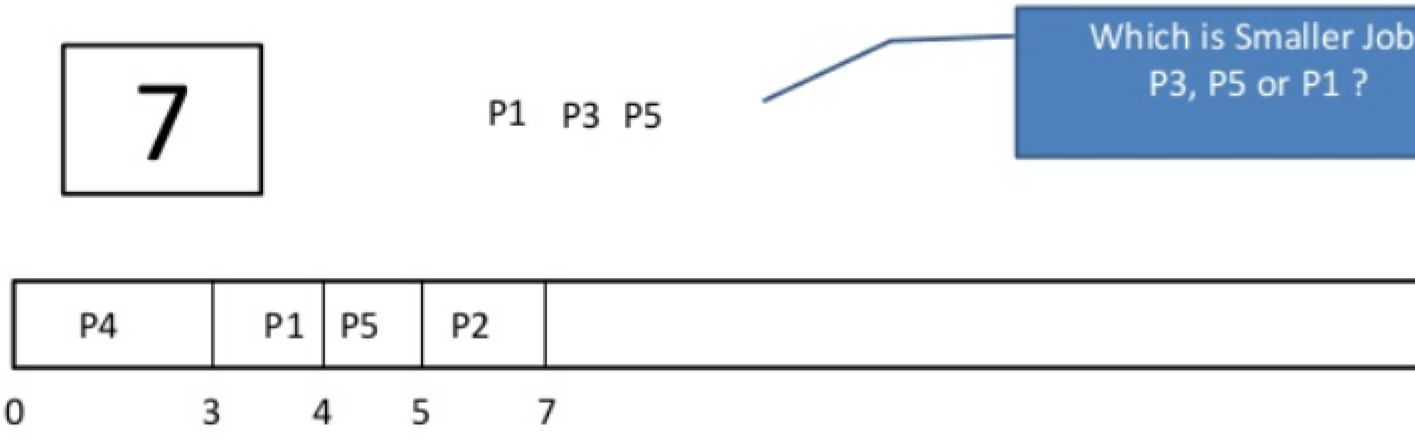


Step 6) At time =6, P2 is executing.

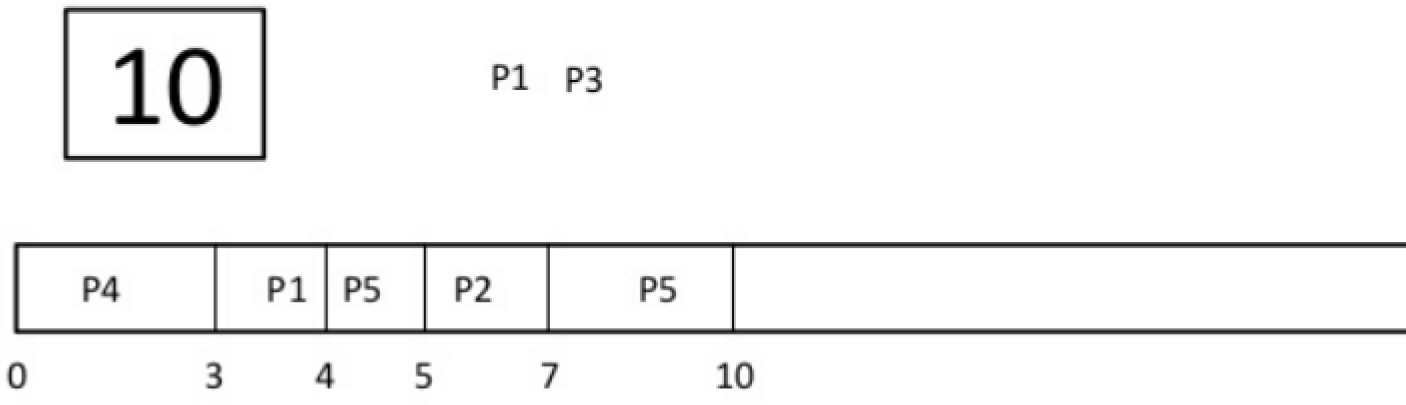


Step 7) At time =7, P2 finishes its execution. The burst time of P1, P3, and P5 is compared. Process P5 is executed because its burst time is lesser.

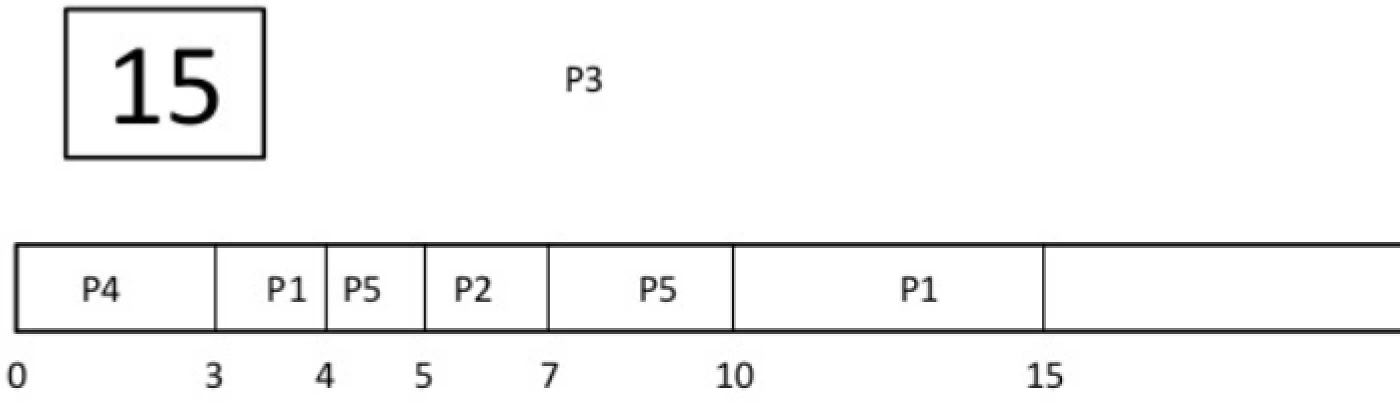
Process Queue	Burst time	Arrival time
P1	5 out of 6 is remaining	2
P2	2	5
P3	8	1
P4	3	0
P5	3 out of 4 is remaining	4



