IARE '24 DISCORD TEAM

II-II

OS MODULE 1 SOLUTIONS

SUHRUTH • ANUSHKA • VISHAL • IKRAM • UJJWAL

INTRODUCTION TO OS

OPERATING SYSTEMS

MODULE 1

PART-A

1. **Distinguish between batch systems and time sharing systems.**

   Batch Systems:

   - There is no direct interaction between the user and the computer.
   - Users have to submit a job to a computer operator.
   - Then the computer operator places a batch of several jobs on an I/P device.
   - Jobs are batched together by type of language and requirements.
   - Then a special program, the monitor, manages the execution of each program in the batch.
   - The monitor is always a main memory and available for execution.

   Time Sharing Systems:

   - These are very similar to multiprogramming systems and an extension of multiprogramming systems.
   - The prime focus is on minimising the response time, while in multiprogramming the prime focus is to maximise the CPU usage.

- It allows multiple users to share computers simultaneously.
- A program loaded into memory and executed, it performs a short period of time either before completion or to complete I/O.
- This short period of time during which the user gets the CPU is known as time slice, time slot or quantum.

The main difference between Multiprogrammed Batch Systems and Time-Sharing Systems is that in case of Multiprogrammed batch systems, the objective is to maximise processor use, whereas in Time-Sharing Systems, the objective is to minimise response time.

2. **Distinguish between hard real time systems and soft real time systems**

S.NO	Hard Real-Time System	Soft Real-Time System
1.	A hard-real time system is a system in which a failure to meet even a single deadline may lead to complete or appalling system failure.	A soft real-time system is a system in which one or more failures to meet the deadline are not considered complete system failure, but that performance is considered to be degraded.
2.	In a hard real-time system, the size of a data file is small or medium.	In a soft real-time system, the size of the data file is large.
	In this system, response time	In this system, response time is

3.	is predefined that is in a millisecond.	higher.
4.	A hard-real time system has more utility.	A soft real-time system has less utility.
5.	A hard real-time system has short databases.	A soft real-time system has enlarged databases.
6.	Peak load performance should be predictable.	In a soft real-time system, peak load can be tolerated.
7.	In this system, safety is critical.	In this system, safety is not critical.
8.	Hard real-time systems have short term data integrity.	Soft real-time systems have long term data integrity.
9.	A hard real-time system is very restrictive.	A Soft real-time system is less restrictive.
10.	In case of an error in a hard real-time system, the computation is rolled back.	In a soft real-time system, computation is rolled back to a previously established checkpoint to initiate a recovery action.
11.	All users of hard real-time systems get validation when needed.	All users of soft real-time systems do not get validation.
12.	Satellite launch, Railway signalling systems, and	DVD players, telephone switches, electronic games, Linux, and

	Safety-critical systems are good examples of a hard real-time system.	many other OS provide a soft real-time system.

3. **What are the three main purposes of an operating system?**

   The purpose of an operating system is to provide a platform on which a user can execute programs in a convenient and efficient manner. An operating system is a piece of software that manages the allocation of computer hardware.

   Security : The operating system uses password protection to protect user data and similar nother techniques. It also prevents unauthorised access to programs and user data.

   Control over system performance : Monitors overall system health to help improve performance, records the response time between service requests and system response to have a complete view of the system health.

   Job accounting : Operating system Keeps track of time and resources used by various tasks and users, this information can be used to track resource usage for a particular user or group of users.

   Error detecting aids : Operating system constantly monitors the system to detect errors and avoid the malfunctioning of computer systems.

   Coordination between other software and users: Operating systems also coordinate and assign interpreters, compilers, assemblers and other software to the various users of the computer systems.

4. **What is a distributed operating system? What are the advantages of a distributed operating system?**

   DOS:

   - A distributed operating system (DOS) is an essential type of operating system. Distributed systems use many central processors to serve multiple real-time applications and users. As a result, data processing jobs are distributed between the processors.

   - It connects multiple computers via a single communication channel. Furthermore, each of these systems has its own processor and memory. Additionally, these CPUs communicate via high-speed buses or telephone lines. Individual systems that communicate via a single channel are regarded as a single entity. They're also known as loosely coupled systems.

   - This operating system consists of numerous computers, nodes, and sites joined together via LAN/WAN lines. It enables the distribution of full systems on a couple of centre processors, and it supports many real-time products and different users. Distributed operating systems can share their computing resources and I/O files while providing users with virtual machine abstraction.

Advantages:

There are various advantages of the distributed operating system. Some of them are as follow:

- It may share all resources (CPU, disk, network interface, nodes, computers, and so on) from one site to another, increasing data availability across the entire system.
- It reduces the probability of data corruption because all data is replicated across all sites; if one site fails, the user can access data from another operational site.
- The entire system operates independently of one another, and as a result, if one site crashes, the entire system does not halt.
- It increases the speed of data exchange from one site to another site.
- It is an open system since it may be accessed from both local and remote locations.
- It helps in the reduction of data processing time.
- Most distributed systems are made up of several nodes that interact to make them fault-tolerant. If a single machine fails, the system remains operational.

5. Explain why you think that idleness in CPU occurs.

A computer processor is described as idle when it is not being used by any program. Every program or task that runs on a computer system occupies a certain amount of processing time on the CPU. If the CPU has completed all tasks it is idle.

Modern processors use idle time to save power. Some programs are designed to appear to make use of CPU idle time, meaning that they run at a low priority (but slightly higher than idle priority) so as not to impact programs that run at normal priority. This allows non-crucial background programs to only run when it would not affect the performance of other applications. Many operating systems, for example Windows, Linux, and macOS will run an idle task, which is a

special task loaded by the OS scheduler on a CPU when there is nothing for the CPU to do.

6. **Explain the difference between interrupt and exception.**

   Interrupt is one of the classes of Exception. There are 4 classes of Exception- interrupt, trap, fault and abort. Though, interrupt belongs to exceptions still there are many differences between them.

   Interrupt:

   The term Interrupt is usually reserved for hardware interrupts. They are program control interruptions caused by external hardware events. Here, external means external to the CPU. Hardware interrupts usually come from many different sources such as timer chip, peripheral devices (keyboards, mouse, etc.), I/O ports (serial, parallel, etc.), disk drives, CMOS clock, expansion cards (sound card, video card, etc). That means hardware interrupts almost never occur due to some event related to the executing program.

   Exception:

   Exception is a software interrupt, which can be identified as a special handler routine. Exceptions can be identified as an automatically occurring trap. Generally, there are no specific instructions associated with exceptions (traps are generated using a specific instruction). So, an exception occurs due to an “exceptional” condition that occurs during program execution

Interrupt	Exception
These are Hardware interrupts.	These are Software interrupts.
Occurrences of hardware	This is not a true case in terms

interrupts usually disable other hardware interrupts.	of Exception.
These are asynchronous external requests for service (like keyboard or printer needs service).	These are synchronous internal requests for service based upon abnormal events (think of illegal instructions, illegal address, overflow etc).
Being asynchronous, interrupts can occur at any place in the program.	Being synchronous, exceptions occur when there is an abnormal event in your program like, divide by zero or illegal memory location.
These are normal events and shouldn't interfere with the normal running of a computer.	These are abnormal events and often result in the termination of a program

7. **Differentiate between tightly coupled systems and loosely coupled systems.**

   Loosely Coupled Multiprocessor System

   - In this system, every processor has its own memory module.
   - It is efficient when there is less interaction between tasks running on different processors.
   - There are no memory conflicts in general.
   - It is considered as a Message transfer system (MTS).
   - It is less expensive.
   - It has a low data rate. It has distributed memory.
   - They are usually seen in distributed computing systems

   Tightly Coupled Multiprocessor System

   - In this system, the processors share memory modules.

- It is efficient when used with real-time processing.
- It provides high speed.
- It has memory conflicts.
- They are connected through networks such as PMIN, IOPIN, ISIN.
- It has a high data rate.
- It is expensive.
- It is usually seen in parallel processing systems.

8. Explain. OS is a resource manager. If so justify your answer

- Operating system is a resource allocator which manages all resources and decides between conflicting requests for efficient and fair resources.
- The operating system provides for an orderly and controlled allocation of the processors, memories, and I/O devices among the various programs in the bottom-up view.
- Now-a-days all modern computers consist of processors, memories, timers, network interfaces, printers, and so many other devices.
- Operating system allows multiple programs to be in memory and run at the same time. Resource management includes multiplexing or sharing resources in two different ways: in time and in space.
- In time multiplexed, different programs take a chance of using CPU. First one tries to use the resource, then the next one that is ready in the queue and so on. For example: Sharing the printer one after another.
- In space multiplexing, Instead of the customers taking a chance, each one gets part of the resource. For example –

Main memory is divided into several running programs, so each one can be resident at the same time.

Refer lecture notes pg.no 2-3

9. **Discuss the view of an operating system as a resource manager.**
    - Operating system is a resource allocator manages all resources and decides between conflicting requests for efficient and fair resources.
    - The operating system provides for an orderly and controlled allocation of the processors, memories, and I/O devices among the various programs in the bottom-up view.
    - Now-a-days all modern computers consist of processors, memories, timers, network interfaces, printers, and so many other devices.
    - Operating system allows multiple programs to be in memory and run at the same time. Resource management includes multiplexing or sharing resources in two different ways: in time and in space.
    - In time multiplexed, different programs take a chance of using CPU. First one tries to use the resource, then the next one that is ready in the queue and so on. For example: Sharing the printer one after another.
    - In space multiplexing, Instead of the customers taking a chance, each one gets part of the resource. For example – Main memory is divided into several running programs, so each one can be resident at the same time.

    Refer lecture notes pg.no 2-3

10. **Define essential properties of the following types of Operating system:**

i) Batch operating system

In a Batch operating system, the user does not have direct access to the computer and cannot directly interact with it either.

In this type of OS, jobs are prepared for each user, and all those jobs have been imprinted or stored in the punch card-like structure, which is submitted to the computer operator.

This kind of operating system mostly works on offline devices and once the punch card is submitted to the computer operator, the computer works according to the code or program written on the card.

ii) Interactive operating system

In an Interactive operating system, there is a direct interaction between the user and the computer. Mostly, all personal computers use Interactive operating systems. In this kind of operating system, the user enters some command in the system and the system works according to it.

iii) Time sharing operating system

It is similar to the multiprogramming system with some additional extensions and also known as Multitasking OS. In a Time sharing OS, the system is capable of handling multiple jobs simultaneously and here the processing time is shared among all the users.

With Time sharing OS, users at different locations or terminals can access the same computer at the same time. Here, the CPU uses the switching mechanism that helps it to switch from one job to another so that each job gets equal processing time.

iv) Real time operating system

An RTOS is a data processing system whose response time to the input is very short. RTOS is also known as the brain of the real-time system because of its immediate response to the input.

The response to the input in RTOS is displayed in a specific time period. Though the time period is very short it does not show any kind of disparity. There are 2 types of Real-Time operating systems:

- Hard Real-Time System: In a Hard Real-Time system, if the response takes more time than the specified time interval, the system will fail. The secondary storage is also limited in these systems.
- Soft Real-Time System: The Soft Real-Time system does not fail the program even if the response takes more time than the specified time. It would just show the output, however, it can compromise the accuracy of the response.

v) Distributed operating system

In this operating system, different computers interact with one another and communicate in order to exchange data. The Internet works on this system where everyone is linked with each other to communicate. It can also be termed as the Networking operating system, which supports a high level of communication.

A Distributed OS uses multiple processors to perform multiple real-time applications on the user terminal. In this system, the processor does not share the memory because each CPU has its own local memory.

PART-B

1. **Define an operating system. State and explain the basic functions or services of an operating system**

   A program that acts as an intermediary between a user of a computer and the computer hardware.

   Operating system is a resource allocator manages all resources and decides between conflicting requests for efficient and fair resources.

   Operating system is a control program that controls execution of programs to prevent errors and improper use of the computer.

   Functions:

   - Memory Management: It is also an important function of the operating system. The memory cannot be managed without an operating system.
   - Loading and execution: A program is loaded in the memory before it can be executed. Operating systems provide the facility to load programs in memory easily and then execute them.
   - Data security: data is an important part of a computer system. The operating system protects the data stored on the computer from illegal use, modification or deletion.
   - Disk management: operating system manages the disk space. It manages the stored files and folders in a proper way.
   - Process management: CPU can perform one task at one time. If there are many tasks, the operating system decides which task should get the CPU.
   - Device controlling: the hardware devices are controlled with the help of small software called device drivers. It keeps track of all devices connected to the system.

- Job accounting : Operating system Keeps track of time and resources used by various tasks and users, this information can be used to track resource usage for a particular user or group of users.
- Error detecting aids : Operating system constantly monitors the system to detect errors and avoid the malfunctioning of computer systems.

2. **Explain the differences between multiprogramming and time-sharing systems**

   Multiprogramming systems:

   Multiprogramming OS is an ability of an operating system that executes more than one program using a single processor machine.

   More than one task or program or jobs are present inside the main memory at one point of time.

   Buffering and spooling can overlap I/O and CPU tasks to improve the system performance but it has some limitations that a single user cannot always keep CPU or I/O busy all the time

   · The operating system picks up and begins to execute one of the jobs from memory.

   · Once this job needs an input and output operation the operating system switches to another job.

   Time sharing systems:

   · These are very similar to multiprogramming systems and an extension of multiprogramming systems.

· The prime focus is on minimising the response time, while in multiprogramming the prime focus is to maximise the CPU usage.

· It allows multiple users to share computers simultaneously.

· A program loaded into memory and executes, it performs a short period of time either before completion or to complete I/O.

· This short period of time during which the user gets the CPU is known as time slice, time slot or quantum.

3. **Explain how operating system services are provided by system calls.**

A system call is a method for a computer program to request a service from the kernel of the operating system on which it is running. A system call is a method of interacting with the operating system via programs. A system call is a request from computer software to an operating system's kernel.

There are various situations where you must require system calls in the operating system. Following of the situations are as follows:

I. It is required when a file system wants to create or delete a file.

II. Network connections require the system calls to send and receive data packets.

III. If you want to read or write a file, you need to make system calls.

IV. If you want to access hardware devices, including a printer, scanner, you need a system call.

V. System calls are used to create and manage new processes.

4. **Describe the operating system structures.**

General purpose operating system is a very large program.

· Various ways to structure ones:

. Simple structure: MS-DOS is written to provide the most functionality in the least space. Not divided into modules. Although MS-DOS has come to be structured, its interfaces and levels of functionality are not well separated.

. More complex: The UNIX OS consists of 2 parts:

Systems programs.

The kernel consists of everything below the sys-call interface & above the hardware. A large no.of function for one level.

. Layered (an abstraction): The OS is divided into a no. of layers, each built on top of lower layers. The bottom layer is the hardware, the highest is the user interface. With modularity, layers are selected such that each uses functions and services of only lower level layers.

. Microkernel: Moves as much from kernel into user interface. Mach is an example of a microkernel. Communication takes place between user modules using message passing.

5. **Distinguish between user mode and kernel mode operations of the operating System**

. User mode: When a computer application is running, it is in the user mode. These are application programs so the computer is in user mode. When the process is in user mode and requires any hardware

resource, that request is sent to the kernel. As there is a limited access to hardware in this mode, it is known as less privileged mode, slave mode or restricted mode.

· Kernel mode: A kernel is a software program which is used to access hardware components of a computer system. Kernel works as a middleware software for hardware and application software/user programs. Kernel mode is generally reserved for low level trusted functions of the operating system. When the process is executing in user mode and if that process requires hardware resources such as RAM, printer etc, that process should send a request to the kernel. These requests are sent through system calls.

6. **Define the essential properties of the operating systems.**

The essential properties of the different types of operating systems are as follows –

- Batch Operating system

  In a Batch operating system, the user does not have direct access to the computer and cannot directly interact with it either.

  In this type of OS, jobs are prepared for each user, and all those jobs have been imprinted or stored in the punch card-like structure, which is submitted to the computer operator.

  This kind of operating system mostly works on offline devices

  Jobs with similar needs are batched together and run through the computer as a group by an operator or automatic job sequencer. Performance is increased by attempting to keep

CPU and I/O devices busy at all times through buffering, off line operation, spooling and multiprogramming.

- Interactive operating system

  The system is composed of many short transactions where the results of the next transaction may be predictable. The response time needs to be short because the user submits and waits for the result.

- Time sharing Operating system

  This system uses CPU scheduling and multiprogramming to provide economical interactive use of the system. The CPU switches rapidly from one user to another. Instead of having a job defined by spooled card images, each program reads its next control card from the terminal, and output is normally printed immediately to the screen.

- Real-time operating system

  Often used in a dedicated application, this system reads information from sensors and must respond within a fixed amount of time to ensure correct performance.

- Network operating system

  It provides operating system features across a network such as file sharing.

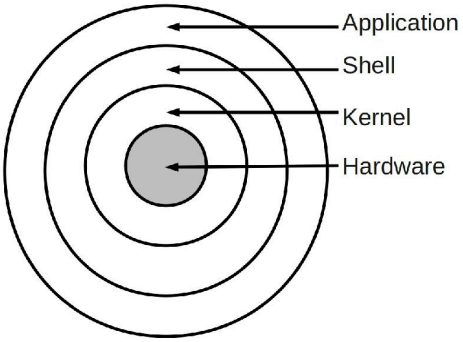
- Symmetric Multiprocessing (SMP)

  It is used in systems where there are multiple CPUs each running the same copy of the operating system. Communication takes place across the system bus.

- Distributed Operating system

This system distributed computation among several physical processors. The processors do not share a memory or a clock. Instead, each processor has its own local memory. They communicate with each other through various communication lines, like high-speed buses or LAN.

7. **Explain the architecture of an operating system.**



An operating system is a program that acts as an interface between a user of a computer and the computer resources. The purpose of an operating system is to provide an environment in which a user may execute programs.

Hardware

The hardware consists of the memory, CPU, arithmetic-logic unit, various bulk storage devices, I/O, peripheral devices and other physical devices.

Kernel

In computing, the kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level. The kernel's responsibilities include managing the system's resources (the communication between hardware and software components). Usually as a basic component of an operating system, a kernel can

provide the lowest-level abstraction layer for the resources (especially processors and I/O devices) that application software must control to perform its function. It typically makes these facilities available to application processes through inter-process communication mechanisms and system calls.

Shell

A shell is a piece of software that provides an interface for users to an operating system which provides access to the services of a kernel. The name shell originates from shells being an outer layer of interface between the user and the innards of the operating system (the kernel). [Wikipedia]

Operating system shells generally fall into one of two categories: command-line and graphical. Command-line shells provide a command-line interface (CLI) to the operating system, while graphical shells provide a graphical user interface (GUI)

Refer here

8. **Distinguish between multiprogramming, multitasking and multiprocessing.**

Multiprogramming: In this the operating system picks up and begins to execute one of the jobs from memory. Once this job needs an I/O operation, the operating system switches to another job (CPU and OS are always busy). Jobs in the memory are always less than the number of jobs on disk(Job Pool). If several jobs are ready to run at the same time, then the system chooses which one to run through the process of CPUScheduling.

In a Non-multiprogrammed system, there are moments when the CPU sits idle and does not do any work. In a Multiprogramming system, the CPU will never be idle and keeps on processing.

Multitasking: A multitasking operating system (OS) is one that can work on more than one task at a time by switching between the tasks very rapidly. The tasks may all pertain to a single user or to multiple users. A multitasking OS can save the current state of each user and task so that it does not lose its place when it comes back to a task to resume its work. This allows the system to switch smoothly between tasks.

Multiprocessing: consists of several processors that share a common physical memory. Higher computing power & speed. All processors operate under a single OS. Multiplicity of the processor & how they act together are transparent to the others. It enhanced performance. Execution of several tasks by different processors concurrently. System divides tasks into many subtasks and then these subtasks can be executed in parallel in different processors.

9. Describe briefly about Batch programming

There is no direct interaction between the user and the computer.

· Users have to submit a job to a computer operator.

· Then the computer operator places a batch of several jobs on an I/P device.

· Jobs are batched together by type of language and requirements.

· Then a special program, the monitor, manages the execution of each program in the batch.

· The monitor is always a main memory and available for execution.

10. Define MultiTasking and MultiThreading?

- Multi Tasking: A multitasking operating system (OS) is one that can work on more than one task at a time by switching between the tasks very rapidly. The tasks may all pertain to a single user or to multiple users. A multitasking OS can save the current state of each user and task so that it does not lose its place when it comes back to a task to resume its work. This allows the system to switch smoothly between tasks.
- Multi Threading: Multithreading is the ability of a program or an operating system program to manage its use by more than one user at a time and to even manage multiple requests by the same user without having to have multiple copies of the programming running in the computer. Majority of programs written nowadays run as a single thread. These tasks cannot be executed by the program at the same time. A process is a program being executed. A process can be further divided into independent units known as threads.

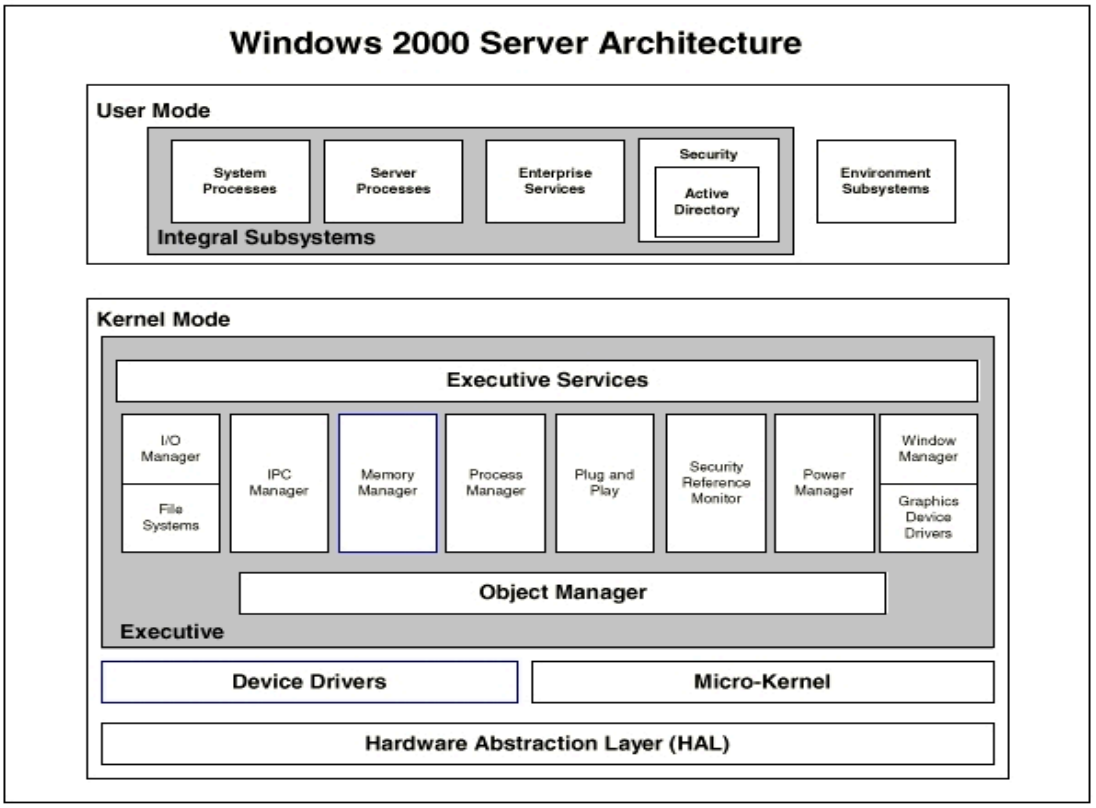
11. **Draw and explain the architecture of Windows 2000 and traditional UNIX.**

Windows 2000 is an operating system first released by Microsoft in February 2000.Code-named Janus during development, Windows 2000 is based on the Windows NT kernel, and is sometimes referred to as Windows NT 5.0. Windows 2000 contains over 29 million lines of code, mainly written in C++ with over 8 million of those lines written for drivers. Windows 2000 is by far one of the largest commercial projects ever built.

Windows 2000 included advanced features and had overall better support for computer hardware.
Some of the significant features of Windows 2000 Professional are:

- Support for FAT16, FAT32, and NTFS.

- Increased uptime of the system and significantly fewer OS reboot scenarios.
- Windows Installer tracks applications and recognizes and replaces missing components.
- Protects memory of individual apps and processes to avoid a single app bringing the system down.
- Encrypted file systems protect sensitive data.



12. **State the differences between system call and system program.**

System program: These are the programs that are used and required to run the system - machine, input output devices and other connected peripherals. They are also known as System softwares.

System calls: System calls are the calls made by the applications or the processors for a particular execution of a code block; also known as interrupts in a computer. You can call the CPU to execute your program with a high priority and execute other commands later.

System Call	System Program
Allow the user process to request the services of the operating system.	Creates an environment for programs to develop and execute.
Defines interface to the services of the operating system.	Defines a user interface of the operating system.
It satisfies the low-level request of the user program.	It satisfies the high-level request of the user program.
Invokes the services of the operating system.	Initiates a sequence of system calls to satisfy a user request.
Usually written in C and C++. But where direct hardware access is required the call is written using assembly level language.	System programs are written in high-level languages only.
Process control, file manipulation, device manipulation, information maintenance, communications, and protection.	File management, status information, file modification, Programming-language support, program loading and execution, communication

13. **Distinguish between the client-server and peer-to-peer models of distributed systems.**

Client Server	Peer-to-peer
There is a specific server and specific clients connected to the server.	Clients and servers are not distinguished; each node acts as client and server.

The client requests for service and the server responds with the service.	Each node can request for services and can also provide the services.
The data is stored in a centralised server.	Each peer has its own data.
When several clients request for the services simultaneously, a server can get bottlenecked.	As the services are provided by several servers distributed in the peer-to-peer system, a server is not bottlenecked.
The client-server is expensive to implement.	Peer-to-peer are less expensive to implement.
Client-Server is more stable and scalable.	Peer-to-Peer suffers if the number of peers increases in the system.

14. **Explain types of System calls.**

Here are the types of system calls –

- 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.

15. **Describe the kernel structure of the operating system.**

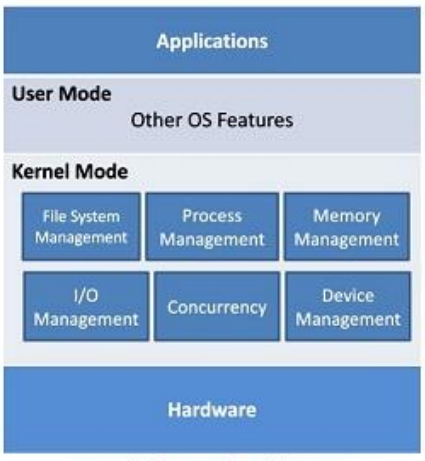
Kernel is the core of the operating system. Kernel controls everything in a computer.

