Roll No. 2102086 | Total No. of Pages : 02

Total No. of Questions : 09

B.Tech. (Civil Engineering / Electrical Engineering / Electronics & Communication Engineering) (Sem.–6)

OPERATING SYSTEMS

Subject Code : BTCS402/18

M.Code : 79262

Date of Examination 21-05-2024

Time : 3 Hrs. | Max. Marks : 60

INSTRUCTIONS TO CANDIDATES :

1. SECTION-A is COMPULSORY consisting of TEN questions carrying TWO marks each.
2. SECTION-B contains FIVE questions carrying FIVE marks each and students have to attempt any FOUR questions.
3. SECTION-C contains THREE questions carrying TEN marks each and students have to attempt any TWO questions.

SECTION-A

1. **Write briefly :**

(i) Discuss access methods of a file.

(ii) Differentiate between physical and logical address space.

(iii) Discuss criteria used to measure CPU performance.

(iv) What is real time operating system? Explain briefly.

(v) Briefly explain the functions performed by an Operating System.

(vi) What is multithreading?

(vii) Discuss directory structure.

(viii) What is dining philosopher problem?

(ix) Discuss EDF real time scheduling method.

(x) What is the importance of PCB (Process Control Block)?

(S2) 962

SECTION-B

2. What do you understand by a process? Draw the state transition diagram and explain the purpose of each state.

3. Explain with example FCFS and Round Robin scheduling algorithms.

4. Compare static and dynamic contiguous partitioned memory management schemes.

5. What are semaphores? How can they be used to implement mutual exclusion?

6. Explain Multi programming and Time Sharing operating systems.

SECTION-C

7. What is a deadlock? Explain necessary conditions for deadlock occurrence. Discuss any method used for deadlock avoidance with example.

8. What do you mean by virtual memory? How it is implemented? Explain-various techniques used to manage the virtual memory.

9. What do you mean by disk scheduling? Explain in detail various disk scheduling algorithm with the help of suitable example.

NOTE : Disclosure of Identity by writing Mobile No. or Making of passing request on any page of Answer Sheet will lead to UMC against the Student.

Operating System

Solved Question paper 2024

Section - A

Q1. (i) Discuss access methods of a file.

Ans:- File Access Methods :- It determine how data is organized, retrieved, and modified within a file system. There are three primary methods:

1.) Sequential Access :- Files are processed in order, one record after another.

→ Commonly used by editors & compilers.

2.) Direct Access :- files consist of fixed-length logical records.

→ Allows reading & writing records in any order.

3.) Index Sequential Method :- Combines sequential & direct access.

→ An index is created for the file, containing pointers to various blocks.

---

ii) Differentiate between physical & logical address space.

Ans:-

	Physical Address space	Logical Address space
Definition	Actual address in memory Hardware.	Address generated by the CPU during a program execution.
Visibility	Visible to the memory unit & used to access the physical RAM	Used by CPU & translated to physical address by the MMU.

	Physical Address space	Logical Address space
Generation	Directly generated by the Hardware.	Generated by CPU & translated by the operating system.
Example	If the physical address of a byte in RAM is 0x1000, then this is the actual location in memory hardware.	If the logical adress is 0x0010, the MMU might translate it to a physical address like 0x1000 in the memory.

(iii) Discuss criteria used to measure CPU performance.

Ans:- Criteria to Measure CPU performance

1.) Clock Speed :- Higher clock speed means more instructions per second.
→ Measured in GHz.

2.) Instructions Per Cycle :- Number of instructions executed per clock cycle.
→ Higher Instructions per cycle improves performance

3.) Throughput :- Number of processes completed per unit time.
→ Higher throughput indicates better performance.