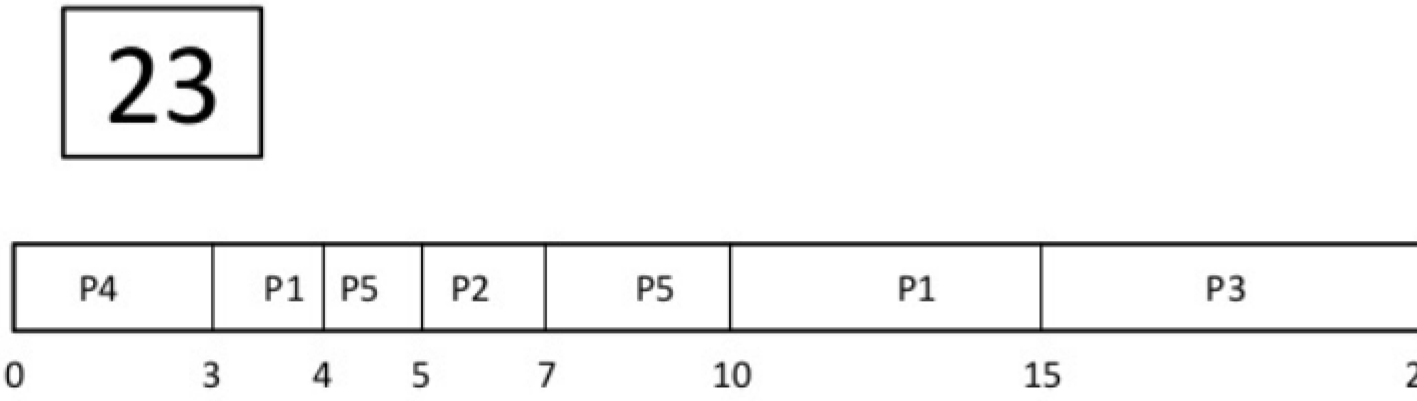
Step 8) At time =10, P5 will finish its execution. The burst time of P1 and P3 is compared. Process P1 is executed because its burst time is less.



Step 9) At time =15, P1 finishes its execution. P3 is the only process left. It will start execution.



Step 10) At time =23, P3 finishes its execution.



Step 11) Let’s calculate the average waiting time for above example.

Wait time

P4= 0-0=0
P1= (3-2) + 6 =7
P2= 5-5 = 0
P5= 4-4+2 =2
P3= 15-1 = 14
Average Waiting Time = 0+7+0+2+14/5 = 23/5 =4.6

Advantages of SJF

Here are the benefits/pros of using SJF method:

- SJF is frequently used for long term scheduling.
- It reduces the average waiting time over FIFO (First in First Out) algorithm.
- SJF method gives the lowest average waiting time for a specific set of processes.
- It is appropriate for the jobs running in batch, where run times are known in advance.
- For the batch system of long-term scheduling, a burst time estimate can be obtained from the job description.
- For Short-Term Scheduling, we need to predict the value of the next burst time.
- Probably optimal with regard to average turnaround time.

Disadvantages/Cons of SJF

Here are some drawbacks/cons of SJF algorithm:

- Job completion time must be known earlier, but it is hard to predict.
- It is often used in a batch system for long term scheduling.
- SJF can't be implemented for CPU scheduling for the short term. It is because there is no specific method to predict the length of the upcoming CPU burst.
- This algorithm may cause very long turnaround times or starvation.
- Requires knowledge of how long a process or job will run.
- It leads to the starvation that does not reduce average turnaround time.
- It is hard to know the length of the upcoming CPU request.
- Elapsed time should be recorded, that results in more overhead on the processor.

The Preemptive version of Shortest Job First(SJF) scheduling is known as Shortest Remaining Time First (SRTF). With the help of the SRTF algorithm, the process having the smallest amount of time remaining until completion is selected first to execute.

However, the SRTF algorithm involves more overheads than the Shortest job first (SJF)scheduling, because in SRTF OS is required frequently in order to monitor the CPU time of the jobs in the **READY** queue and to perform context switching.

In the **SRTF scheduling algorithm**, the execution of any process can be stopped after a certain amount of time. On arrival of every process, the short-term scheduler schedules those processes from the list of available processes & running processes that have the least remaining burst time.

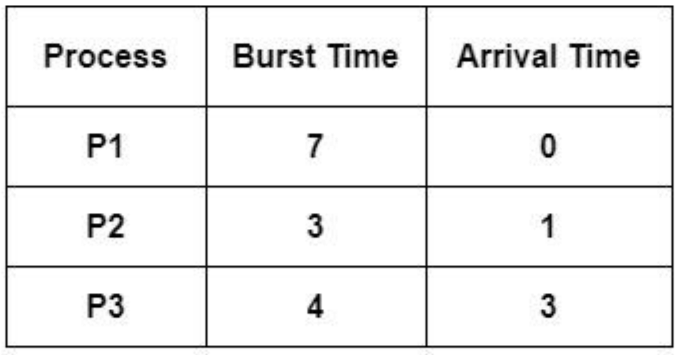
Advantages of SRTF

The main advantage of the SRTF algorithm is that it makes the processing of the jobs faster than the SJF algorithm, mentioned it's overhead charges are not counted.

Disadvantages of SRTF

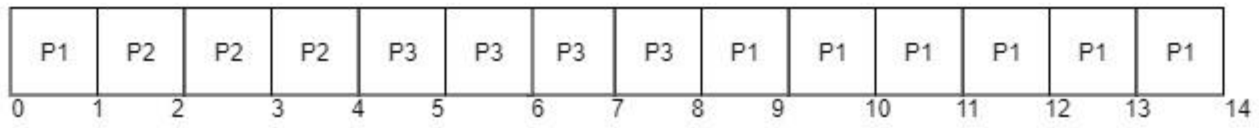
In SRTF, the context switching is done a lot more times than in SJN due to more consumption of the CPU's valuable time for processing. The consumed time of CPU then adds up to its processing time and which then diminishes the advantage of fast processing of this algorithm.

Example



Process	Burst Time	Arrival Time
P1	7	0
P2	3	1
P3	4	3

The Gantt Chart for SRTF will be:



As Arrival Time and Burst time for three processes P1, P2, P3 are given in the above diagram. Let us calculate Turn around time, completion time, and waiting time.

