UR ENGINEERING FRIEND

#1 stop Destination for Diploma & Degree

Live Seminar

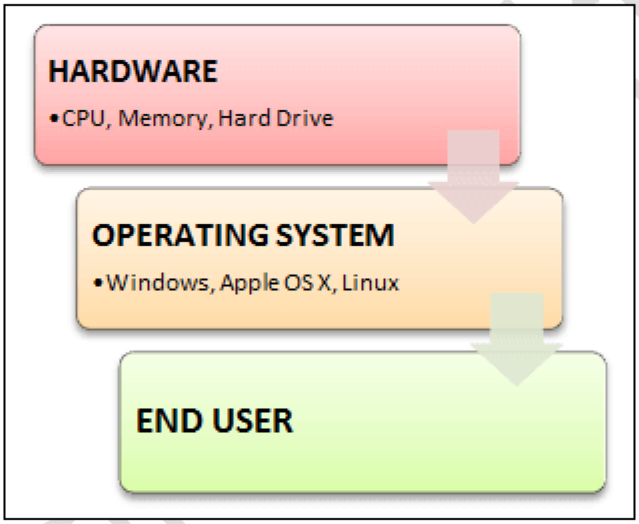
Subject – Operating System

Date – 07th Dec , 2022

Contents

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.



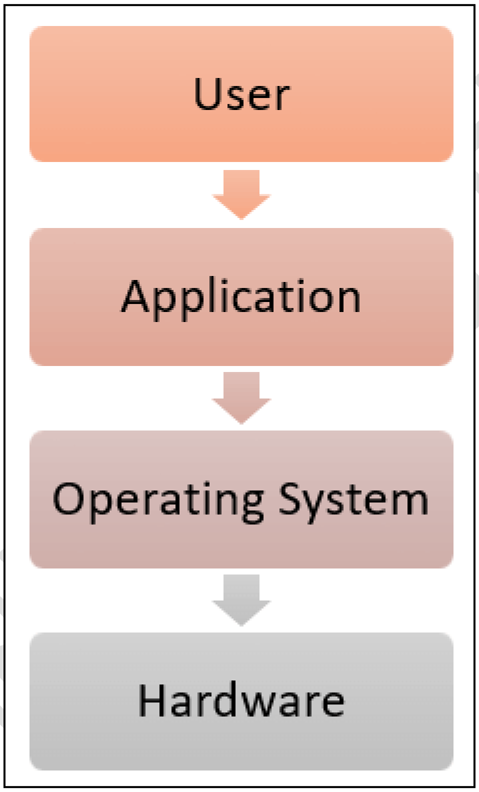
The OS helps you to communicate with the computer without knowing how to speak the computer's language. It is not possible for the user to use any computer or mobile device without having an operating system.

Features of Operating System (OS)

Here is a list important features of 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



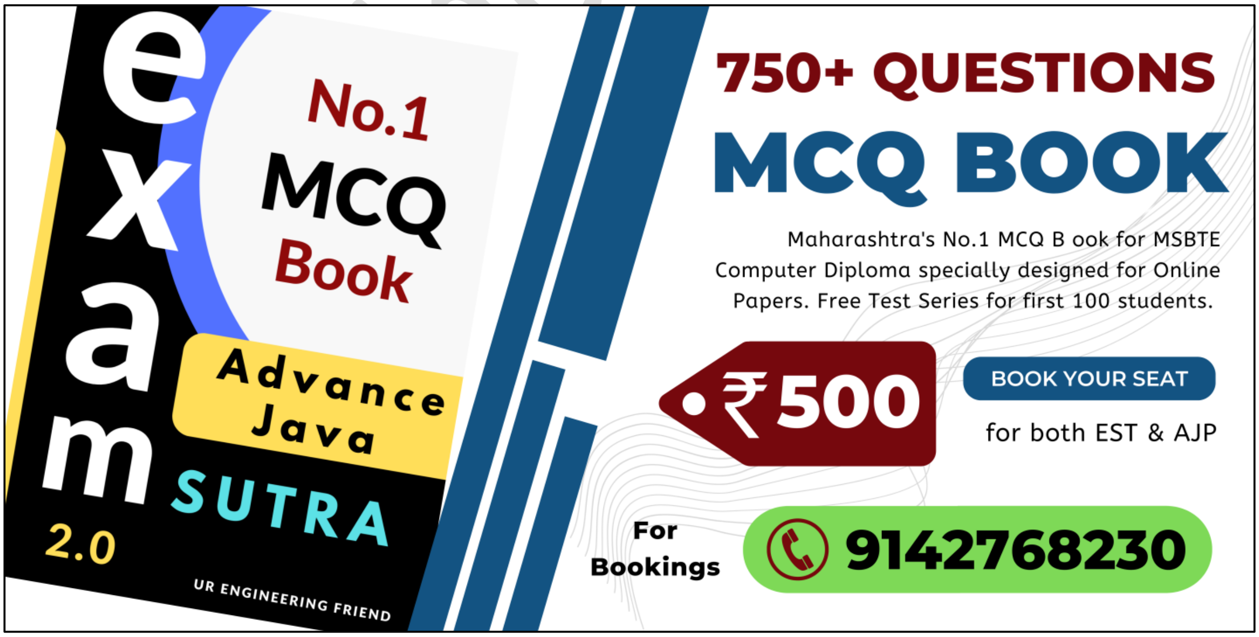
Advantage of Operating System

- 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 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

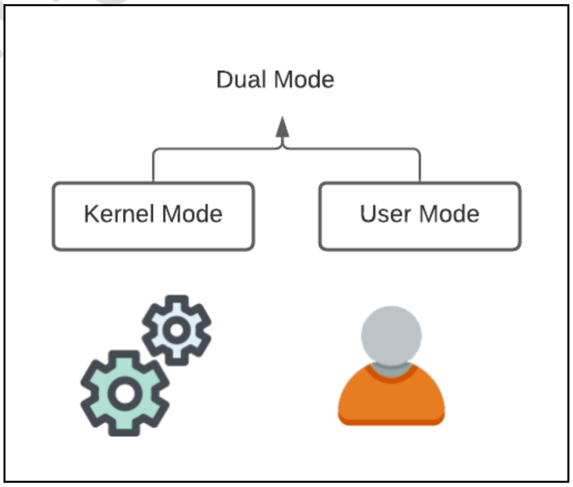


Dual mode operation

The dual-mode operations in the operating system protect the operating system from illegal users. We accomplish this defense by designating some of the system instructions as privileged instructions that can cause harm.

- The hardware only allows for the execution of privileged instructions in kernel mode.
- An example of a privileged instruction is the command to switch to user mode. Other examples include monitoring of I/O, controlling timers and handling interruptions.

An error in one program can adversely affect many processes, it might modify data of another program, or also can affect the operating system. For example, if a process stuck in the infinite loop then this infinite loop could affect the correct operation of other processes. So to ensure the proper execution of the operating system, there are two modes of operation:



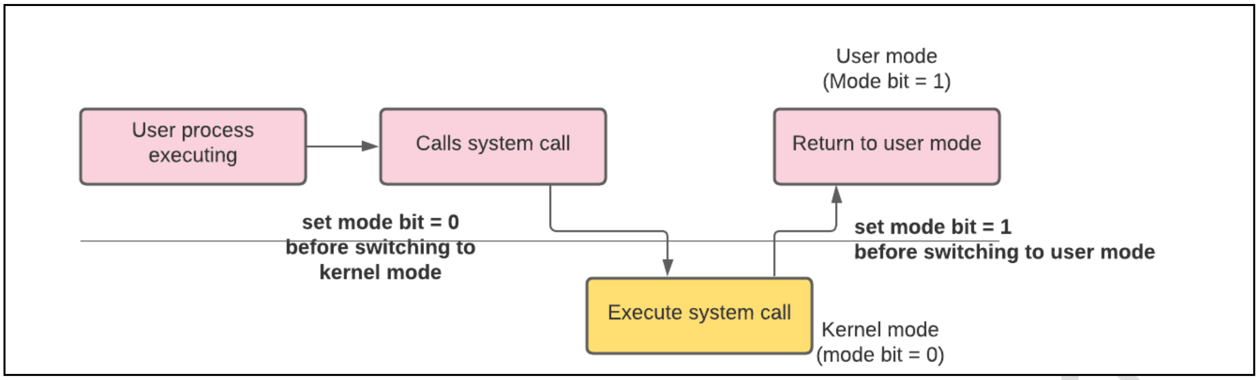
User mode

When the computer system is run by user applications like creating a text document or using any application program, then the system is in user mode. When the user application requests for a service from the operating system or an interrupt occurs or system calls, then there will be a transition from user to kernel mode to fulfill the requests.

Kernel Mode

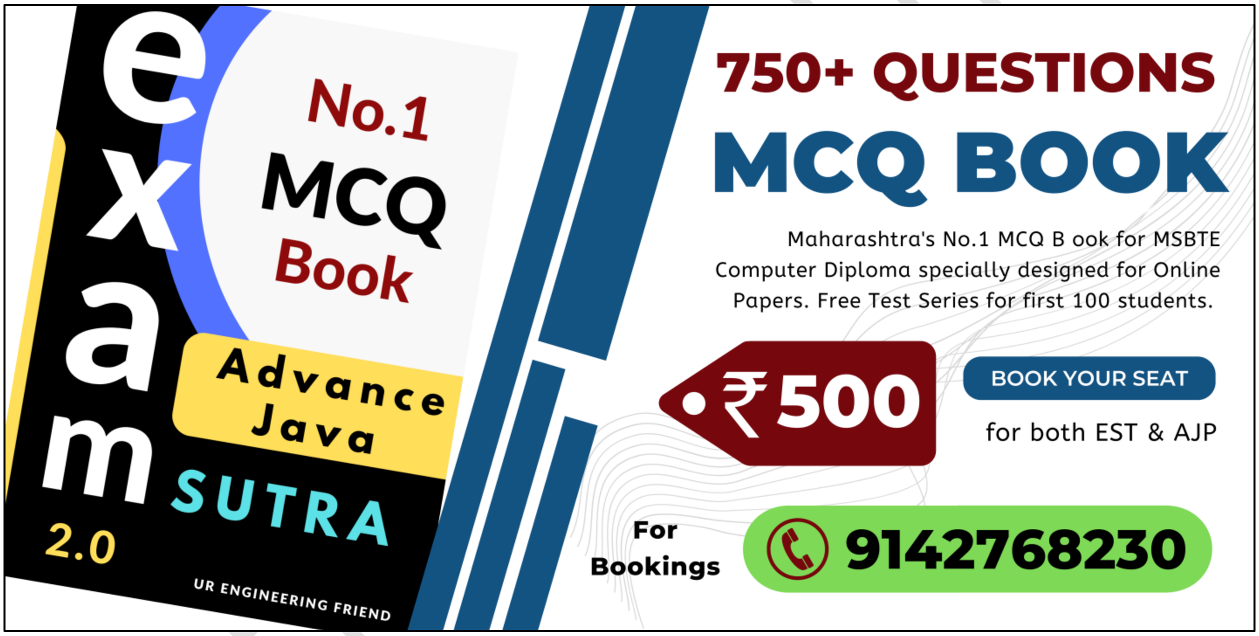
When the system boots, hardware starts in kernel mode and when the operating system is loaded, it starts user application in user mode. To provide protection to the hardware, we have privileged instructions which execute only in kernel mode. If the user attempts to run privileged instruction in user mode then it will treat instruction as illegal and traps to OS. Some of the privileged instructions are:

1. Handling Interrupts
2. To switch from user mode to kernel mode.
3. Input-Output management.



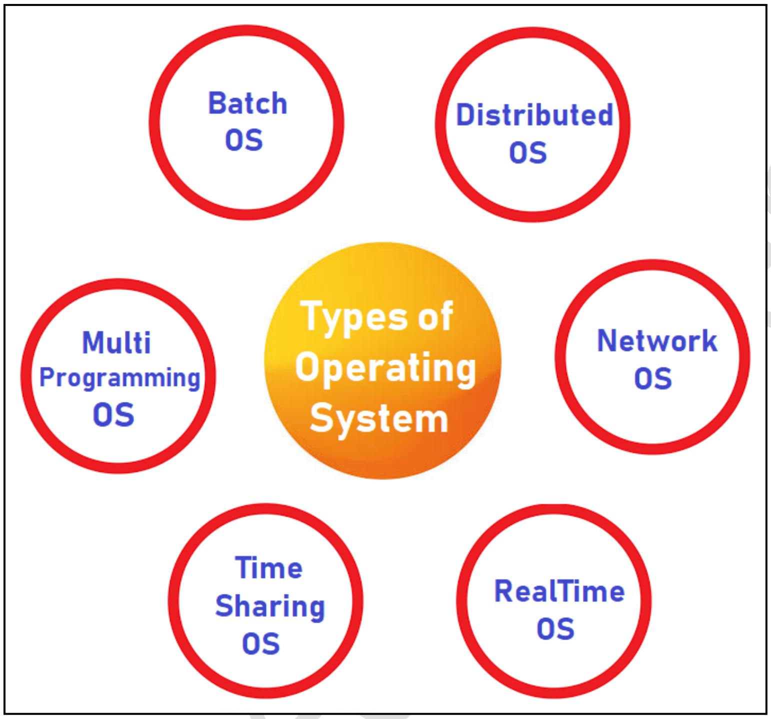
Terms	User Mode	Kernel Mode
Definition	User Mode is a restricted mode in which application programs execute and begin.	Kernel Mode is the privileged mode that the computer enters when it accesses the hardware resources.
Modes	User Mode is also known as slave mode or restricted mode.	Kernel mode is also known as system mode, master mode, or privileged mode.
Interruptions	In User Mode, if an interrupt occurs, only that process fails.	In Kernel Mode, if an interrupt occurs, the operating system might fail.
Address	A process in User mode has its	In Kernel Mode, processes get

space	own address space.	single address space.
Restrictions	In user mode, there are restrictions to access kernel programs. They cannot be accessed directly.	In kernel mode, both user programs and kernel programs can be accessed.