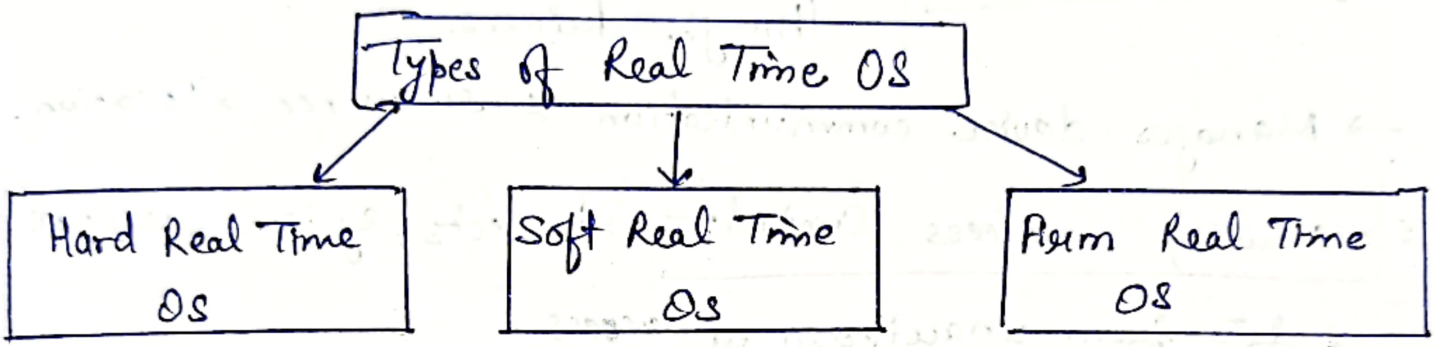
4.) Latency :- Time taken to complete a single task.
→ Lower latency means faster task completion.

5.) Cache performance :- Efficiency of CPU cache memory.
→ Lower cache miss rates improve performance.

iv) What is real time Operating system? Explain briefly.

Ans:- Real-Time Operating System :-

→ Used in environments with many external events needing quick processing.
→ processing time measured in tenths of seconds.
→ Must meet fixed deadlines to avoid system failure.
→ Applications:- Flight Control, Airline traffic Control etc.



→ Hard Real Time OS :- Guarantees critical tasks are completed within strict time constraints.

→ Soft-Real Time OS:- Provides some flexibility in time limits, with priority-based scheduling.

→ Firm RealTime OS :- Deadlines must be met, with minor impacts if missed, affecting product quality.

V) Briefly explain the functions performed by an Operating system.

Ans:- Functions of an OS :-

1) Process Management :- Manages creation, scheduling & termination of processes.
→ Ensures efficient execution & multitasking.

2) Memory Management :- Allocates & deallocates memory space as needed by programs.
→ Manages memory hierarchy & virtual memory.

3) File System Management :- Controls creation, deletion, & access of files.
→ Manages storage, retrieval & data integrity.

4) Device Management :- Handles input / output devices through drivers.
→ Manages device communication & resource allocation.

5) Security & Access Control :- Protects system resources & data from unauthorized access.
→ Implements authentication & encryption mechanisms

6) User Interface :- Provides user interface (CLI or GUI) for interaction.
→ Facilitates command execution & user friendly operation

---

vi) What is multithreading ?

Ans :- Multithreading :-
→ Allows a program to perform multiple tasks simultaneously.
→ Each task, called as thread, works concurrently to complete parts of a job faster.
→ Improves performance for applications like web browsers, games & more.