Process	**Arrival Time**	**Burst Time**	**Completion time**	**Turn around Time** **Turn Around Time = Completion Time – Arrival Time**	**Waiting Time** **Waiting Time = Turn Around Time – Burst Time**
P1	0	7	14	14-0=14	14-7=7
P2	1	3	4	4-1=3	3-3=0
P3	3	4	8	8-3=5	5-4=1

Average waiting time is calculated by adding the waiting time of all processes and then dividing them by no. of processes.

average waiting time = waiting for time of all processes/ no.of processes

average waiting time=7+0+1=8/3 = **2.66ms**

Unit 3:Memory Management:

1)Memory Hierarchy:

The Computer memory hierarchy looks like a pyramid structure which is used to describe the differences among memory types. It separates the computer storage based on hierarchy.

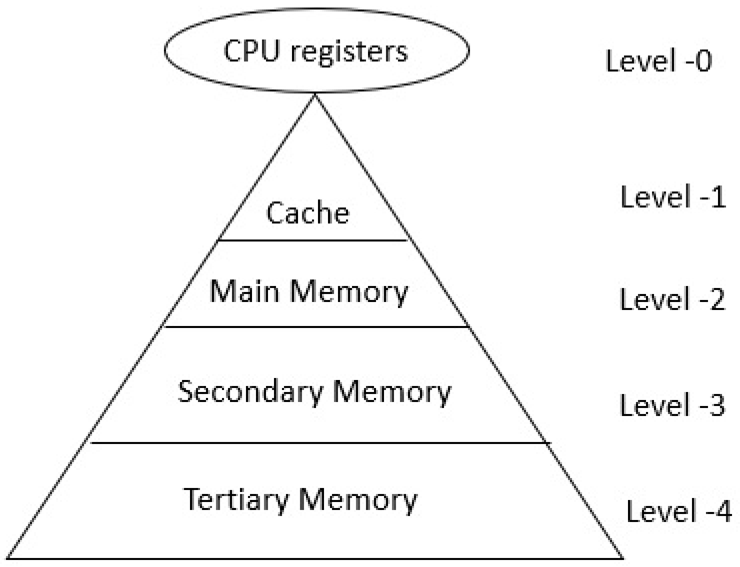
Level 0: CPU registers

Level 1: Cache memory

Level 2: Main memory or primary memory

Level 3: Magnetic disks or secondary memory

Level 4: Optical disks or magnetic types or tertiary Memory



In Memory Hierarchy the cost of memory, capacity is inversely proportional to speed. Here the devices are arranged in a manner Fast to slow, that is form register to Tertiary memory.

Let us discuss each level in detail:

Level-0 − Registers

The registers are present inside the CPU. As they are present inside the CPU, they have least access time. Registers are most expensive and smallest in size generally in kilobytes. They are implemented by using Flip-Flops.

Level-1 − Cache

Cache memory is used to store the segments of a program that are frequently accessed by the processor. It is expensive and smaller in size generally in Megabytes and is implemented by using static RAM.

Level-2 − Primary or Main Memory

It directly communicates with the CPU and with auxiliary memory devices through an I/O processor. Main memory is less expensive than cache memory and larger in size generally in Gigabytes. This memory is implemented by using dynamic RAM.

Level-3 − Secondary storage

Secondary storage devices like Magnetic Disk are present at level 3. They are used as backup storage. They are cheaper than main memory and larger in size generally in a few TB.

Level-4 − Tertiary storage

Tertiary storage devices like magnetic tape are present at level 4. They are used to store removable files and are the cheapest and largest in size (1-20 TB).

Why memory Hierarchy is used in systems?

Memory hierarchy is arranging different kinds of storage present on a computing device based on speed of access. At the very top, the highest performing storage is CPU registers which are the fastest to read and write to. Next is cache memory followed by conventional DRAM memory, followed by disk storage with different levels of performance including SSD, optical and magnetic disk drives.

2)Memory function:

- Allocate and de-allocate memory before and after process execution.
- To keep track of used memory space by processes.
- To minimize fragmentation issues.
- To proper utilization of main memory.
- To maintain data integrity while executing of process.

3)Mono Programming model: Monoprogramming, also called monoexecution, is an operating system in which only one program is executed at a time and no other is executed until the previous one is finished.In other words, only one program is executed at a time within a single address space.This type of execution practically does not exist anymore because it has been replaced by multiprogramming.
In the monoprogramming, the memory only contains one program at any time. In the case of monoprogramming, when the CPU is running the program and it find an I/O operation (input-output), then the program goes to the I/O devices, during that time the CPU remains idle.Therefore, in the monogramming the CPU is not used effectively, that is, CPU utilization is poor.

4)Multi Programming model:

A multiprogramming operating system may run many programs on a single processor computer. If one program must wait for an input/output transfer in a multiprogramming operating system, the other programs are ready to use the CPU. As a result, various jobs may share CPU time. However, the execution of their jobs is not defined to be at the same time period.

When a program is being performed, it is known as a **"Task", "Process"**, and **"Job"**. Concurrent program executions improve system resource consumption and throughput as compared to serial and batch processing systems.

The primary goal of multiprogramming is to manage the entire system's resources. The key components of a multiprogramming system are the file system, command processor, transient area, and I/O control system. As a result, multiprogramming operating systems are designed to store different programs based on sub-segmenting parts of the transient area. The resource management routines are linked with the operating system core functions.

Advantages

There are various advantages of the multiprogramming operating system. Some of the advantages are as follows:

1. It provides less response time.
2. It may help to run various jobs in a single application simultaneously.
3. It helps to optimize the total job throughput of the computer.
4. Various users may use the multiprogramming system at once.
5. Short-time jobs are done quickly in comparison to long-time jobs.
6. It may help to improve turnaround time for short-time tasks.
7. It helps in improving CPU utilization and never gets idle.

8. The resources are utilized smartly.

Disadvantages

There are various disadvantages of the multiprogramming operating system. Some of the disadvantages are as follows:

1. It is highly complicated and sophisticated.
2. The CPU scheduling is required.
3. Memory management is needed in the operating system because all types of tasks are stored in the main memory.
4. The harder task is to handle all processes and tasks.
5. If it has a large number of jobs, then long-term jobs will require a long wait.

5)Sharing and Protection:

Sharing – A protection mechanism must have to allow several processes to access the same portion of main memory. Allowing each processes access to the same copy of the program rather than have their own separate copy has an advantage.

For example, multiple processes may use the same system file and it is natural to load one copy of the file in main memory and let it shared by those processes. It is the task of Memory management to allow controlled access to the shared areas of memory without compromising the protection. Mechanisms are used to support relocation supported sharing capabilities.