Resource Management

OS Types



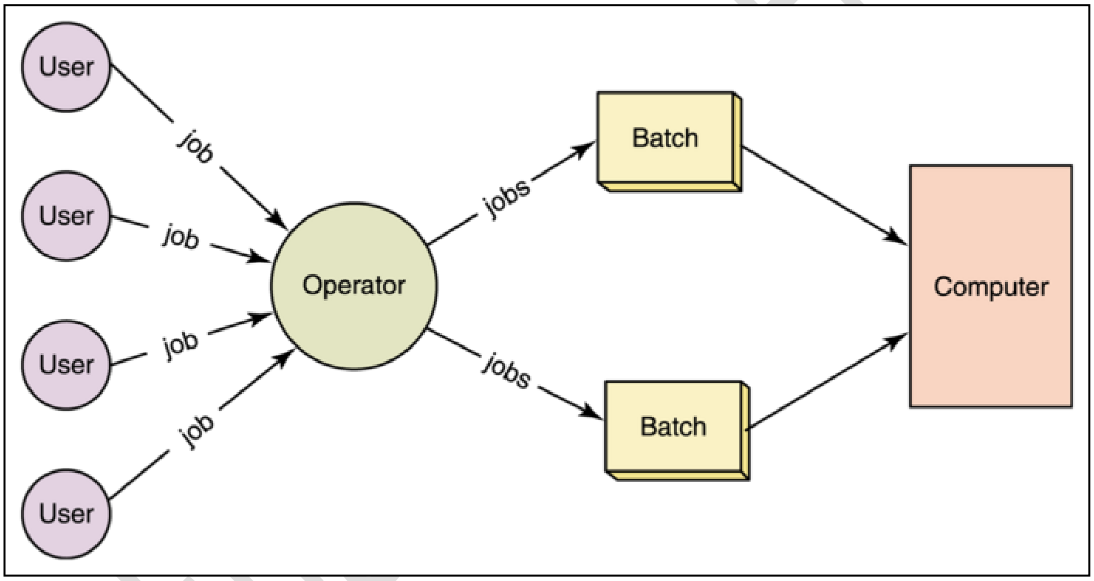
Batch operating System -:

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

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

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

The system put all of the jobs in a queue on the basis of first come first serve and then executes the jobs one by one. The users collect their respective output when all the jobs get executed.



The problems with Batch Systems are as follows −

- Lack of interaction between the user and the job.
- CPU is often idle, because the speed of the mechanical I/O devices is slower than the CPU.
- Difficult to provide the desired priority.

Steps in Batch Operating System -:

Step 1 − Using punch cards the user prepares his job.

Step 2 − After that the user submits the job to the programmer.

Step 3 − The programmer collects the jobs from different users and sorts the jobs into batches with similar needs.

Step 4 − Finally, the programmer submits the batches to the processor one by one.

Step 5 − All the jobs of a single batch are executed together.

Advantages

- The time taken by the system to execute all the programs will be reduced.
- It can be shared between multiple users.

Disadvantages

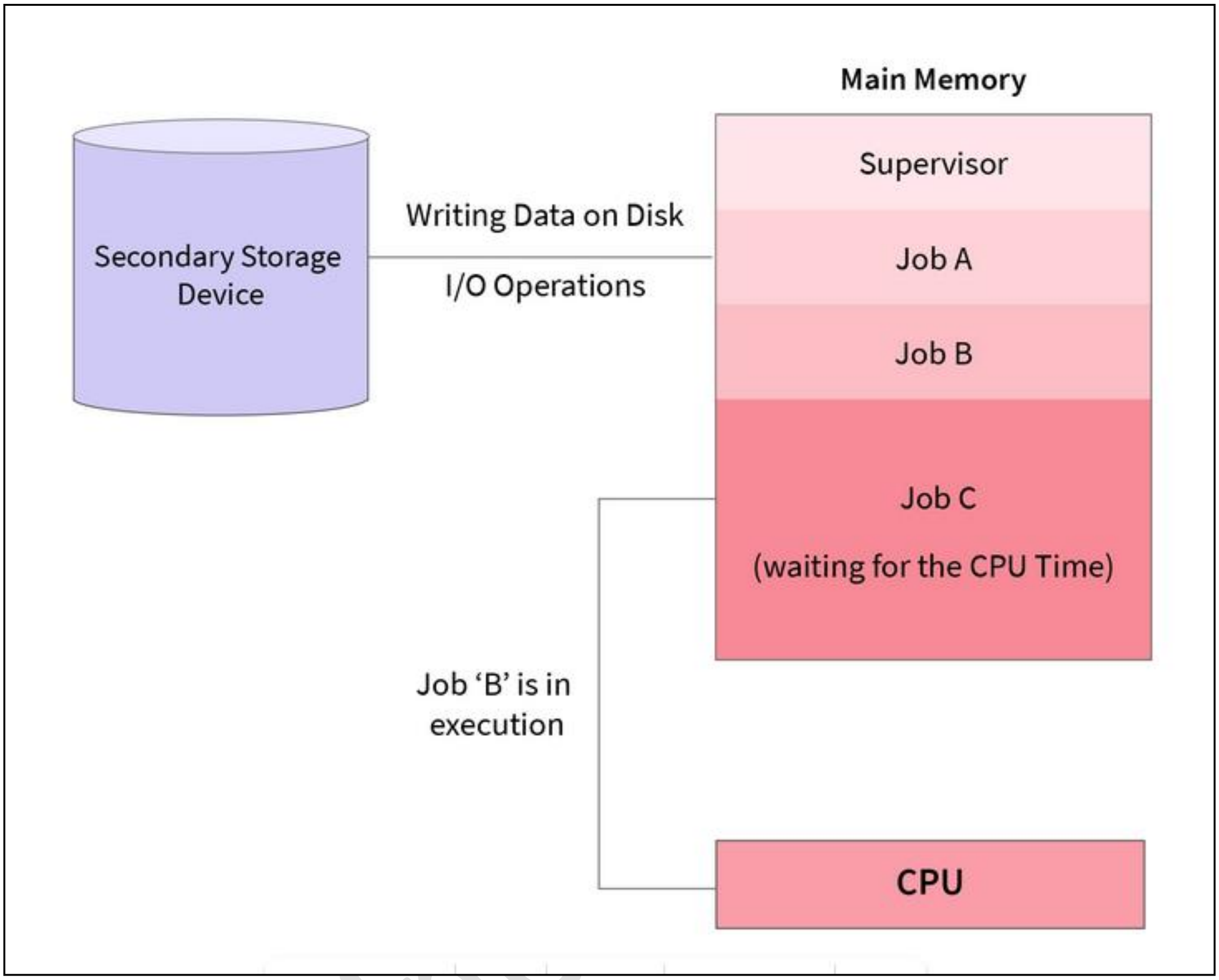
- Manual interrupts are required between two batches.
- Priority of jobs is not set, they are executed sequentially.
- It may lead to starvation.

- The CPU utilization is low and it has to remain ideal for a long time because the time taken in loading and unloading of batches is very high as compared to execution time.

Multi-programming OS

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.
- 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.



Examples of Multiprogramming Operating Systems

- Desktop operating systems, such as Linux distributions, Windows, macOS, etc. These are modern operating systems and incorporate various multiprogramming concepts in their functioning.
- On Android, iOS, and other mobile operating systems on the phone, one can listen to music while also sending and receiving text messages.

- Application software, such as Office, popular web browsers, media players, etc. Any modern web browser would let a user visit multiple websites at the same time by opening as many windows/tabs as required.

Advantages

- CPU utilization is high because the CPU is never goes to idle state.
- Memory utilization is efficient.
- CPU throughput is high and also supports multiple interactive user terminals.

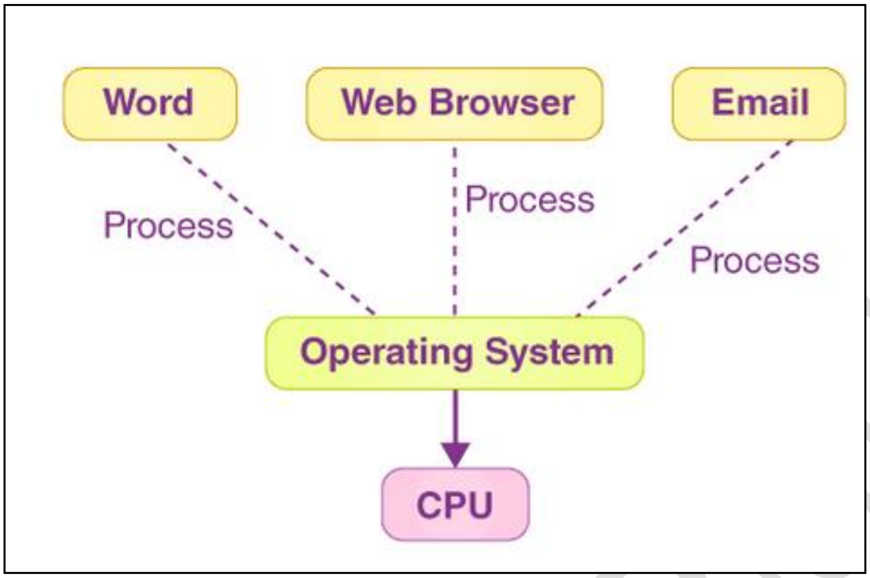
Disadvantages

- CPU scheduling is compulsory because lots of jobs are ready to run on CPU simultaneously.
- User is not able to interact with jobs when it is executing.
- Programmers also cannot modify a program that is being executed.

Time-shared OS

Time sharing is a logical extension of multiprogramming. The CPU performs many tasks by switches are so frequent that the user can interact with each program while it is running. A time shared operating system allows multiple users to share computers simultaneously.

- A **time shared operating system** uses CPU scheduling and multi-programming to provide each user with a small portion of a shared computer at once. Each user has at least one separate program in memory.
- This short period of time during which user gets attention of CPU is known as **time slice, time slot or quantum**.
- It is typically of the order of 10 to 100 milliseconds.
- Time shared operating systems are more complex than multiprogrammed operating systems.
- In both, multiple jobs must be kept in memory simultaneously, so the system must have memory management and security.
- The main difference between Time-Sharing Systems and Multiprogrammed Batch Systems is that in case of Multiprogrammed batch systems, the objective is to maximize processor use, whereas in Time-Sharing Systems, the objective is to minimize response time.
- An operating system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time.



Advantages :

1. Each task gets an equal opportunity.
2. Less chances of duplication of software.
3. CPU idle time can be reduced.

Disadvantages :

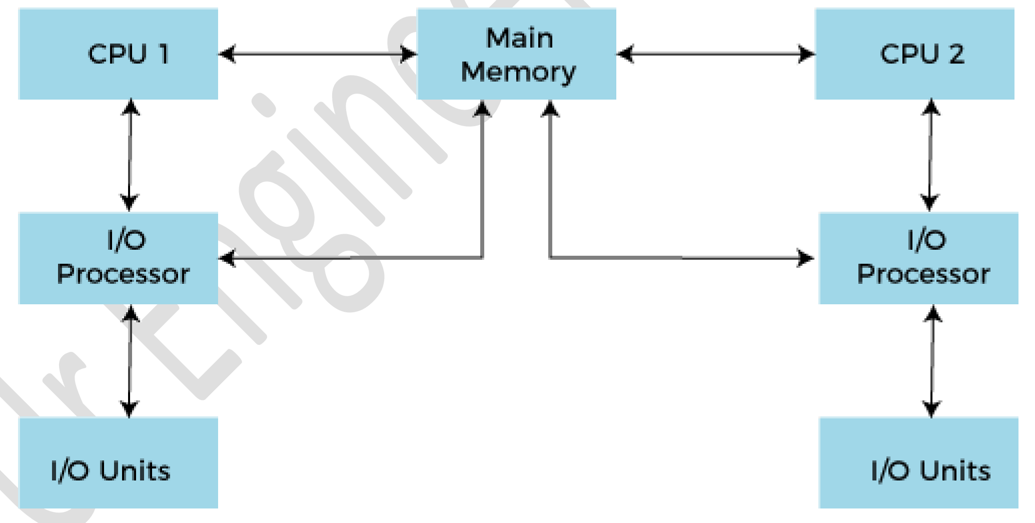
1. Reliability problem.
2. One must have to take of security and integrity of user programs and data.

3. Data communication problem.

Multi-processor OS

In operating systems, to improve the performance of more than one CPU can be used within one computer system called Multiprocessor operating system.

Multiple CPUs are interconnected so that a job can be divided among them for faster execution. When a job finishes, results from all CPUs are collected and compiled to give the final output. Jobs needed to share main memory and they may also share other system resources among themselves. Multiple CPUs can also be used to run multiple jobs simultaneously.



Working of Multiprocessor System

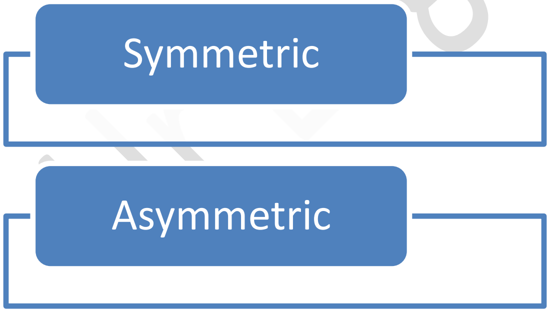
Advantages

- **Increased reliability:** Due to the multiprocessing system, processing tasks can be distributed among several processors. This increases reliability as if one processor fails; the task can be given to another processor for completion.
- **Increased throughout:** As several processors increase, more work can be done in less
- **The economy of Scale:** As multiprocessors systems share peripherals, secondary storage devices, and power supplies, they are relatively cheaper than single-processor systems.

Disadvantages

- Operating system of multiprocessing is more complex and sophisticated as it takes care of multiple CPUs at the same time.

Types of Multi-processor OS -:



Symmetric

Asymmetric

Asymmetric Multiprocessor

Every processor is given seeded tasks in this operating system, and the master processor has the power for running the entire system. In the course, it uses the master-slave relationship.

Symmetric Multiprocessor

In this system, every processor owns a similar copy of the OS, and they can make communication in between one another. All processors are connected with peering relationship nature, meaning it won’t be using master & slave relation.