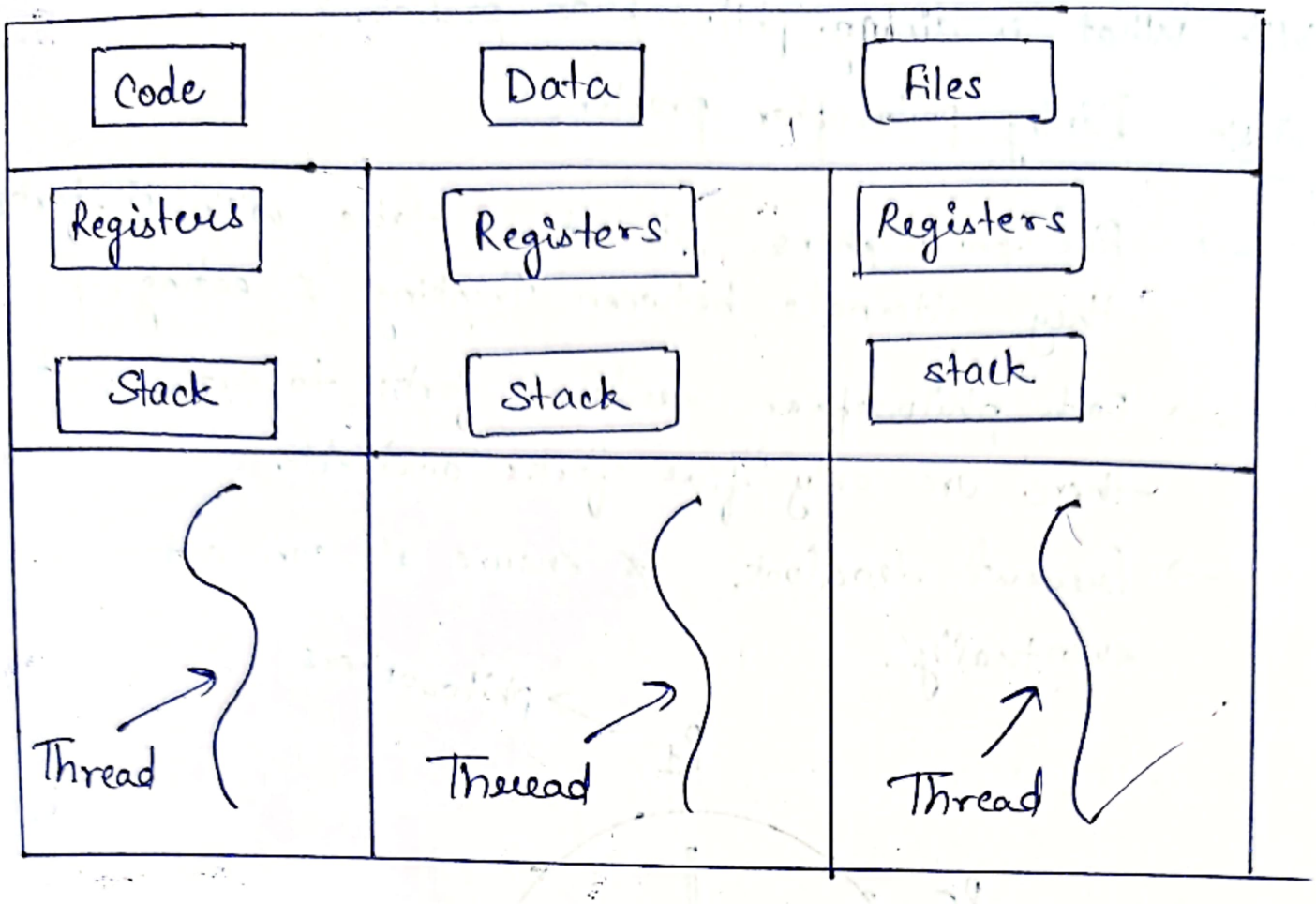


fig:- Multithreaded

VII) Discuss Directory structure.

Ans:- Directory structure:-

→ A directory is a container that is used to contain folders & files.

→ It organizes files & folders in a hierarchical manner.