Protection – There is always a danger when we have multiple programs at the same time as one program may write to the address space of another program. So every process must be protected against unwanted interference when other process tries to write in a process whether accidental or incidental. Between relocation and protection requirement a trade-off occurs as the satisfaction of relocation requirement increases the difficulty of satisfying the protection requirement.

Prediction of the location of a program in main memory is not possible, that's why it is impossible to check the absolute address at compile time to assure protection. Most of the programming language allows the dynamic calculation of address at run time. The memory protection requirement must be satisfied by the processor rather than the operating system because the operating system can hardly control a process when it occupies the processor. Thus it is possible to check the validity of memory references.

6)Static and dynamic partition:

Fixed(Static) Partitioning:

The earliest and one of the simplest technique which can be used to load more than one processes into the main memory is Fixed partitioning or Contiguous memory allocation.
In this technique, the main memory is divided into partitions of equal or different sizes. The operating system always resides in the first partition while the other partitions can be used to store user processes. The memory is assigned to the processes in contiguous way.

In fixed partitioning:

1. The partitions cannot overlap.
2. A process must be contiguously present in a partition for the execution.

There are various cons of using this technique.

1. Internal Fragmentation

If the size of the process is lesser then the total size of the partition then some size of the partition get wasted and remain unused. This is wastage of the memory and called internal fragmentation.

As shown in the image below, the 4 MB partition is used to load only 3 MB process and the remaining 1 MB got wasted.

2. External Fragmentation

The total unused space of various partitions cannot be used to load the processes even though there is space available but not in the contiguous form.

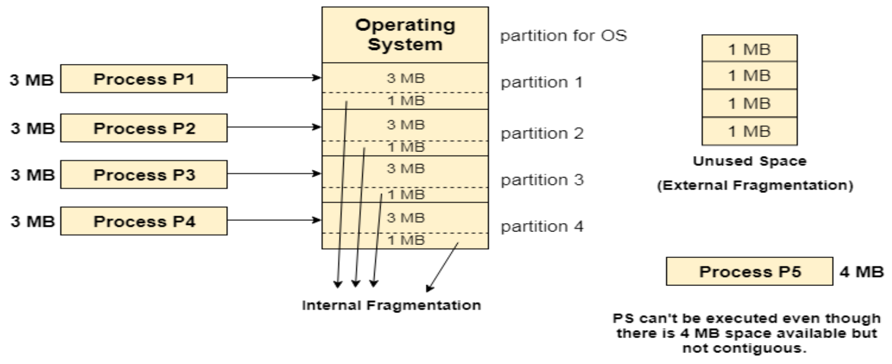
As shown in the image below, the remaining 1 MB space of each partition cannot be used as a unit to store a 4 MB process. Despite of the fact that the sufficient space is available to load the process, process will not be loaded.

3. Limitation on the size of the process

If the process size is larger than the size of maximum sized partition then that process cannot be loaded into the memory. Therefore, a limitation can be imposed on the process size that is it cannot be larger than the size of the largest partition.

4. Degree of multiprogramming is less

By Degree of multi programming, we simply mean the maximum number of processes that can be loaded into the memory at the same time. In fixed partitioning, the degree of multiprogramming is fixed and very less due to the fact that the size of the partition cannot be varied according to the size of processes.



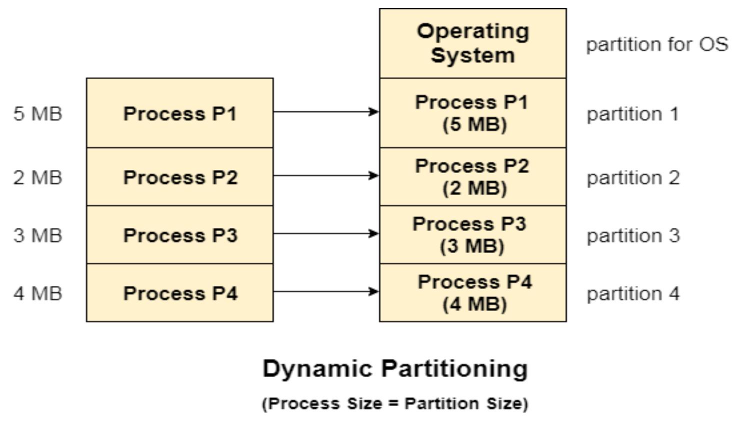
Fixed Partitioning

(Contiguous memory allocation)

Dynamic Partitioning:

Dynamic partitioning tries to overcome the problems caused by fixed partitioning. In this technique, the partition size is not declared initially. It is declared at the time of process loading.

The first partition is reserved for the operating system. The remaining space is divided into parts. The size of each partition will be equal to the size of the process. The partition size varies according to the need of the process so that the internal fragmentation can be avoided.



Advantages of Dynamic Partitioning over fixed partitioning:

1. No Internal Fragmentation

Given the fact that the partitions in dynamic partitioning are created according to the need of the process, It is clear that there will not be any internal fragmentation because there will not be any unused remaining space in the partition.

2. No Limitation on the size of the process

In Fixed partitioning, the process with the size greater than the size of the largest partition could not be executed due to the lack of sufficient contiguous memory. Here, In Dynamic partitioning, the process size can't be restricted since the partition size is decided according to the process size.

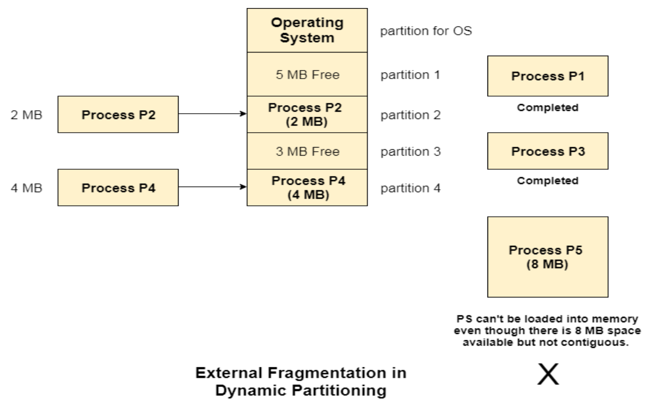
3. Degree of multiprogramming is dynamic

Due to the absence of internal fragmentation, there will not be any unused space in the partition hence more processes can be loaded in the memory at the same time.