Real-time OS

Real-time **operating systems (RTOS)** are used in environments where a large number of events, mostly external to the computer system, must be accepted and processed in a short time or within certain deadlines.

such applications are industrial control, telephone switching equipment, flight control, and real-time simulations.

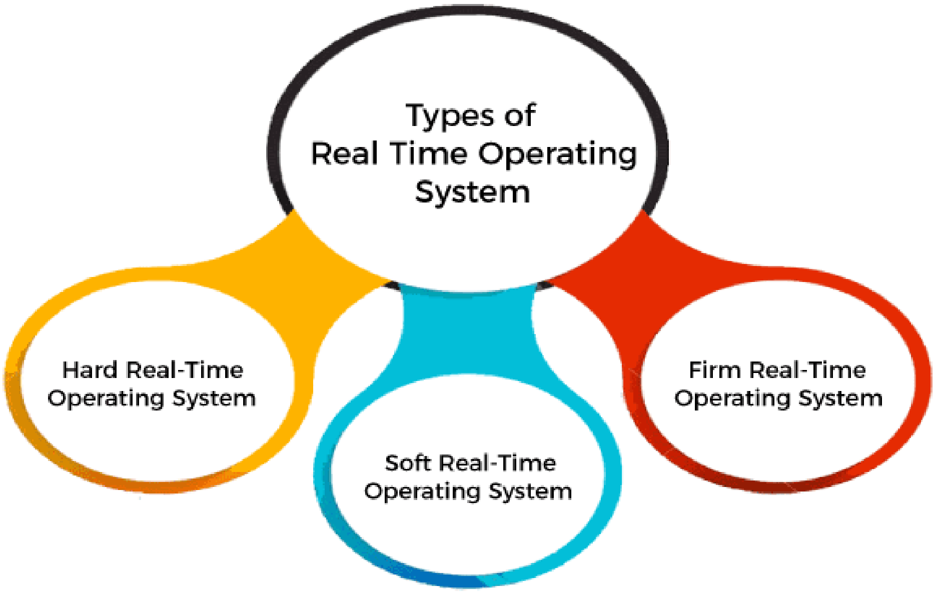
With an RTOS, the processing time is measured in tenths of seconds. This system is time-bound and has a fixed deadline.

The processing in this type of system must occur within the specified constraints

Examples of the real-time operating systems: Airline traffic control systems, Command Control Systems, Airlines reservation system, Heart Pacemaker, Network Multimedia Systems, Robot etc.

Applications of Real-time operating system (RTOS):

- Real-time running structures are used inside the Radar gadget.
- Real-time running structures are utilized in Missile guidance.
- Real-time running structures are utilized in on line inventory trading.
- Real-time running structures are used inside the cell phone switching gadget.
- Real-time running structures are utilized by Air site visitors to manipulate structures.
- Real-time running structures are used in Medical Imaging Systems.
- Real-time running structures are used inside the Fuel injection gadget.
- Real-time running structures are used inside the Traffic manipulate gadget.
- Real-time running structures are utilized in Autopilot travel simulators.



Hard Real-Time operating system -:

In Hard RTOS, all critical tasks must be completed within the specified time duration, i.e., within the given deadline. Not meeting the deadline would result in critical failures such as damage to equipment or even loss of human life.

For Example,

Let's take an example of airbags provided by carmakers along with a handle in the driver's seat. When the driver applies brakes at a particular instance, the airbags grow and prevent the driver's head from hitting the handle. Had there been some delay even of milliseconds, then it would have resulted in an accident.

Soft Real-Time operating system -:

Soft RTOS accepts a few delays via the means of the Operating system. In this kind of RTOS, there may be a closing date assigned for a particular job, but a delay for a small amount of time is acceptable. So, cut off dates are treated softly via means of this kind of RTOS.

For Example,

This type of system is used in Online Transaction systems and Livestock price quotation Systems.

Firm Real-Time operating system:

In Firm RTOS additionally want to observe the deadlines. However, lacking a closing date might not have a massive effect, however may want to purposely undesired effects, like a massive discount within the fine of a product.

For Example, this system is used in various forms of Multimedia applications.

Advantages

- Easy to layout, develop and execute real-time applications under the real-time operating system.
- The real-time working structures are extra compact, so those structures require much less memory space.
- In a Real-time operating system, the maximum utilization of devices and systems.
- Focus on running applications and less importance to applications that are in the queue.
- Since the size of programs is small, RTOS can also be embedded systems like in transport and others.

Disadvantages

- Real-time operating systems have complicated layout principles and are very costly to develop.
- Real-time operating systems are very complex and can consume critical CPU cycles.

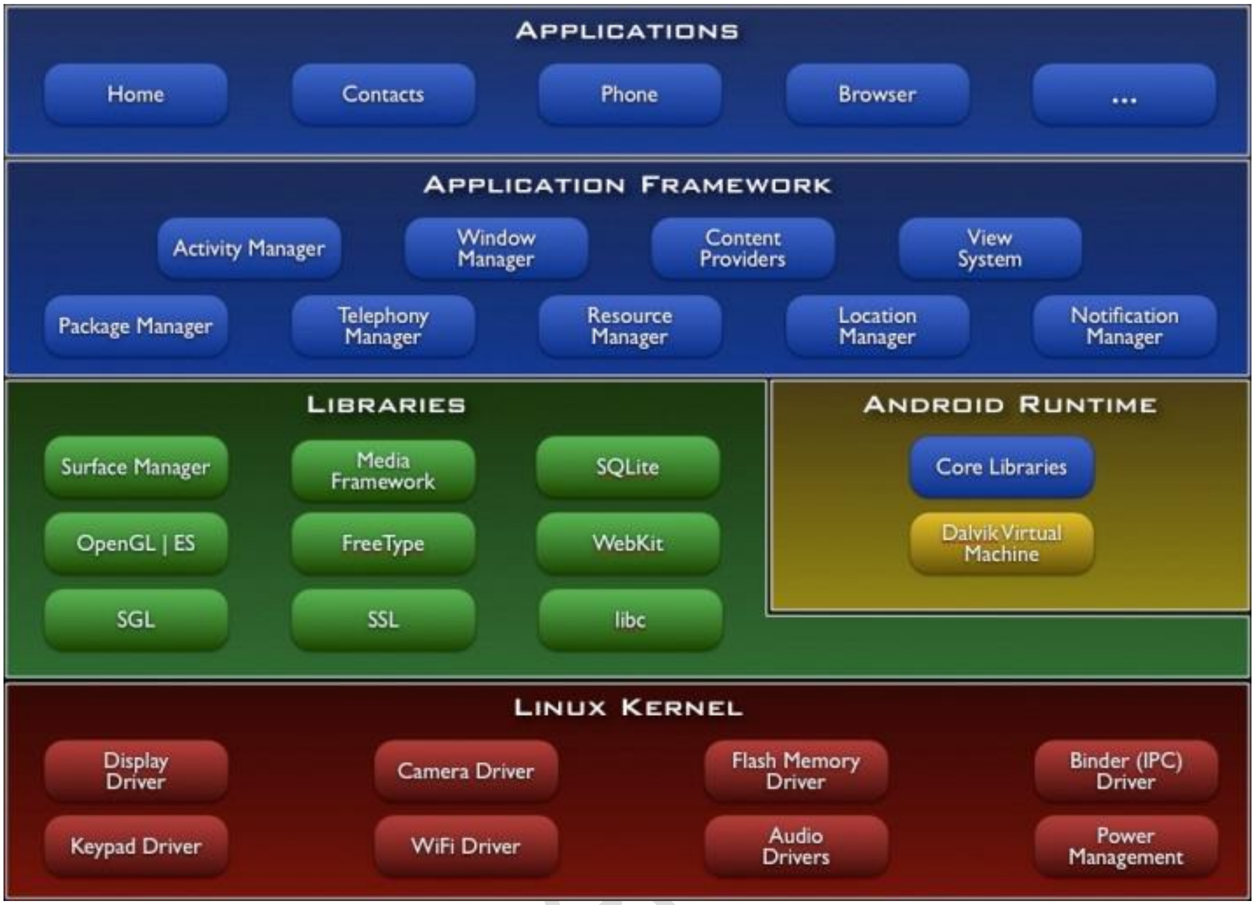
Android Operating System

Android architecture contains different number of components to support any android device needs. Android software contains an open-source Linux Kernel having collection of number of C/C++ libraries which are exposed through an application framework services.

Among all the components Linux Kernel provides main functionality of operating system functions to smartphones and Dalvik Virtual Machine (DVM) provide platform for running an android application.

The main components of android architecture are following:-

- Applications
- Application Framework
- Android Runtime
- Platform Libraries
- Linux Kernel



Applications

Applications is the top layer of android architecture. The pre-installed applications like home, contacts, camera, gallery etc and third party applications downloaded from the play store like chat applications, games etc. will be installed on this layer only.

Application framework

Application Framework provides several important classes which are used to create an Android application. It provides a generic abstraction for hardware access and also helps in managing the user interface with application resources.

Generally, it provides the services with the help of which we can create a particular class and make that class helpful for the Applications creation.

Application runtime

Android Runtime environment is one of the most important part of Android. It contains components like core libraries and the Dalvik virtual machine(DVM). Mainly, it provides the base for the application framework and powers our application with the help of the core libraries.

Like Java Virtual Machine (JVM), **Dalvik Virtual Machine (DVM)** is a register-based virtual machine and specially designed and optimized for android to ensure that a device can run multiple instances efficiently. It depends on the layer Linux kernel for threading and low-level memory management. The core libraries enable us to implement android applications using the standard JAVA or Kotlin programming languages.

Platform libraries

The Platform Libraries includes various C/C++ core libraries and Java based libraries such as Media, Graphics, Surface Manager, OpenGL etc. to provide a support for android development.

- **Media** library provides support to play and record an audio and video formats.
- **Surface manager** responsible for managing access to the display subsystem.
- **SGL** and **OpenGL** both cross-language, cross-platform application program interface (API) are used for 2D and 3D computer graphics.
- **SQLite** provides database support and **FreeType** provides font support.

- **Web-Kit** This open source web browser engine provides all the functionality to display web content and to simplify page loading.
- **SSL (Secure Sockets Layer)** is security technology to establish an encrypted link between a web server and a web browser.

Linux Kernel

Linux Kernel is heart of the android architecture. It manages all the available drivers such as display drivers, camera drivers, Bluetooth drivers, audio drivers, memory drivers, etc. which are required during the runtime.

Command Line Based OS

Dos Operating System

MS-DOS is one of the oldest and widely used operating system. DOS is a set of computer programs, the major functions of which are file management, allocation of system resources, providing essential features to control hardware devices.

DOS commands can be typed in either upper case or lower case.

Features of DOS

Following are the significant features of DOS −

- It is a single user system.
- It controls program.
- It is machine independence.
- It manages (computer) files.
- It manages input and output system.
- It manages (computer) memory.
- It provides command processing facilities.
- It operates with Assembler.

Types of DOS Commands

Following are the major types of DOS Command −

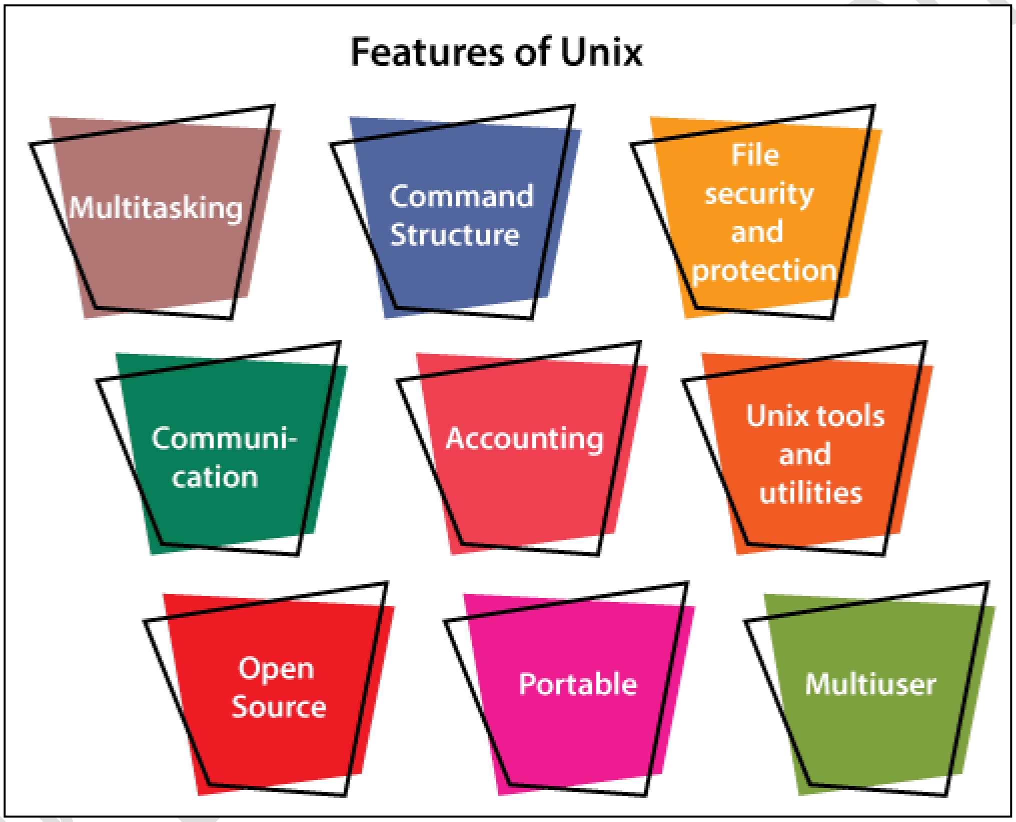
- **Internal Commands** − Commands such as DEL, COPY, TYPE, etc. are the internal commands that remain stored in computer memory.
- **External Commands** − Commands like FORMAT, DISKCOPY, etc. are the external commands and remain stored on the disk.

UNIX Operating System

UNIX is a powerful Operating System initially developed by Ken Thompson, Dennis Ritchie at AT&T Bell laboratories in 1970. It is prevalent among scientific, engineering, and academic institutions due to its most appreciative features like multitasking, flexibility, and many more. In UNIX, the file system is a hierarchical structure of files and directories where users can store and retrieve information using the files.