Different types of Kernel Architecture are:

Monolithic architecture

In Monolithic kernel mode, the operating system runs in a single address space. Monolithic kernel has all the operating system functions or services within a single kernel. This single kernel will run as a single process in a single address space in memory. Monolithic architecture enables higher performance, however less flexible for modifications to add new features or enhance existing features.

Ex: All Linux distributions, Android uses a modified Linux Kernel.

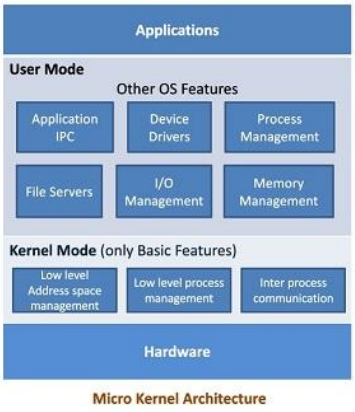


Monolithic Kernel Architecture

Microkernel architecture

Microkernel architecture is an architecture with kernel having the basic interaction with hardware and the basic Inter-Process Communication mechanisms. All the other Operating System services exist outside the Kernel. Microkernel provides the flexibility to add new features or modify existing features while slightly affecting performance as it increases the amount of interactions between kernel and user mode features.

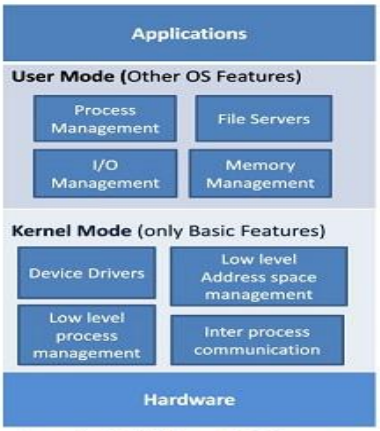
Ex: Mach, OKL4, Codezero, Fiasco.OC, PikeOS, seL4, QNX



Micro Kernel Architecture

Hybrid Kernel architecture

Hybrid architecture tries to get the best features of both monolithic kernels as well as microkernels. Hybrid kernel aims to have optimal performance and the flexibility to modify and upgrade kernel services.



Example Hybrid Kernel Architecture. Implementations may differ

Ex: Apple IOS, Apple macOS for desktop machines uses hybrid architecture

16. **Discuss about evolution of operating system**

Operating system is divided into four generations, which are explained as follows –

- First Generation (1945-1955) It is the beginning of the development of electronic computing systems which are substitutes for mechanical computing systems. Because of the drawbacks in mechanical computing systems like, the speed of humans to calculate is limited and humans can easily make mistakes. In this generation there is no operating system, so the computer system is given instructions which must be done directly.

  Example – Type of operating system and devices used is Plug Boards.

- Second Generation (1955-1965) The Batch processing system was introduced in the second generation, where a job or a task that can be done in a series, and then executed sequentially. In this generation, the computer system is not equipped with an

operating system, but several operating system functions exist like FMS and IBSYS.

Example – Type of operating system and devices used is Batch systems.

- Third Generation (1965-1980) The development of the operating system was developed to serve multiple users at once in the third generation. Here the interactive users can communicate through an online terminal to a computer, so the operating system becomes multi-user and multiprogramming. Example – Type of operating system and devices used is Multiprogramming.
- Fourth Generation (1980-Now) In this generation the operating system is used for computer networks where users are aware of the existence of computers that are connected to one another.

  At this generation users are also comforted with a Graphical User Interface (GUI), which is an extremely comfortable graphical computer interface, and the era of distributed computing has also begun.

17. **Explain about system calls between user mode and kernel mode.**

BASIS	KERNEL MODE	USER MODE
Permission	Unrestricted and full permissions to access the system's hardware	Restricted and limited permissions to access the system's hardware
Memory reference	It can reference to both the memory spaces	It can only reference to memory space that is dedicated to user mode
Access	Only core functionality can be allowed to operate in this very mode	User applications can access this mode for a particular system and is allowed to operate in this particular mode
System crash	Fatal and increases the complexity	Recoverable and can simply restart the session
Also Known	Privileged mode or Supervisor mode	Restricted mode

18. **Compare Multiprogramming and Multitasking**

Multiprogramming	Multitasking
In multiprogramming, multiple programs execute at a same time on a single device.	In Multitasking, a single resource is used to process multiple tasks.
The process resides in the main memory.	The process resides in the same CPU.
It uses batch OS. The CPU is utilized completely while execution.	It is time sharing as the task assigned switches regularly.
The processing is slower, as a single job resides in the main memory while execution.	Multitasking follows the concept of context switching.

19. **Explain various types of computer systems?**

A computer is a device that transforms data into meaningful information. It processes the input according to the set of instructions provided to it by the user and gives the desired output. Computers are of various types and they can be categorised in two ways on the basis of size and on the basis of data handling capabilities.

So, on the basis of size, there are five types of computers:

1. Supercomputer
2. Mainframe computer
3. Minicomputer
4. Workstation
5. PC (Personal Computer)

1. Supercomputer:

When we talk about speed, then the first name that comes to mind when thinking of computers is supercomputers. They are the biggest and fastest computers (in terms of speed of processing data). Supercomputers are designed such that they can process a huge amount of data, like processing trillions of instructions or data just in a second. This is because of the thousands of interconnected processors in supercomputers. It is basically used in scientific and engineering applications such as weather forecasting, scientific simulations, and nuclear energy research. It was first developed by Roger Cray in 1976.

2. Mainframe computer:

Mainframe computers are designed in such a way that it can support hundreds or thousands of users at the same time. It also supports multiple programs simultaneously. So, they can execute different processes simultaneously. All these features make the mainframe computer ideal for big organisations like banking, telecom sectors, etc., which processes a high volume of data in general.

3. Minicomputer:

Minicomputer is a medium size multiprocessing computer. In this type of computer, there are two or more processors, and it supports 4 to 200 users at one time. Minicomputers are used in places like

institutes or departments for different work like billing, accounting, inventory management etc. It is smaller than a mainframe computer but larger in comparison to the microcomputer.

4. Workstation:

Workstation is designed for technical or scientific applications. It consists of a fast microprocessor, with a large amount of RAM and high speed graphic adapter. It is a single-user computer. It generally used to perform a specific task with great accuracy.

5. PC (Personal Computer):

It is also known as a microcomputer. It is basically a general-purpose computer and designed for individual use. It consists of a microprocessor as a central processing unit(CPU), memory, input unit, and output unit. This kind of computer is suitable for personal work such as making an assignment, watching a movie, or at the office for office work, etc. For example, Laptops and desktop computers.

20. **Compare user mode and kernel mode.**

What is the Difference Between User Mode and Kernel Mode?

User Mode vs Kernel Mode	
User Mode is a restricted mode, which the application programs are executing and starts out.	Kernel Mode is the privileged mode, which the computer enters when accessing hardware resources.
Modes	
User Mode is considered as the slave mode or the restricted mode.	Kernel mode is the system mode, master mode or the privileged mode.
Address Space	
In User mode, a process gets their own address space.	In Kernel Mode, processes get single address space.
Interruptions	
In User Mode, if an interrupt occurs, only one process fails.	In Kernel Mode, if an interrupt occurs, the whole operating system might fail.
Restrictions	
In user mode, there are restrictions to access kernel programs. Cannot access them directly.	In kernel mode, both user programs and kernel programs can be accessed.

PART-C

1. Define Operating Systems.
2. Define Distributed Systems.
3. How is user mode different from kernel mode?
4. Define a Multiprocessor system?
5. Define System call?
6. Define interrupt?
7. Define Time Sharing Systems?
8. Write the various types of OS components?

9. List any four functions of the Operating system?

10. Define Kernel.

11. Define a real time operating system.

12. Define Virtual Machine?

13. Explain how protection is provided for the hardware resources by the operating system.

14. Discuss about batch systems?

15. Explain about a parallel distributed system?

16. Discuss about OS architecture?

17. Discuss about protection and security by OS?

18. What are the advantages of layered structure?

19. Define Virtual Memory? Describe the use of fork () and exec () system calls?

IARE '24 DISCORD TEAM

II-II

OS MODULE 2 SOLUTIONS

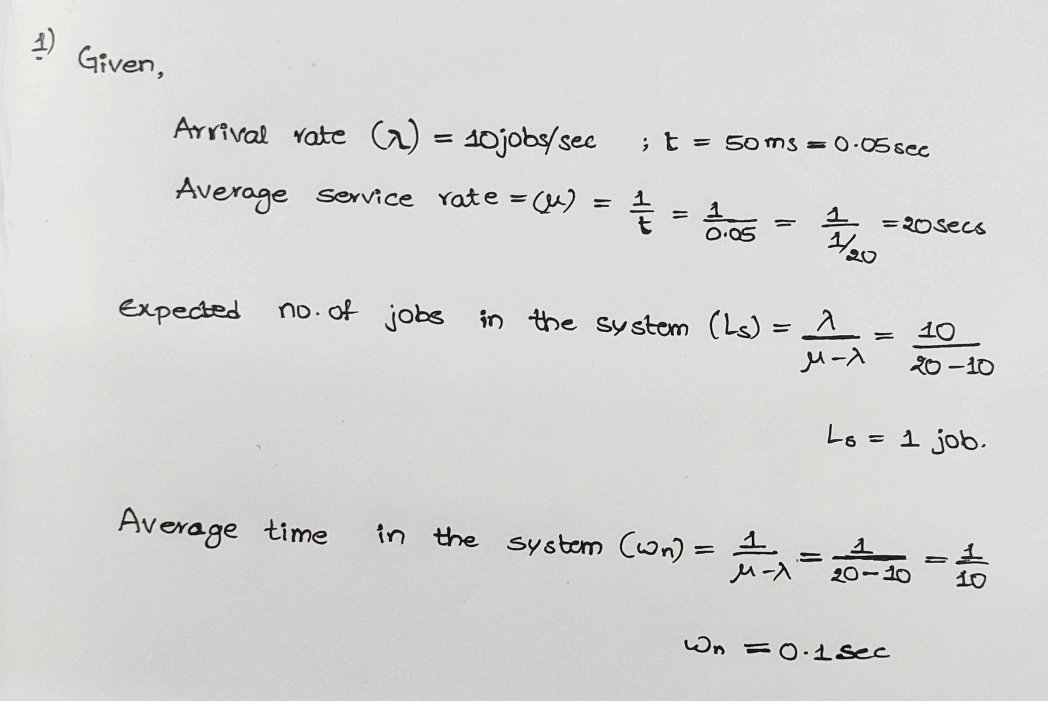
SUHRUTH • ANUSHKA • IKRAM • ABHIRAMI • UJJWAL

PROCESS AND CPU SCHEDULING,
PROCESS COORDINATION

MODULE - 2

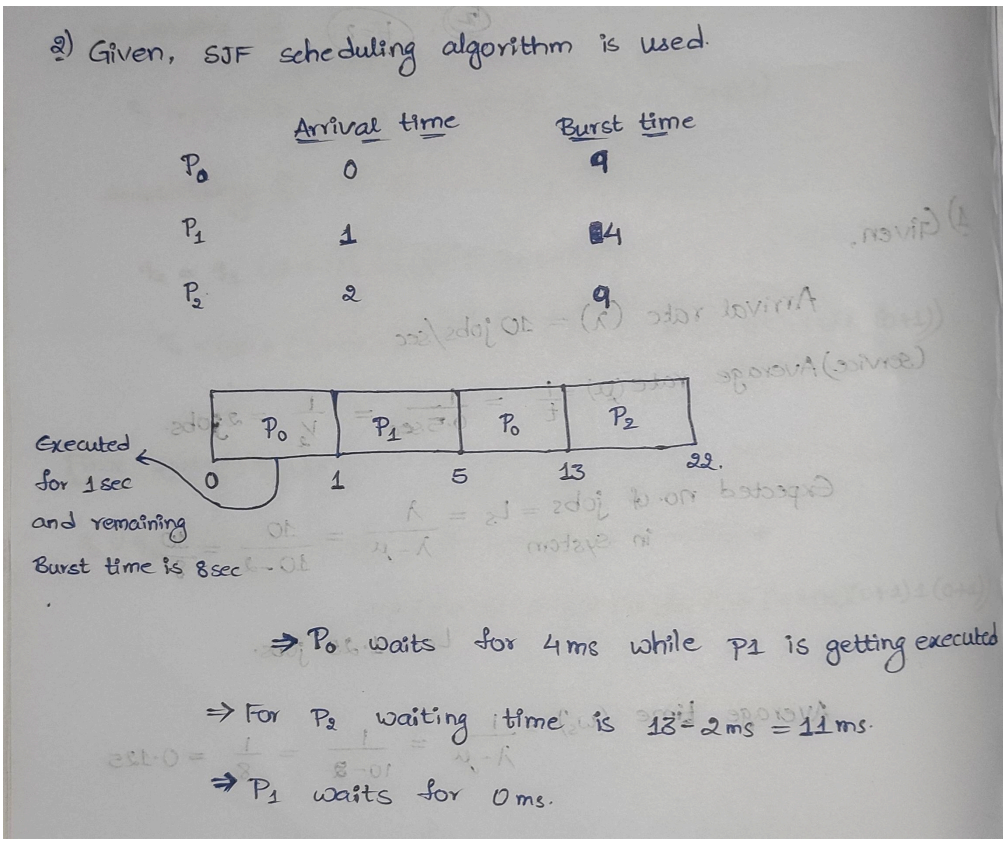
PART - A

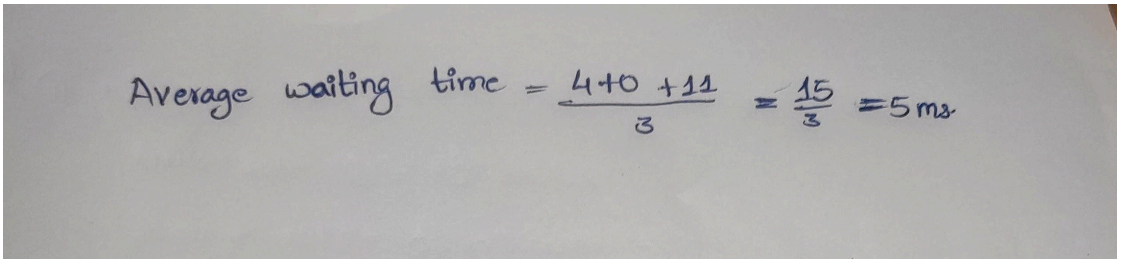
1. Suppose we have a single processor system and jobs arrive at a rate of 10 jobs/sec, suppose each job takes an average of 50 milliseconds to complete. Assure that both distributions are exponential. State the expected number of jobs in the system and the average time in the system.



Answer Verified ✅

2. Consider the following table of arrival time and burst time for three processes P0, P1 and P2. Process Arrival time and Burst Time P0-0ms, 9ms; P1-1ms, 4ms; P2-2ms, 9ms. The preemptive shortest job first scheduling algorithm is used. Scheduling is carried out only at arrival or completion of processes. What is the average waiting time for the three processes?





Answer Verified ✅

3. What are the 3 different types of scheduling queues?

The 3 types of scheduling queues are:

- Job Queue
- Ready Queue
- Device Queue (or) Waiting Queue

i) Job Queue: Starting, all the processes get stored in the job queue. It is maintained in the secondary memory. The long term scheduler(Job scheduler) picks some of the jobs and puts them in the primary memory.

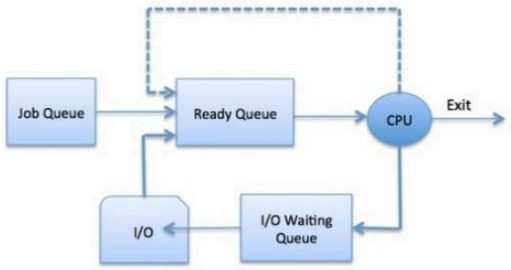
ii) Ready Queue: Is maintained in primary memory. The short term scheduler picks the job from the ready queue and dispatches it to the CPU for the execution.

iii) Waiting Queue: When the process needs I/O operation in order to complete its execution, the OS changes the state of the process from running to waiting. The context (PCB) associated with the process gets stored on the waiting queue which will be used by the processor when the process finishes the I/O.

state queue.

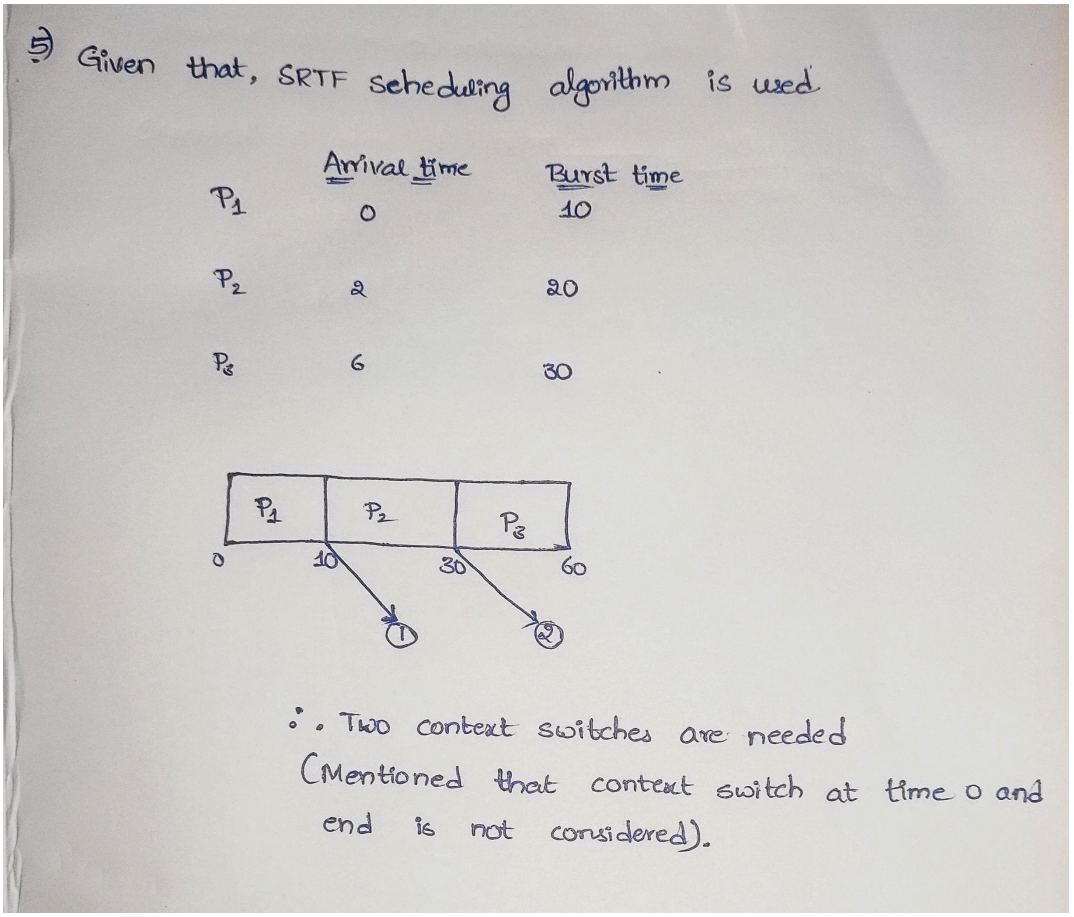
The Operating System maintains the following important process scheduling queues −

- **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.



4. Explain the advantage of using semaphores over Test and Set () and Swap() functions. Describe the use of wait() and signal() functions on semaphore and how these can provide the solution to the critical section problem.

5. Consider three CPU - intensive processes which require 10, 20 and 30 time units to arrive at times 0, 2 and 6 respectively. How many context switches are needed if the operating system implements a shortest remaining time first scheduling algorithm. Do not count the context switches at time zero and at the end.

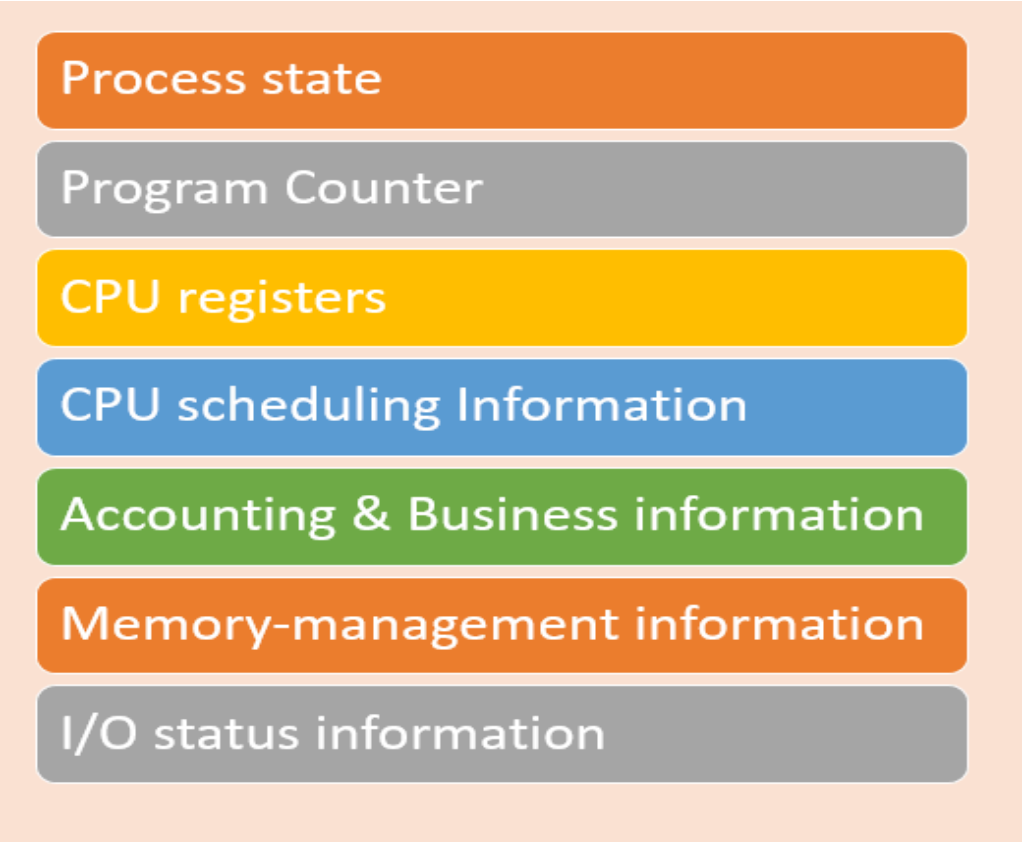


6. Construct Process Control Block for any given example.

A process control block (PCB) is a data structure which contains information about the process, i.e. registers, quantum, priority, etc. The process table is an array of PCBs that logically contains a PCB for all of the current processes in the 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 **–**

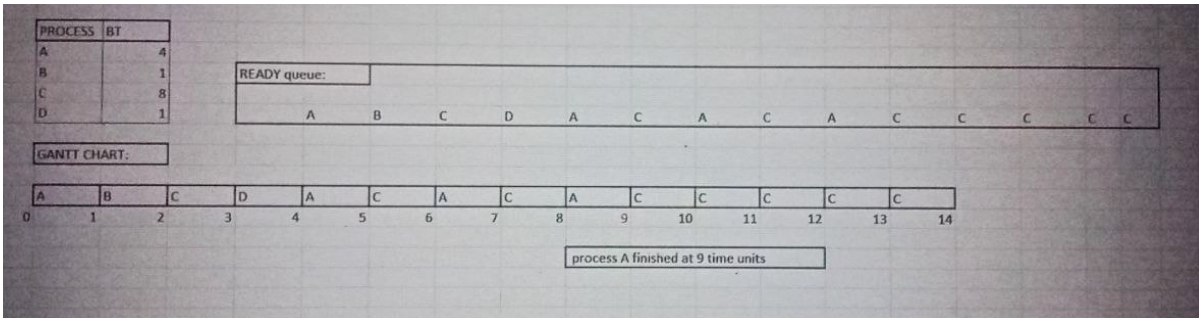


- **Process state:** A process can be new, ready, running, waiting, etc.
- **Program counter:** The program counter lets you know the address of the next instruction, which should be executed for that process.
- **CPU registers:** This component includes accumulators, index and general-purpose registers, and information of condition code.
- **CPU scheduling information:** This component includes a process priority, pointers for scheduling queues, and various other scheduling parameters.
- **Accounting and business information:** It includes the amount of CPU and time utilities like real time used, job or process numbers, etc.
- **Memory-management information:** This information includes the value of the base and limit registers, the page, or segment tables. This depends on the memory system, which is used by the operating system.

- **I/O status information:** This block includes a list of open files, the list of I/O devices that are allocated to the process, etc.

Refer- What is Process Control Block (PCB)?
or
Process Table and Process Control Block (PCB) - GeeksforGeeks

7. Explain Four jobs to be executed on a single processor system arrive at time 0 in the order A, B, C, D their burst CPU time requirements are 4, 1, 8, 1 time units respectively. The completion time of A under round robin scheduling with a time slice of one time unit is?



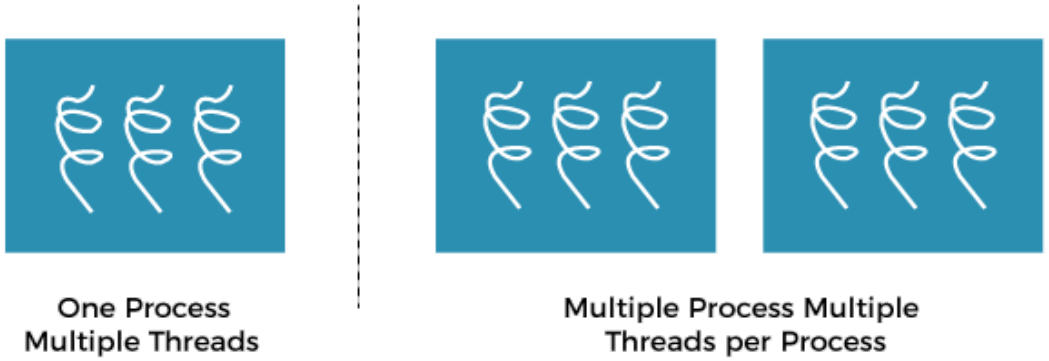
8. Explain the concept of multi threading. Discuss the following multi threading models.

a) **Many-to-one.**
b) **One-to-one.**
c) **Many-to-many.**
d) **Two-level**

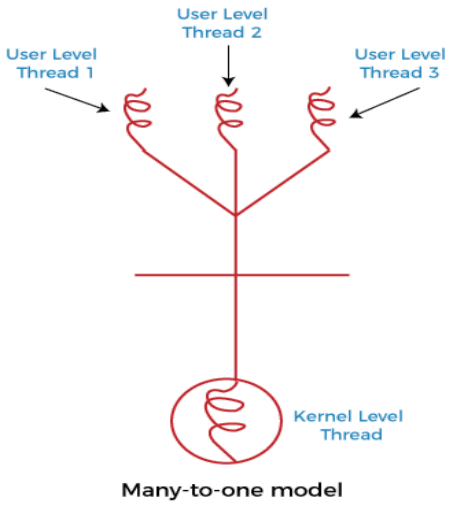
Multithreading Model:

Multithreading allows the application to divide its task into individual threads. In multi-threads, the same process or task can be done by the

number of threads, or we can say that there is more than one thread to perform the task in multithreading. With the use of multithreading, multitasking can be achieved.