Disadvantages of dynamic partitioning

a)External Fragmentation

Absence of internal fragmentation doesn't mean that there will not be external fragmentation.
Let's consider three processes P1 (1 MB) and P2 (3 MB) and P3 (1 MB) are being loaded in the respective partitions of the main memory.
After some time P1 and P3 got completed and their assigned space is freed. Now there are two unused partitions (1 MB and 1 MB) available in the main memory but they cannot be used to load a 2 MB process in the memory since they are not contiguously located.
The rule says that the process must be contiguously present in the main memory to get executed. We need to change this rule to avoid external fragmentation.



External Fragmentation in Dynamic Partitioning

b)Complex Memory Allocation

In Fixed partitioning, the list of partitions is made once and will never change but in dynamic partitioning, the allocation and deallocation is very complex since the partition size will be varied every time when it is assigned to a new process. OS has to keep track of all the partitions.

Due to the fact that the allocation and deallocation are done very frequently in dynamic memory allocation and the partition size will be changed at each time, it is going to be very difficult for OS to manage everything.

7)Internal and External fragmentation:

Fragmentation

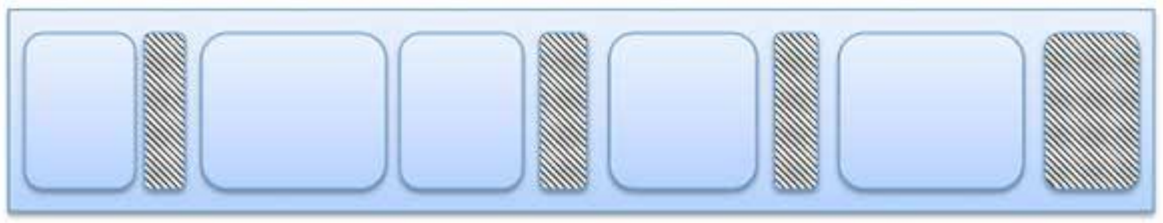
As processes are loaded and removed from memory, the free memory space is broken into little pieces. It happens after sometimes that processes cannot be allocated to memory blocks considering their small size and memory blocks remains unused. This problem is known as Fragmentation.

Fragmentation is of two types −

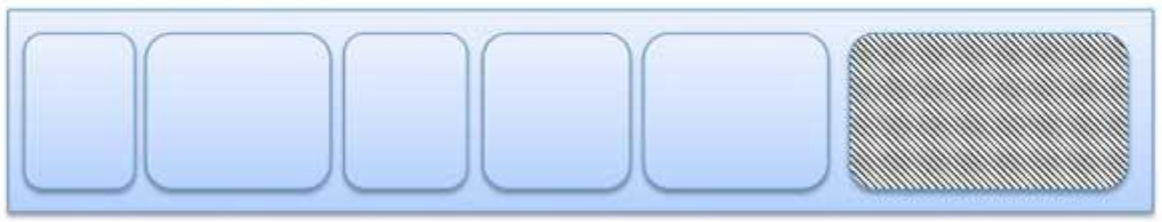
S.N.	Fragmentation & Description
1	**External fragmentation** Total memory space is enough to satisfy a request or to reside a process in it, but it is not contiguous, so it cannot be used.
2	**Internal fragmentation** Memory block assigned to process is bigger. Some portion of memory is left unused, as it cannot be used by another process.

The following diagram shows how fragmentation can cause waste of memory and a compaction technique can be used to create more free memory out of fragmented memory −

Fragmented memory before compaction



Memory after compaction



External fragmentation can be reduced by compaction or shuffle memory contents to place all free memory together in one large block. To make compaction feasible, relocation should be dynamic.

The internal fragmentation can be reduced by effectively assigning the smallest partition but large enough for the process.

8)Concept of Virtual memory:

A computer can address more memory than the amount physically installed on the system. This extra memory is actually called **virtual memory** and it is a section of a hard disk that's set up to emulate the computer's RAM.

The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory serves two purposes. First, it allows us to extend the use of physical memory by using disk. Second, it allows us to have memory protection, because each virtual address is translated to a physical address.

Following are the situations, when entire program is not required to be loaded fully in main memory.

- User written error handling routines are used only when an error occurred in the data or computation.
- Certain options and features of a program may be used rarely.
- Many tables are assigned a fixed amount of address space even though only a small amount of the table is actually used.
- The ability to execute a program that is only partially in memory would counter many benefits.
- Less number of I/O would be needed to load or swap each user program into memory.
- A program would no longer be constrained by the amount of physical memory that is available.
- Each user program could take less physical memory, more programs could be run the same time, with a corresponding increase in CPU utilization and throughput.

Paging:

A computer can address more memory than the amount physically installed on the system. This extra memory is actually called virtual memory and it is a section of a hard that's set up to emulate the computer's RAM. Paging technique plays an important role in implementing virtual memory.
Paging is a memory management technique in which process address space is broken into blocks of the same size called pages (size is power of 2, between 512 bytes and 8192 bytes). The size of the process is measured in the number of pages.

Similarly, main memory is divided into small fixed-sized blocks of (physical) memory called frames and the size of a frame is kept the same as that of a page to have optimum utilization of the main memory and to avoid external fragmentation.

Advantages and Disadvantages of Paging

Here is a list of advantages and disadvantages of paging −

- Paging reduces external fragmentation, but still suffer from internal fragmentation.
- Paging is simple to implement and assumed as an efficient memory management technique.
- Due to equal size of the pages and frames, swapping becomes very easy.
- Page table requires extra memory space, so may not be good for a system having small RAM.

Unit 4: Deadlock Management

1)Introduction to deadlock:

Every process needs some resources to complete its execution. However, the resource is granted in a sequential order.

1. The process requests for some resource.
2. OS grant the resource if it is available otherwise let the process waits.
3. The process uses it and release on the completion.

A Deadlock is a situation where each of the computer process waits for a resource which is being assigned to some another process. In this situation, none of the process gets executed since the resource it needs, is held by some other process which is also waiting for some other resource to be released.

Let us assume that there are three processes P1, P2 and P3. There are three different resources R1, R2 and R3. R1 is assigned to P1, R2 is assigned to P2 and R3 is assigned to P3.

After some time, P1 demands for R1 which is being used by P2. P1 halts its execution since it can't complete without R2. P2 also demands for R3 which is being used by P3. P2 also stops its execution because it can't continue without R3. P3 also demands for R1 which is being used by P1 therefore P3 also stops its execution.