Features of UNIX Operating System:

Let's discuss the features of UNIX OS one by one in detail.



Multitasking: A UNIX operating system is a multitasking operating system that allows you to initiate more than one task from the same terminal so that one task is performed as a foreground and the other task as a background process.

Multi-user: UNIX operating system supports more than one user to access computer resources like main memory, hard disk, tape drives, etc. Multiple users can log on to the system from different terminals and run different jobs that share the resources of a command terminal. It deals with the principle of time-sharing. Time-sharing is done by a scheduler that divides the CPU time into several segments also called a time slice, and each segment is assigned to each user on a scheduled basis. This time slice is tiny. When this time is expired, it passes control to the following user on the system. Each user executes their set of instructions within their time slice.

Portability: This feature makes the UNIX work on different machines and platforms with the easy transfer of code to any computer system. Since a significant portion of UNIX is written in C language, and only a tiny portion is coded in assembly language for specific hardware.

File Security and Protection: Being a multi-user system, UNIX makes special consideration for file and system security. UNIX has different levels of security using assigning username and password to individual users ensuring the authentication, at the level providing file access permission viz. read, write and execute and lastly file encryption to change the file into an unreadable format.

Command Structure: UNIX commands are easy to understand and simple to use. Example: "cp", mv etc. While working in the UNIX environment, the UNIX commands are case-sensitive and are entered in lower case.

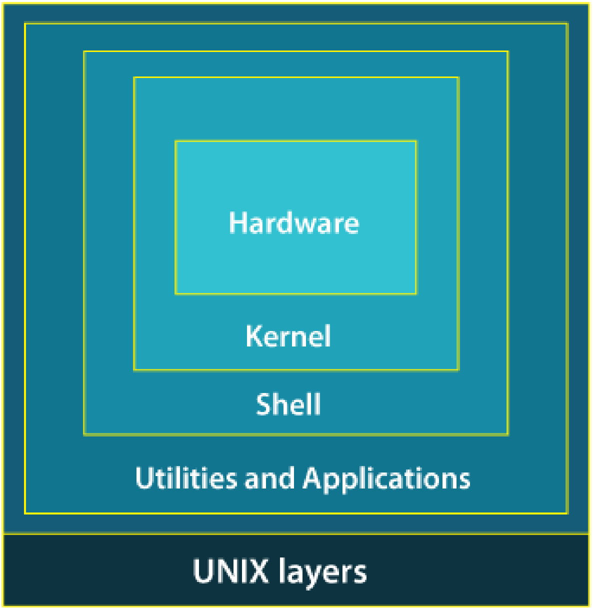
Communication: In UNIX, communication is an excellent feature that enables the user to communicate worldwide. It supports various communication facilities provided using the write command, mail command, talk command, etc.

Open Source: UNIX operating system is open source it means it is freely available to all and is a community-based development project.

Accounting: UNIX keeps an account of jobs created by the user. This feature enhances the system performance in terms of CPU monitoring and disk space checking. It allows you to keep an account of disk space used by each user, and the disk space can be limited by each other. You can assign every user a different disk quota. The root user can perform these accounting tasks using various commands such as quota, df, du, etc.

UNIX Tools and Utilities: UNIX system provides various types of tools and utilities facilities such as UNIX grep, sed and awk, etc. Some of the general-purpose tools are compilers, interpreters, network applications, etc. It also includes various server programs which provide remote and administration services.

The structure of Unix OS Layers are as follows:



Services of Operating System

An Operating System provides services to both the users and to the programs.

- It provides programs an environment to execute.
- It provides users the services to execute the programs in a convenient manner.

Following are a few common services provided by an operating system −

- Program execution
- I/O operations
- File System manipulation
- Communication
- Error Detection
- Resource Allocation
- Protection

Program execution

Operating systems handle many kinds of activities from user programs to system programs like printer spooler, name servers, file server, etc. Each of these activities is encapsulated as a process.

A process includes the complete execution context (code to execute, data to manipulate, registers, OS resources in use).

- Loads a program into memory.
- Executes the program.
- Handles program's execution.
- Provides a mechanism for process synchronization.
- Provides a mechanism for process communication.
- Provides a mechanism for deadlock handling.

I/O Operation

An I/O subsystem comprises of I/O devices and their corresponding driver software. Drivers hide the peculiarities of specific hardware devices from the users.

An Operating System manages the communication between user and device drivers.

- I/O operation means read or write operation with any file or any specific I/O device.
- Operating system provides the access to the required I/O device when required.

File system manipulation

A file represents a collection of related information. Computers can store files on the disk (secondary storage), for long-term storage purpose. Examples of storage media include magnetic tape, magnetic disk and optical disk drives like CD, DVD. Each of these media has its own properties like speed, capacity, data transfer rate and data access methods.

A file system is normally organized into directories for easy navigation and usage. These directories may contain files and other directions. Program needs to read a file or write a file.

- The operating system gives the permission to the program for operation on file.
- Permission varies from read-only, read-write, denied and so on.
- Operating System provides an interface to the user to create/delete files.
- Operating System provides an interface to the user to create/delete directories.
- Operating System provides an interface to create the backup of file system.

Communication

In case of distributed systems which are a collection of processors that do not share memory, peripheral devices, or a clock, the operating system manages communications between all the processes. Multiple processes communicate with one another through communication lines in the network.

The OS handles routing and connection strategies, and the problems of contention and security. Following are the major activities of an operating system with respect to communication −

- Two processes often require data to be transferred between them
- Both the processes can be on one computer or on different computers, but are connected through a computer network.
- Communication may be implemented by two methods, either by Shared Memory or by Message Passing.

Error handling

Errors can occur anytime and anywhere. An error may occur in CPU, in I/O devices or in the memory hardware. Following are the major activities of an operating system with respect to error handling −

- The OS constantly checks for possible errors.
- The OS takes an appropriate action to ensure correct and consistent computing.

Resource Management

In case of multi-user or multi-tasking environment, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management −

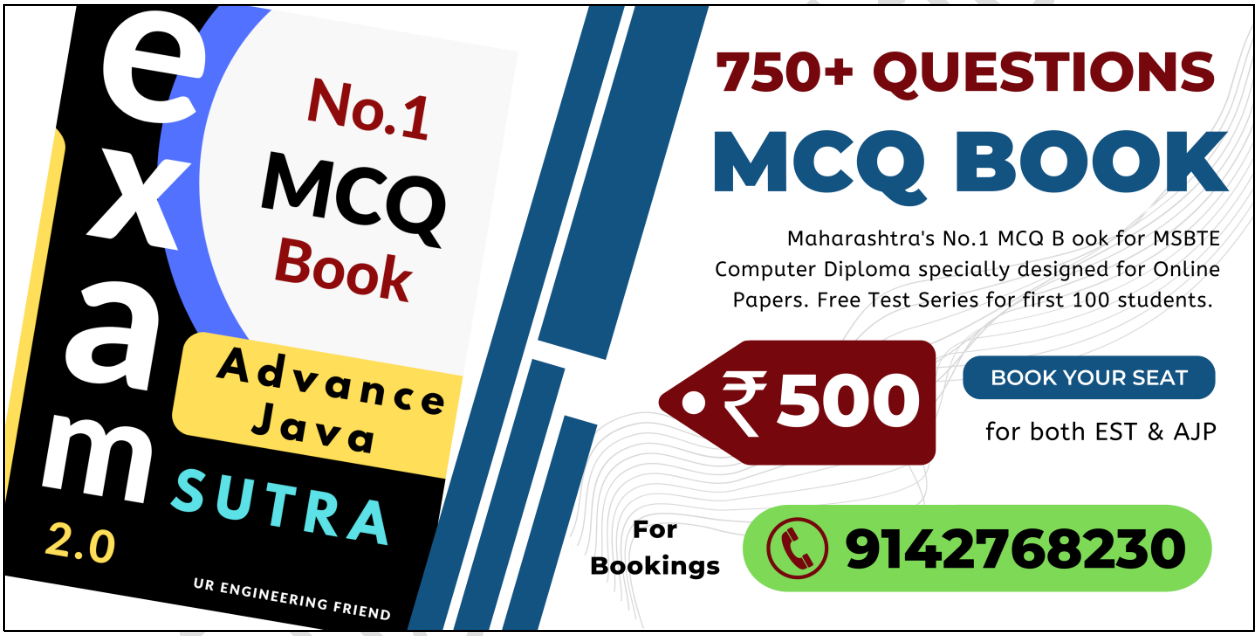
- The OS manages all kinds of resources using schedulers.
- CPU scheduling algorithms are used for better utilization of CPU.

Protection

Considering a computer system having multiple users and concurrent execution of multiple processes, the various processes must be protected from each other's activities.

Protection refers to a mechanism or a way to control the access of programs, processes, or users to the resources defined by a computer system. Following are the major activities of an operating system with respect to protection −

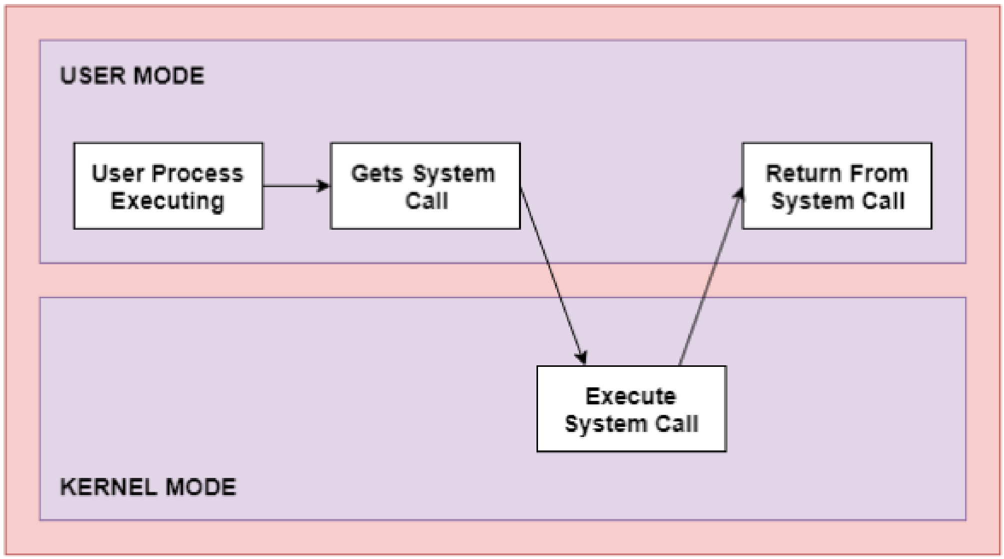
- The OS ensures that all access to system resources is controlled.
- The OS ensures that external I/O devices are protected from invalid access attempts.
- The OS provides authentication features for each user by means of passwords.



System Calls in Operating System

The interface between a process and an operating system is provided by system calls. In general, system calls are available as assembly language instructions. They are also included in the manuals used by the assembly level programmers. System calls are usually made when a process in user mode requires access to a resource. Then it requests the kernel to provide the resource via a system call.

A figure representing the execution of the system call is given as follows −



As can be seen from this diagram, the processes execute normally in the user mode until a system call interrupts this. Then the system call is executed on a priority basis in the kernel mode. After the execution of the system call, the control returns to the user mode and execution of user processes can be resumed.

In general, system calls are required in the following situations −

- If a file system requires the creation or deletion of files. Reading and writing from files also require a system call.
- Creation and management of new processes.
- Network connections also require system calls. This includes sending and receiving packets.
- Access to a hardware devices such as a printer, scanner etc. requires a system call.

Types of System Calls

There are mainly five types of system calls. These are explained in detail as follows −

Process Control

These system calls deal with processes such as process creation, process termination etc.

File Management

These system calls are responsible for file manipulation such as creating a file, reading a file, writing into a file etc.

Device Management

These system calls are responsible for device manipulation such as reading from device buffers, writing into device buffers etc.

Information Maintenance

These system calls handle information and its transfer between the operating system and the user program.

Communication

These system calls are useful for interprocess communication. They also deal with creating and deleting a communication connection.

Process	Windows	Unix

Process Control	CreateProcess()
	ExitProcess()
	WaitForSingleObject()
		Fork()
		Exit()
		Wait()

File Manipulation	CreateFile()
	ReadFile()
	WriteFile()
		Open()
		Read()
		Write()

	CloseHandle()	Close()

Device Management	SetConsoleMode()
	ReadConsole()
	WriteConsole()
		Ioctl()
		Read()
		Write()

Information Maintenance	GetCurrentProcessID()
	SetTimer()
	Sleep()
		Getpid()
		Alarm()
		Sleep()

Communication	CreatePipe()
	CreateFileMapping()
	MapViewOfFile()
		Pipe()
		Shmget()
		Mmap()

Protection	SetFileSecurity()
	InitializeSecurityDescriptor()
	SetSecurityDescriptorgroup()
		Chmod()
		Umask()
		Chown()

Operating System Components

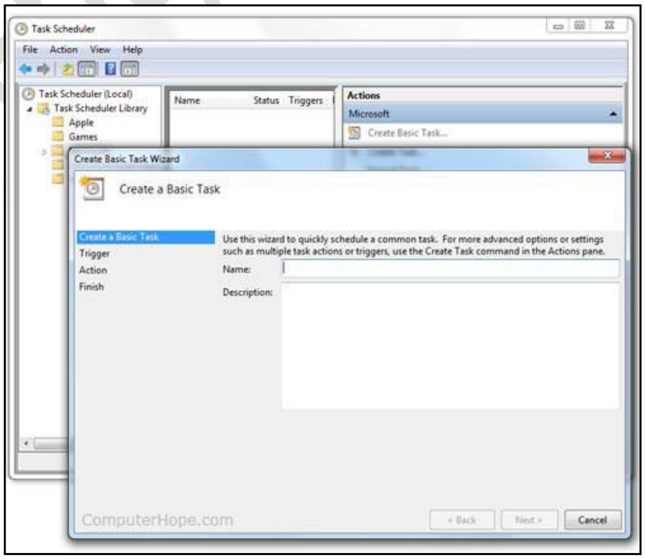
OS Tools

- Task Scheduler
- Performance Monitor

Task Scheduler

The **Task Scheduler** is a tool included with Windows that allows predefined actions to be automatically executed whenever a certain set of conditions is met. For example, you can schedule a task to run a backup script every night, or send you an e-mail whenever a certain system event occurs.