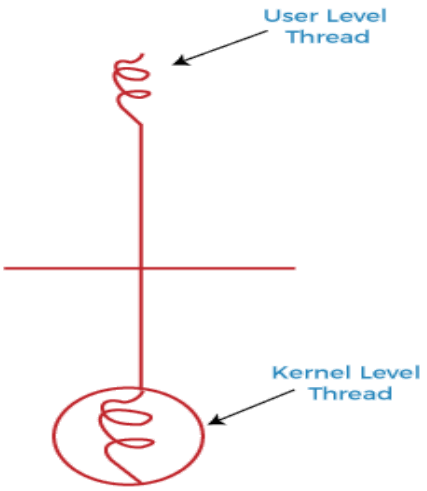


The main drawback of single threading systems is that only one task can be performed at a time, so to overcome the drawback of this single threading, there is multithreading that allows multiple tasks to be performed.



One to one multithreading model

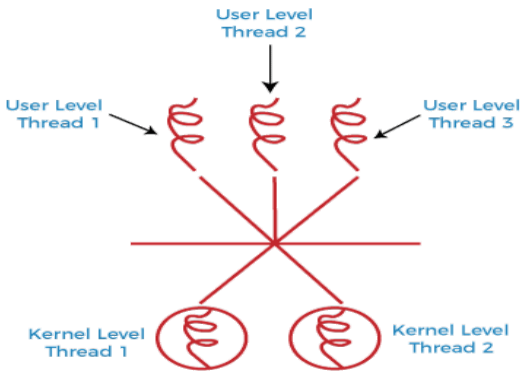
The one-to-one model maps a single user-level thread to a single kernel-level thread. This type of relationship facilitates the running of multiple threads in parallel. However, this benefit comes with its drawback. The generation of every new user thread must include creating a corresponding kernel thread causing an overhead, which can hinder the performance of the parent process



One-to-one model

Many to Many Model multithreading model

In this type of model, there are several user-level threads and several kernel-level threads. The number of kernel threads created depends upon a particular application. The developer can create as many threads at both levels but may not be the same. The many to many model is a compromise between the other two models. In this model, if any thread makes a blocking system call, the kernel can schedule another thread for execution



Many-to-Many model

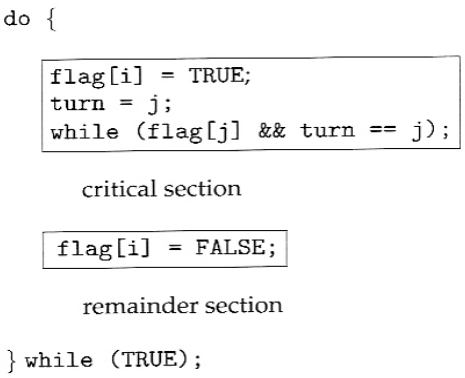
9. Write about Peterson's solution.

Peterson's solution is a widely used solution to critical section problems. This algorithm was developed by a computer scientist Peterson that's why it is named as Peterson's solution.

In this solution, when a process is executing in a critical state, then the other process only executes the rest of the code, and the opposite can happen. This method also helps to make sure that only a single process runs in the critical section at a specific time.

Peterson's algorithm (or Peterson's solution) is a concurrent programming algorithm for mutual exclusion that allows two or more processes to share a single-use resource without conflict, using only shared memory for communication.

REFER– Peterson's Problem



10. Distinguish between monitor and semaphore. Explain in detail a monitor with notify and broadcast functions using an example.

Some of the main differences between semaphore and monitor are as follows:

Features	Semaphore	Monitor
Definition	A semaphore is an integer variable that allows many processes in a parallel system to manage access to a common resource like a multitasking OS.	It is a synchronisation process that enables threads to have mutual exclusion and the wait() for a given condition to become true.

Syntax	`// Wait Operation`
	`wait(Semaphore S) {`
	`while (S<=0);`
	`S--;`
	`}`
	`// Signal Operation`
	`signal(Semaphore S) {`
	`S++;`
	`}`
		`monitor {`
		`//shared variable declarations`
		`data variables;`
		`Procedure P1() { ... }`
		`Procedure P2() { ... }`
		`.`
		`.`
		`.`
		`Procedure Pn() { ... }`
		`}`

Basic	Integer variable	Abstract data type
Access	When a process uses shared resources, it calls the wait() method on S, and when it releases them, it uses the signal() method on S.	When a process uses shared resources in the monitor, it has to access them via procedures.
Action	The semaphore's value shows the number of shared resources available in the system.	The Monitor type includes shared variables as well as a set of procedures that operate on them.
Condition Variable	No condition variables.	It has condition variables.

notify():

The notify() method chooses one thread that is waiting on the monitor held by the current thread and wakes it up. Typically, the waiting thread will grab the monitor and proceed.

Example:

```
public synchronized int get() {
    while (available == false) {
        try {
```

```
            wait();
        }
        catch (InterruptedException e) {
        }
    }
    available = false;
    notify();
    return contents;
}
```

broadcast()

The broadcast operation wakes up every thread waiting for a particular resource. This generally makes sense only with sharable resources. Perhaps a writer just completed so all of the readers can be awakened.

Example:

```
void broadcast (condition *c)
  {
    thread_id tid;
    mutex_acquire (c->listLock);
    while (&c->next){
      tid = dequeue(&c->next, &c->prev);
      thr_continue (tid);
    }
   mutex_release (c->listLock); /* done with the queue */
  }
```

PART - B

1. Describe process scheduling. Explain the various levels of 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 system. 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.

Various levels of process scheduling are:

1) **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.

2) **Short term scheduler**

   It is also called a 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. The 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.

3) **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 process cannot make any progress towards completion.

2. Explain the process state transition diagram with examples.

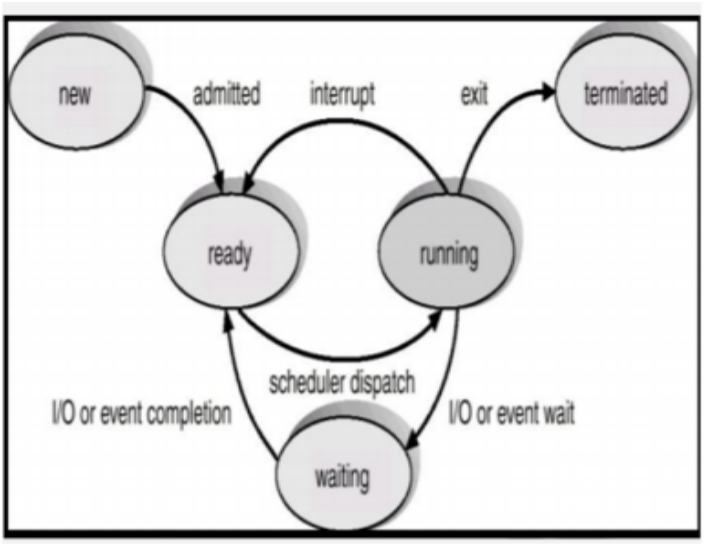


Figure 2.1: Process state diagram

1. Whenever a new process is created, it is admitted into ready state.

2. If no other process is present at running state, it is dispatched to running based on scheduler dispatcher.

3. If any higher priority process is ready, the uncompleted process will be sent to the waiting state from the running state.

4. Whenever I/O or event is completed the process will be sent back to ready state based on the interrupt signal given by the running state.

5. Whenever the execution of a process is completed in running state, it will exit to terminate state, which is the completion of process.

3. Explain handling pruning in detail with examples.

A handle is a substring that connects a right-hand side of the production rule in the grammar. The process of discovering a handle & reducing it to the appropriate left-hand side is called handle pruning. Handle pruning

forms the basis for a bottom-up parsing method to construct a rightmost derivation.

Left sentential and right sentential form:

- A left-sentential form is a sentential form that occurs in the leftmost derivation of some sentence.
- A right-sentential form is a sentential form that occurs in the rightmost derivation of some sentence.

Handle contains two things:

- Production
- Position

Removing the children of the left-hand side non-terminal from the parse tree is called Handle Pruning. A rightmost derivation in reverse can be obtained by handling pruning.

Example:

Right Sequential Form	Handle	Reducing Production
id + id * id	id	E ⇒ id
E + id * id	id	E ⇒ id
E + E * id	id	E ⇒ id
E + E * E	E + E	E ⇒ E + E
E * E	E * E	E ⇒ E * E
E (Root)		

4. What do you mean by PCB? Where is it used? What are its contents? Explain.

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.

PCB must be kept in an area of memory protected from normal process access. In some operating systems the PCB is placed at the bottom of the process stack.

- CPU registers Accumulators, index registers, stack pointer, general purpose registers

- CPU scheduling information Process priority, scheduling parameters

- Memory-management information Value of base and limit registers

- Accounting information Amount of CPU used, time limits, process numbers

- I/O status information List of I/O devices allocated to the process, list of open files and so on.

5. Explain direct and indirect communication of messages passing systems.

Direct communication:

With direct communication each process that requires communication must explicitly name the recipient or sender of the communication. The send and receive primitives are

- Send (P,message) - Send a message to process P.
- Receive (Q,message) - Receive a message from process Q.

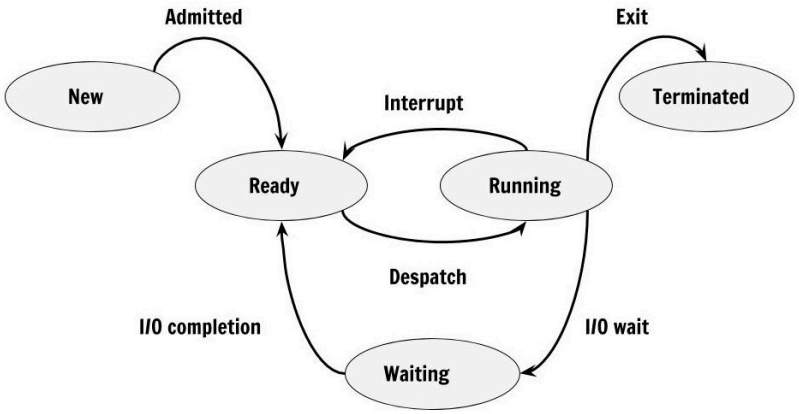
Indirect communication:

With indirect communication, the messages are sent to and received from mailboxes or ports. A mailbox can be viewed abstractly as an object into which messages can be placed by processes and from which messages can be removed. Every mailbox has a unique identification. Two processes can communicate only if they share a mailbox. The primitives are

- Send (A, message) - Send a message to mailbox A.
- Receive (A, message) - Receive a message from mailbox A.

6. What is a process? Draw and explain the process state diagram.

A process is a program in execution. The execution of a process progresses in a sequential fashion. A program is a passive entity while a process is an active entity. A process includes much more than just the program code. A process includes the text section, stack, data section, program counter, register contents and so on.



Process state:

As a process executes, it changes state. The state of a process refers to what the process currently does. A process can be in one of the following states during its lifetime:

New: The process is being created.

Running: Instructions are being executed.

Waiting: The process is waiting for some event to occur or waiting for the I/O operation.
Ready: The process is waiting to be assigned to a processor.
Terminated: The process has finished execution.

7. Distinguish between monitor and semaphore.

Monitor:
Monitors are used for process synchronisation. With the help of programming languages, we can use a monitor to achieve mutual exclusion among the processes. In other words, monitors are defined as the construct of programming language, which helps in controlling shared data access. The Monitor is a module or package which encapsulates shared data structure, procedures, and the synchronisation between the concurrent procedure invocations.
Semaphores:
Semaphores are integer variables that are used to solve the critical section problem by using two atomic operations, wait and signal that are used for process synchronisation. The definitions of wait and signal are as follows –
• Wait :The wait operation decrements the value of its argument S, if it is positive. If S is negative or zero, then no operation is performed.
• Signal: The signal operation increments the value of its argument S.

8. Discuss the attributes of the process. Describe the typical elements of the process control block.

1. Process ID

When a process is created, a unique id is assigned to the process which is used for unique identification of the process in the system.

2. Program counter

A program counter stores the address of the last instruction of the process on which the process was suspended. The CPU uses this address when the execution of this process is resumed.

3. Process State

The Process, from its creation to the completion, goes through various states which are new, ready, running and waiting. We will discuss them later in detail.

4. Priority

Every process has its own priority. The process with the highest priority among the processes gets the CPU first. This is also stored on the process control block.

5. General Purpose Registers

Every process has its own set of registers which are used to hold the data which is generated during the execution of the process.

6. List of open files

During the Execution, Every process uses some files which need to be present in the main memory. OS also maintains a list of open files in the PCB.

7. List of open devices

OS also maintains the list of all open devices which are used during the execution of the process.

Components of Process Control Block(PCB):

- Process Privileges–allowed/disallowed access to system resources
- Process State–new, ready, running, waiting, dead

- Process Number (PID)–unique identification number for each process (also known as Process ID)
- Program Counter (PC)–A pointer to the address of the next instruction to be executed for this process.

9. What is the purpose of the system calls and system programs?

Purpose of System Calls:

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.

Purpose of System Programs:

The system program serves as a part of the operating system. It traditionally lies between the user interface and the system calls. The user view of the system is actually defined by system programs and not system calls because that is what they interact with and system programs are closer to the user interface. System programs as well as application programs form a bridge between the user interface and the system calls.

10. List out the various process states and briefly explain the same with a state diagram.

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 puts all of them in the main memory.

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.

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 behaviour of the process.

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 and the process will be terminated by the Operating system.

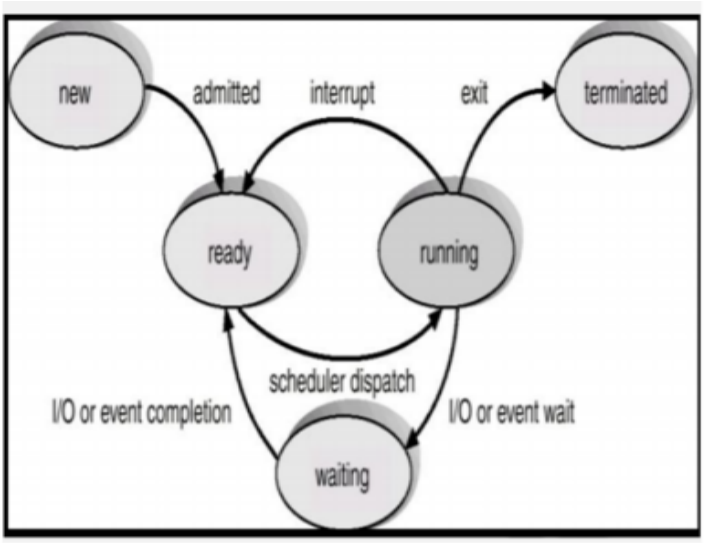
6. Suspend ready

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 suspended ready state.

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.

State Diagram:



11. What is the purpose of a command interpreter? Why is it usually separate from the kernel?

A command interpreter allows the user to interact with a program using commands in the form of text lines. It reads commands from the user or from a file of commands and executes them, usually by turning them into one or more system calls. The main purpose of it is to understand the command inputs and then turn it into system calls. Users can directly enter the command in the form of text lines. It contains code to execute a command or uses a command to read an external file having a code. It is usually not part of the kernel since the command interpreter is subject to changes. The Unix Shell and Windows command prompt are some examples of command interpreters.

12. What are the differences between user level and kernel supported threads?

User-level threads	Kernel-level threads
User-level threads are managed without kernel support by the run-time system.	Kernel-level threads are supported and managed by the operating system.
The scheduler cannot schedule the process properly as the kernel is unaware of user-level threads.	The scheduler handles the process better as the kernel is fully aware of kernel-level threads.
User-level threads are faster to create.	Kernel-level threads are slower to create.
User-level threads are more efficient.	Kernel-level threads are not so efficient.
User-level threads are easy to manage.	Kernel-level threads are not as easy to manage as user-level threads.
The context switching time is less.	The context switching time is more.
User-level threads are generic and can run on any Operating System.	Kernel-level threads are specific to the Operating System.
When one user-level thread performs a blocking operation, the entire process gets blocked.	The kernel can still schedule another thread for execution if one thread is blocked.
User-level threads cannot take full advantage of multiprocessing.	Kernel-level threads take full advantage of multiprocessing.
User-level threads are designed as dependent threads.	Kernel-level threads are designed as independent threads.
Hardware support is needed for context switches.	No hardware support is needed.

13. What is a Scheduler? What is a dispatcher?

Scheduler:

Schedulers in Operating Systems are the process which decides which task and process should be accessed and run at what time by the system resources. It is required to maintain the multi-tasking capabilities of a computer and to keep its performance at the highest level by scheduling the process according to their preferences and needs. The Schedulers in

Operating System are the algorithms which help in the system optimisation for maximum performance.

There are three types of schedulers, they are:

1. Long term scheduler
2. Short term scheduler
3. Middle term scheduler

Dispatcher:
A dispatcher is a special program which comes into play after the scheduler. When the scheduler completes its job of selecting a process, it is the dispatcher which takes that process to the desired state/queue. The dispatcher is the module that gives a process control over the CPU after it has been selected by the short-term scheduler. This function involves the following:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program to restart that program

14. Discuss the following CPU scheduling algorithms

a) Round Robin

b) Shortest job

Round-Robin Scheduling:
Round robin is the oldest, simplest scheduling algorithm. The name of this algorithm comes from the round-robin principle, where each person gets an equal share of something in turn. It is mostly used for scheduling algorithms in multitasking. This algorithm method helps for starvation free execution of processes.

Shortest Job First:

SJF is a full form of (Shortest job first) is a scheduling algorithm in which the process with the shortest execution time should be selected for execution next. This scheduling method can be preemptive or non-preemptive. It significantly reduces the average waiting time for other processes awaiting execution.

15. Discuss the following CPU scheduling algorithms

a) **First come First serve**
b) **Priority**

First Come First Serve:

First Come First Serve is the full form of FCFS. It is the easiest and most simple CPU scheduling algorithm. In this type of algorithm, the process which requests the CPU gets the CPU allocation first. This scheduling method can be managed with a FIFO queue.

As the process enters the ready queue, its PCB (Process Control Block) is linked with the tail of the queue. So, when the CPU becomes free, it should be assigned to the process at the beginning of the queue.

Priority Based Scheduling:

Priority scheduling is a method of scheduling processes based on priority. In this method, the scheduler selects the tasks to work as per the priority.
Priority scheduling also helps OS to involve priority assignments. The processes with higher priority should be carried out first, whereas jobs with equal priorities are carried out on a round-robin or FCFS basis. Priority can be decided based on memory requirements, time requirements, etc.

16. Discuss the following.

a) CPU-i/O burst cycle

b) CPU schedule

c) Preemptive and Nonpreemptive scheduling

d) Dispatcher

a) CPU i/o Burst Cycle

Process execution consists of a cycle of CPU execution and I/O wait. The state of process under execution is called CPU burst and the state of process under I/O request & its handling is called I/O burst. Processes alternate between these two states. Process execution begins with a CPU burst. That is followed by an I/O burst and repeats so on.

b) CPU Schedule

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

c) 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. 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 switches to ready state.

Non-preemptive Scheduling is a CPU scheduling technique where 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.

d) Dispatcher

A dispatcher is a special program which comes into play after the scheduler. When the scheduler completes its job of selecting a process, it is the dispatcher which takes that process to the desired state/queue. The dispatcher is the module that gives a process control over the CPU after it has been selected by the short-term scheduler.

17. Compare between Job-scheduling and CPU-scheduling.

Job Scheduling	CPU Scheduling
The job scheduling is the mechanism to select which process has to be brought into the ready queue.	The CPU scheduling is the mechanism to select which process has to be executed next and allocates the CPU to that process.
The job scheduling is also known as long-term scheduling.	The CPU scheduling is also known as short-term scheduling
The job scheduling is done by the long-term scheduler or the job scheduler.	The CPU scheduling is done by the short-term scheduler or the CPU scheduler
The process transfers from new state to ready state in job scheduling.	The process transfers from ready state to running state in CPU scheduling.
More control over multiprogramming in Job Scheduling.	Less control over multiprogramming in CPU Scheduling.

It regulates the programs which are selected to system for processing.	It ensures which program is suitable or important for processing.
Speed is less than the short-term scheduler.	Speed is very fast as compared to a long-term scheduler.

18. Discuss the attributes of the process. Describe the typical elements of the process control block.

The Attributes of the process are used by the Operating System to create the process control block (PCB) for each of them. This is also called context of the process. Attributes which are stored in the PCB are described below:

1. Process ID

   When a process is created, a unique id is assigned to the process which is used for unique identification of the process in the system.

2. Program counter

   A program counter stores the address of the last instruction of the process on which the process was suspended. The CPU uses this address when the execution of this process is resumed.

3. Process State

   The Process, from its creation to the completion, goes through various states which are new, ready, running and waiting. We will discuss them later in detail.

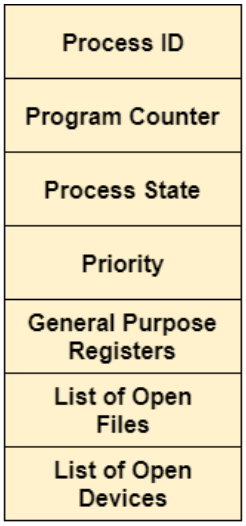
4. Priority

   Every process has its own priority. The process with the highest priority among the processes gets the CPU first. This is also stored on the process control block.

5. General Purpose Registers
   Every process has its own set of registers which are used to hold the data which is generated during the execution of the process.

6. List of open files
   During the Execution, Every process uses some files which need to be present in the main memory. OS also maintains a list of open files in the PCB.

7. List of open devices
   OS also maintains the list of all open devices which are used during the execution of the process.



Process Attributes

19. Define semaphore. Explain the method of applications of semaphore for process synchronisation.

Semaphore:
Semaphore is simply an integer variable that is shared between threads. This variable is used to solve the critical section problem and to achieve process synchronisation in the multiprocessing environment.

Semaphores are of two types:

1. Binary Semaphore –

   This is also known as mutex lock. It can have only two values – 0 and 1. Its value is initialised to 1. It is used to implement the solution of critical section problems with multiple processes.

2. Counting Semaphore –

   Its value can range over an unrestricted domain. It is used to control access to a resource that has multiple instances.

20. Discuss about the following?

a) Process

b) Components of process

c) Program versus process

d) Process states

a) Process

A process is defined as an entity which represents the basic unit of work to be implemented in the system. It is basically a program in execution. The execution of a process must progress in a sequential fashion.

b) Components of Process

S.N.	Component & Description
1	Stack The process Stack contains the temporary data such as method/function parameters, return address and local variables.

2	Heap This is dynamically allocated memory to a process during its run time.
3	Text This includes the current activity represented by the value of Program Counter and the contents of the processor's registers.
4	Data This section contains the global and static variables.

c) Program vs Process

Program	Process
Set of instruction	Process is active.
It is static	It is program in execution
The contents are stored on the Hard Disk	Every process occurs at a different memory location

d) Process States

The process executes when it changes the state. The state of a process is defined by the current activity of the process. Each process may be in any one of the following states –

- New – The process is being created.
- Running – In this state the instructions are being executed.
- Waiting – The process is in waiting state until an event occurs like I/O operation completion or receiving a signal.
- Ready – The process is waiting to be assigned to a processor.
- Terminated – the process has finished execution.

PART - C

1. Define process.

2. Define thread.

3. Describe context switching.

4. What is the information maintained in a PCB?

5. Define the process state and mention the various states of a process.

6. Define CPU scheduling.

7. State critical section problem

8. Distinguish between thread and process.

9. Distinguish between user threads and kernel threads.

10. Define turnaround time.

11. List the various scheduling criteria for CPU scheduling.

12. Describe entry and exit sections of a criteria section.

13. Define semaphores.

14. Explain different ways in which a thread can be cancelled.

15. Write about Scheduling queues.

16. Explain the use of job queues, ready queues and device queues.

17. Explain bounded waiting in critical regions.

18. State the factors on which the performance of the Round Robin CPU scheduling algorithm depends.

19. Distinguish between semaphore and binary semaphore.

20. Distinguish between preemptive and nonpreemptive scheduling techniques.

CIE 2 DONE

OS MODULE 3

PART A

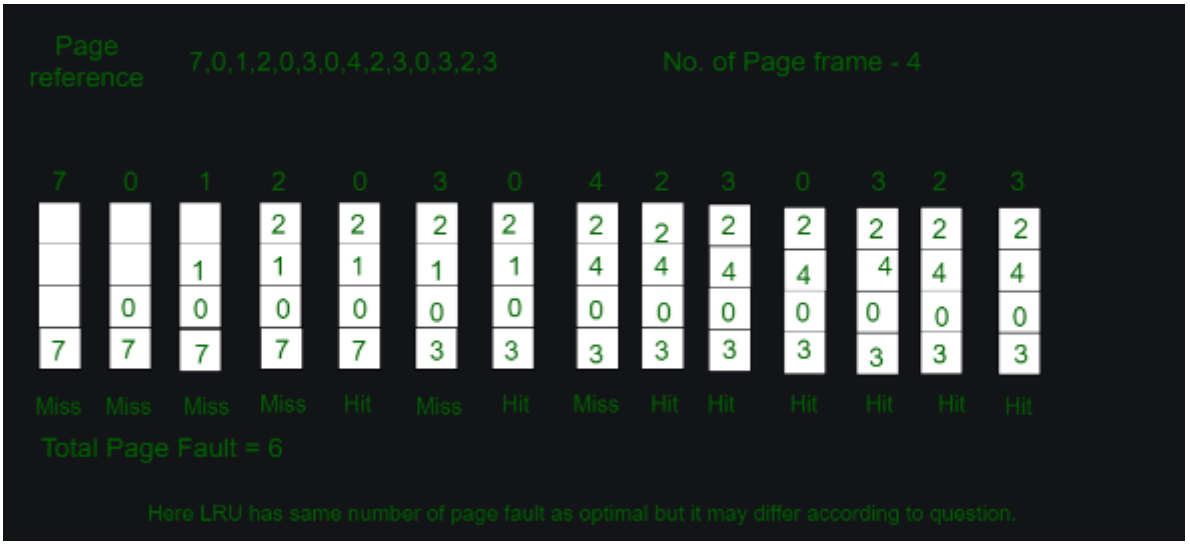
1. Memory partitions of 100kb,500 kb,200 kb,300kb,600 kb are available . How would 7 best ,worst, first fit algorithms place processes 212,417,112,426 in order. Which is the best algorithm?