2)Necessary Conditions for Deadlock:

a)Mutual Exclusion: A resource can only be shared in mutually exclusive manner. It implies, if two process cannot use the same resource at the same time.

b)Hold and Wait: A process waits for some resources while holding another resource at the same time.

c)No Preemption: The process which once scheduled will be executed till the completion. No other process can be scheduled by the scheduler meanwhile.

d)Circular wait: All the processes must be waiting for the resources in a cyclic manner so that the last process is waiting for the resource which is being held by the first process.

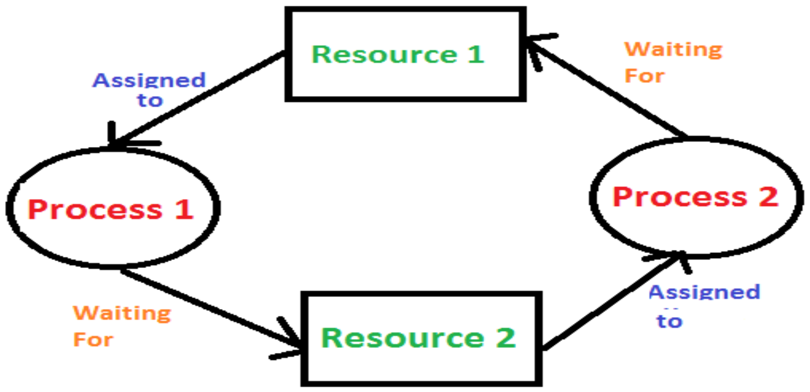
3)Methods for handling deadlock:

a)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.

b)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.

c)Deadlock detection:

Deadlock Detection :
1. If resources have a single instance –
In this case for Deadlock detection, we can run an algorithm to check for the cycle in the Resource Allocation Graph. The presence of a cycle in the graph is a sufficient condition for deadlock.



2. In the above diagram, resource 1 and resource 2 have single instances. There is a cycle R1 → P1 → R2 → P2. So, Deadlock is Confirmed.

3. If there are multiple instances of resources –
Detection of the cycle is necessary but not sufficient condition for deadlock detection, in this case, the system may or may not be in deadlock varies according to different situations.

d)Recovery from deadlock:
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.

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.

Unit 5: Concept of File Management

5.1 Introduction to file management

A file management system is used for file maintenance (or management) operations. It is a type of software that manages data files in a computer system.

A file management system has limited capabilities and is designed to manage individual or group files, such as special office documents and records. It may display report details, like owner, creation date, state of completion and similar features useful in an office environment.

5.2 File naming:

In File naming typically involves giving a name to a file or folder to make it easy to identify and access. Each OS has its own rules for file naming conventions, but there are some general guidelines that apply to most OSs:

a) Use letters, numbers, and underscores: Avoid using special characters such as slashes, backslashes, colons, or question marks, as these can cause issues when working with files.

b) Be mindful of file extensions: File extensions are typically used to identify the type of file and determine which application is used to open it. In some OSs, the extension is automatically added when the file is saved.

c) Keep file names short: Some OSs have limits on the length of file names, so it's best to keep them short and meaningful.

d) Use descriptive names: Names that accurately describe the contents of the file or folder will help you and others find it more easily.

e) Be consistent: Follow a consistent naming convention for all files and folders to keep them organized.

f) Avoid using reserved words: Some OSs have reserved words that are used for system files or commands. Avoid using these words in your file names.

5.3 File operation:

A file is a collection of logically related data that is recorded on the secondary storage in the form of sequence of operations. The content of the files are defined by its creator who is creating the file. The various operations which can be implemented on a file such as read, write, open and close etc. are called file operations. These operations are performed by the user by using the commands provided by the operating system. Some common operations are as follows:

1.Create operation:

This operation is used to create a file in the file system. It is the most widely used operation performed on the file system. To create a new file of a particular type the associated application program calls the file system. This file system allocates space to the file. As the file system knows the format of directory structure, so entry of this new file is made into the appropriate directory.

2. Open operation:

This operation is the common operation performed on the file. Once the file is created, it must be opened before performing the file processing operations. When the user wants to open a file, it provides a file name to open the particular file in the file system. It tells the operating system to invoke the open system call and passes the file name to the file system.

3. Write operation:

This operation is used to write the information into a file. A system call write is issued that specifies the name of the file and the length of the data has to be written to the file. Whenever the file length is increased by specified value and the file pointer is repositioned after the last byte written.

4. Read operation:

This operation reads the contents from a file. A Read pointer is maintained by the OS, pointing to the position up to which the data has been read.

5. Re-position or Seek operation:

The seek system call re-positions the file pointers from the current position to a specific place in the file i.e. forward or backward depending upon the user's requirement. This operation is generally performed with those file management systems that support direct access files.

6. Delete operation:

Deleting the file will not only delete all the data stored inside the file it is also used so that disk space occupied by it is freed. In order to delete the specified file the directory is searched. When the directory entry is located, all the associated file space and the directory entry is released.

7. Truncate operation:

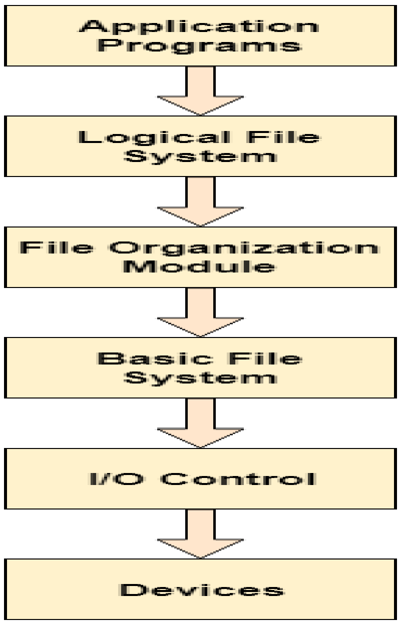
Truncating is simply deleting the file except deleting attributes. The file is not completely deleted although the information stored inside the file gets replaced.