The picture below is an example of what the Task Scheduler looks like in Microsoft Windows 7.



Use this wizard to quickly schedule a common task. For more advanced options or settings such as multiple task actions or triggers, use the Create Task command in the Actions pane.

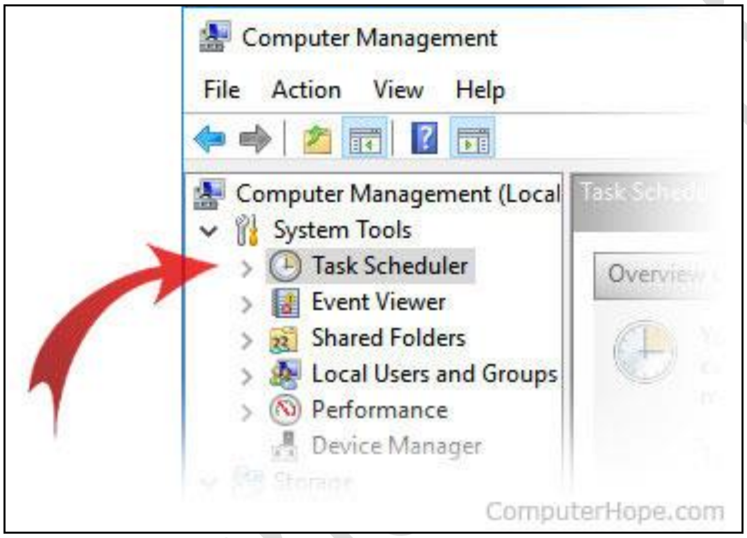
Name:

Description:

How to open Task Scheduler

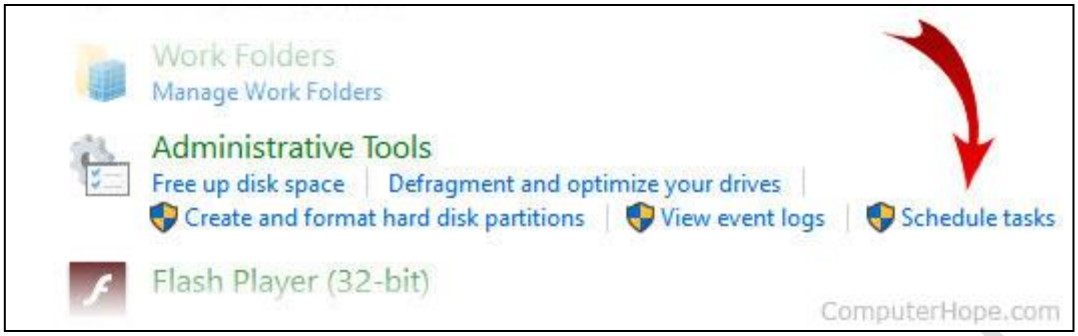
Windows 10 and Windows 8

1. Press windows key +X to open the Power User Tasks Menu.
2. Select **Computer Management** (or press G).
3. In the Computer Management window, select **Task Scheduler** in the pane on the left.



– OR –

1. Open the Control Panel.
2. Select **System and Security**.
3. Under **Administrative Tools**, select **Schedule Tasks**.



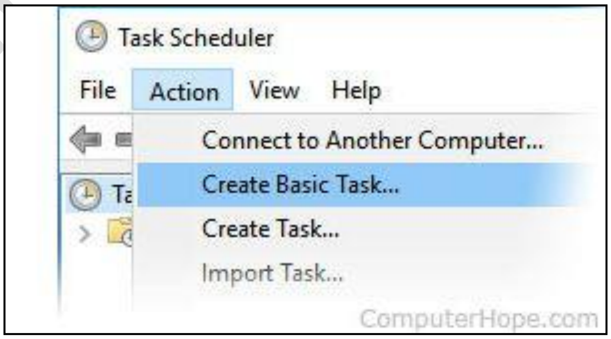
Windows 7

1. Access the Start menu.
2. In the Search / Run box, type **task scheduler** and in the returned results, select **Task Scheduler**.

– OR –

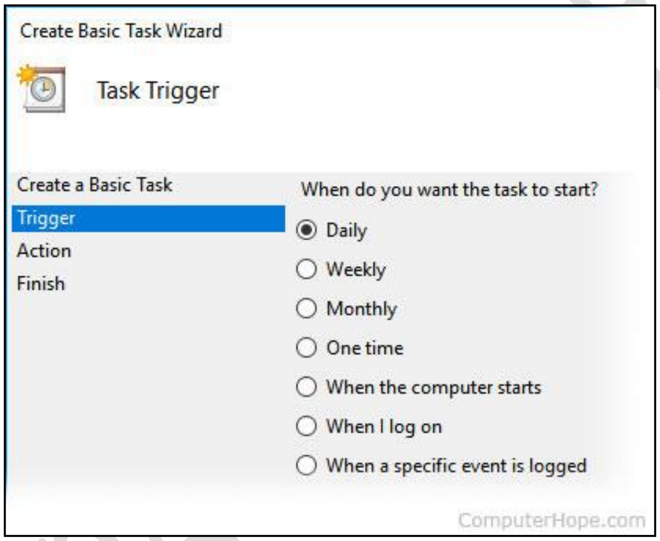
1. Access the Start menu
2. Select **Programs** or **All Programs**.
3. Select **Accessories**, and then **System Tools**.
4. In the System Tools folder, click **Task Scheduler**.

How to create a new task in Task Scheduler



For most purposes, creating a **Basic Task** will serve your needs. To create one, follow these steps.

1. In the *Action* menu (pictured), select **Create basic task** to open the *Create Basic Task* wizard.
2. In the *Name:* field, enter a name for your task. You may optionally add a description of the task in the *Description:* text box below. When this is done, click **Next**.
3. Select a trigger from the options presented (pictured below) to define when you want your task to run. When you're ready to continue, click **Next**. Depending on your trigger selection, you may be prompted to choose a specific time, day, or event. If so, make your choice and then click **Next**.



Create Basic Task Wizard

Task Trigger

When do you want the task to start?

● Daily
○ Weekly
○ Monthly
○ One time
○ When the computer starts
○ When I log on
○ When a specific event is logged

4. Select the *Action* you want to perform when the trigger occurs (**Start a program**, **Send an e-mail**, or **Display a message**), then click **Next**.
5. Depending on the *Action* you chose in the previous step, fill out the relevant information, then click **Next**.
6. Finally, the *Finish* screen displays your task as you configured it. If you need to make changes, click the **Back** button to return to a previous step, make your changes, then click **Next** until you return to the *Finish* screen.

7. To finish configuring your task, click **Finish**. The next time your trigger occurs, the task runs.

Performance Monitor

A system performance monitor (SPM) is a type of application that identifies, collects, monitors and reports on the overall operational health of a computer system.

It is a performance monitoring tool that enables end users, administrators and organizations to gauge and evaluate the performance of a given system.

A system performance monitor primarily collects and reports key performance indicators and metrics into the operational state of a system.

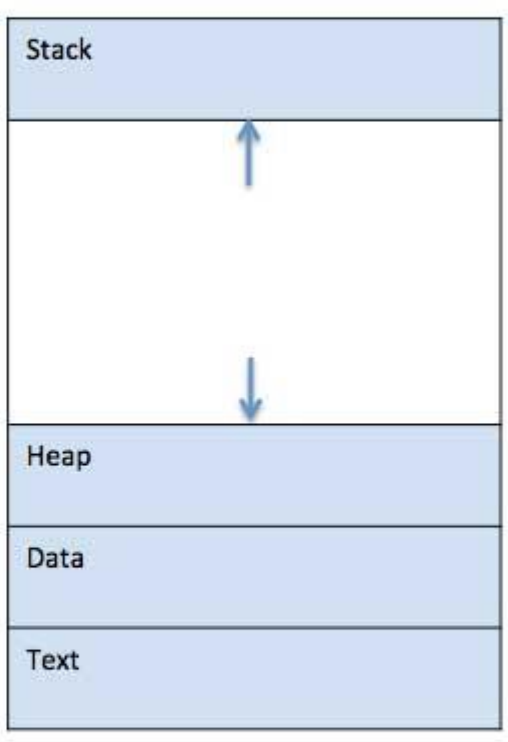
Most operating systems have a native SPM application/component that displays factual and graphical stats for system performance.

For example, a system might have an over-utilized CPU/memory and based on past trends provided by the SPM, a system administrator can schedule a system upgrade accordingly.

Basic Concepts of Process

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

- A process is defined as an entity which represents the basic unit of work to be implemented in the system.
- To put it in simple terms, we write our computer programs in a text file and when we execute this program, it becomes a process which performs all the tasks mentioned in the program.
- When a program is loaded into the memory and it becomes a process, it can be divided into four sections ─ stack, heap, text and data.



Stack	The process Stack contains the temporary data such as method/function parameters, return address and local variables.
Heap	This is dynamically allocated memory to a process during its run time.
Text	This includes the current activity represented by the value of Program Counter and the contents of the processor's registers.
Data	This section contains the global and static variables.

Program

A program is a piece of code which may be a single line or millions of lines. A computer program is usually written by a computer programmer in a programming language. For example, here is a simple program written in C programming language −

```
#include <stdio.h>


int main() {
   printf("Hello, World! \n");
   return 0;
}
```

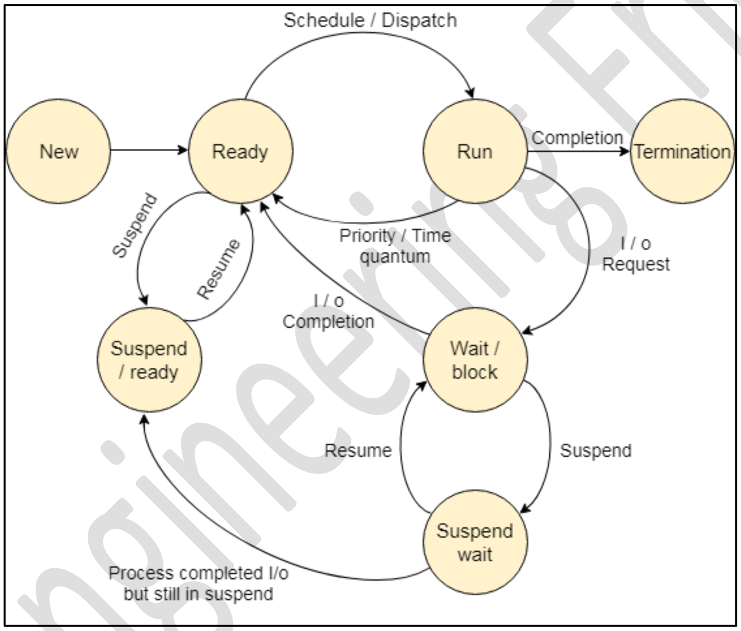
A computer program is a collection of instructions that performs a specific task when executed by a computer. When we compare a program with a process, we can conclude that a process is a dynamic instance of a computer program.

A part of a computer program that performs a well-defined task is known as an **algorithm**. A collection of computer programs, libraries and related data are referred to as a **software**.

Process States

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.

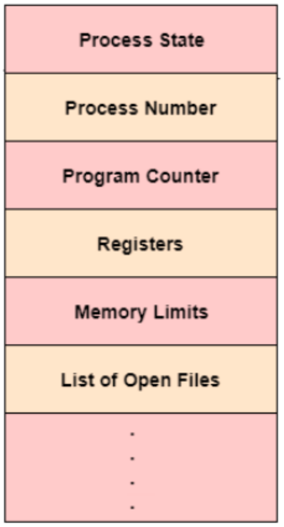
Process Control Block

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)

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.

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

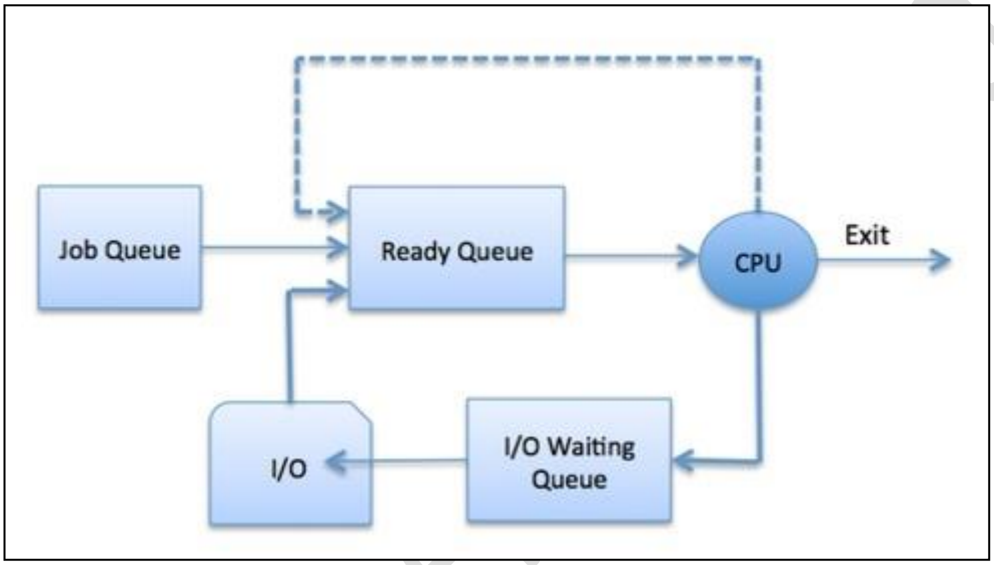
Process Scheduling Queues

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.

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.

Context Switching

Context Switching involves storing the context or state of a process so that it can be reloaded when required and execution can be resumed from the same point as earlier. This is a feature of a multitasking operating system and allows a single CPU to be shared by multiple processes.