First-fit:
212K is put in 500K partition
417K is put in 600K partition
112K is put in 288K partition (new partition 288K = 500K - 212K)
426K must wait

Best-fit:
212K is put in 300K partition
417K is put in 500K partition
112K is put in 200K partition
426K is put in 600K partition

Worst-fit:
212K is put in 600K partition
417K is put in 500K partition
112K is put in 388K partition
426K must wait
In this example, best-fit turns out to be the best

2. Describe the LRU page replacement algorithm, assuming there are 3 frames and the page reference string is 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1.Find the number of page faults.



- ➢ Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots → 4 Page faults
- ➢ 0 is already there so → 0 Page fault.
- ➢ when 3 came it will take the place of 7 because it is least recently used →1 Page fault
- ➢ 0 is already in memory so → 0 Page fault.
- ➢ 4 will takes place of 1 → 1 Page Fault
- ➢ Now for the further page reference string → 0 Page fault because they are already available in the memory.

3. Consider the following page reference string 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6 ,3 ,2 ,1 ,2 ,3 ,6 .Find out the number of page faults a) LRU b) FIFO

Number of frames is not given in question

LRU:

```
3 frames:
        1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6
Frame 1: 1        4        5        1        7        2     -
Frame 2:    2        -        6           3        -           -
Frame 3:       3        1        2     -        6        1        6
   15 faults

4 frames:
        1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6
Frame 1: 1              -           -           6                    -
Frame 2:    2        -           -     -              -     -
Frame 3:       3           5              3        -           -
Frame 4:          4           6              7           1
   10 faults
```

FIFO:

```
3 frames:
         1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6
Frame 1: 1        4           6           3        -  2     -     6
Frame 2:    2        -  1        2     -     7           1
Frame 3:       3           5        1           6              3
   16 faults

4 frames:
         1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6
Frame 1: 1              -  5              3        -     1
Frame 2:    2        -        6              7                 3
Frame 3:       3                 2     -        6                    -
Frame 4:          4                 1                 2     -
   14 faults
```

4. The queue of requests in FIFO is 86,147,91,177,94,150,102,175,130 What is the total head movement needed to satisfy the requests for the following Scheduling algorithms FCFS, SJF, SCAN

Draw table and do

(a) FCFS: 565.
$(143 \rightarrow 86 \rightarrow 147 \rightarrow 91 \rightarrow 177 \rightarrow 94 \rightarrow 150 \rightarrow 102 \rightarrow 175 \rightarrow 130)$

(b) SSTF: 162.
$143 \rightarrow 147 \rightarrow 150 \rightarrow 130 irighrarrow 102 \rightarrow 94 \rightarrow 91 \rightarrow 86 \rightarrow 175 \rightarrow 177)$

(c) SCAN: 169.
$143 \rightarrow 147 \rightarrow 150 \rightarrow 175 \rightarrow 177 \rightarrow 199 \rightarrow 130 \rightarrow 102 \rightarrow 94 \rightarrow 91 \rightarrow 86)$

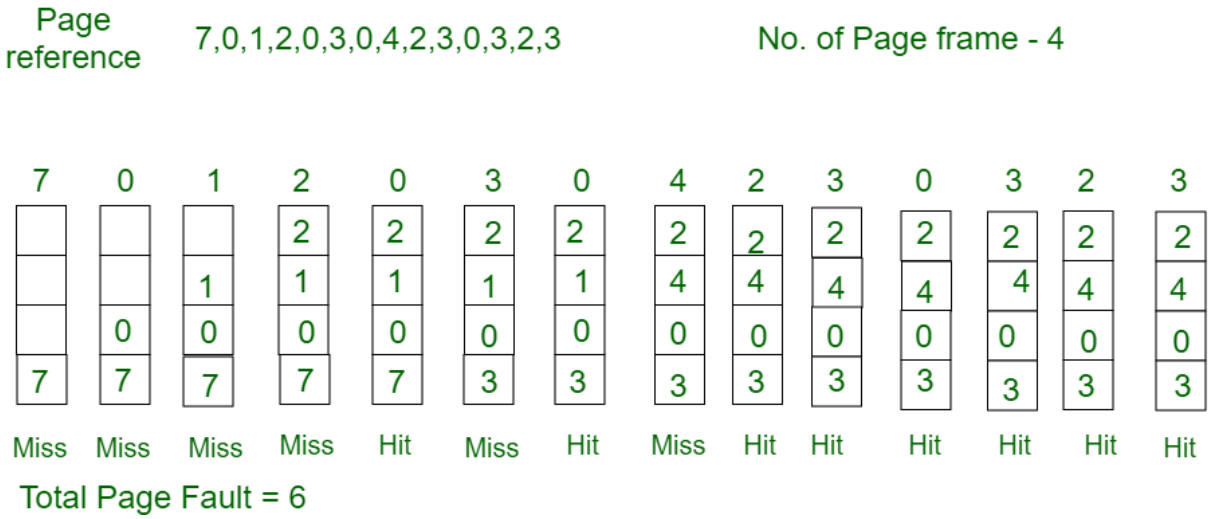
Refer https://brainly.in/question/28658541?msp_srt_exp=4

5. Discuss the following page replacement algorithm with an example i) Optimal ii) LRU

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

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.

Example-2:Consider the page references 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, with 4 page frames. Find number of page faults.



Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots → 4 Page faults

0 is already there so → 0 Page fault.

when 3 comes it will take the place of 7 because it is not used for the longest duration of time in the future.→1 Page fault.

0 is already there so → 0 Page fault..

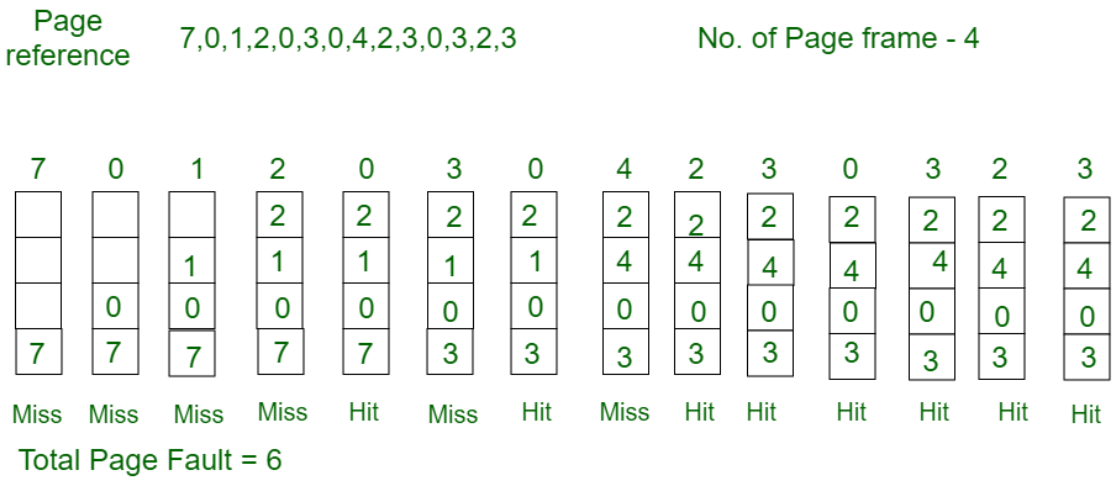
4 will take place of 1 → 1 Page Fault.

Now for the further page reference string → 0 Page fault because they are already available in the memory.

Optimal page replacement is perfect, but not possible in practice as the operating system cannot know future requests. The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analysed against it.

Least recently used (LRU) page replacement algorithm → 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 the future.

In this algorithm, a page will be replaced which is least recently used.
Example-3Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2 with 4 page frames.Find number of page faults.



Here LRU has same number of page fault as optimal but it may differ according to question.

Initially, all slots are empty, so when 7 0 1 2 are allocated to the empty slots → 4 Page faults

0 is already there so → 0 Page fault.

when 3 came it will take the place of 7 because it is least recently used →1 Page fault

0 is already in memory so → 0 Page fault.

4 will takes place of 1 → 1 Page Fault

Now for the further page reference string → 0 Page fault because they are already available in the memory.

6. A virtual memory system has the following specification: Size of the virtual address space=64k Size of the physical address space=4k Page size=512 Virtual page# physical frame# 0 0 3 1 7 2 4 3 10 4 12 5 30 6 31 7 i)find all the virtual addresses that will generate a page fault and compute the main memory addresses for the following virtual addresses. 24, 3784, 10250, 30780

Not getting exact answer, but refer here

In a virtual memory system, the size of the virtual address is 32-bit, size of physical address is 30-bit, page size is 4 Kbyte and size of each page table entry is 32-bit. The main memory is byte addressable. Which one of the following is the maximum number of bits that can be used for storing protection and other information in each page table entry?

(A) 2

(B) 10

(C) 12

(D) 14

Answer: (D)

Explanation:

Virtual memory = 2^{32} bytes

Physical memory = 2^{30} bytes

Page size = Frame size = $4 * 10^3$ bytes = $22 * 2^{10}$ bytes = 2^{12} bytes

Number of frames = Physical memory / Frame size = $2^{30}/2^{12} = 2^{18}$

Therefore, Numbers of bits for frame = 18 bits

Page Table Entry Size = Number of bits for frame + Other information

Other information = 32 – 18 = 14 bits

Thus, option (D) is correct.

7 A process references 5 pages A,B, C, D, E in the following order A, B, C, D, A, E, B, C, E, D Assuming that the replacement algorithm is LRU and FIFO, find out the number of page faults during the sequence of references, starting with an empty main memory With 3 frames.

LRU **(Under Review)**

		C	C	C	E	E	E	E
	B	B	B	A	A	A	C	C
A	A	A	D	D	D	B	B	D
Hit	Hit	Hit	Hit	Hit	Hit	Hit	Miss	Hit

Page Faults = 8

FIFO **(Under Review)**

Same as LRU in this case

Refer-

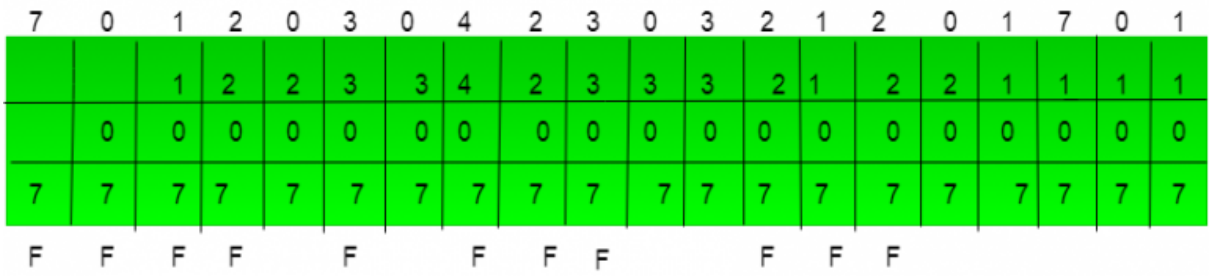
https://compsciedu.com/Operating-System/Linux-System/discussion/19796

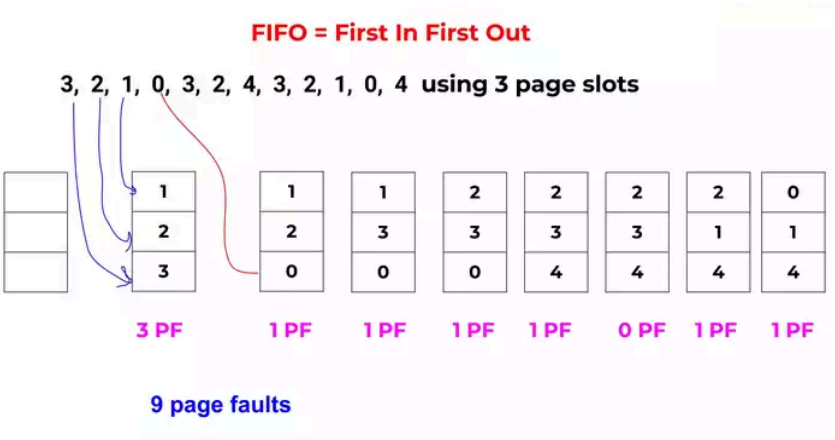
8. Consider the following page reference string 7,0, 1,2,0,3,0,4,2,3,0,3,2, 1,2,0, 1, 7, 10, 0, 1. How many page faults would occur for FIFO page replacement algorithms, assuming three frames?

FIFO → in this algorithm, a queue is maintained. The page which is assigned the frame first will be replaced first. In other words, the page which resides at the rare end of the queue will be replaced on every page fault.

If u take this ref string **7,0, 1,2,0,3,0,4,2,3,0,3,2, 1,2,0, 1, 7, 10 0, 1 and 3 frames there will be 16 misses and 5 hits**

In LIFO algorithm 11 page fault will occur -Its an example for





9. Given memory partitions of 100 K, 500 K, 200 K, 300 K and 600 K (in order) how Would each of the first fit, best fit and worst fit algorithms work with 212 K, 417K, 112 K and 426 K (in order)? Which algorithm makes the most efficient use of memory?

Solution:

First-Fit:
212K is put in 500K partition.
417K is put in 600K partition.
112K is put in 288K partition (new partition 288K = 500K - 212K).
426K must wait.

Best-Fit:
212K is put in 300K partition.
417K is put in 500K partition.
112K is put in 200K partition.
426K is put in 600K partition.

Worst-Fit:
212K is put in 600K partition.
417K is put in 500K partition.
112K is put in 388K partition.
426K must wait.

In this example, Best-Fit turns out to be the best.

10 Consider a logical address space of eight pages of 1024 words each mapped onto a physical memory of 32 frames a) How many bits are in the logical address. b) How many bits are in the physical address

Logical address space has 8 pages, size of each page, offset = 1024 words
Number if bits in page# field of the logical address = log2 8 bits = 3 bits
Offset bits = log2 1024 = 10 bits
So, Logical address = 3 + 10 = **13 bits**
Physical memory has 32 frames, offset = 1024 words
Number of bits in frame# field of the Physical address = log2 32 bits = 5 bits
Physical address = 5 + 10 = **15 bits**

PART B

1 Describe the following. a) Virtual Memory b) Cache Memory

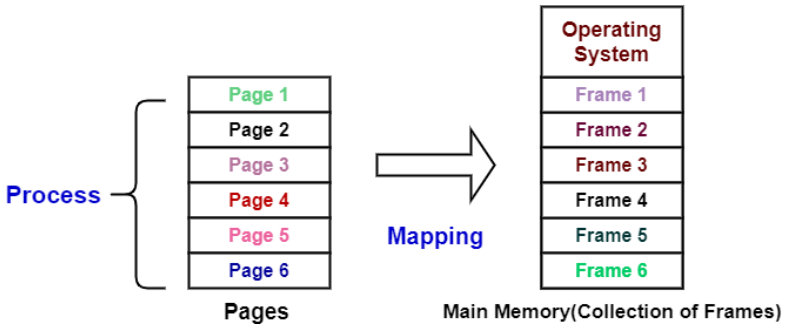
- Virtual Memory is a storage allocation scheme in which secondary memory can be addressed as though it were part of the main memory. The addresses a program may use to reference memory are distinguished from the addresses the memory system uses to identify physical storage sites, and program-generated addresses are translated automatically to the corresponding machine addresses.
- The size of virtual storage is limited by the addressing scheme of the computer system and the amount of secondary memory is available not by the actual number of the main storage locations.
- It is a technique that is implemented using both hardware and software.

Cache Memory

- Cache Memory is a special very high-speed memory. It is used to speed up and synchronise with high-speed CPUs. Cache memory is costlier than main memory or disk memory but economical than CPU registers. Cache memory is an extremely fast memory type that acts as a buffer between RAM and the CPU. It holds frequently requested data and instructions so that they are immediately available to the CPU when needed.
- Cache memory is used to reduce the average time to access data from the Main memory. The cache is a smaller and faster memory which stores copies of the data from frequently used main memory locations. There are various different independent caches in a CPU, which store instructions and data.

2. What is paging and swapping?

Paging is a storage mechanism that allows the OS to retrieve processes from the secondary storage into the main memory in the form of pages. In the Paging method, the main memory is divided into small fixed-size blocks of physical memory, which is called frames. The size of a frame should be kept the same as that of a page to have maximum utilisation of the main memory and to avoid external fragmentation. Paging is used for faster access to data, and it is a logical concept.

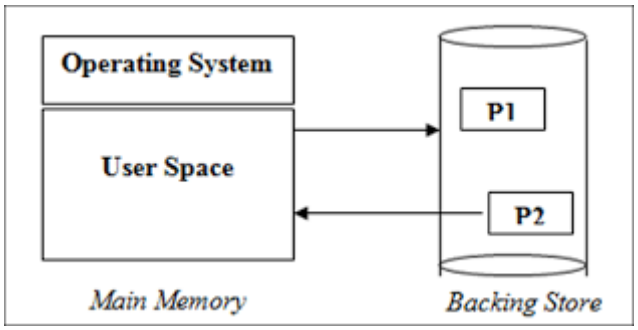


Swapping

- Swapping is a memory management scheme in which any process can be temporarily swapped from main memory to secondary

memory so that the main memory can be made available for other processes. It is used to improve main memory utilisation. In secondary memory, the place where the swapped-out process is stored is called swap space.

- The purpose of the swapping in the operating system is to access the data present in the hard disk and bring it to RAM so that the application programs can use it. The thing to remember is that swapping is used only when data is not present in RAM.



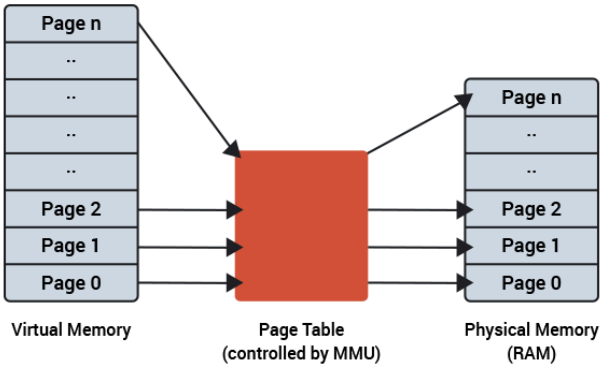
3. With a diagram, discuss the steps involved in handling a page fault.

A Page Fault happens when you access a page that has been marked as invalid. The paging hardware would notice that the invalid bit is set while translating the address across the page table, which will cause an operating system trap. The trap is caused primarily by the OS's failure to load the needed page into memory.

Now, let's understand the procedure of page fault handling in the OS:

1. Firstly, an internal table for this process to assess whether the reference was valid or invalid memory access.
2. If the reference becomes invalid, the system process would be terminated. Otherwise, the page will be paged in.
3. After that, the free-frame list finds the free frame in the system.

4. Now, the disk operation would be scheduled to get the required page from the disk.
5. When the I/O operation is completed, the process's page table will be updated with a new frame number, and the invalid bit will be changed. Now, it is a valid page reference.
6. If any page fault is found, restart these steps from starting.



4. Describe a) Paging b) Page table structure

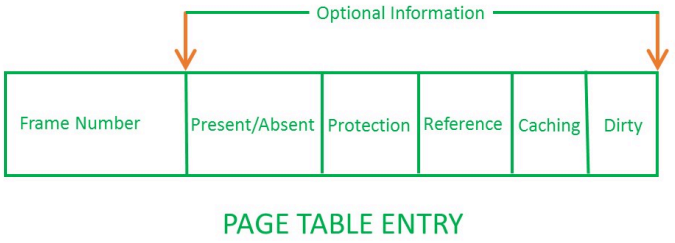
Paging

Paging is a storage mechanism that allows the OS to retrieve processes from the secondary storage into the main memory in the form of pages. In the Paging method, the main memory is divided into small fixed-size blocks of physical memory, which is called frames. The size of a frame should be kept the same as that of a page to have maximum utilisation of the main memory and to avoid external fragmentation. Paging is used for faster access to data, and it is a logical concept.

Page Table

- Page Table is a data structure used by the virtual memory system to store the mapping between logical addresses and physical addresses.
- Logical addresses are generated by the CPU for the pages of the processes therefore they are generally used by the processes.

- Physical addresses are the actual frame address of the memory. They are generally used by the hardware or more specifically by RAM subsystems.



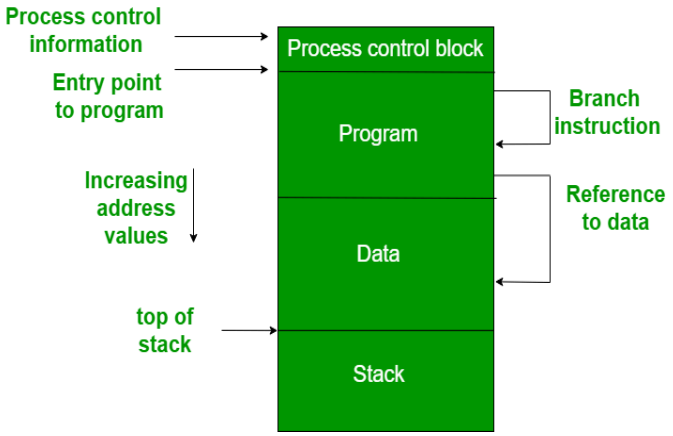
PAGE TABLE ENTRY

5. Explain in detail the requirements that memory management technique needs to satisfy.

Memory management keeps track of the status of each memory location, whether it is allocated or free. It allocates the memory dynamically to the programs at their request and frees it for reuse when it is no longer needed. Memory management is meant to satisfy some requirements that we should keep in mind.

These Requirements of memory management are:

1. **Relocation** – The available memory is generally shared among a number of processes in a multiprogramming system, so it is not possible to know in advance which other programs will be resident in main memory at the time of execution of this program. Swapping the active processes in and out of the main memory enables the operating system to have a larger pool of ready-to-execute processes.



2. **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 another process tries to write in a process whether accidental or incidental.
3. **Sharing** – A protection mechanism must have to allow several processes to access the same portion of main memory. Allowing each process access to the same copy of the program rather than have their own separate copy has an advantage.
4. **Logical organisation** – Main memory is organised as linear or it can be a one-dimensional address space which consists of a sequence of bytes or words. Most of the programs can be organised into modules, some of those are unmodifiable (read-only, execute only) and some of those contain data that can be modified.
5. **Physical organisation** – The structure of computer memory has two levels referred to as main memory and secondary memory. Main memory is relatively very fast and costly as compared to secondary memory. Main memory is volatile. Thus secondary memory is provided for storage of data on a long-term basis while the main memory holds currently used programs. The major system concern between main memory and secondary memory is the flow of information and it is impractical for programmers to understand

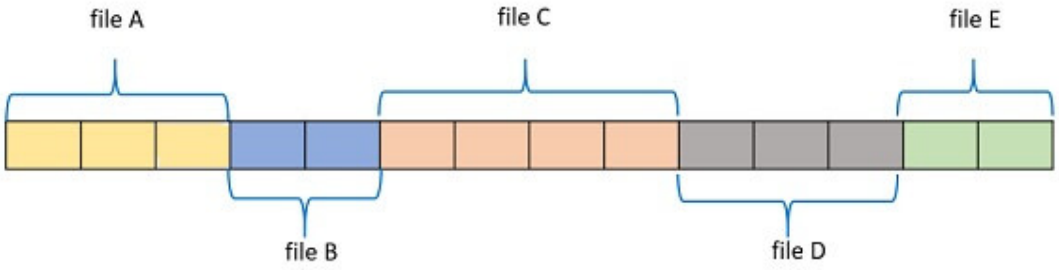
6 Describe a) Translation look-aside buffer b) Segmentation

For a) Translation look-aside buffer refer 15q part B

For b) Segmentation refer 18q part B

7. Describe contiguous memory allocation concept with advantages and disadvantages.

Contiguous memory allocation is a memory allocation method that allocates a single contiguous section of memory to a process or a file.
Taking into account the future growth of the file and its request for memory, the operating system allocates sufficient contiguous memory blocks to that file. Considering this future expansion and the file's request for memory, the operating system will allocate those many contiguous blocks of memory to that file.



- In Contiguous memory allocation, when the process arrives from the ready queue to the main memory for execution, the contiguous memory blocks are allocated to the process according to its requirement. Now, to allocate the contiguous space to user processes, the memory can be divided either in the fixed-sized partition or in the variable-sized partition.

Advantages and Disadvantages

- The main disadvantage of contiguous memory allocation is memory wastage and inflexibility. As the memory is allocated to a file or a process keeping in mind that it will grow during the run. But until a process or a file grows, many blocks allocated to it remain unutilized.

And they even cannot be allocated to the other process leading to wastage of memory.