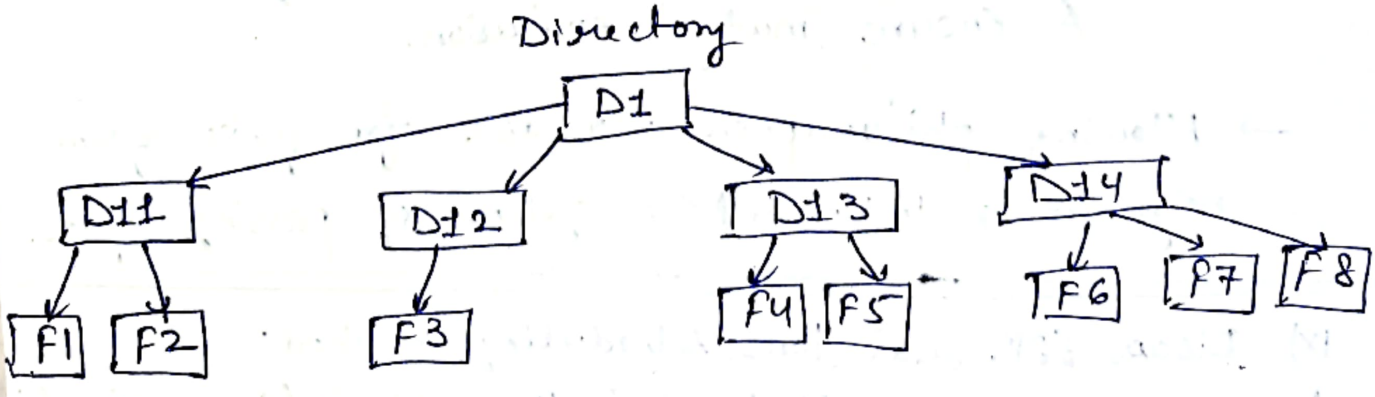


fig:- Directory structure

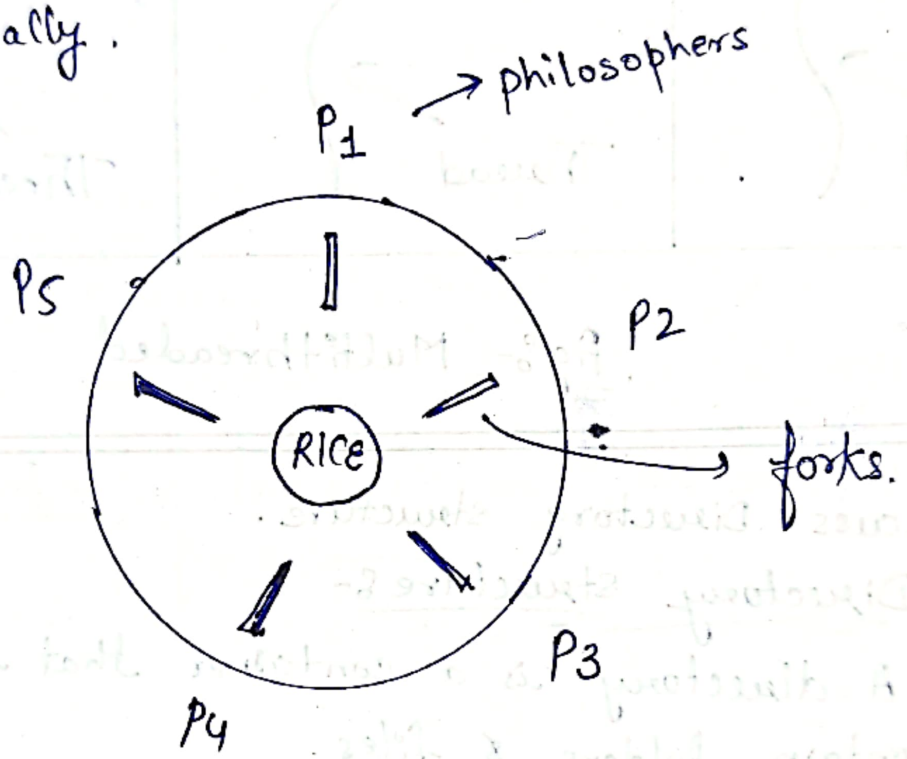
VIII) What is dining philosopher problem?

Ans:- Dining philosopher problem

→ Five philosophers sit at a table with 5 forks, they alternate between thinking & eating.

→ Each philosopher needs 2 forks to eat, but there are only five forks available.

→ Prevent deadlock & ensure all can eat eventually.



→ Solution :- Semaphore-based Approach :-
Use semaphores to control fork access & ensure mutual exclusion.

→ Allowing philosophers to ask for forks from neighbors & implement a system of passing forks.

IX) Discuss EDF real time scheduling method.

Ans:- Earliest Deadline First Real time scheduling method :-

→ Optimal dynamic priority scheduling for real-

time systems.

→ Based on absolute deadline; closest deadline gets higher priority.

→ Tasks do not need to be periodic; require fixed CPU burst time.

→ Allows preemption if a task with an earlier deadline becomes runnable.

→ Limitations :- Resource sharing Problem
→ Efficient Implementation Problem.

X) What is importance of PCB (Process Control Block)?

Ans:- Importance of PCB :-

→ Process Management :- Stores information about each process, including process state, program counter, & registers.

→ Resource allocation :- Manages resources allocated to processes, such as CPU time, memory, & I/O devices.

→ Context switching :- Facilitates switching between processes efficiently by storing & restoring execution context.

→ Scheduling :- Holds scheduling information like process priority & scheduling state.