Context Switching Steps

The steps involved in context switching are as follows −

- Save the context of the process that is currently running on the CPU. Update the process control block and other important fields.
- Move the process control block of the above process into the relevant queue such as the ready queue, I/O queue etc.
- Select a new process for execution.
- Update the process control block of the selected process. This includes updating the process state to running.
- Update the memory management data structures as required.
- Restore the context of the process that was previously running when it is loaded again on the processor. This is done by loading the previous values of the process control block and registers.

Threads in OS

A thread is a single sequential flow of execution of tasks of a process so it is also known as thread of execution or thread of control.

- There is a way of thread execution inside the process of any operating system. Apart from this, there can be more than one thread inside a process.
- Each thread of the same process makes use of a separate program counter and a stack of activation records and control blocks. Thread is often referred to as a lightweight process.

Need of Thread

- It takes far less time to create a new thread in an existing process than to create a new process.
- Threads can share the common data, they do not need to use Inter- Process communication.
- Context switching is faster when working with threads.
- It takes less time to terminate a thread than a process.

Types of Threads

- Kernel level thread.
- User-level thread.

User - Level Threads

The user-level threads are implemented by users and the kernel is not aware of the existence of these threads. It handles them as if they were single-threaded processes. User-level threads are small and much faster than kernel level threads. They are represented by a program counter(PC), stack, registers and a small process control block. Also, there is no kernel involvement in synchronization for user-level threads.

Advantages of User-Level Threads

- User-level threads are easier and faster to create than kernel-level threads. They can also be more easily managed.
- User-level threads can be run on any operating system.
- There are no kernel mode privileges required for thread switching in user-level threads.

Disadvantages of User-Level Threads

- Multithreaded applications in user-level threads cannot use multiprocessing to their advantage.
- The entire process is blocked if one user-level thread performs blocking operation.

Kernel-Level Threads

Kernel-level threads are handled by the operating system directly and the thread management is done by the kernel. The context information for the process as well as the process threads is all managed by the kernel. Because of this, kernel-level threads are slower than user-level threads.

Advantages of Kernel-Level Threads

- Multiple threads of the same process can be scheduled on different processors in kernel-level threads.
- The kernel routines can also be multithreaded.
- If a kernel-level thread is blocked, another thread of the same process can be scheduled by the kernel.

Disadvantages of Kernel-Level Threads

Some of the disadvantages of kernel-level threads are as follows −

- A mode switch to kernel mode is required to transfer control from one thread to another in a process.
- Kernel-level threads are slower to create as well as manage as compared to user-level threads.

Preemptive & Non-Preemptive *scheduling*

Preemptive Scheduling

Preemptive scheduling is used when a process switches from running state to ready state or from the waiting state to ready state. The resources (mainly CPU cycles) are allocated to the process for a limited amount of time and then taken away, and the process is again placed back in the ready queue if that process still has CPU burst time remaining. That process stays in the ready queue till it gets its next chance to execute.

Non-Preemptive Scheduling

Non-preemptive Scheduling is used when a process terminates, or a process switches from running to the waiting state. In this scheduling, once the resources (CPU cycles) are allocated to a process, the process holds the CPU till it gets terminated or reaches a waiting state. In the case of non-preemptive scheduling does not interrupt a process running CPU in the middle of the execution. Instead, it waits till the process completes its CPU burst time, and then it can allocate the CPU to another process.

Scheduling Algorithms

There are various algorithms which are used by the Operating System to schedule the processes on the processor in an efficient way.

The Purpose of a Scheduling algorithm

1. Maximum CPU utilization
2. Fare allocation of CPU
3. Maximum throughput
4. Minimum turnaround time
5. Minimum waiting time
6. Minimum response time

FCFS Scheduling

First come first serve (FCFS) scheduling algorithm simply schedules the jobs according to their arrival time. The job which comes first in the ready queue will get the CPU first. The lesser the arrival time of the job, the sooner will the job get the CPU. FCFS scheduling may cause the problem of starvation if the burst time of the first process is the longest among all the jobs.

Advantages of FCFS

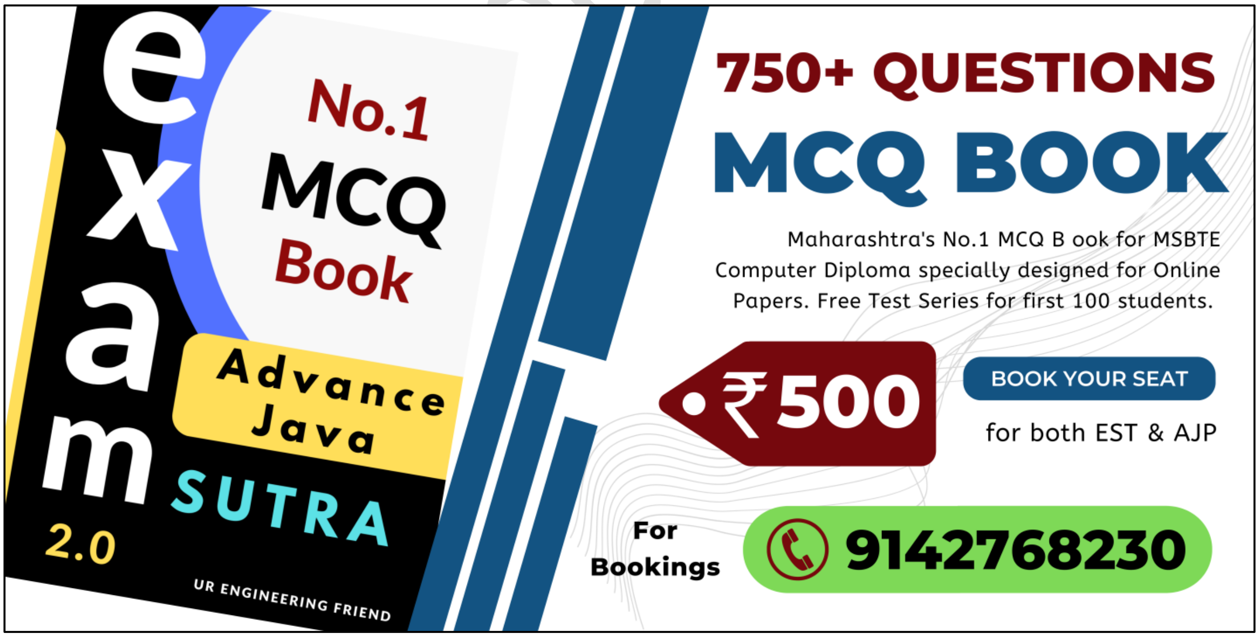
- Simple

- Easy
- First come, First serve

Disadvantages of FCFS

1. The scheduling method is non preemptive, the process will run to the completion.
2. Due to the non-preemptive nature of the algorithm, the problem of starvation may occur.
3. Although it is easy to implement, but it is poor in performance since the average waiting time is higher as compare to other scheduling algorithms.

Time for Problem -:



Shortest Job First (SJF) Scheduling

Till now, we were scheduling the processes according to their arrival time (in FCFS scheduling). However, SJF scheduling algorithm, schedules the processes according to their burst time.

In SJF scheduling, the process with the lowest burst time, among the list of available processes in the ready queue, is going to be scheduled next.

However, it is very difficult to predict the burst time needed for a process hence this algorithm is very difficult to implement in the system.

Advantages of SJF

1. Maximum throughput
2. Minimum average waiting and turnaround time

Disadvantages of SJF

1. May suffer with the problem of starvation
2. It is not implementable because the exact Burst time for a process can't be known in advance.

Time for Problem -:

Priority Scheduling

Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm, the scheduler selects the tasks to work as per the priority.

Characteristics of Priority Scheduling

- A CPU algorithm that schedules processes based on priority.
- It used in Operating systems for performing batch processes.
- If two jobs having the same priority are READY, it works on a FIRST COME, FIRST SERVED basis.
- In priority scheduling, a number is assigned to each process that indicates its priority level.
- Lower the number, higher is the priority.
- In this type of scheduling algorithm, if a newer process arrives, that is having a higher priority than the currently running process, then the currently running process is preempted.

Time for Problem -:

Round robin Scheduling

Round Robin scheduling algorithm is one of the most popular scheduling algorithm which can actually be implemented in most of the operating systems.

- This is the **preemptive version** of first come first serve scheduling. The Algorithm focuses on Time Sharing. In this algorithm, every process gets executed in a **cyclic way**.
- A certain time slice is defined in the system which is called time **quantum**.
- Each process present in the ready queue is assigned the CPU for that time quantum, if the execution of the process is completed during that time then the process will **terminate** else the process will go back to the **ready queue** and waits for the next turn to complete the execution.

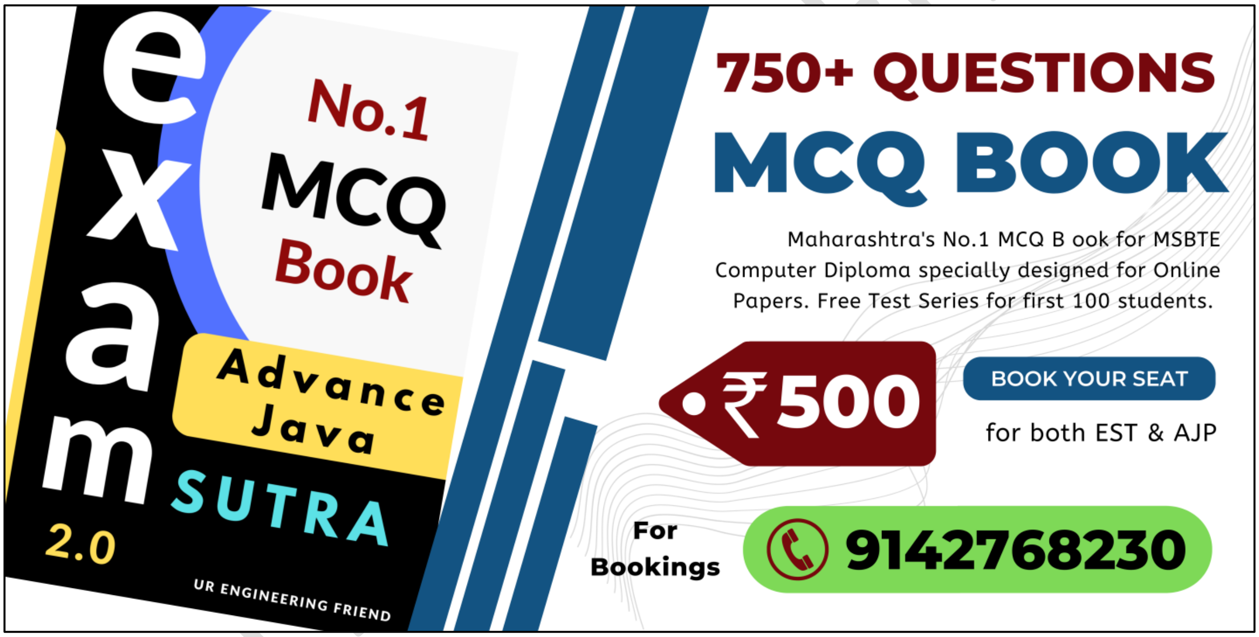
Advantages

1. It can be actually implementable in the system because it is not depending on the burst time.
2. It doesn't suffer from the problem of starvation or convoy effect.
3. All the jobs get a fare allocation of CPU.

Disadvantages

1. The higher the time quantum, the higher the response time in the system.
2. The lower the time quantum, the higher the context switching overhead in the system.
3. Deciding a perfect time quantum is really a very difficult task in the system.

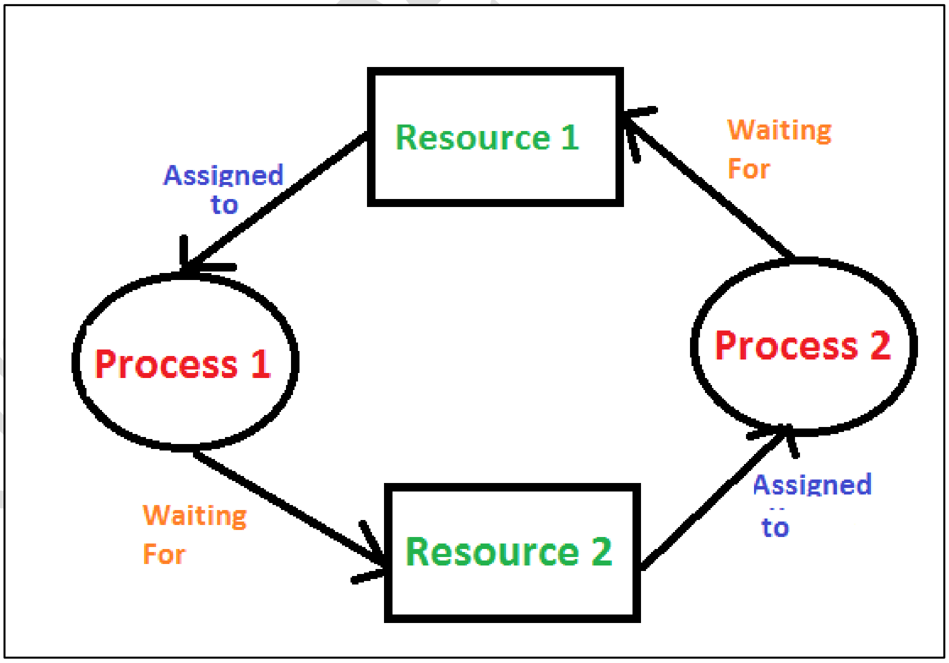
Time for Problem -:



Deadlock

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

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



Necessary conditions for Deadlocks

1. **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.

2. **Hold and Wait**

   A process waits for some resources while holding another resource at the same time.

3. **No preemption**

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

4. **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.

Advantages of Deadlock

- This situation works well for processes which perform a single burst of activity
- No preemption needed for Deadlock.