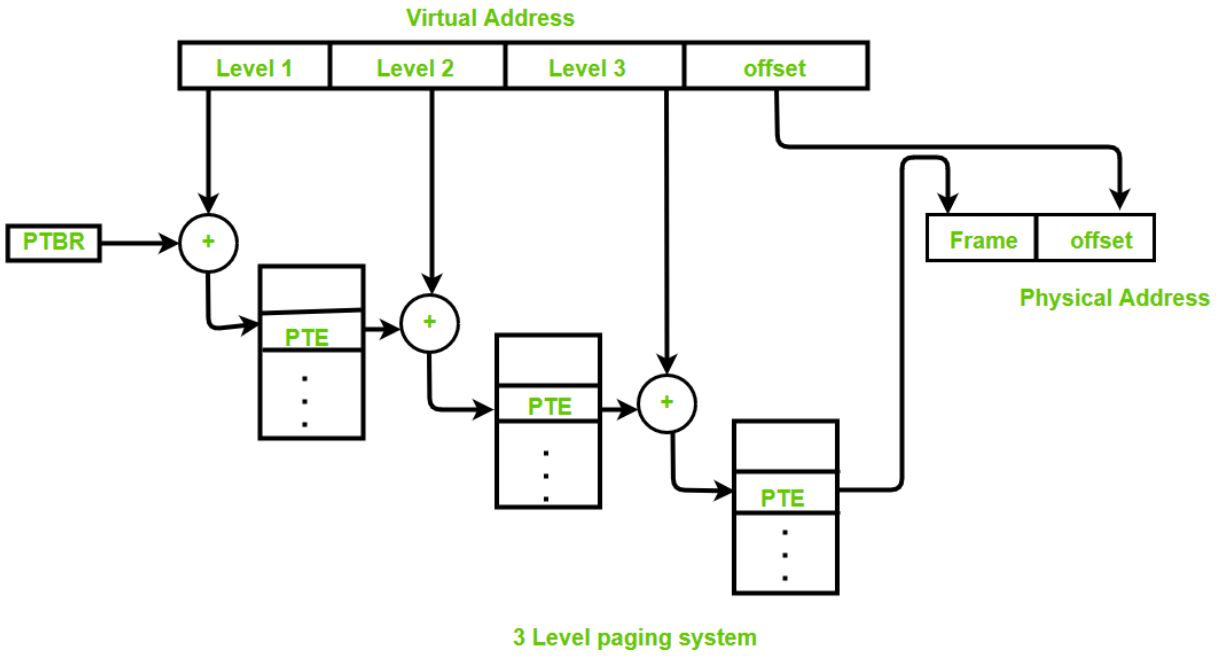
- The advantage of contiguous memory allocation is it increases the processing speed. As the operating system uses the buffered I/O and reads the process memory blocks consecutively it reduces the head movements. This speed ups the processing.

8. Describe Hierarchical paging .

Multilevel Paging is a paging scheme that consists of two or more levels of page tables in a hierarchical manner. It is also known as hierarchical paging. The entries of the level 1 page table are pointers to a level 2 page table and entries of the level 2 page tables are pointers to a level 3 page table and so on. The entries of the last level page table store actual frame information. Level 1 contains a single-page table and the address of that table is stored in PTBR (Page Table Base Register).
If the frame size of the main memory is smaller than the page size and the process cannot fit that way then we divide the pages into further pages, and this concept is known as multilevel paging.

Virtual address:

In multilevel paging, whatever may be levels of paging, all the page tables will be stored in the main memory. So it requires more than one memory access to get the physical address of the page frame. One access for each level is needed. Each page table entry except the last level page table entry contains the base address of the next level page table.



3 Level paging system

9. Describe Inverted page Table

Inverted Page Table is the global page table which is maintained by the Operating System for all the processes. In an inverted page table, the number of entries is equal to the number of frames in the main memory. It can be used to overcome the drawbacks of page tables.

There is always a space reserved for the page regardless of the fact that whether it is present in the main memory or not. However, this is simply the wastage of the memory if the page is not present.

Pages	Frames
0	X
1	X
2	F1
3	F3
4	F6
5	X
6	F5

Page Table of P1

Pages	Frames
0	F2
1	F4
2	F7
3	X
4	X
5	X
6	F0

Page Table of P2

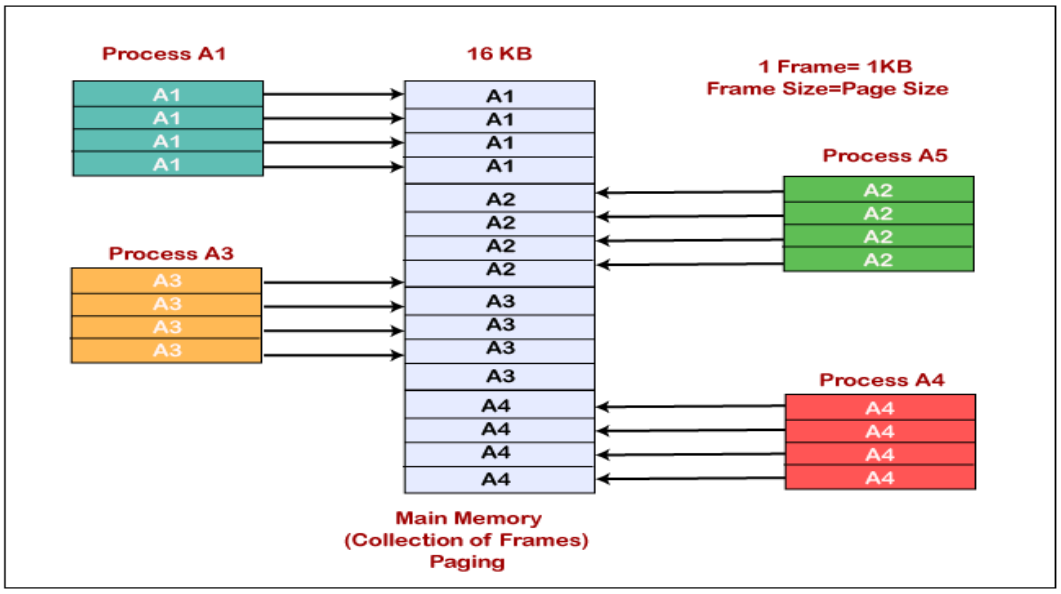
We can save this wastage by just inverting the page table. We can save the details only for the pages which are present in the main memory. Frames are the indices and the information saved inside the block will be Process ID and page number.

Pages	Frames
0	OS
1	P1 p2
2	P2 p0
3	P1 p3
4	P2 p1
5	P1 p6
6	P1 p4
7	P2 p2

Inverted Page Table

10 Explain briefly about paging with a neat diagram.

Paging is a storage mechanism that allows OS to retrieve processes from the secondary storage into the main memory in the form of pages. In the Paging method, the main memory is divided into small fixed-size blocks of physical memory, which is called frames. The size of a frame should be kept the same as that of a page to have maximum utilization of the main memory and to avoid external fragmentation. Paging is used for faster access to data, and it is a logical concept.



11 Explain the uses of the following: a. Mutex object b. Semaphore object

Mutex

- Mutex or Mutual Exclusion Object is used to give access to a resource to only one process at a time.
- The mutex object allows all the processes to use the same resource but at a time, only one process is allowed to use the resource.
- Mutex uses the lock-based technique to handle the critical section problem.
- Whenever a process requests for a resource from the system, then the system will create a mutex object with a unique name or ID. So, whenever the process wants to use that resource, then the process occupies a lock on the object.
- After locking, the process uses the resource and finally releases the mutex object. After that, other processes can create the mutex object in the same manner and use it.

Use of Mutex

A mutex provides mutual exclusion, which can be either producer or consumer that can have the key (mutex) and proceed with their work. As

long as producer fills the buffer, the user needs to wait, and vice versa. In Mutex lock, all the time, only a single thread can work with the entire buffer.

Semaphore

- Semaphore is an integer variable S, that is initialized with the number of resources present in the system and is used for process synchronization.
- It uses two functions to change the value of S i.e. wait() and signal().
- Both these functions are used to modify the value of semaphore but the functions allow only one process to change the value at a particular time i.e. no two processes can change the value of semaphore simultaneously.

Use of Semaphore

In the case of a single buffer, we can separate the 4 KB buffer into four 1 KB buffers. Semaphore can be associated with these four buffers. This allows users and producers to work on different buffers at the same time.

12. Define page fault. When does a page fault occur?

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.

A page fault trap occurs if the requested page is not loaded into memory. The page fault primarily causes an exception, which is used to notify the operating system to retrieve the "pages" from virtual memory to continue operation. Once all of the data has been placed into physical memory, the program resumes normal operation. The Page fault process occurs in the background, and thus the user is unaware of it.

The computer's hardware track to the kernel and the program counter is often saved on the stack. The CPU registers hold information about the current state of instruction.
An assembly program is started, which saves the general registers and other volatile data to prevent the Operating system from destroying it.

13 Describe the action taken by the OS when page fault occurs.

Page Fault Handling

- A Page Fault happens when you access a page that has been marked as invalid. The paging hardware would notice that the invalid bit is set while translating the address across the page table, which will cause an operating system trap. The trap is caused primarily by the OS's failure to load the needed page into memory.
- Firstly, an internal table for this process to assess whether the reference was valid or invalid memory access.
- If the reference becomes invalid, the system process would be terminated. Otherwise, the page will be paged in.
- After that, the free-frame list finds the free frame in the system.
- Now, the disk operation would be scheduled to get the required page from the disk.
- When the I/O operation is completed, the process's page table will be updated with a new frame number, and the invalid bit will be changed. Now, it is a valid page reference.
- If any page fault is found, restart these steps from starting.

14. Write a note on file types and file structures

File Type
File type refers to the ability of the operating system to distinguish different types of file such as text files, source files and binary files etc. Many

operating systems support many types of files. Operating system like MS-DOS and UNIX have the following types of files –

Ordinary files

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

Directory files

- These files contain a list of file names and other information related to these files.
- Special files these files are also known as device files.
- These files represent physical devices like disks, terminals, printers, networks, tape drive etc.
- A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the files creator and user.

File Structure

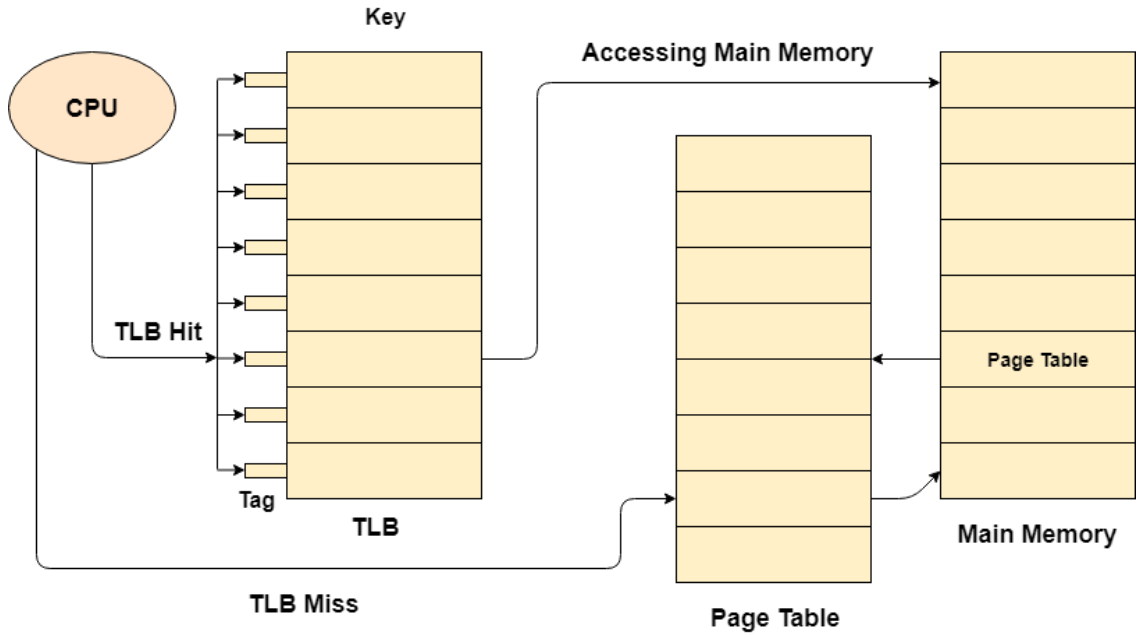
- A File Structure should be according to a required format that the operating system can understand.
- A file has a certain defined structure according to its type.
- A text file is a sequence of characters organized into lines.
- A source file is a sequence of procedures and functions.
- An object file is a sequence of bytes organized into blocks that are understandable by the machine.

When an operating system defines different file structures, it also contains the code to support these file structures. Unix, MS-DOS support a minimum number of file structures.

15. Explain with the help of supporting diagrams how TLB improves the performance of a demand paging system.

A Translation look aside buffer can be defined as a memory cache which can be used to reduce the time taken to access the page table again and again.
It is a memory cache which is closer to the CPU and the time taken by CPU to access TLB is lesser than that taken to access main memory.

In other words, we can say that TLB is faster and smaller than the main memory but cheaper and bigger than the register.
TLB follows the concept of locality of reference which means that it contains only the entries of those many pages that are frequently accessed by the CPU.
OS Translation Lookaside buffer



In translation lookaside buffers, there are tags and keys with the help of which, the mapping is done.

TLB hit is a condition where the desired entry is found in translation Lookaside Buffer. If this happens then the CPU simply accesses the actual location in the main memory.
However, if the entry is not found in TLB (TLB miss) then the CPU has to access the page table in the main memory and then access the actual frame in the main memory.
Therefore, in the case of a TLB hit, the effective access time will be lesser as compared to the case of TLB miss.

If the probability of TLB hit is P% (TLB hit rate) then the probability of TLB miss (TLB miss rate) will be (1-P) %.

16. Differentiate between global and local replacement algorithms

Global page replacement
In global replacement a process can select even those frames which are currently allocated to some other process. Thus, in this strategy, one process can take a frame from another process. In global replacement, the number of frames allocated to a process may increase. One problem with this method is that a process cannot control its own page fault rate. The main advantage is that it results in greater system throughput.

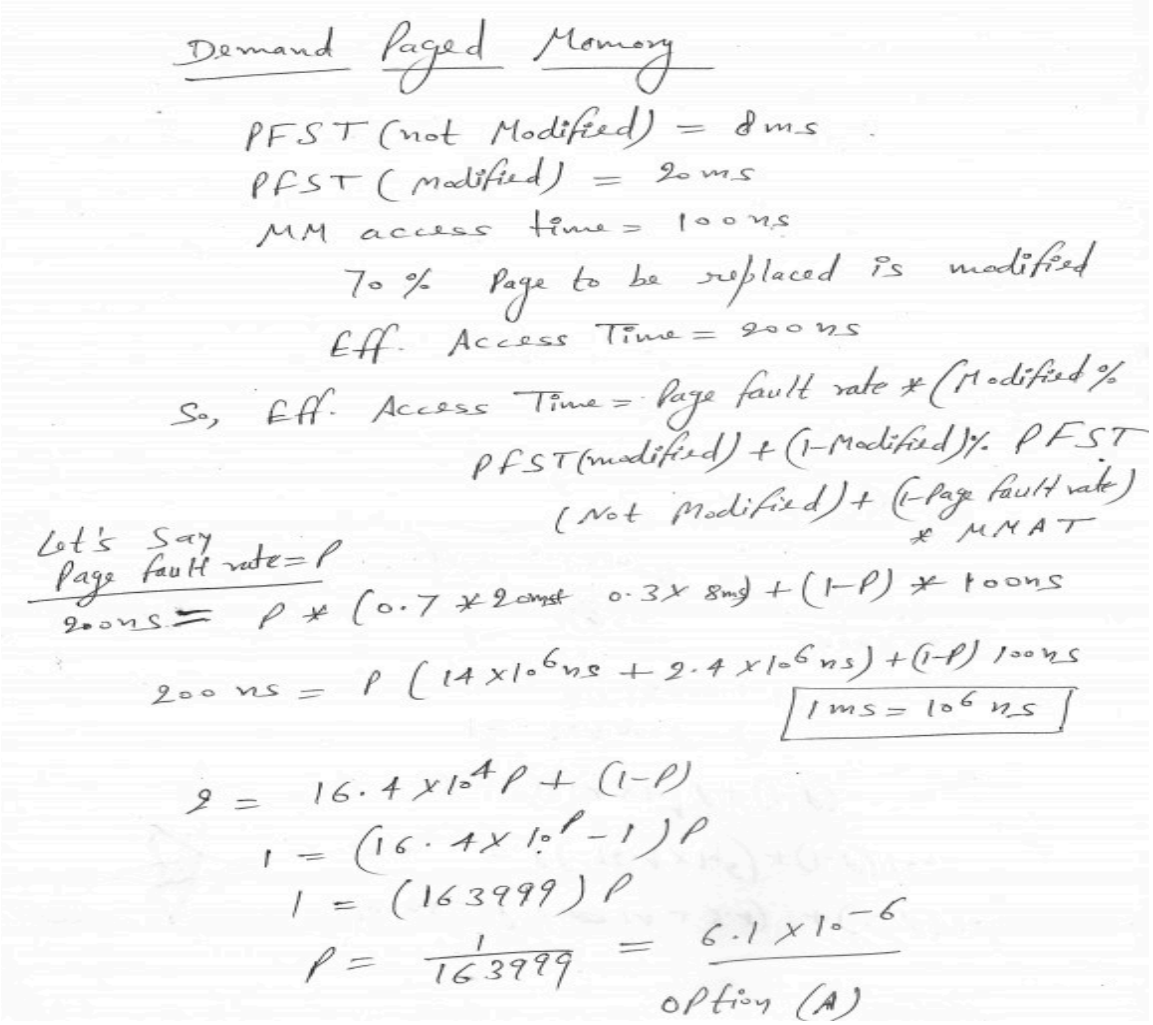
Local page replacement
In this algorithm, the replacement frame selected by a process can only be from the set of frames which are allocated to it. The process cannot select the frame which is currently allocated to another process. Thus, one process cannot take a frame from another process in local replacement. In local replacement, the number of frames allocated to a process does not change.

Global page replacement	Local page replacement
In this algorithm, the replacement frame selected by a process can be any frame from the set of all frames.	In this algorithm, the replacement frame selected by a process can only be from the set of frames which are allocated to it.
A process can select even those frames which are currently allocated to some other process.	A process cannot select the frame which is currently allocated to another process.

17. What is virtual memory? Explain: Suppose we have a demand paged memory. The page table is held in registers. It takes 8ms to service a page fault if an empty page is available or the replaced page is not modified, and 20ms if the replaced page is modified. memory access time is 100ns. Assume that the page to be replaced is modified 70% of the time. What is the maximum acceptable page fault rate for an effective access time of no more than 200ns?

- Virtual memory is a memory management technique where secondary memory can be used as if it were a part of the main memory. Virtual memory is a common technique used in a computer's operating system (OS).
- Virtual memory uses both hardware and software to enable a computer to compensate for physical memory shortages, temporarily transferring data from random access memory (RAM) to disk storage.

Mapping chunks of memory to disk files enables a computer to treat secondary memory as though it were main memory.



18. Explain the basic concepts of segmentation with neat diagrams

Segmentation

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

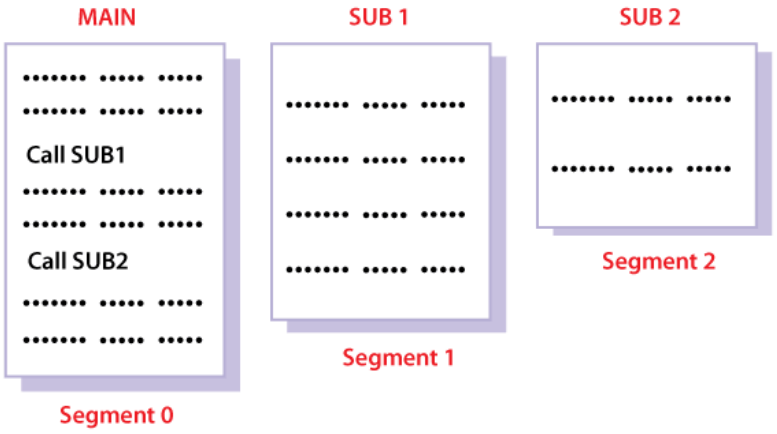
The details about each segment are stored in a table called a segment table. Segment table is stored in one (or many) of the segments.

Segment table contains mainly two information about segment:

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

Till now, we were using Paging as our main memory management technique. Paging is more close to the Operating system rather than the User. It divides all the processes into the form of pages regardless of the fact that a process can have some relative parts of functions which need to be loaded in the same page.
Operating system doesn't care about the User's view of the process. It may divide the same function into different pages and those pages may or may not be loaded at the same time into the memory. It decreases the efficiency of the system.

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



19 Explain the uses of the following: a) Mutex object b) Semaphore object c) Waitable timer object.

Mutex

- Mutex or Mutual Exclusion Object is used to give access to a resource to only one process at a time.
- The mutex object allows all the processes to use the same resource but at a time, only one process is allowed to use the resource.
- Mutex uses the lock-based technique to handle the critical section problem.
- Whenever a process requests for a resource from the system, then the system will create a mutex object with a unique name or ID. So, whenever the process wants to use that resource, then the process occupies a lock on the object.
- After locking, the process uses the resource and finally releases the mutex object. After that, other processes can create the mutex object in the same manner and use it.

Semaphore

- Semaphore is an integer variable S, that is initialized with the number of resources present in the system and is used for process synchronization.
- It uses two functions to change the value of S i.e. wait() and signal().

- Both these functions are used to modify the value of semaphore but the functions allow only one process to change the value at a particular time i.e. no two processes can change the value of semaphore simultaneously.

A waitable timer object is a synchronization object whose state is set to signaled when the specified due time arrives. There are two types of waitable timers that can be created: manual-reset and synchronization. A timer of either type can also be a periodic timer.

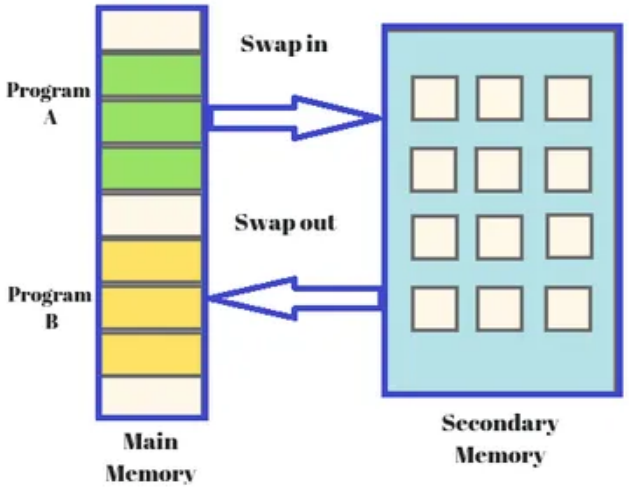
A thread uses the CreateWaitableTimer or CreateWaitableTimerEx function to create a timer object. The creating thread specifies whether the timer is a manual-reset timer or a synchronization timer. The creating thread can specify a name for the timer object. Threads in other processes can open a handle to an existing timer by specifying its name in a call to the OpenWaitableTimer function.

20 Explain briefly the performance of demand paging with necessary examples

Demand Paging

According to the concept of Virtual Memory, in order to execute some process, only a part of the process needs to be present in the main memory which means that only a few pages will only be present in the main memory at any time.

However, deciding which pages need to be kept in the main memory and which need to be kept in the secondary memory, is going to be difficult because we cannot say in advance that a process will require a particular page at a particular time.



Therefore, to overcome this problem, a concept called Demand Paging is introduced. It suggests keeping all pages of the frames in the secondary memory until they are required. In other words, it says that do not load any page in the main memory until it is required.

Whenever any page is referred for the first time in the main memory, then that page will be found in the secondary memory.

IARE '24 DISCORD TEAM

II-II

OS MODULE 4 SOLUTIONS

AD • UJJWAL • ASRITHA • PRANAV

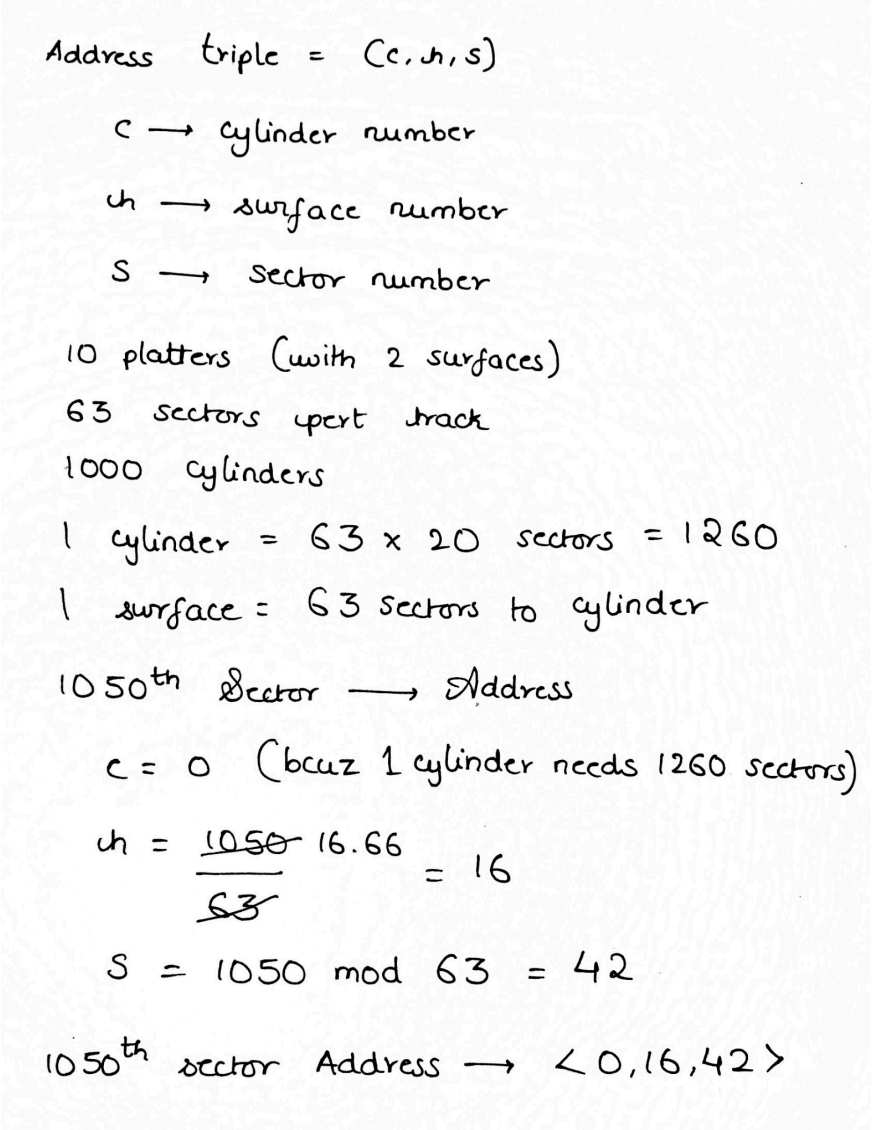
FILE SYSTEM INTERFACE, MASS-STORAGE STRUCTURE

Few Pending

OS MODULE 4

PART A

1. A hard disk has 63 sectors per track, 10 platters each with 2 recording surfaces and 1000 cylinders. The address of a sector is given as a triple ⟨c,h,s⟩, where c is the cylinder number, h is the surface number and s is the sector number. Thus, the 0th sector is addresses as ⟨0,0,0⟩, the 1st sector as ⟨0,0,1⟩, and so on. The address of the 1050th sector is



2. Explain the maximum file size supported by a file system with 16 direct blocks, single, double, and triple indirection. The block size is 512 bytes. Disk block numbers can be stored in 4 bytes.