8. Close operation:

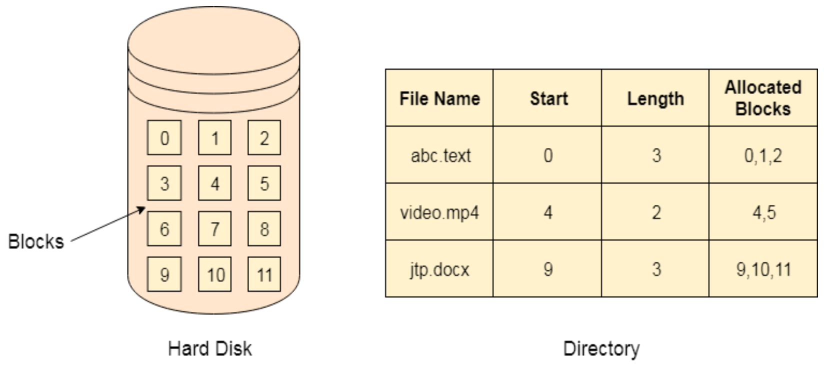
When the processing of the file is complete, it should be closed so that all the changes made permanent and all the resources occupied should be released. On closing it deallocates all the internal descriptors that were created when the file was opened.

9. Append operation:
This operation adds data to the end of the file.
10. Rename operation:
This operation is used to rename the existing file.

5.4 File extension: A file extension is an identifier used as a suffix to a name of the computer file in an operating system such as Microsoft Windows. It can be categorized as a type of metadata. A file extension helps the operating system to understand the characteristics of the file, and to some extent, its intended use.
A complete filename includes the name of the file as well as its extension. It is usually three or four characters long, although in rare cases it could be one or two, and it is used as a suffix to the file name. It is often separated from the filename with the help of the dot (.) symbol. Some operating systems include the file extension as part of the file system itself, thus limiting the format and length of the extension. In the case of Windows operating systems, file extensions are usually hidden from users. File extensions can be renamed, however simply renaming a file extension does not necessarily convert one file format to another.
5.5 File system layout:
File System provide efficient access to the disk by allowing data to be stored, located and retrieved in a convenient way. A file System must be able to store the file, locate the file and retrieve the file.
Most of the Operating Systems use layering approach for every task including file systems. Every layer of the file system is responsible for some activities.
The image shown below, elaborates how the file system is divided in different layers, and also the functionality of each layer.



5.6 **File allocation**:
a)Contiguous Allocation
If the blocks are allocated to the file in such a way that all the logical blocks of the file get the contiguous physical block in the hard disk then such allocation scheme is known as contiguous allocation.
In the image shown below, there are three files in the directory. The starting block and the length of each file are mentioned in the table. We can check in the table that the contiguous blocks are assigned to each file as per its need.



File Name	Start	Length	Allocated Blocks
abc.text	0	3	0,1,2
video.mp4	4	2	4,5
jtp.docx	9	3	9,10,11

Contiguous Allocation

Advantages

I. It is simple to implement.
II. We will get Excellent read performance.
III. Supports Random Access into files.

Disadvantages

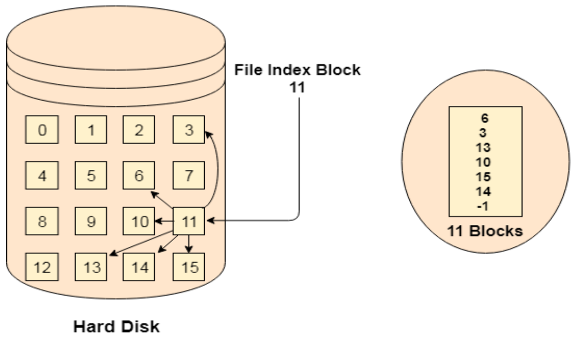
I. The disk will become fragmented.
II. It may be difficult to have a file grow.

b)Indexed Allocation

Limitation in the existing technology causes the evolution of a new technology. Till now, we have seen various allocation methods each of them was carrying several advantages and disadvantages.
File allocation table tries to solve as many problems as possible but leads to a drawback. The more the number of blocks, the more will be the size of FAT.

Indexed Allocation Scheme

Instead of maintaining a file allocation table of all the disk pointers, Indexed allocation scheme stores all the disk pointers in one of the blocks called as indexed block. Indexed block doesn't hold the file data, but it holds the pointers to all the disk blocks allocated to that particular file. Directory entry will only contain the index block address.



Advantages

I. Supports direct access
II. A bad data block causes the lost of only that block.

Disadvantages

I. A bad index block could cause the lost of entire file.
II. Size of a file depends upon the number of pointers, a index block can hold.
III. Having an index block for a small file is totally wastage.
IV. More pointer overhead

5.7 Free space management:

A file system is responsible to allocate the free blocks to the file therefore it has to keep track of all the free blocks present in the disk. There are mainly two approaches by using which, the free blocks in the disk are managed.

1. Bit Vector

In this approach, the free space list is implemented as a bit map vector. It contains the number of bits where each bit represents each block.

If the block is empty then the bit is 1 otherwise it is 0. Initially all the blocks are empty therefore each bit in the bit map vector contains 1.

2. Linked List

It is another approach for free space management. This approach suggests linking together all the free blocks and keeping a pointer in the cache which points to the first free block.

Therefore, all the free blocks on the disks will be linked together with a pointer. Whenever a block gets allocated, its previous free block will be linked to its next free block.

Unit 6: Linux

6.1 Introduction to Linux:

Linux is a free and open-source operating system based on the Unix operating system. It was created by Linus Torvalds in 1991 and has since grown into a widely-used operating system for servers, desktops, and other computing devices. Linux is known for its stability, security, and flexibility, and is popular among developers and tech enthusiasts. It comes in many different "flavors" or distributions, each with its own set of tools and user interface. Some popular distributions include Ubuntu, Debian, Red Hat, and Fedora. Linux is typically used through a command-line interface, although many distributions also offer a graphical user interface.

6.2 Features of Linux

Some of the key features of Linux are:

I. Open-source: Linux is an open-source operating system, which means that its source code is freely available for anyone to view, modify and distribute.
II. Customizability: Linux is highly customizable, allowing users to modify the operating system to fit their specific needs and preferences.
III. Stability: Linux is known for its stability and reliability, with many servers and critical systems running on Linux.
IV. Security: Linux has a strong security model, with built-in features like user account controls, file permissions, and network security.
V. Flexibility: Linux can be run on a wide range of hardware, from small embedded devices to large servers.
VI. Command-line interface: Linux is typically used through a command-line interface, which allows for more powerful and precise control over the system.
VII. Compatibility: Linux can run a wide range of software, including many popular applications and programming languages.
VIII. Community support: Linux has a large and active community of users and developers, who provide support and contribute to the ongoing development of the operating system.

6.3 Advantages Linux:

1. Pen Source: As it is open-source, its source code is easily available. Anyone having programming knowledge can customize the operating system. One can contribute, modify, distribute, and enhance the code for any purpose.