- Convenient method when applied to resources whose state can be saved and restored easily
- Feasible to enforce via compile-time checks
- Needs no run-time computation since the problem is solved in system design

Disadvantages of Deadlock

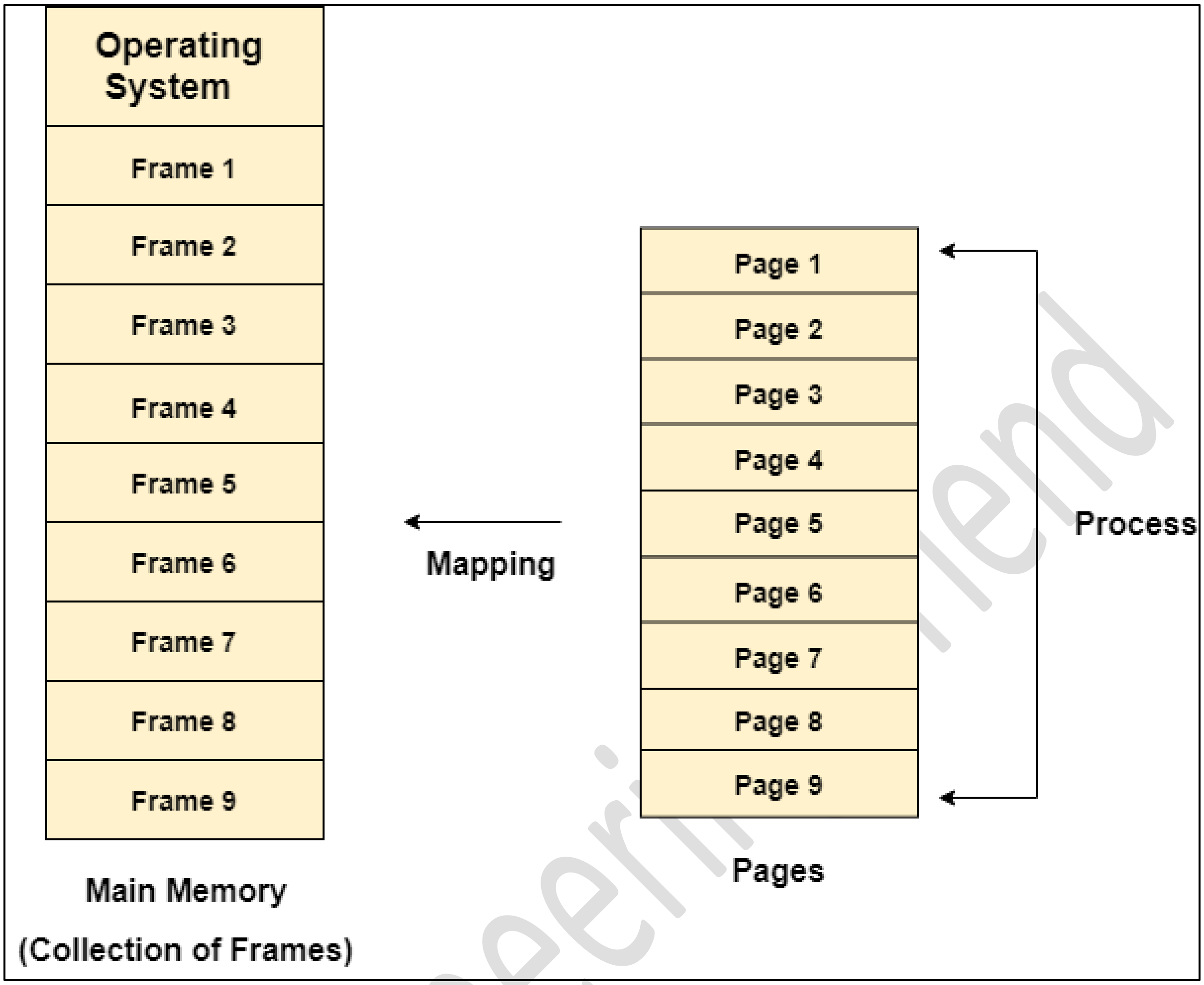
- Delays process initiation
- Processes must know future resource need
- Pre-empts more often than necessary
- Dis-allows incremental resource requests
- Inherent preemption losses.

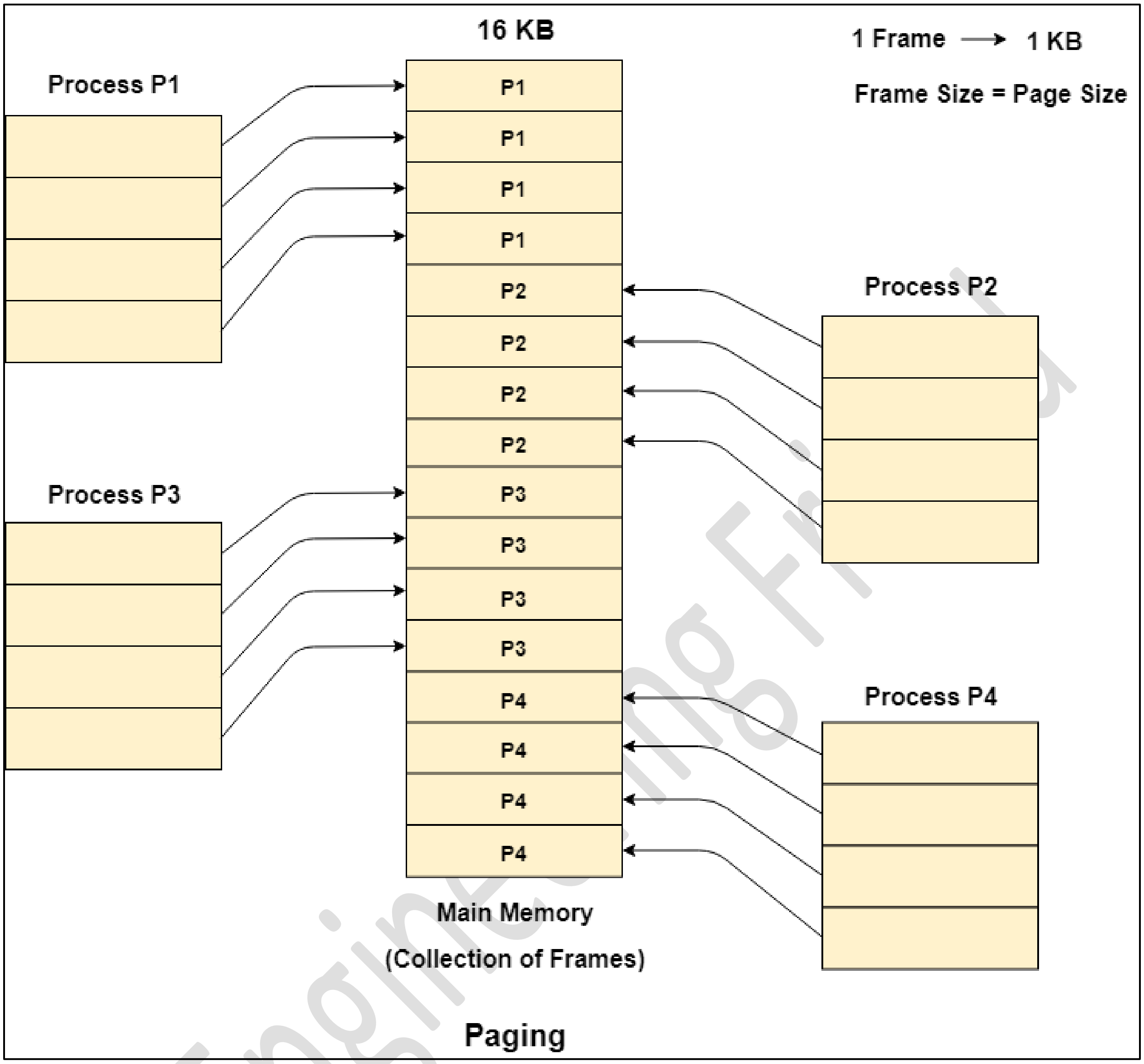
Page Replacement Algorithm

Paging -:

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory. The process of retrieving processes in the form of pages from the secondary storage into the main memory is known as paging. The basic purpose of paging is to separate each procedure into pages. Additionally, frames will be used to split the main memory.This scheme permits the physical address space of a process to be non – contiguous.

- The main idea behind the paging is to divide each process in the form of pages. The main memory will also be divided in the form of frames.
- Pages of the process are brought into the main memory only when they are required otherwise they reside in the secondary storage.
- Different operating system defines different frame sizes. The sizes of each frame must be equal. Considering the fact that the pages are mapped to the frames in Paging, page size needs to be as same as frame size.


Operating System
Frame 1
Frame 2
Frame 3
Frame 4
Frame 5
Frame 6
Frame 7
Frame 8
Frame 9
Main Memory
(Collection of Frames)
Mapping
Page 1
Page 2
Page 3
Page 4
Page 5
Page 6
Page 7
Page 8
Page 9
Pages
Process



What is Page Fault in Operating System?

Page faults dominate more like an **error**. A page fault will happen if a program tries to access a piece of memory that does not exist in physical memory (main memory). The fault specifies the operating system to trace all data into virtual memory management and then relocate it from secondary memory to its primary memory, such as a hard disk.

Page Hit

When the CPU attempts to obtain a needed page from main memory and the page exists in **main memory (RAM)**, it is referred to as a **"PAGE HIT"**.

FIFO (First In First Out)

This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.

Time for Problem -:

LRU (Least recently used)

This algorithm replaces the page which has not been referred for a long time. This algorithm is just opposite to the optimal page replacement algorithm. In this, we look at the past instead of staring at future.

The primary purpose of any page replacement algorithm is to reduce the number of *page faults*. When a page replacement is required, the **LRU page replacement algorithm** replaces the least recently used page with a new page. This algorithm is based on the assumption that among all pages, the least recently used page will not be used for a long time. It is a popular and efficient page replacement technique.

Time for Problem -:

Optimal Page Replacement

This algorithms replaces the page which will not be referred for so long in future. Although it can not be practically implementable but it can be used as a benchmark. Other algorithms are compared to this in terms of optimality.

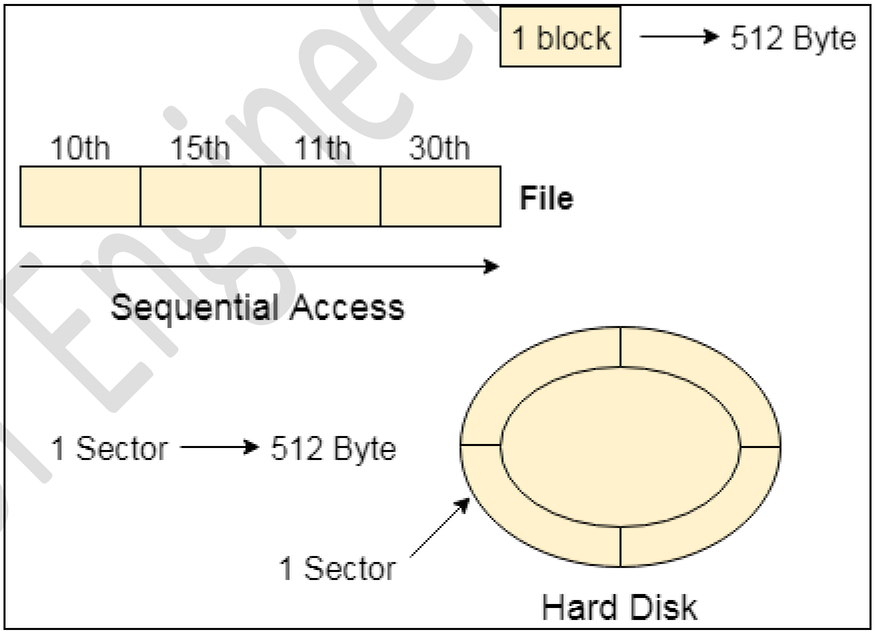
Optimal page replacement is the best page replacement algorithm as this algorithm results in the least number of page faults. In this algorithm, the pages are replaced with the ones that will not be used for the longest duration of time in the future. In simple terms, the pages that will be referred farthest in the future are replaced in this algorithm.

Time for Problem -:

Basic concept of File

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

Sequential Access



Most of the operating systems access the file sequentially. In other words, we can say that most of the files need to be accessed sequentially by the operating system.

In sequential access, the OS read the file word by word. A pointer is maintained which initially points to the base address of the file. If the user wants to read first word of the file then the pointer provides that word to the user and increases its value by 1 word. This process continues till the end of the file.

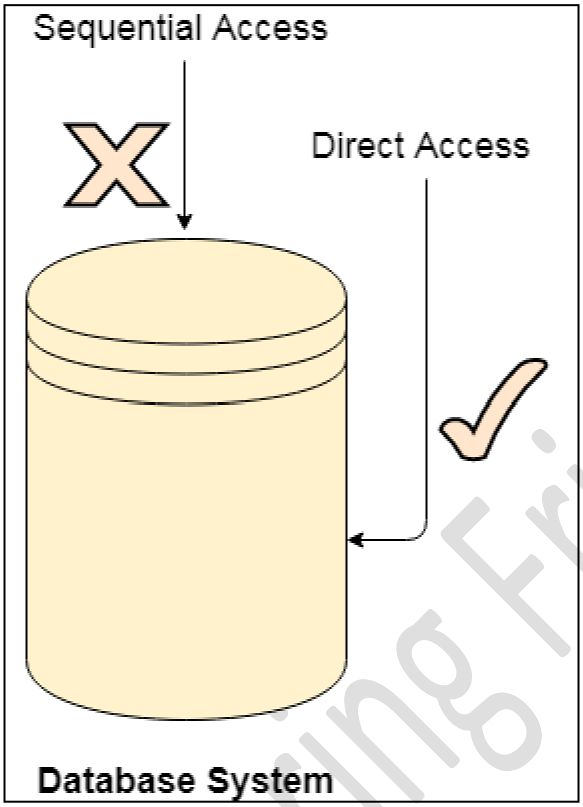
Modern word systems do provide the concept of direct access and indexed access but the most used method is sequential access due to the fact that most of the files such as text files, audio files, video files, etc need to be sequentially accessed.

Direct Access

The Direct Access is mostly required in the case of database systems. In most of the cases, we need filtered information from the database. The sequential access can be very slow and inefficient in such cases.

Suppose every block of the storage stores 4 records and we know that the record we needed is stored in 10th block. In that case, the sequential access will not be implemented because it will traverse all the blocks in order to access the needed record.