We have, block size = 512
number of block numbers in an indirection block
= block size / 4
= 128
number of blocks for file data in that file object
$= 16 + 128 + 128^2 + 128^3$
Maximum file size:
(direct + single indirect + double indirect + triple indirect) * (blocksize)
$= (16 + 512/4 + (512/4)^2 + (512/4)^3) * (512)$
= 68853964800 bytes, ~64 GB

3. Discuss the reasons why the operating system might require accurate information on how blocks are stored on disk. how could operating system improves file system performance with this knowledge

While allocating blocks for a file, the operating system could allocate blocks that are geometrically close by on the disk if it had more information regarding the physical location of the blocks on the disk. In particular, it could allocate a block of data and then allocate the second block of data in the same cylinder but on a different surface at a rotationally optimal place so that the access to the next block could be made with minimal cost.

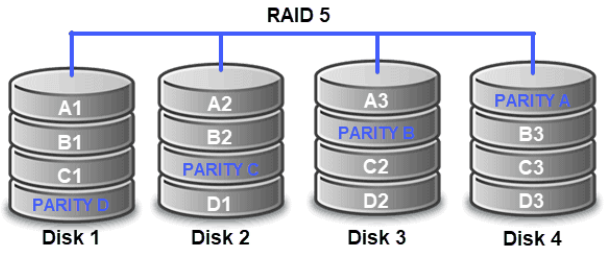
4. Discuss how the OS could maintain a free-space list for a tape-resident file system. Assume that the tape technology is append-only and that it uses EOT marks and locate, space and read position command

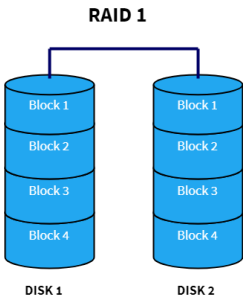
- Since this tape technology is append-only, all the free space is at the end of the tape.

- The location of this free space does not need to be stored at all, because the space command can be used to position to the EOT mark.
- The amount of available free space after the EOT mark can be represented by a single number.
- It may be desirable to maintain a second number to represent the amount of space occupied by files that have been logically deleted (but their space has not been reclaimed since the tape is append-only) so that we can decide when it would pay to copy the non-deleted files to a new tape in order to reclaim the old tape for reuse.
- We can store the free and deleted space numbers on disk for easy access. Another copy of these numbers can be stored at the end of the tape as the last data block.
- We can overwrite this last data block when we allocate new storage on the tape.

5. Compare the performance of write operations achieved by a RAID level 5 organization with that achieved by a RAID level 1 organization.

RAID Level 1 organization can perform writes by simply issuing the writes to mirrored data concurrently. RAID Level 5, on the other hand, would require the old contents of the parity block to be read before it is updated based on the new contents of the target block. This results in more overhead for the write operations on a RAID Level 5 system.

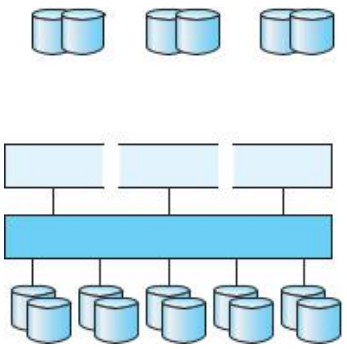




6. Is there any way to implement truly stable storage. Explain your answer.

By definition, information residing in the Stable-Storage is never lost. Even if the disk and CPU have some errors, it will never lose any data.

Stable-Storage Implementation :
To achieve such storage, we need to replicate the required information on multiple storage devices with independent failure modes. The writing of an update should be coordinated in such a way that it would not delete all the copies of the state and that, when we are recovering from a failure, we can force all the copies to a consistent and correct value, even if another failure occurs during the recovery. Truly stable storage would never lose data. The fundamental technique for stable storage is to maintain multiple copies of the data, so that if one copy is destroyed, some other copy is still available for use. But for any scheme, we can imagine a large enough disaster that all copies are destroyed.



Multiple Storages in Stable-Storage Implementation

7. What are file protection methods?

Protection mechanisms provide controlled access by limiting the types of file access that can be made. Access is permitted or denied depending on several factors, one of which is the type of access requested. Several different types of operations may be controlled:

- **Read:** Read from the file.
- **Write:** Write or rewrite the file.
- **Execute:** Load the file into memory and execute it.
- **Append:** Write new information at the end of the file.
- **Delete:** Delete the file and free its space for possible reuse.
- **List:** List the name and attributes of the file.

Other Protection Approaches:
The access to any system is also controlled by the password. If the use of password is random and it is changed often, this may result in limiting the effective access to a file.

The use of passwords has a few disadvantages:

- The number of passwords is very large so it is difficult to remember the large passwords.
- If one password is used for all the files, then once it is discovered, all files are accessible; protection is on all-or-none basis.

8. Explain different types of files.
The types of files recognized by the system are either regular, directory, or special. However, the operating system uses many variations of these basic types.
The following basic types of files exist:

Item	Description
regular	Stores data (text, binary, and executable)
directory	Contains information used to access other files
special	Defines a FIFO (first-in, first-out) pipe file or a physical device

BIOS	Contains information related to Basic Input Output System

Regular file: It stores data of text, binary information these are the most common files. It is also known as ordinary files.

This is of two types:

(i) Text file: It contains information that is readable by the user. We can display and print these files.

(ii) Binary files: It contains information that is readable by the computer. These files may be executable files that instruct the system to accomplish a job.

Directory files: It contains information about a system that needs to access all types of files, but they do not contain the actual file data.

It occupies less space than a regular file.

Special files: These are the temporary files created by process.

These are of three types:

(i) FIFO

(ii) Block

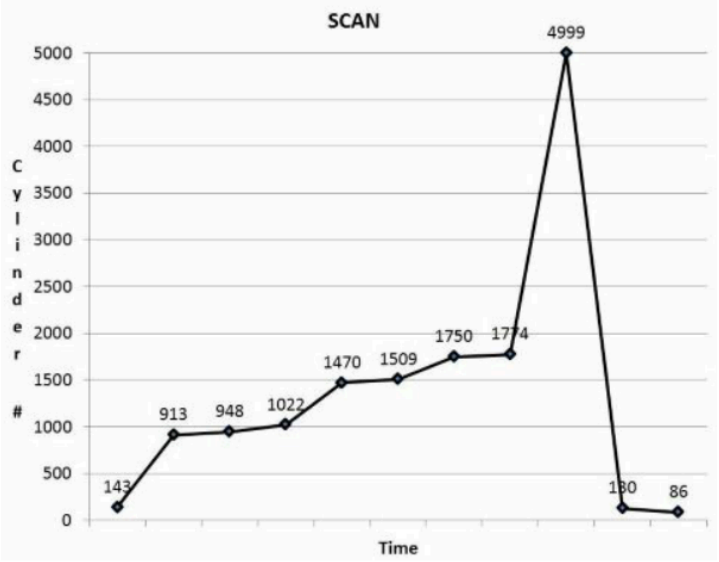
(iii) Character

9. Suppose we have files F1 to F4 in sizes of 7178, 572, 499 and 1195 bytes.Our disks have fixed physical block size of 512 bytes for allocation. Explain how many physical blocks would be needed to store these four files if we were to use a chained allocation strategy assuming that we need 5 bytes of information to determine the next block in the link. Which file results in the maximum internal fragmentation (measured as a percentage of the file size itself).

10 Consider that a disk drive has 5,000 cylinders, numbered 0 to 4,999. The drive is currently serving requests at cylinder 143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO

order, is: 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130 Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all pending requests for each of the following disk scheduling algorithms. A. FCFS B. SSTF C. SCAN D. C-SCAN E. LOOK F. C-LOOK

The FCFS schedule is 143, 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130. The total seek distance is 7081.
The SSTF schedule is 143, 130, 86, 913, 948, 1022, 1470, 1509, 1750, 1774. The total seek distance is 1745.

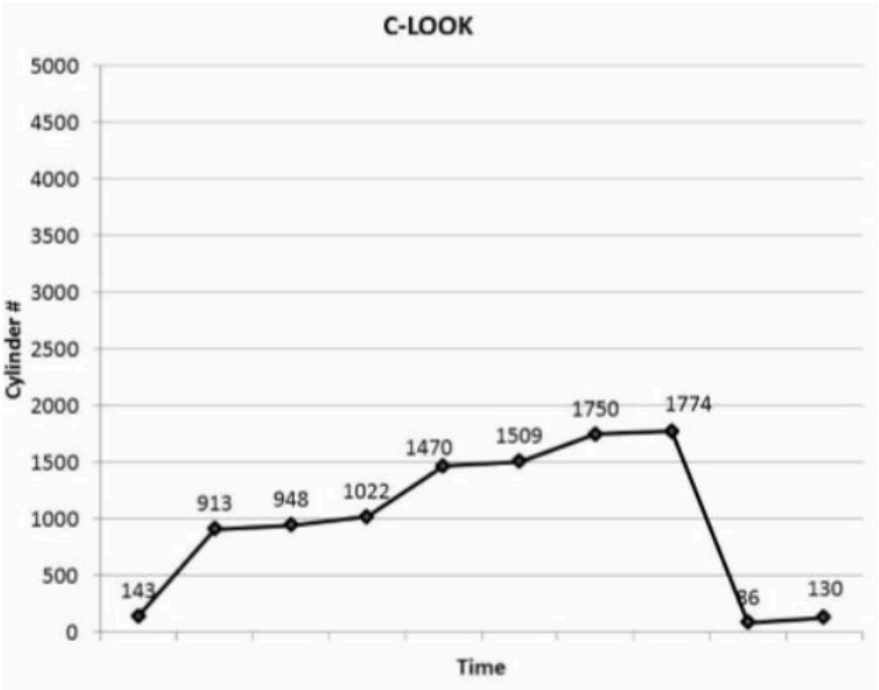


Total Distance travelled =
`|143−913| + |913−948| + |948−1022| + |1022−1470| + |1470−1509| + |1509−1750| + | 1750−1774 | + |1774−4999| + |4999−180| + |180−86| = 9769`

The SCAN schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774,4999, 130, 86. The total seek distance is 9769.
The LOOK schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 130, 86. The total seek distance is 3319.

The C-SCAN schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 4999, 0, 86, 130. The total seek distance is 9985.

Total distance travelled =
|143−913| + |913−948| + |948−1022| + |1022−1470| + |1470−1509| + |1509−1750| + |1 750−1774| + |1774−86| + |86−130|
= 3363



Total distance travelled=
|143−913| + |913−948| + |948−1022| + |1022−1470| + |1470−1509| + |1509−1750| + |1 750−1774| + |1774−86| + |86−130|
= 3363

The C-LOOK schedule is 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 86, 130. The total seek distance is 3363.

PART B

1. Explain in detail the interrupts and interrupt handling features.

Interrupt: It is an event that alters the sequence in which the processor executes.

Interrupt handler: It is the function that the Kernel runs in response to a specific interrupt.

- Each device that generates interrupts has an associated interrupt handler.
- The interrupt handler for a device is a part of the device drivers.
- In Linux, interrupt handlers are normal C functions, which match a specific prototype and thus enables the Kernel to pass the handler information in standard way.
- The difference between interrupt handles from other Kernel functions is that the Kernel invokes them in response to interrupts and that they run in a special context called interrupt context.
- An interrupt can occur at any time, an interrupt handler can be executed at any time.
- It is imperative that the handler runs quickly, to resume execution of the interrupted code as soon as possible.
- It is important that

  To the hardware: The OS services the interrupt without delay.

  To the rest of system: The interrupt handler executes in as short a period as possible.

2. Discuss about Disk space management.

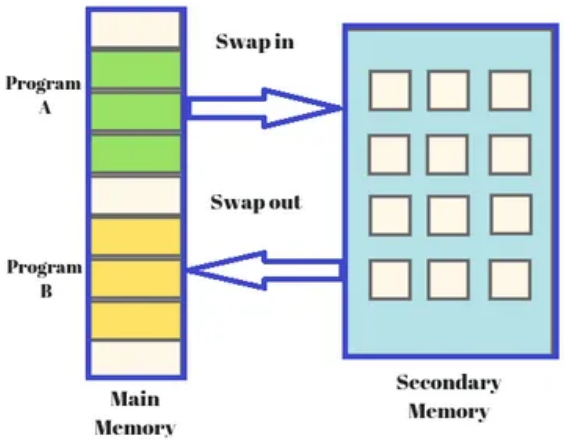
In order to ensure that your system and temporary partitions have enough storage space, you must manage your disk space. You need to maintain separate file systems, provide sufficient disk space, use shared devices for storage and manage temporary files in order to manage your disk space.

Disks serve as the main medium for storing files. A file can be defined as a collection of related information stored on a secondary device under a name. Means, a file can be viewed as a collection of bytes stored in certain memory locations. Since all these bytes represent the information related to a common specific program or job, they have to be linked together in some way or other. The simplest and easiest way to provide this linkage is to store the file in continuous memory locations.

3. Discuss about Swap – space management.

Swap-space management:

- A swap file is a space on a hard disk used as the virtual memory extension of computers real memory.
- Having a swap file allows your computer OS to pretend that you have more RAM than you actually do.
- The least recently used files in RAM can be swapped out to your hard disk until they are needed later so that new files can be swapped in to RAM.
- In larger OS the units that are moved are called pages and the swapping is called paging.



4. Describe the following Directory Implementation methods. a)Linear List b) Hash Table

Directory Implementation:

- Directories need to be fast to search, insert, and delete, with a minimum of wasted disk space.

Linear List:

- A linear list is the simplest and easiest directory structure to set up, but it does have some drawbacks.
- Finding a file (or verifying one does not already exist upon creation) requires a linear search.
- Deletions can be done by moving all entries, flagging an entry as deleted, or by moving the last entry into the newly vacant position.
- Sorting the list makes searches faster, at the expense of more complex insertions and deletions.
- A linked list makes insertions and deletions into a sorted list easier, with overhead for the links.
- More complex data structures, such as B-trees, could also be considered.

Hash Table:

- A hash table can also be used to speed up searches.
- Hash tables are generally implemented in addition to a linear or other structure.

5. Explain the concept of file sharing. What are the criteria to be followed in systems which implement file sharing.

File sharing:

It is the practice of sharing or offering access to digital information or resources, including documents, multimedia, graphics, computer programs, images and e-books. It is the private or public distribution of data or resources in a network with different levels of sharing privileges.

- File-sharing tasks use two basic sets of network criteria.

(i) Peer-to-Peer File Sharing: This is the most popular, but controversial, method of file sharing because of the use of peer-to-peer software.

- Peer-to-Peer file sharing allows users to directly access, download and edit files.
- Some 3rd party softwares facilitates Peer-to-Peer sharing by collecting large files into smaller pieces.

(ii) File Hosting Services: This is an alternative for peer-to-peer and it is a popular online material.

- These services are quite often used with Internet Collaboration Methods, including email, blogs etc., where direct download links from the file hosting services can be included.

6. Explain the following file concepts: a) File attributes b) File operations

File Attributes

Different OSes keep track of different file attributes, including:

- Name - Some systems give special significance to names, and particularly extensions (.exe, .txt, etc.), and some do not. Some extensions may be of significance to the OS (.exe), and others only to certain applications (.jpg)
- Identifier (e.g. inode number)
- Type - Text, executable, other binary, etc.
- Location - on the hard drive.
- Size
- Protection
- Time & Date
- User ID

File Operations

The file ADT supports many common operations:

- Creating a file
- Writing a file
- Reading a file
- Repositioning within a file
- Deleting a file
- Truncating a file.

Most OS's require that files be opened before access and closed after all access is complete. Normally the programmer must open and close files explicitly, but some rare systems open the file automatically at first access. Information about currently open files is stored in an open file table, containing for example:

- File pointer - records the current position in the file, for the next read or write access.
- File-open count - How many times has the current file been opened (simultaneously by different processes) and not yet closed? When this counter reaches zero the file can be removed from the table. o Disk location of the file.
- Access rights: Some systems provide support for file locking.
- A shared lock is for reading only.
- An exclusive lock is for writing as well as reading.
- An advisory lock is informational only, and not enforced. (A "Keep Out" sign, which may be ignored.)
- A mandatory lock is enforced. (A truly locked door.) o UNIX used advisory locks, and Windows used mandatory locks.

7. Explain the following file concepts: a) File types b) Internal file structure

File Types:

There are a large number of file types. Each has a particular purpose. The type of a file indicates its use cases, contents, etc. Some common types are:

1. Media:
   Media files store media data such as images, audio, icons, video, etc. Common extensions: img, mp3, mp4, jpg, png, flac, etc.

2. Programs:
   These files store code, markup, commands, scripts, and are usually executable. Common extensions: c, cpp, java, xml, html, css, js, ts, py, sql, etc.

3. Operating System Level:
   These files are present with the OS for its internal use. Common extensions: bin, sh, bat, dl, etc.

4. Document:
   These files are used for managing office programs such as documents, spreadsheets, etc. Common extensions: xl, doc, docx, pdf, ppt, etc.

5. Miscellaneous:
   Generic text file(.txt), canvas files, proprietary files, etc.

Internal File Structure

- Disk files are accessed in units of physical blocks, typically 512 bytes or some power-of-two multiple thereof. (Larger physical disks use larger block sizes, to keep the range of block numbers within the range of a 32-bit integer.)

- Internally files are organised in logical units, which may be as small as a single byte, or may be a larger size corresponding to some data record or structure size.
- The number of logical units which fit into one physical block determines its packing, and has an impact on the amount of internal fragmentation (wasted space) that occurs.
- As a general rule, half a physical block is wasted for each file, and the larger the block sizes the more space is lost to internal fragmentation.

8. Discuss the following a) File system mounting b)Thrashing

a) Before you can access the files on a file system, you need to mount the file system. Mounting a file system attaches that file system to a directory (mount point) and makes it available to the system. The root (/) file system is always mounted. Any other file system can be connected or disconnected from the root (/) file system.
When you mount a file system, any files or directories in the underlying mount point directory are unavailable as long as the file system is mounted. These files are not permanently affected by the mounting process, and they become available again when the file system is unmounted. However, mount directories are typically empty, because you usually do not want to obscure existing files.

b) Thrashing is computer activity that makes little or no progress, usually because memory or other resources have become exhausted or too limited to perform needed operations. When this happens, a pattern typically develops in which a request is made of the operating system by a process or program, the operating system tries to find resources by taking them from some other process, which in turn makes new requests that can't be satisfied. In a virtual storage system (an operating system that manages its

logical storage or memory in units called pages), thrashing is a condition in which excessive paging operations are taking place.
A system that is thrashing can be perceived as either a very slow system or one that has come to a halt.

9. Explain caching.

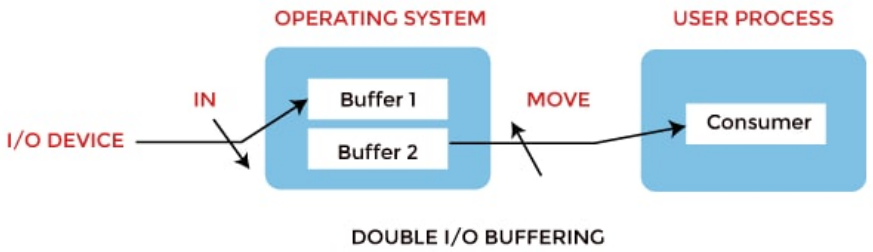
- Caching transparently stores data in a component called Cache, so that future requests for that data can be served faster.
- A special high-speed storage mechanism. It can be either a reserved section of main memory or an independent high-speed storage device.
- The data that is stored within a cache might be values that have been computed earlier or duplicates of original values that are stored elsewhere.
- E.g: Memory Caching, Disk Caching, Web Caching(used in browser), Database Caching etc.
- The data which is to be used many times results in wastage of time if it is in hard disk, but storing the data in cache reduces this time wastage.

A disk cache is a mechanism for improving the time it takes to read from or write to a hard disk. Today, the disk cache is usually included as part of the hard disk. A disk cache can also be a specified portion of random access memory (RAM). The disk cache holds data that has recently been read and, in some cases, adjacent data areas that are likely to be accessed next. Write caching is also provided with some disk caches.

10. Define buffering.

- Preloading data into a reserved area of memory (the buffer).

- It temporarily stores input or output data in an attempt to better match the speeds of two devices such as a fast CPU and a slow disk drive.
- Buffers may be used in between when moving data between two processes within a computer. Data is stored in a buffer as it is retrieved from one process or just before it is sent to another process.
- With spooling, the disk is used as a very large buffer. Usually complete jobs are queued on disk to be completed later.
- It is mostly used for input, output, and sometimes temporary storage of data either when transfer of data takes place or data that may be modified in a non-sequential manner.

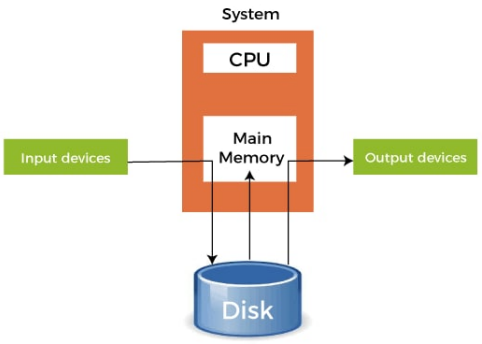


DOUBLE I/O BUFFERING

11. Write about spooling.

- Acronym for “Simultaneous Peripheral Operation On-Line”.
- It's a process of placing data in a temporary working area for another program to process. E.g: Print spooling and Mail spools etc.
- When there is a resource (like printer) to be accessed by two or more processes(or devices), spooling comes handy to schedule the tasks. Data from each process is put on the spool (print queue) and processed in FIFO(first in first out) manner.
- With spooling all processes can access the resource without waiting.
- After writing the data on a spool, the process can perform other tasks. And the printing process operates separately.
- Without spooling, the process would be tied up until the printing finished.

- Spooling is useful for devices which have differing data access rates. Used mainly when processes share some resource and need to have synchronisation.



12. Explain the techniques used for performing I/O Operations.

Programmed I/O: The processor issues an I/O command, on behalf of a process, to an I/O module; that process then busy-waits for the operation to be completed before proceeding.

Interrupt-driven I/O: The processor issues an I/O command on behalf of a process, continues to execute subsequent instructions, and is interrupted by the I/O module when the latter has completed its work. The subsequent instructions may be in the same process, if it is not necessary for that process to wait for the completion of the I/O. Otherwise, the process is suspended pending the interrupt and other work is performed.

Direct memory access (DMA): A DMA module controls the exchange of data between main memory and an I/O module. The processor sends a request for the transfer of a block of data to the DMA module and is interrupted only after the entire block has been transferred.

13. Give an example of an application in which data in a file should be accessed in the following order: i. Sequential ii. Random.

Sequential Access to a data file means that the computer system reads or writes information to the file sequentially, starting from the beginning of the file and proceeding step by step.

On the other hand, Random Access to a file means that the computer system can read or write information anywhere in the data file.

A more modern example is a **cassette tape** (sequential—you have to fast-forward through earlier songs to get to later ones) and a CD (random access—you can skip to the track you want).

14. Explain the following in detail with respect to the disk. a) Seek time b) Latency.

Seek time:

- A disk is divided into many circular tracks.
- Seek time is defined as the time required by the read/write head to move from one track to another.

Latency:

- The disk is divided into many circular tracks and these tracks are further divided into blocks known as sectors.
- The time required by the read/write head to rotate to the requested sector from the current position is called Rotational Latency.

15. Explain the following in detail with respect to disk. a)Access time b) Transfer time.

Access time : It is defined as the setup time before the actual data transfer takes place.

Access time is the summation of

a) Seek time

b) Rotational latency
c) Command processing time
d) Settle time

NOTE: Command processing time and settle time are taken as zero.

Transfer time: It is defined as the time required to transfer data between the system and the disk.

Transfer time is of two types:

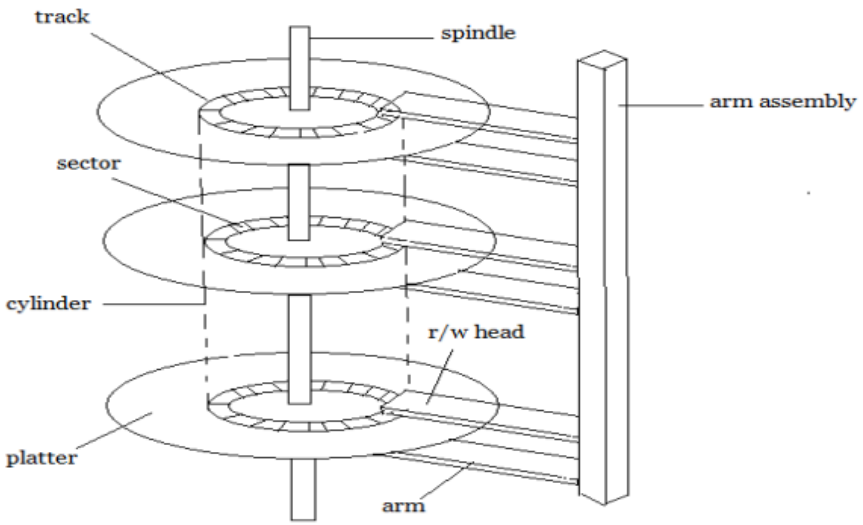
a) Internal transfer rate
b) External transfer rate

a) Internal transfer rate: It is defined as the time required to move data between disk surface and hard disk cache.

b) External transfer rate: It is defined as the time required to move data between hard disk cache and system.

16. Define magnetic disk structure and its management.

In modern computers, most of the secondary storage is in the form of magnetic disks. Hence, knowing the structure of a magnetic disk is necessary to understand how the data in the disk is accessed by the computer.



Structure of a magnetic disk

A magnetic disk contains several platters. Each platter is divided into circular shaped tracks. The length of the tracks near the center is less than the length of the tracks farther from the center. Each track is further divided into sectors, as shown in the figure.

Tracks of the same distance from the center form a cylinder. A read-write head is used to read data from a sector of the magnetic disk.

The speed of the disk is measured as two parts:

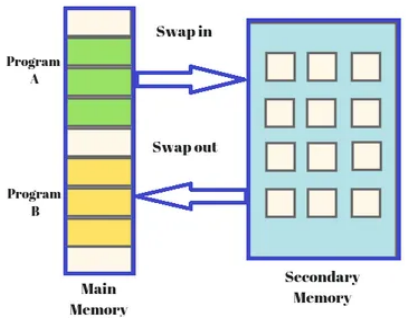
- Transfer rate: This is the rate at which the data moves from disk to the computer.
- Random access time: It is the sum of the seek time and rotational latency.