2. Security: The Linux security feature is the main reason that it is the most favorable option for developers. It is not completely safe, but it is less vulnerable than others. Each application needs to authorize by the admin user. The virus is not executed until the administrator provides the access password. Linux systems do not require any antivirus program.

3. Lightweight: Linux is lightweight. The requirements for running Linux are much less than other operating systems. In Linux, the memory footprint and disk space are also lower. Generally, most of the Linux distributions required as little as 128MB of RAM around the same amount for disk space.

4. Performance: Linux system provides high performance over different networks. It is capable of handling a large number of users simultaneously.
5. Flexibility: Linux operating system is very flexible. It can be used for desktop applications, embedded systems, and server applications too. It also provides various restriction options for specific computers. We can install only necessary components for a system.
6. Graphical User Interface: Linux is a command-line based OS but, it provides an interactive user interface like Windows.
7. Networking: Linux facilitates with powerful support for networking. The client-server systems can be easily set to a Linux system. It provides various command-line tools such as ssh, ip, mail, telnet, and more for connectivity with the other systems and servers.
8. Multitasking: It is a multitasking operating system as it can run multiple tasks simultaneously without affecting the system speed.

Disadvantages of Linux Operating System
1. Adaptation: For people who are less expertise in computers it can be hard to understand Linux. Most users find it difficult to adopt to Linux due to the terminals used. Terminals are command line interface where you need to enter specific command in order to complete tasks.
2. Software Compatibility: Popular applications which are made for Windows and Mac are not available for Linux. Many developers are not interested in making softwares for Linux due to its small market value.
3. Gaming: Similar to sofwares, games too doesn't natively support Linux. Because Linux is not a platform which is not widely used, gaming developers are not much interested in Linux. So you cannot expect your favorite game to run on Linux.
4. Hardware Compatibility: Almost all the hardwares can be connected to a Linux operating system. But the problem comes with the drivers. The concept of drivers in Linux is very different from other operating system. The drivers here are inbuilt inside the Kernel.
5. Technical Support: Since Linux is an open source operating system it lacks on the side of technical support. The problem cannot be rectified unless you find a solution yourself.

6.4 Linux family:

a) Debian: Debian is renowned for being a mother to popular Linux distributions such as **Deepin**, **Ubuntu,** and **Mint** which have provided solid performance, stability, and unparalleled user experience. The latest stable release is **Debian 10.5**, an update of **Debian 10** colloquially known as **Debian Buster**.

b)Ubuntu:Created and maintained by **Canonical**, **Ubuntu** is one of the most popular Linux distros enjoyed across the globe by beginners, intermediate users, and professionals alike. **Ubuntu** was specifically designed for beginners in Linux or those transitioning from mac and Windows.

c)Kali Linux:Kali Linux (formerly known as BackTrack) is a free and open source Linux operating system specially designed for penetration testing (computer system security) using a wide range of available security tools.

d) Gentoo
Gentoo is a distro built for professional use and experts who take into consideration what packages they are working with from the word go. This category includes developers, system & network administrators. As such, it’s not ideal for beginners in Linux. **Gentoo** comes recommended for those who want to have a deeper understanding of the ins and outs of the Linux operating system.

e) Red Hat Enterprise Linux:
Abbreviated as **RHEL**, Red Hat Enterprise Linux is a Linux distro designed for Enterprise or commercial purposes. It’s one of the leading open-source alternatives to other proprietary systems such as **Microsoft**. **Red Hat** is usually a top choice for server environments given its stability and regular security patches which boost its overall security.

6.5 Difference between windows and Linux

There are several key differences between Windows and Linux, including:

I. Licensing: Windows is a proprietary operating system developed and sold by Microsoft, while Linux is open-source software that can be freely used, modified, and distributed by anyone.
II. User interface: Windows has a more consistent and unified user interface, while Linux offers a wider variety of desktop environments and window managers, giving users more flexibility in how they interact with the system.
III. Software availability: Windows has a larger selection of commercial software applications that are designed to run on the platform, while Linux has a vast array of free and open-source software that can be easily downloaded and installed using package management tools.
IV. Security: Linux is generally considered to be more secure than Windows, due in part to its open-source nature, which allows developers to easily identify and fix security vulnerabilities.
V. Hardware support: Windows has a larger market share and is thus more likely to have drivers available for a wider variety of hardware, while Linux may require more effort to find and install drivers for certain devices.
VI. Command line interface: Linux has a powerful and flexible command line interface, which is often preferred by system administrators and developers, while Windows has a more limited command line interface that is primarily used for basic tasks.

6.6 Structure of Linux

The structure of Linux can be thought of as a hierarchical tree-like structure, with the root directory ("/") at the top and subdirectories branching off from there. The following are some of the key directories in the Linux file system:

i. /bin - Contains essential binary files and commands that are required for the system to boot and run.
ii. /sbin - Contains system administration binaries and commands that are used by the system administrator.
iii. /etc - Contains system-wide configuration files that control the behavior of various system programs and services.
iv. /home - Contains the home directories for regular users, where they can store their personal files and configuration settings.
v. /usr - Contains the bulk of the system files, including the majority of user-level binaries, libraries, documentation, and source code.

6.7 Linux Basic Commands

i. cd: Change directory. Used to change the current working directory. Example: cd /home/user changes the current directory to "/home/user" .

ii. ls: List. Used to list the files and directories in the current working directory. Example: ls lists the files and directories in the current working directory.

iii. pwd: Print working directory. Used to print the current working directory. Example: pwd prints the current working directory.

iv. mkdir: Make directory. Used to create a new directory. Example: mkdir new_directory creates a new directory named "new_directory" .

v. rm: Remove. Used to remove files or directories. Example: rm file.txt removes the file named "file.txt" . Use the "-r" flag to remove a directory and its contents recursively: rm -r directory.

vi. cp: Copy. Used to copy files or directories. Example: cp file.txt new_file.txt creates a copy of "file.txt" called "new_file.txt" .

vii. mv: Move. Used to move or rename files or directories. Example: mv file.txt new_directory/ moves "file.txt" to the directory "new_directory" . Use the "rename" functionality by using mv oldname.txt newname.txt.

viii. cat: Concatenate. Used to display the contents of a file. Example: cat file.txt displays the contents of "file.txt".

ix. man: Manual. Used to display the manual page for a command. Example: man ls displays the manual page for the "ls" command.