Direct access will give the required result despite of the fact that the operating system has to perform some complex tasks such as determining the desired block number. However, that is generally implemented in database applications.



Allocation Methods

The allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.

- Contiguous Allocation
- Linked Allocation
- Indexed Allocation

The main idea behind these methods is to provide:

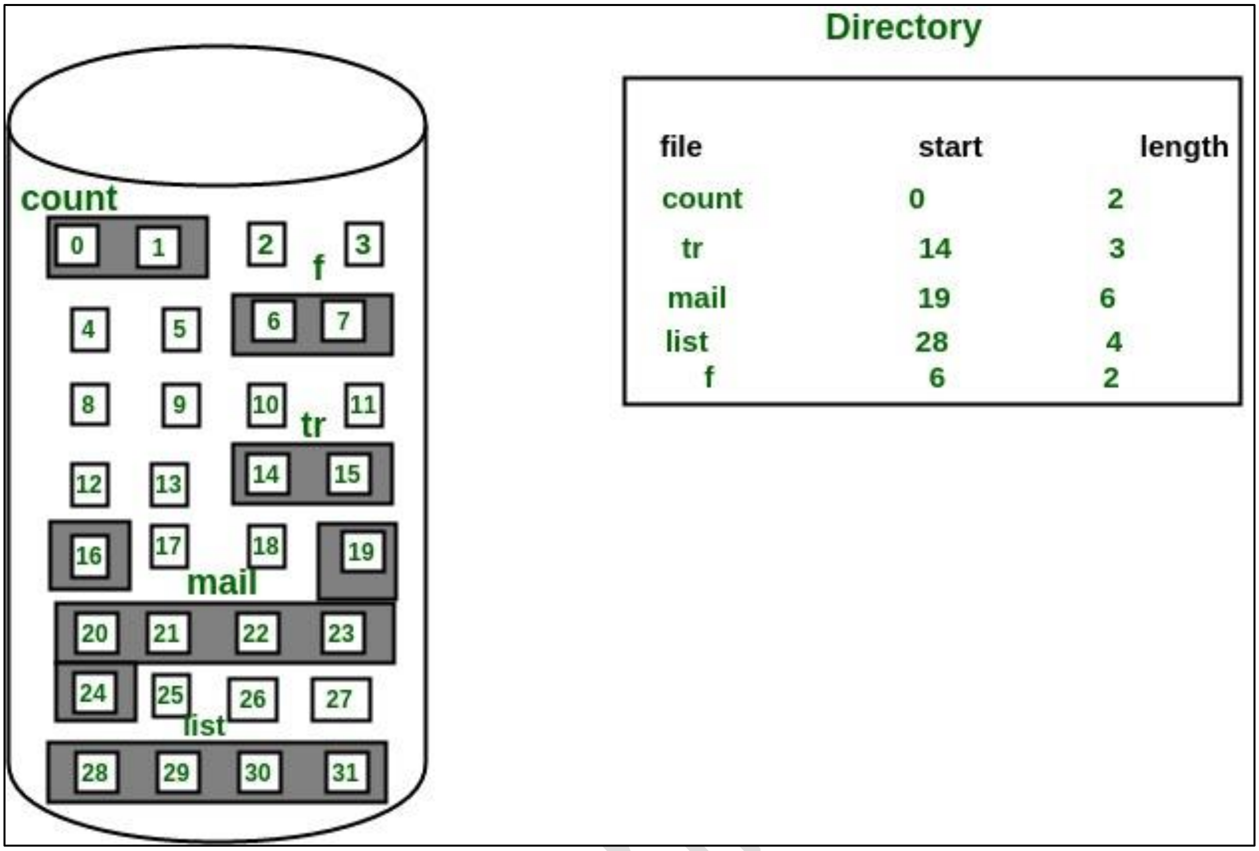
- Efficient disk space utilization.
- Fast access to the file blocks.

Contiguous Allocation

In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: *b, b+1, b+2,……b+n-1*. This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file. The directory entry for a file with contiguous allocation contains

- Address of starting block
- Length of the allocated portion.

The *file 'mail'* in the following figure starts from the block 19 with length = 6 blocks. Therefore, it occupies *19, 20, 21, 22, 23, 24* blocks.



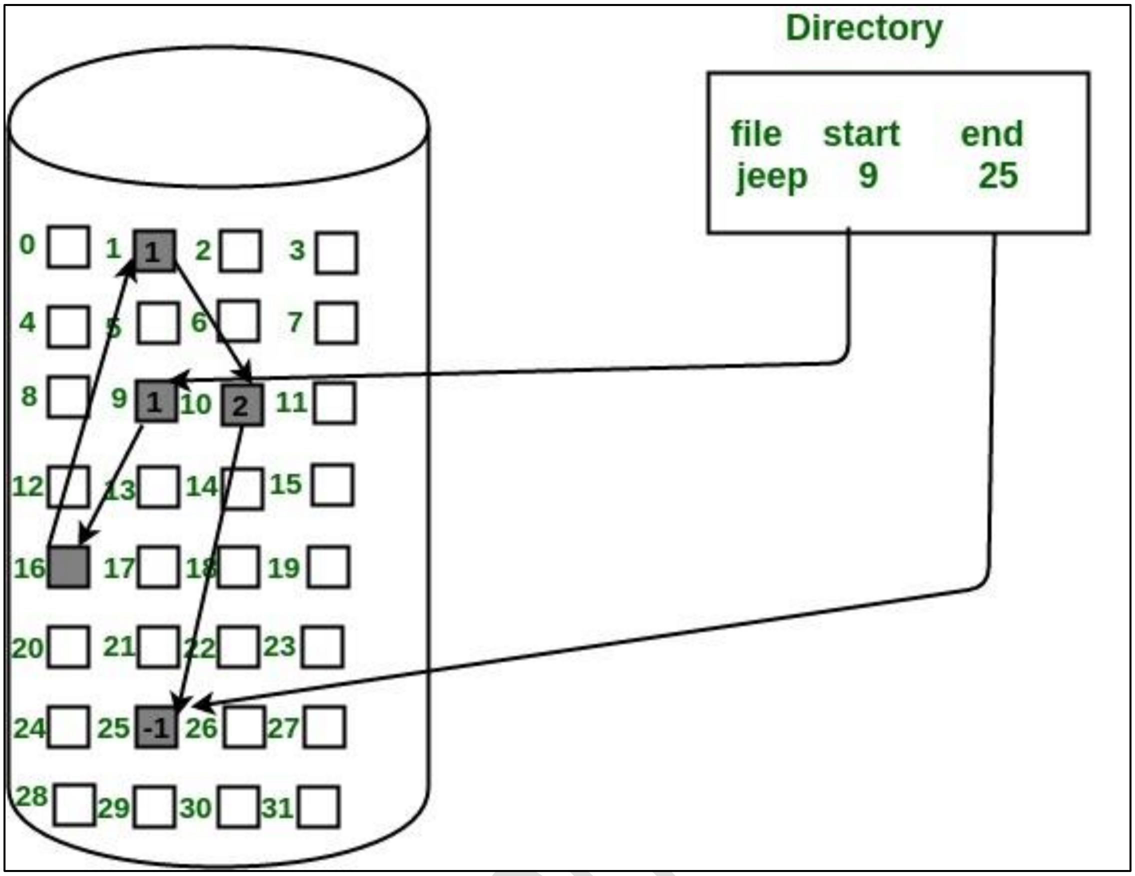
Directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Linked List Allocation

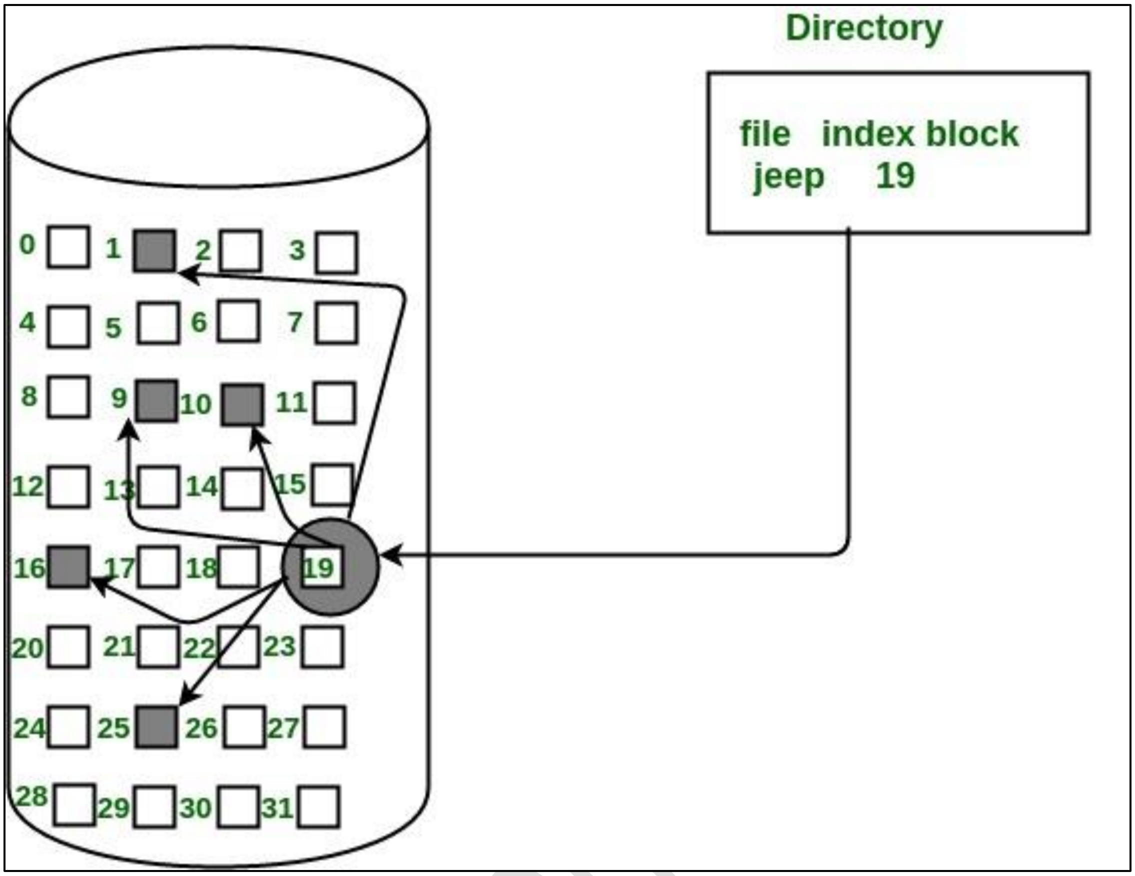
In this scheme, each file is a linked list of disk blocks which **need not be** contiguous. The disk blocks can be scattered anywhere on the disk. The directory entry contains a pointer to the starting and the ending file block. Each block contains a pointer to the next block occupied by the file.

The file 'jeep' in following image shows how the blocks are randomly distributed. The last block (25) contains -1 indicating a null pointer and does not point to any other block.

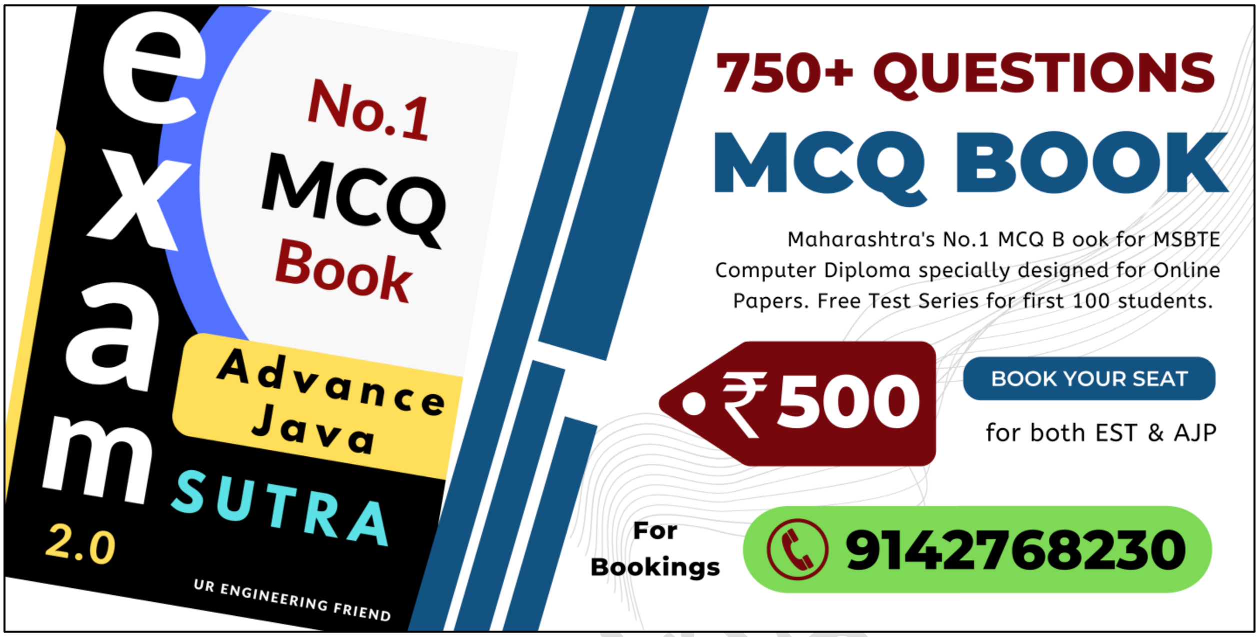


Indexed Allocation

In this scheme, a special block known as the **Index block** contains the pointers to all the blocks occupied by a file. Each file has its own index block. The ith entry in the index block contains the disk address of the ith file block. The directory entry contains the address of the index block as shown in the image:


Directory
file index block
jeep 19
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31

End .



Maharashtra's No.1 MCQ B ook for MSBTE Computer Diploma specially designed for Online Papers. Free Test Series for first 100 students.

How to enroll ?

Send us your details to our official whatsapp number -: +91 9142768230

What you will get ?

Fees – 500 for MCQ Set

In this set you will get Advance Java and Environmental Studies MCQ Book with more than 1500+ Questions