  Seek time is the time taken by the arm to move to the required track. Rotational latency is defined as the time taken by the arm to reach the required sector in the track.

  Even though the disk is arranged as sectors and tracks physically, the data is logically arranged and addressed as an array of blocks of fixed size. The size of a block can be 512 or 1024 bytes. Each logical block is mapped with a sector on the disk, sequentially. In this way, each sector in the disk will have a logical address.

17. Explain swap space management.

A swap file (or swap space or, in Windows NT, a pagefile) is a space on a hard disk used as the virtual memory extension of a computer's real memory (RAM). Having a swap file allows your computer's operating system to pretend that you have more RAM than you actually do. The least recently used files in RAM can be "swapped out" to your hard disk until they are needed later so that new files can be "swapped in" to RAM. In larger operating systems (such as IBM's OS/390), the units that are moved are called pages and the swapping is called paging.



18. Differentiate among the following disk scheduling algorithms. a) FCFS b) SSTF

1. FCFS: FCFS is the simplest of all the Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.

Advantages:

- Every request gets a fair chance
- No indefinite postponement

Disadvantages:

- Does not try to optimise seek time
- May not provide the best possible service

2. SSTF: In SSTF (Shortest Seek Time First), requests having the shortest seek time are executed first. So, the seek time of every request is calculated in advance in the queue and then they are scheduled according to their calculated seek time. As a result, the request near the disk arm will get executed first. SSTF is certainly an improvement over FCFS as it decreases the average response time and increases the throughput of the system.

Advantages:

- Average Response Time decreases
- Throughput increases

Disadvantages:

- Overhead to calculate seek time in advance.

- Can cause Starvation for a request if it has higher seek time as compared to incoming requests.
- High variance of response time as SSTF favours only some requests

Also refer Difference between FCFS and SSTF Disk Scheduling Algorithm - GeeksforGeeks

19. Differentiate among the following disk scheduling algorithms. a) SCAN b) C-SCAN

SCAN: In SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path. So, this algorithm works like an elevator and hence is also known as elevator algorithm. As a result, the requests at the midrange are serviced more and those arriving behind the disk arm will have to wait.

Advantages:

▪ High throughput

▪ Low variance of response time

▪ Average response time

Disadvantages:

▪ Long waiting time for requests for locations just visited by disk arm

CSCAN: In SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction. So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending at the scanned area.

These situations are avoided in the SAN algorithm in which the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the requests from there. So, the disk arm moves in a circular fashion and this algorithm is also similar to the SCAN algorithm and hence it is known as C-SCAN (Circular SCAN).

Advantages:

▪ Provides more uniform wait time compared to SCAN

Also refer Difference between SCAN and CSCAN Disk scheduling algorithms - GeeksforGeeks

20. Differentiate among the following disk scheduling algorithms. a)LOOK b) C-LOOK

LOOK: It is similar to the SCAN disk scheduling algorithm except the difference is that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only. Thus it prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

CLOOK: As LOOK is similar to SCAN algorithm, CLOOK is similar to CSCAN disk scheduling algorithm. In CLOOK, the disk arm in spite of going to the end goes only to the last request to be serviced in front of the head and then from there goes to the other end's last request. Thus, it also prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

PART C

1. Describe various file access method

File Access Methods:

1. Sequential Access Method. A sequential access is that in which the records are accessed in some sequence, i.e., the...
2. Direct or Random Access Methods. Sometimes it is not necessary to process every record in a file. It is not necessary...

3. Index Access Method. An indexed file is a computer file with an index that allows easy random access to any record...
4. Index sequential Access Method. The index sequential access method is a modification of the direct access method.

2. Explain the following i)file types ii) file operation iii) file attributes.

https://eduladder.com/viewquestions/556/Explain%20the%20following%20i)file%20types%20ii)file%20operation%20iii)file%20attributes

3. Define the terms – file, file path, directory?

File: A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks.
File path:
Directory :

4. Describe UFD and MFD.?

In the two-level directory structure, each user has her own user file directory (UFD).
Each UFD has a similar structure, but lists only the files of a single user. When a job starts the system's master file directory (MFD) is searched. The MFD is indexed by the user name or account number, and each entry points to the UFD for that user.

5. Describe file system mounting?

In computers, to mount is to make a group of files in a file system structure accessible to a user or user group. In some usages, it means to make a device physically accessible. For instance, in data storage, to mount is to place a data medium (such as a tape cartridge) on a drive in a position to operate. Macintosh calls it mounting when a user inserts a disc into the machine.

6. List the various layers of a file system.

A file system installed on an operating system consists of three layers:
Physical file system
Virtual file system
Logical file system

7. Explain the functions of the virtual file system (VFS)?

It has two functions:

1. It separates file-system-generic operations from their implementation defining a clean VFS interface. It allows transparent access to different types of file systems mounted locally.

2. VFS is based on a file representation structure, called a vnode. It contains a numerical value for a network-wide unique file .The kernel maintains one vnode structure for each active file or directory.

8. Explain the allocation methods of a disk space?

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.

9. List the various layers of a file system?

▤ Application programs

▤ Logical file system

▤ File-organization module

▤ Basic file system

▤ I/O control

▤ Devices

10. Describe the logical formatting of the disk.

Disk formatting is the process of preparing a data storage device such as a hard disk drive, solid-state drive, floppy disk or USB flash drive for initial use. In some cases, the formatting operation may also create one or more new file systems. The first part of the formatting process that performs basic medium preparation is often referred to as "low-level formatting". Partitioning is the common term for the second part of the process, dividing the device into several sub-devices and, in some cases, writing information to the device allowing an operating system to be booted from it. The third part of the process, usually termed "high-level formatting" most often refers to the process of generating a new file system. In some operating systems all or parts of these three processes can be combined or repeated at different levels and the term "format" is understood to mean an operation in which a new disk medium is fully prepared to store files. Some formatting utilities allow distinguishing between a quick format, which does not erase all existing data and a long option that does erase all existing data.

11. List any four common file types and their extensions.

File type	Usual extension
Archive	arc, zip, tar
Multimedia	mpeg, mov, rm
Markup	xml, html, tex
Library	lib, a ,so, dll

12) Explain any four common file attributes?

- Read-only - Allows a file to be read, but nothing can be written to the file or changed.
- Archive - Tells Windows Backup to backup the file.
- System - System file.
- Hidden - File will not be shown when doing a regular dir from DOS.

13) Explain any four file operations?

- Copying a file
- Moving a file
- Deleting a file
- Renaming a file

14. Distinguish between shared and exclusive lock?

exclusive	Shared
exclusive locks are sometimes called "write locks".	Shared locks are sometimes called "read locks"
An exclusive lock prohibits other users from reading the locked resource	a shared lock allows other users to read the locked resource, but they cannot update it.

15. List any four secondary storage memory devices.

Secondary storage devices

- Solid-state drives (SSDs).
- Hard disk drives (HDDs).
- Cloud storage.
- CD-ROM drives.
- DVD drives.
- Blu-ray drives.

16. Define the terms with respect to disk I/O - seek time, latency time?

Seek time is the time required to move the disk arm to the required track. latency is the time it takes for the beginning of the required sector to reach the head.

17. Explain the information associated with an open file.

Several pieces of information are associated with an open file which may be:

1. File pointer
2. File open count
3. Disk location of the file
4. Access rights

18. Discuss the advantages of contiguous memory allocation of disk space?

- Both the Sequential and Direct Accesses are supported by this. For direct access, the address of the kth block of the file which starts at block b can easily be obtained as (b+k).
- This is extremely fast since the number of seeks are minimal because of contiguous allocation of file blocks.

(Or)

- Supports direct access
- Supports sequential access
- Number of disk seeks is minimal.

19. Write a short note on procedures?

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

20. Discuss the drawbacks of contiguous allocation of disk space?

- This method suffers from both internal and external fragmentation. This makes it inefficient in terms of memory utilisation.
- Increasing file size is difficult because it depends on the availability of contiguous memory at a particular instance.

(or)

- Suffers from external fragmentation

- Suffers from internal fragmentation
- Difficulty in finding space for a new file
- File cannot be extended
- Size of the file is to be declared in advance

IARE '24 DISCORD TEAM

II-II

OS MODULE 5 SOLUTIONS

VISHAL • SYED IKRAM • PRANAV • UJJWAL • ASRITHA

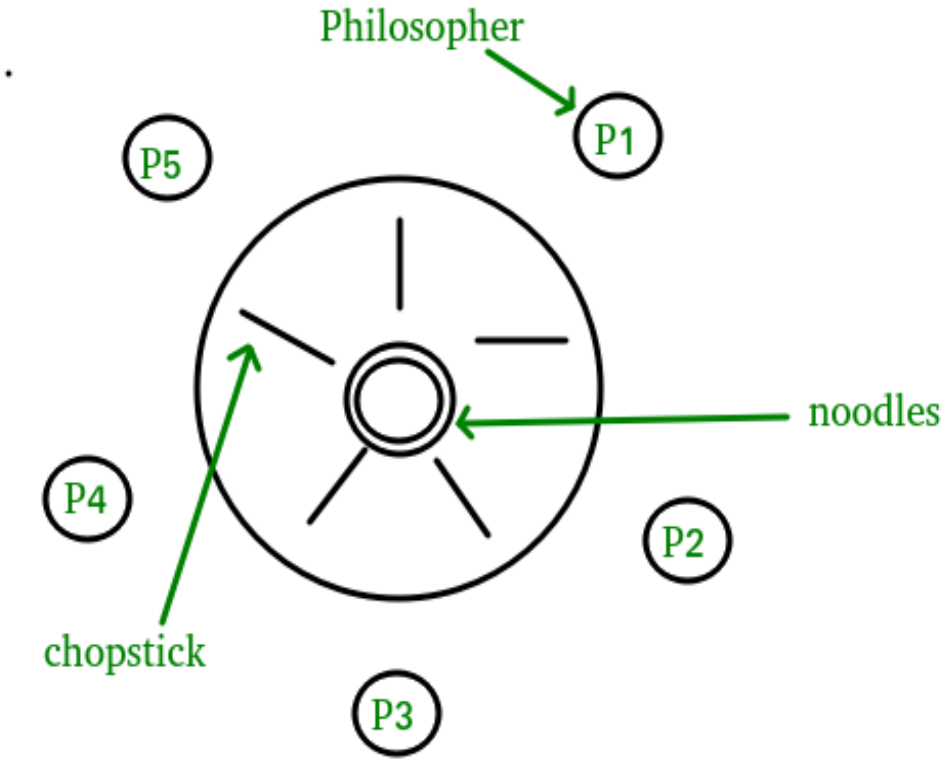
DEADLOCKS, PROTECTION

Completed

OS MODULE 5

PART A

1. Consider the version of the dining-philosophers problem in which the chopsticks are placed at the center of the table and any two of them can be used by a philosopher. Assume that requests for chopsticks are made one at a time. Describe a simple rule for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers



Dining Philosophers Problem

The rule that prevents deadlock philosophers problem is: When one philosopher makes a request for the first chopstick, do not award the

request if there are no other philosophers with two chopsticks and if there is only one chopstick remaining.

2. Consider a system consisting of m resources of the same type being shared by n processes. A process can request or release only one resource at a time. Show that the system is deadlock free if the following two conditions hold: a) The maximum need of each process is between one resource and m resources. b) The sum of all maximum needs is less than m +n.

...Proof:

Suppose N = Sum of all Need(i), A = Sum of all Allocation(i), M = Sum of all Max(i). Use contradiction to prove.

Assume this system is not deadlock free. If there exists a deadlock state, then A = m because there's only one kind of resource and resources can be requested and released only one at a time. From condition b, N + A = M < m + n. So we get N + m < m + n. So we get N < n. It shows that at least one process i that Need(i) = 0. From condition a, Pi can release at least 1 resource. So there are n-1 processes sharing m resources now, condition a and b still hold. Go on the argument, no process will wait permenently, so there's no deadlock.

3. Given 3 processes A,B and C, three resources x,y and z and following events, a. A requests x ii) A requests y iii) B requests y iv) B requests z v) C requests z vi) C requests x vii) C requests y Assume that requested resources should always be allocated to the request process if it is available. Draw the resource allocation graph for the sequences. And also mention whether it is a deadlock? If it is, how to recover the deadlock.

① Tutorial - 3

Q1) Given 3 processes A, B & C, three single instance resources X, Y & Z are following events happen in the sequence as given:
1. A requests X, A requests Y, B requests Z, C requests Z,
C requests X, C requests Y, B requests Y
2. A requests X, B requests Y, C requests Z, A requests Y,
B requests Z, C requests X, C requests Y.

Assuming that initially all resources are free & once requested gets allocated to the requesting process if it is available. Draw the single instance resource allocation graph for the sequences.
And also mention whether it is a deadlock, justify if yes or no?

⇒ 1)

Here, Allocation is represented in Blue.
Requesting is represented in Red.

Basically there are mainly four conditions of deadlock:
(i) Mutual Exclusion: (All resources are non-sharable) ✓
(ii) No-Preemption (Processes should not be preempted) ✓
(iii) Hold & Wait (All processes are holding a resource & waiting for other) ✓
(iv) Circular Wait (X)

Here, in this pattern,
* Hold & Wait condition is not there.
Reason: A is not waiting for any resource.
* Circular Wait is also not there.

[Hence, Deadlock will not happen in this case.]

② 2)

Here, in this case;
* Mutual Exclusion is there ✓
* No-Preemption ✓
* Hold & Wait ✓
* Circular Wait ✓

Here, all our deadlock condition satisfy,
Also, when cycle is present in single instance Allocation graph then deadlock will definetely happen.
[Hence, deadlock will occur in this case.]

4. Explain How does the principle of least privilege aid in the creation of protection systems

The principle of least privilege works by allowing only enough access to perform the required job. In an IT environment, adhering to the principle of least privilege reduces the risk of attackers gaining access to critical systems or sensitive data by compromising a low-level user account, device, or application. Implementing the POLO helps contain compromises to their area of origin, stopping them from spreading to the system at large.
The principle of least privilege prevents the spread of malware on your network. An administrator or superuser with access to a lot of other network resources and infrastructure could potentially spread malware to all those other systems.
Five Benefits of the Least Privilege Principle

1. The least privilege principle reduces liability.
2. Least privilege access limits the possibility of catastrophic damage.
3. The principle of least privilege protects against common attacks, like SQL injections.
4. Data classification creates a healthy, secure network.
5. The least privilege principle enables better security and audit capabilities.

5. Describe how the Java protection model would be compromised if a Java program were allowed to directly alter the annotations of its stack frame.

When a thread issues an access request in a doPrivileged() block, the stack frame of the calling thread is annotated and the stack frame can make subsequent method.Thus, the annotation serves to mark a calling thread as being privileged.

6. Describe the Coffman's conditions that lead to a deadlock.

A deadlock situation can arise if and only if all of the following conditions hold simultaneously in a system:

- **Mutual Exclusion:** At least one resource must be non-shareable. Only one process can use the resource at any given instant of time.
- **Hold and Wait or Resource Holding:** A process is currently holding at least one resource and requesting additional resources which are being held by other processes.
- **No Preemption:** The operating system must not de-allocate resources once they have been allocated; they must be released by the holding process voluntarily.
- **Circular Wait:** A process must be waiting for a resource which is being held by another process, which in turn is waiting for the first process to release the resource. In general, there is a set of waiting processes, P = {P1, P2, ..., PN}, such that P1 is waiting for a resource held by P2, P2 is waiting for a resource held by P3 and so on till PN is waiting for a resource held by P1.

These four conditions are known as the Coffman conditions from their first description in a 1971 article by Edward G. Coffman.

7. A system contains three programs and each requires three tape units for its operation. Explain the minimum number of tape units which the system must have such that deadlocks never arise is

Data:

Programs in system = Y = 3

Tapes per program needed = R = 3

Calculation:

Max tape per program to be in deadlock = needed – 1 = 3 – 1 = 2

For Y program, max resource to be in deadlock = Y × 2 = 2Y

Condition for deadlock free

Total tapes > 2Y

$T > 2 \times 3$

$T_{min} = 7$

The minimum number of tape units which the system must have such that deadlocks never arise is 7

8. A system has 6 identical resources and N processes competing for them. Each process can request at most 2 resources. Explain which one of the following values of N could lead to a deadlock.

Data:

Available identical Resources = R = 6

Max needs per process = 2

Concepts:

Deadlock can occur If any process gets available resource < needed (requested) resource

Max resource per process to be in deadlock = needed − 1 = 2 − 1 = 1

For N process, max resource to be in deadlock = N × 1 = N

Condition for deadlock

N ≥ R

N ≥ 6

values of N could lead to a deadlock is 6

9. Define in detail the technique of deadlock avoidance

In order to avoid deadlocks, the process must tell the OS the maximum number of resources a process can request to complete its execution.
The simplest and most useful approach states that the process should declare the maximum number of resources of each type it may ever need. The Deadlock avoidance algorithm examines the resource allocations so that there can never be a circular wait condition.
Deadlock avoidance can be done with Banker's Algorithm.

Banker's Algorithm
Banker's Algorithm is resource allocation and deadlock avoidance algorithm which test all the request made by processes for resources, it checks for

the safe state, if after granting request system remains in the safe state it allows the request and if there is no safe state it doesn't allow the request made by the process.

10. Explain briefly about the purpose of the banker's algorithm.

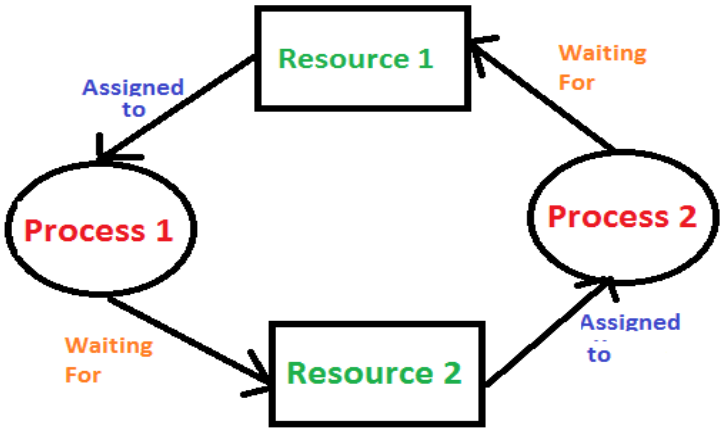
Banker's Algorithm is used majorly in the banking system to avoid deadlock. It helps you to identify whether a loan will be given or not. The banker's algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an "s-state" check to test for possible activities, before deciding whether allocation should be allowed to continue.

Banker's algorithm is named so because it is used in the banking system to check whether a loan can be sanctioned to a person or not. Suppose there are n number of account holders in a bank and the total sum of their money is S. If a person applies for a loan then the bank first subtracts the loan amount from the total money that bank has and if the remaining amount is greater than S then only the loan is sanctioned. It is done because if all the account holders come to withdraw their money then the bank can easily do it.
In other words, the bank would never allocate its money in such a way that it can no longer satisfy the needs of all its customers. The bank would always try to be in a safe state.

PART B

1. What is the deadlock? Explain the necessary conditions for its occurrence

A deadlock is a situation in which more than one process is blocked because it is holding a resource and also requires some resource that is acquired by some other process. Therefore, none of the processes gets executed.



IMG: Example of Deadlock

Four necessary conditions for its occurrence are:

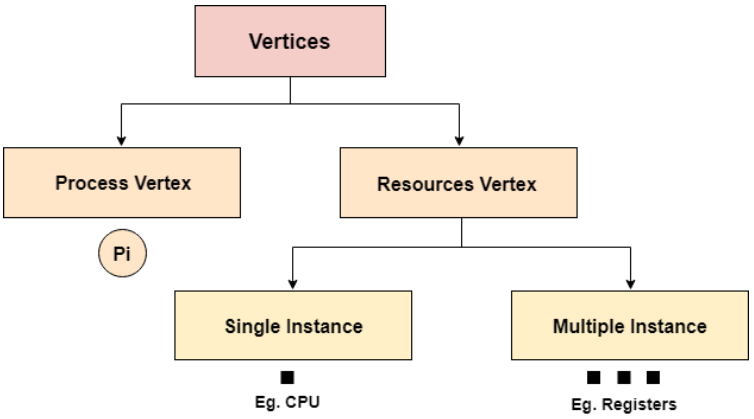
1. Mutual Exclusion: At least one resource must be non-shareable. Only one process can use the resource at any given instant of time.
2. Hold and Wait or Resource Holding: A process is currently holding at least one resource and requesting additional resources which are being held by other processes.
3. No Preemption: The operating system must not de-allocate resources once they have been allocated; they must be released by the holding process voluntarily.
4. Circular Wait: A process must be waiting for a resource which is being held by another process, which in turn is waiting for the first process to release the resource. In general, there is a set of waiting processes, P = {P1, P2, ..., PN}, such that P1 is waiting for a resource held by P2, P2 is waiting for a resource held by P3 and so on till PN is waiting for a resource held by P1.

2. Explain briefly resource allocation graph with examples

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

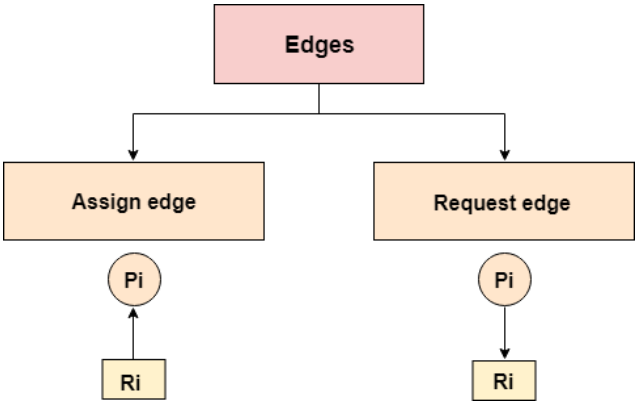
It also contains the information about all the instances of all the resources whether they are available or being used by the processes.

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



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

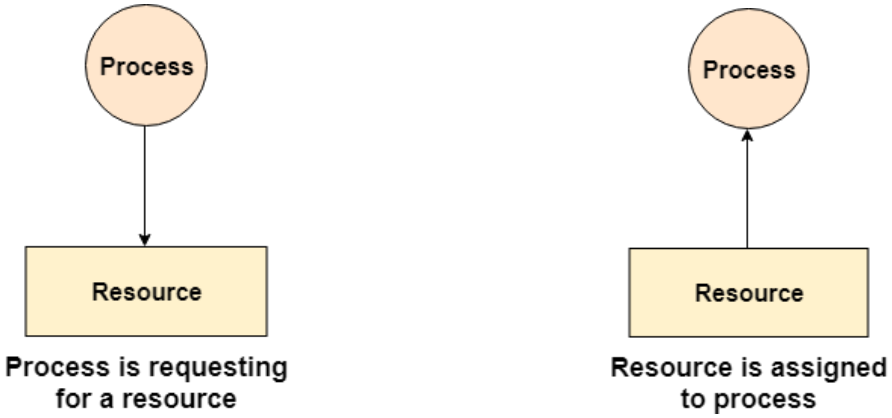
A resource can have more than one instance. Each instance will be represented by a dot inside the rectangle.



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

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

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

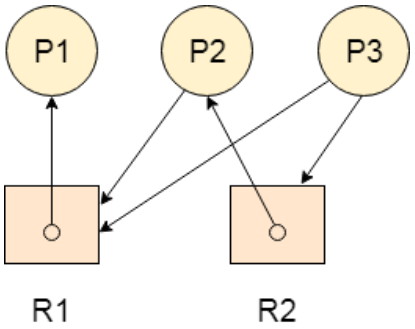


Example

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

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

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



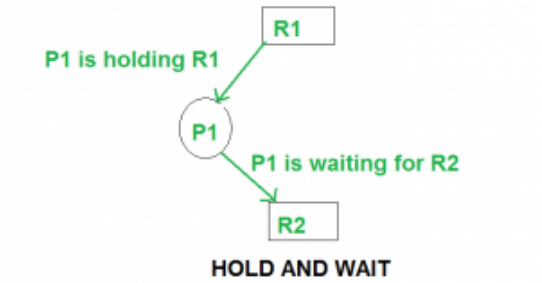
3. Explain the methods for deadlock prevention.

Deadlock prevention is eliminating one of the necessary conditions of deadlock so that only safe requests are made to OS and the possibility of deadlock is excluded before making requests.
As now requests are made carefully, the operating system can grant all requests safely.

Deadlock Prevention Techniques:
Deadlock prevention techniques refer to violating any one of the four necessary conditions. We will see one by one how we can violate each of them to make safe requests and which is the best approach to prevent deadlock.

- Eliminate Mutual Exclusion:
  It is not possible to dis-satisfy the mutual exclusion because some resources, such as the tape drive and printer, are inherently non-shareable.

- Eliminate Hold and wait:
  Allocate all required resources to the process before the start of its execution, this way hold and wait condition is eliminated but it will lead to low device utilisation. For example, if a process requires a printer at a later time and we have allocated a printer before the start of its execution, the printer will remain blocked till it has completed its execution.
  The process will make a new request for resources after releasing the current set of resources. This solution may lead to starvation.



- Eliminate No Preemption:
  Preempt resources from the process when resources are required by other high priority processes.

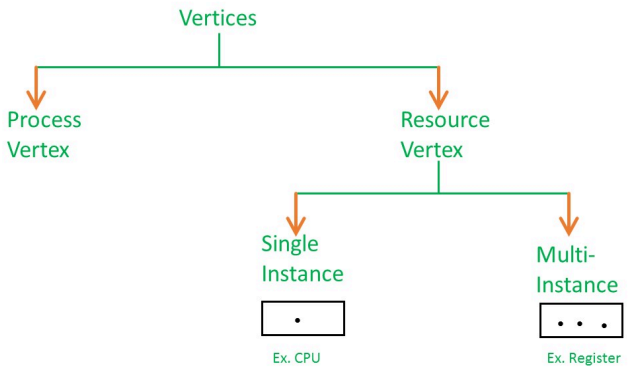
- Eliminate Circular Wait:
  Each resource will be assigned with a numerical number. A process can request the resources to increase/decrease. order of numbering.
  For Example, if P1 process is allocated R5 resources, now next time if P1 ask for R4, R3 less than R5 such request will not be granted, only request for resources more than R5 will be granted.

4. Explain the resource allocation graph.

As Banker's algorithm using some kind of table like allocation, request, available all that thing to understand what is the state of the system. Similarly, if you want to understand the state of the system instead of using those table, actually tables are very easy to represent and understand it, but then still you could even represent the same information in the graph. That graph is called **Resource Allocation Graph (RAG).**
So, resource allocation graph is explained to us what is the state of the system in terms of processes and resources. Like how many resources are available, how many are allocated and what is the request of each process. Everything can be represented in terms of the diagram. One of the advantages of having a diagram is, sometimes it is possible to see a deadlock directly by using RAG, but then you might not be able to know that by looking at the table. But the tables are better if the system contains lots of process, resource and graph is better if the system contains less number of process and resource. We know that any graph contains vertices and edges. So RAG also contains vertices and edges. In RAG vertices are of two types:

1. **Process vertex -** Every process will be represented as a process vertex. Generally, the process will be represented with a circle.

2. **Resource vertex -** Every resource will be represented as a resource vertex. It is also two types:
    - **Single instance type resource -** It represents as a box, there will be one dot. So the number of dots indicate how many instances are present of each resource type.
    - **Multi - resource instance type resource -** It also represents as a box, inside the box, there will be many dots present.



5. Differentiate the deadlock handling methods

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

1. Deadlock Prevention
2. Deadlock avoidance
3. Deadlock detection

These are explained below.

1. Deadlock Prevention : The strategy of deadlock prevention is to design the system in such a way that the possibility of deadlock is excluded. Indirect methods prevent the occurrence of one of three necessary conditions of deadlock i.e., mutual exclusion, no pre-emption and hold and wait. Direct methods prevent the occurrence of circular wait. Prevention techniques – Mutual exclusion – is supported by the OS. Hold and Wait – condition can be prevented by requiring that a process requests all its

required resources at one time and blocking the process until all of its requests can be granted at a same time simultaneously. But this prevention does not yield good result because :

- long waiting time required
- in efficient use of allocated resource
- A process may not know all the required resources in advance

No pre-emption – techniques for 'no pre-emption are'
If a process that is holding some resource, requests another resource that can not be immediately allocated to it, all resources currently being held are released and if necessary, request them again together with the additional resource.
If a process requests a resource that is currently held by another process, the OS may preempt the second process and require it to release its resources. This works only if both the processes do not have the same priority.
Circular wait One way to ensure that this condition never hold is to impose a total ordering of all resource types and to require that each process requests resource in an increasing order of enumeration, i.e., if a process has been allocated resources of type R, then it may subsequently request only those resources of types following R in ordering.

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

- Process initiation denial

- Resource allocation denial

Advantages of deadlock avoidance techniques :

- Not necessary to pre-empt and rollback processes
- Less restrictive than deadlock prevention

Disadvantages :

- Future resource requirements must be known in advance
- Processes can be blocked for long periods
- Exists fixed number of resources for allocation

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

This technique does not limit resources access or restrict process action.
Requested resources are granted to processes whenever possible.
It never delays the process initiation and facilitates online handling.
The disadvantage is the inherent pre-emption losses.

6. Explain Banker's algorithm for deadlock avoidance with an example

Banker's Algorithm
Banker's Algorithm is resource allocation and deadlock avoidance algorithm which test all the request made by processes for resources, it checks for the safe state, if after granting request system remains in the safe state it allows the request and if there is no safe state it doesn't allow the request made by the process.

Banker's Algorithm is used majorly in the banking system to avoid deadlock. It helps you to identify whether a loan will be given or not.

The banker's algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an "s-state" check to test for possible activities, before deciding whether allocation should be allowed to continue.

Banker's algorithm is named so because it is used in the banking system to check whether a loan can be sanctioned to a person or not. Suppose there are n number of account holders in a bank and the total sum of their money is S. If a person applies for a loan then the bank first subtracts the loan amount from the total money that bank has and if the remaining amount is greater than S then only the loan is sanctioned. It is done because if all the account holders come to withdraw their money then the bank can easily do it.
In other words, the bank would never allocate its money in such a way that it can no longer satisfy the needs of all its customers. The bank would always try to be in a safe state.

7. Discuss the various issues that need to be considered through the process of revocation of access rights.

With an access list scheme revocation is easy, immediate, and can be selective, general, partial, total, temporary, or permanent, as desired.
With capabilities lists the problem is more complicated, because access rights are distributed throughout the system. A few schemes that have been developed include:

- Reacquisition - Capabilities are periodically revoked from each domain, which must then re-acquire them.

- Back-pointers - A list of pointers is maintained from each object to each capability which is held for that object.
- Indirection - Capabilities point to an entry in a global table rather than to the object. Access rights can be revoked by changing or invalidating the table entry, which may affect multiple processes, which must then re-acquire access rights to continue.
- Keys - A unique bit pattern is associated with each capability when created, which can be neither inspected nor modified by the process.
    - A master key is associated with each object.
    - When a capability is created, its key is set to the object's master key.
    - As long as the capability's key matches the object's key, then the capabilities remain valid.

Revocation of Access Rights:

The need to revoke access rights dynamically raises several questions:

➔ Immediate versus delayed - If delayed, can we determine when the revocation will take place?

➔ Selective versus general - Does revocation of an access right to an object affect all users who have that right, or only some users?

➔ Partial versus total - Can a subset of rights for an object be revoked, or are all rights revoked at once?

➔ Temporary versus permanent - If rights are revoked, is there a mechanism for processes to re-acquire some or all of the revoked rights?

8. State and explain the methods involved in recovery from deadlocks.

Deadlock recovery performs when a deadlock is detected:

When a deadlock is detected, then our system stops working, and after the recovery of the deadlock, our system starts working again.

Therefore, after the detection of deadlock, a method/way must be required to recover that deadlock to run the system again. The method/way is called deadlock recovery.

1. Deadlock recovery through preemption
2. Deadlock recovery through rollback
3. Deadlock recovery through killing processes

Deadlock Recovery through Preemption

The ability to take a resource away from a process, have another process use it, and then give it back without the process noticing. It is highly dependent on the nature of the resource.

Deadlock recovery through preemption is too difficult or sometimes impossible.

Deadlock Recovery through RollBack

In this case of deadlock recovery through rollback, whenever a deadlock is detected, it is easy to see which resources are needed.

To do the recovery of deadlock, a process that owns a needed resource is rolled back to a point in time before it acquired some other resource just by starting one of its earlier checkpoints.

Deadlock Recovery through Killing Processes

This method of deadlock recovery through killing processes is the simplest way of deadlock recovery.

Sometimes it is best to kill a process that can be returned from the beginning with no ill effects.

9. Describe resource-allocation graph. Explain how resource graph can be used for detecting deadlocks.

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

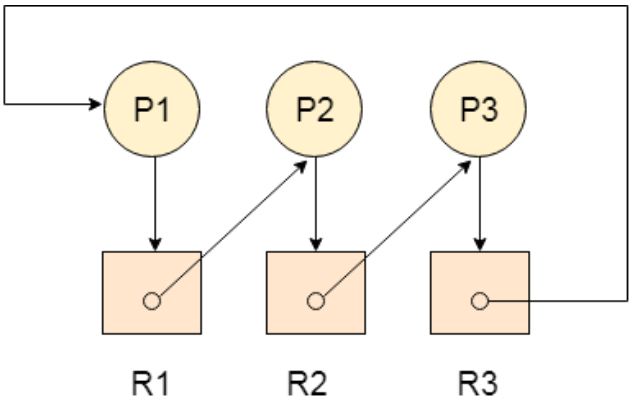
It also contains the information about all the instances of all the resources whether they are available or being used by the processes.

Deadlock Detection using RAG

If a cycle is being formed in a Resource allocation graph where all the resources have the single instance then the system is deadlocked.

In Case of Resource allocation graphs with multi-instanced resource types, Cycle is a necessary condition of deadlock but not the sufficient condition.

The following example contains three processes P1, P2, P3 and three resources R2, R2, R3. All the resources are having single instances each.



If we analyse the graph then we can find out that there is a cycle formed in the graph since the system is satisfying all the four conditions of deadlock.

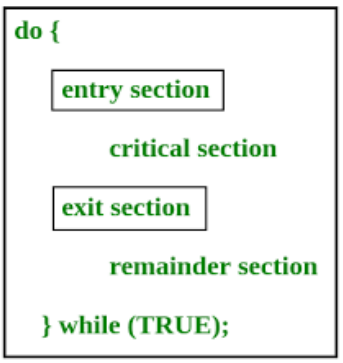
For more click here

10 Describe the terms. a) Race condition b) Atomic transaction c) Critical section d) Mutual exclusion

a) Race condition

A race condition occurs when two or more threads can access shared data and they try to change it at the same time. Because the thread scheduling algorithm can swap between threads at any time, you don't know the order in which the threads will attempt to access the shared data. Therefore, the result of the change in data is dependent on the thread scheduling algorithm, i.e. both threads are "racing" to access/change the data.
A race condition is a situation that may occur inside a critical section. This happens when the result of multiple thread execution in the critical section differs according to the order in which the threads execute.

b) Atomic transaction
An atomic transaction is an indivisible and irreducible series of database operations such that either all occurs, or nothing occurs. A guarantee of atomicity prevents updates to the database occurring only partially, which can cause greater problems than rejecting the whole series outright. As a consequence, the transaction cannot be observed to be in progress by another database client. At one moment in time, it has not yet happened, and at the next it has already occurred in whole (or nothing happened if the transaction was cancelled in progress).
An example of an atomic transaction is a monetary transfer from bank account A to account B. It consists of two operations, withdrawing the money from account A and saving it to account B. Performing these operations in an atomic transaction ensures that the database remains in a consistent state, that is, money is neither lost nor created if either of those two operations fails.
critical section
The critical section refers to the segment of code where processes access shared resources, such as common variables and files, and perform write operations on them.



Since processes execute concurrently, any process can be interrupted mid-execution. In the case of shared resources, partial execution of processes can lead to data inconsistencies. When two processes access and manipulate the shared resource concurrently, and the resulting execution outcome depends on the order in which processes access the resource; this is called a race condition.

Mutual exclusion

A mutual exclusion (mutex) is a program object that prevents simultaneous access to a shared resource. This concept is used in concurrent programming with a critical section, a piece of code in which processes or threads access a shared resource. Only one thread owns the mutex at a time, thus a mutex with a unique name is created when a program starts

Refer example from Mutual Exclusion in Synchronization - GeeksforGeeks

11. Describe how the access matrix facility and role-based access control facility are similar. How do they differ?

Access Matrix is a security model of protection state in computer systems. It is represented as a matrix. Access matrix is used to define the rights of each process executing in the domain with respect to each object. The rows of the matrix represent domains and columns represent objects.

Role-based access control (RBAC), also known as role-based security, is a mechanism that restricts system access. It involves setting permissions and privileges to enable access to authorised users. Most large organisations use role-based access control to provide their employees with varying

levels of access based on their roles and responsibilities. This protects sensitive data and ensures employees can only access information and perform actions they need to do their jobs.

12. Explain why a capability based system such as Hydra provides greater flexibility than the ring- protection scheme in enforcing protection policies.

The ring-based protection scheme requires the modules to be ordered in a strictly hierarchical fashion. It also enforces the restriction that system code in internal rings cannot invoke operations in the external rings. This restriction limits the flexibility in structuring the code andis unnecessarily restrictive. The capability system provided by Hydra not only allows unstructured interactions between different modules,but also enables the dynamic instantiation of new modules as the need arises.

13. Define Goals of protection

Processes in the system must be protected from one another's activities. Else there may be disruption in the normal working. Protection refers to a mechanism for controlling the access of programs, processes, or users to resources defined by the computer system

Goals of protection

1. Provides a means to distinguish between authorised and unauthorised usage.
2. To prevent mischievously, intentional violation of an access restriction by the user.
3. To ensure that each program component which is active in a system uses system resources only in ways consistent with stated policies. (This gives a reliable system).
4. To detect latent errors at the interfaces between the component subsystems. (This can improve reliability). Early detection helps in preventing malfunctioning of subsystems.
5. To enforce policies governing resource usage.

14. Define Principles of protection.

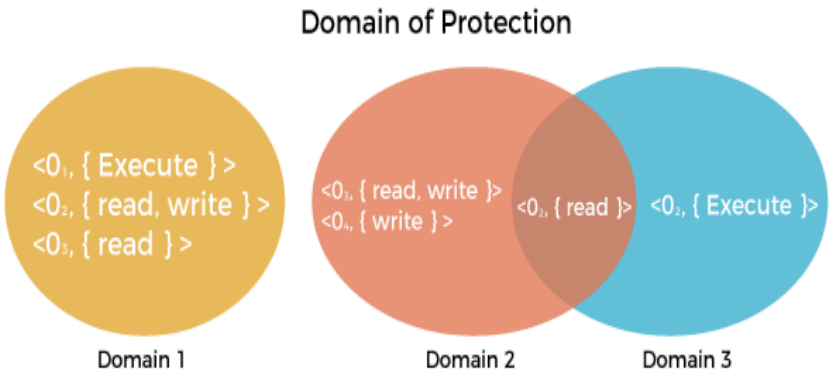
1. The time-tested guiding principle used for protection is called the principle of least privilege. It states that programs, users and even systems be given just enough privileges to perform their tasks.
2. An OS following this principle implements its features, programs, system calls, and data structures so that failure or compromise of a component does the minimum damage and allows minimum damage to be done. Such OS has fine-grained access control.
3. It provides mechanisms to enable privileges when they are needed and to disable them when not needed.
4. Privileged function access have audit trails that enable programmer or systems administrator or law-enforcement officer to trace all protection and security activities of the system.

5. We can create separate accounts for each user with just the privileges that the user needs.

15. Discuss about domain of protection

Various domains of protection in operating system are as follows:

- The protection policies restrict each process's access to its resource handling. A process is obligated to use only the resources necessary to fulfil its task within the time constraints and in the mode in which it is required. It is a process's protected domain.
- Processes and objects are abstract data types in a computer system, and these objects have operations that are unique to them. A domain component is defined as <object, {set of operations on object}>.



System with three protection domains.

- Each domain comprises a collection of objects and the operations that may be implemented on them. A domain could be made up of only one process, procedure, or user. If a domain is linked with a procedure, changing the domain would mean changing the procedure ID. Objects may share one or more common operations.

16. Why do you need to provide protection to the system?

Various needs of protection in the operating system are as follows:

1. There may be security risks like unauthorised reading, writing, modification, or preventing the system from working effectively for authorised users.
2. It helps to ensure data security, process security, and program security against unauthorised user access or program access.
3. It is important to ensure no access rights' breaches, no viruses, no unauthorised access to the existing data.
4. Its purpose is to ensure that only the systems' policies access programs, resources, and data.

17. Discuss the access matrix implementation techniques.

1. Global Table:

- The simplest approach is one big global table with <domain, object, rights> entries.
- Unfortunately this table is very large (even if sparse) and so cannot be kept in memory (without invoking virtual memory techniques.)
- There is also no good way to specify groupings - if everyone has access to some resource, then it still needs a separate entry for every domain.

2. Access Lists for Objects:

- Each column of the table can be kept as a list of the access rights for that particular object, discarding blank entries.
- For efficiency a separate list of default access rights can also be kept and checked first.

3. Capability Lists for Domains:

- In a similar fashion, each row of the table can be kept as a list of the capabilities of that domain.
- Capability lists are associated with each domain, but not directly accessible by the domain or any user process.

- Capability lists are themselves protected resources, distinguished from other data in one of two ways:
- A tag, possibly hardware implemented, distinguishing this special type of data. (other types may be floats, pointers, booleans, etc.)
- The address space for a program may be split into multiple segments, at least one of which is inaccessible by the program itself and used by the operating system for maintaining the process's access right capability list.

4. A Lock-Key Mechanism:

- Each resource has a list of unique bit patterns, termed locks.
- Each domain has its own list of unique bit patterns, termed keys.
- Access is granted if one of the domain's keys fits one of the resource's locks.
- Again, a process is not allowed to modify its own keys.
- It is effective and flexible.

Object / Domain	F_1	F_2	F_3	Laser Printer
D_1	read		read	
D_2				Print
D_3		read	exceute	
D_4	read / write		read / write	

Access Matirx

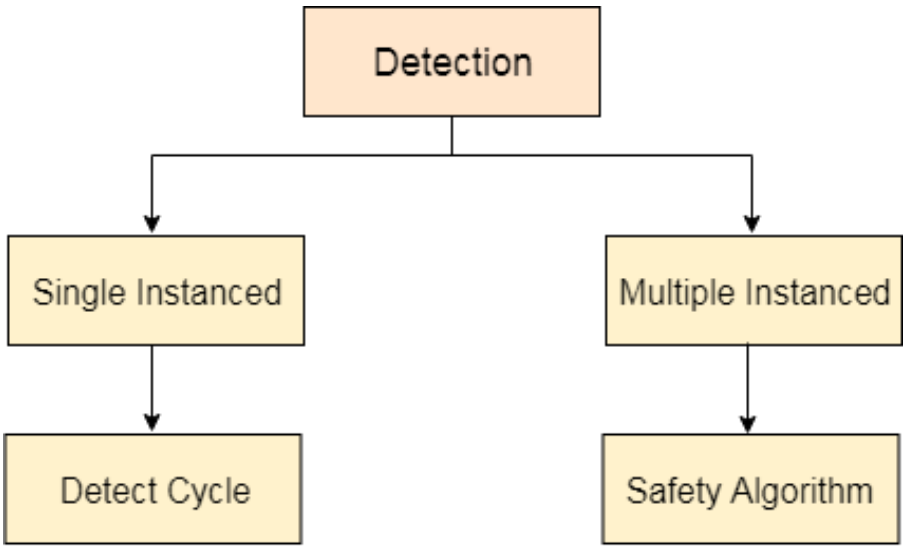
Click here to refer more info

18. Compare the various access matrix implementation techniques

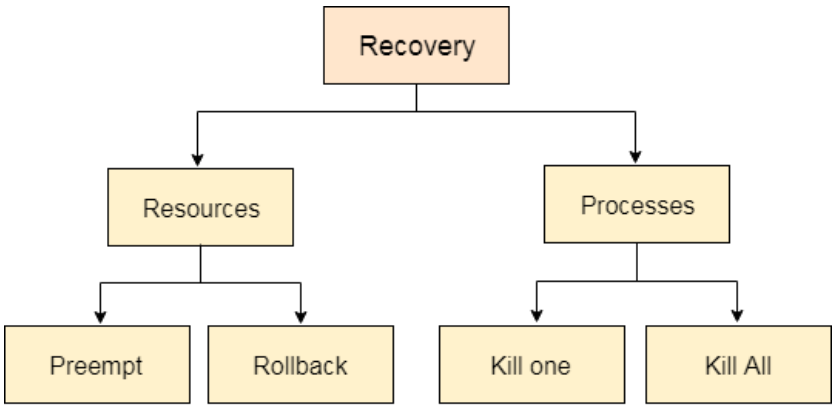
Refer Q.No.17 click here

19. Discuss the deadlock detection method in detail.

In this approach, The OS doesn't apply any mechanism to avoid or prevent the deadlocks. Therefore the system considers that the deadlock will definitely occur. In order to get rid of deadlocks, The OS periodically checks the system for any deadlock. In case, it finds any of the deadlock then the OS will recover the system using some recovery techniques.
The main task of the OS is detecting the deadlocks. The OS can detect the deadlocks with the help of a Resource allocation graph.



In single instanced resource types, if a cycle is being formed in the system then there will definitely be a deadlock.
On the other hand, in multiple instanced resource type graphs, detecting a cycle is not enough. We have to apply the safety algorithm on the system by converting the resource allocation graph into the allocation matrix and request matrix.
In order to recover the system from deadlocks, either OS considers resources or processes.



20. Explain various schemes to implement revocation for capabilities

With an access list scheme revocation is easy, immediate, and can be selective, general, partial, total, temporary, or permanent, as desired.
With capabilities lists the problem is more complicated, because access rights are distributed throughout the system. A few schemes that have been developed include:

- Reacquisition - Capabilities are periodically revoked from each domain, which must then re-acquire them.
- Back-pointers - A list of pointers is maintained from each object to each capability which is held for that object.
- Indirection - Capabilities point to an entry in a global table rather than to the object. Access rights can be revoked by changing or invalidating the table entry, which may affect multiple processes, which must then re-acquire access rights to continue.
- Keys - A unique bit pattern is associated with each capability when created, which can be neither inspected nor modified by the process.
    - A master key is associated with each object.
    - When a capability is created, its key is set to the object's master key.
    - As long as the capability's key matches the object's key, then the capabilities remain valid.

- The object master key can be changed with the set-key command, thereby invalidating all current capabilities.
- More flexibility can be added to this scheme by implementing a list of keys for each object, possibly in a global table.

PART C

1. Define Deadlock

A **deadlock** is a situation in which two computer programs sharing the same resource are effectively preventing each other from accessing the resource, resulting in both programs ceasing to function. The earliest computer operating systems ran only one program at a time.

2.Define resource.

(1) Generally, any item that can be used. Devices such as printers and disk drives are resources, as is memory.
(2) In many operating systems, including Microsoft Windows and the Macintosh operating system, the term resource refers specifically to data or routines that are available to programs.

3.List some resources that a process might need for its execution.

Resource Allocation of Operating System
When multiple jobs are running concurrently, resources must need to be allocated to each of them. Resources can be CPU cycles, main memory storage, file storage and I/O devices.

4.Explain safe state

A state is safe if the system can allocate resources to each process (up to its maximum) in some order and still avoid a deadlock. More formally, a system is in a safe state only if there exists a safe sequence.

5. Explain unsafe state

If there is no allocation sequence that allows the processes to finish executing, then the system is in an unsafe state. This is not equivalent to the "If the system is in an unsafe state, then there is no allocation sequence that allows the processes to finish executing.

6.Define the purpose of banker's algorithm.

By using the Banker's algorithm, **the bank ensures that when customers request money the bank never leaves a safe state**. If the customer's request does not cause the bank to leave a safe state, the cash will be allocated, otherwise the customer must wait until some other customer deposits enough.

For more info click here

7.Describe the techniques for recovery from deadlock.

Killing the process – Killing all the processes involved in the deadlock. Killing process one by one. After killing each process check for deadlock again keep repeating the process till the system recovers from deadlock. Killing all the processes one by one helps a system to break circular wait condition.

Resource Preemption – Resources are preempted from the processes involved in the deadlock, preempted resources are allocated to other

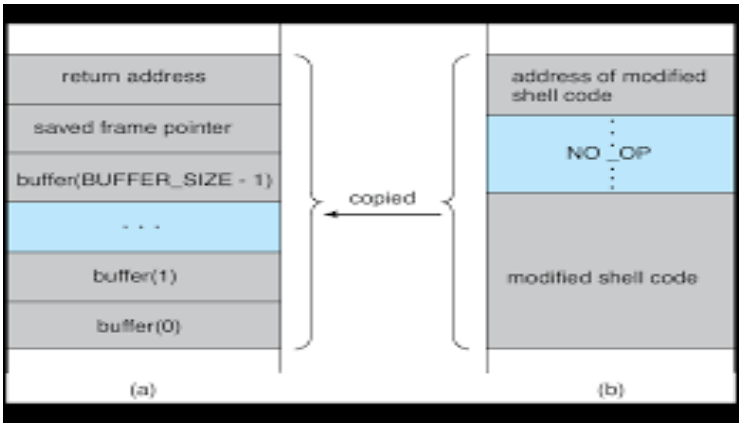
processes so that there is a possibility of recovering the system from deadlock. In this case, the system goes into starvation.

8.List the goals of protection.

Refer part b Q.no.13 click here

9. Describe any one language-based protection schemes.

In computer science, language-based security (LBS) is **a set of techniques that may be used to strengthen the security of applications on a high level by using the properties of programming languages**.



10. State principle of least privilege.

The principle of least privilege (POLP) is a concept in computer security that **limits users' access rights to only what are strictly required to do their jobs.** Users are granted permission to read, write or execute only the files or resources necessary to do their jobs.

11. Describe role-based access control.

Role-based access control (RBAC) **is a method of restricting network access based on the roles of individual users within an enterprise**. RBAC ensures employees access only information they need to do their jobs and prevents them from accessing information that doesn't pertain to them.

12. List the schemes that implement revocation of capabilities

Schemes that implement revocation for capabilities include the following:

• **Reacquisition**. Periodically, all capabilities are deleted from each domain. If a process wants to use a capability, it may find that that capability has been deleted. The process may then try to reacquire the capability. If access has been revoked, the process will not be able to reacquire the capability.

• **Back-pointers.** A list of pointers is maintained with each object, pointing to all capabilities associated with that object. When revocation is required, we can follow these pointers, changing the capabilities as necessary.

• **Indirection.** The capabilities point indirectly to the objects. Each capability points to a unique entry in a global table, which in turn points to the object. We implement revocation by searching the global table for the desired entry and deleting it. Then, when an access is attempted, the capability is found to point to an illegal table entry.

13. Explain the sequence in which a process may utilize the resources in normal mode of operation.

Under normal mode of operation, a process may utilize a resource in the following sequence: Request: If the request cannot be granted immediately,then the requesting process must wait until it can acquire the

resource. Use: The process can operate on the resource. Release: The process releases the resource.

14.Define the terms : a)object b)domain c) access right.

Object: Each domain defines a set of objects and the types of operations that may be invoked on each object.

Access right: The ability to execute an operation on an object.

Domain: A set of < object, { access right set } > pairs, as shown below. Note that some domains may be disjoint while others overlap.

15.Write the main differences between capability lists and access lists

An access list is a list for each object consisting of the domains with a nonempty set of access rights for that object. A capability list is a list of objects and the operations allowed on those objects for each domain.

16. Distinguish between deadlock avoidance and prevention strategies.

The main difference between deadlock prevention and deadlock avoidance is that deadlock prevention ensures that at least one of the necessary conditions to cause a deadlock will never occur while deadlock avoidance ensures that the system will not enter an unsafe state.

17. Why is deadlock state more critical than starvation?

Deadlock state is more critical than starvation because deadlock is said to happen when there is a limited resource but there are multiple processes in the CPU who are competing against each other to get that limited resource.

18. What are two options for breaking deadlock?

There are two options for breaking a deadlock. One is simply to abort one or more processes to break the circular wait. The other is to preempt some resources from one or more of the deadlocked processes.

19. What is meant by Starvation?

Starvation is **the problem that occurs when low priority processes get jammed for an unspecified time as the high priority processes keep executing.** A steady stream of higher-priority methods will stop a low-priority process from ever obtaining the processor.

20. Write the format of an access matrix.

Access Matrix is a security model of protection state in computer system. It is represented as a matrix. Access matrix is used to define the rights of each process executing in the domain with respect to each object. The rows of the matrix represent domains and columns represent objects.