Section- B

Q2. What do you understand by a process? Draw the state transition diagram & explain the purpose of each state.

Ans:- **Process** :- An instance of a program in execution, comprising code, data & resources.

→ Has its own memory space, registers, & execution state.

→ Managed by the operating system scheduler for efficient CPU utilization.

State transition :-

The process can be in any one of the following three possible states.

1.) Running (actually using the CPU at that time & running).

2.) Ready (runnable; temporarily stopped to allow another process run).

3.) Blocked (unable to run until some external event happens).

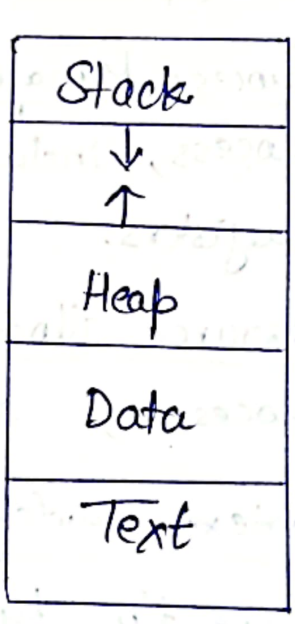


Fig:- Process in the Memory

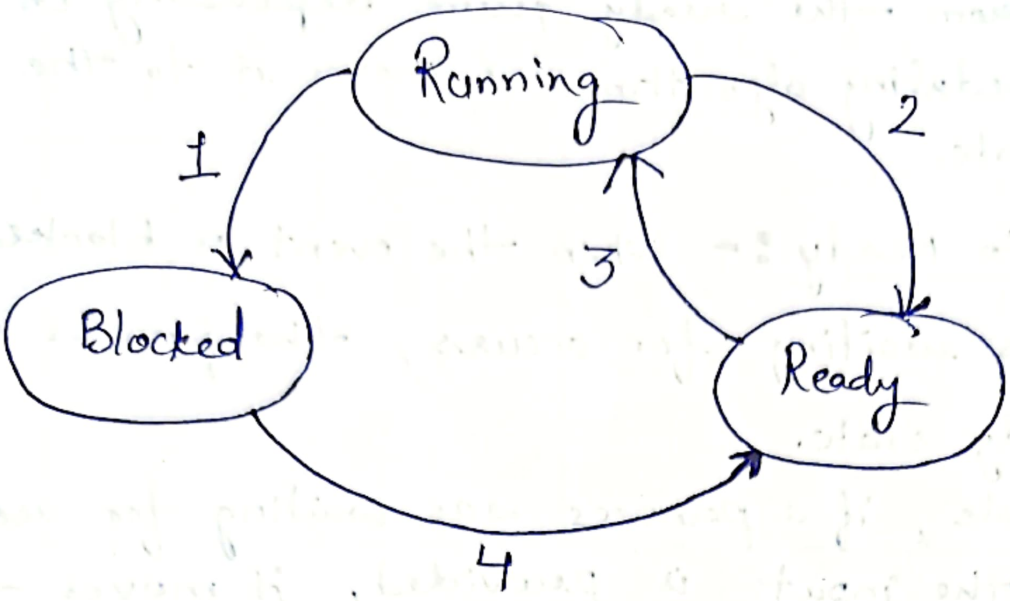


Fig :- State transition.

1> Running to Blocked :- When a process needs to wait for an event to occur (I/O operation or System Call), it moves to the blocked state.

→ for Example, if a process needs to wait for user input, it moves to the blocked state until the user provides the input.

2> Running to Ready :- when a running process is preempted by the OS, it moves to the ready state.

→ for Example: if a higher priority process becomes ready, the OS may preempt the running process & move it to the ready state.

3> Ready to Running :- When the CPU become available, the OS selects

a process from the ready queue depending on various scheduling algorithms & moves it to the running state.

4.) Blocked to Ready :- When the event a blocked process was waiting for occurs, the process moves to the ready state.

→ For example, if a process was waiting for user input & the input is provided, it moves to the ready state.

Q3. Explain with example FCFS & Round Robin scheduling algorithms.

Ans:- FCFS :- First Come First Serve :-

→ Non preemptive scheduling algorithm where the first process arriving at the CPU is allocated first.

→ Processes are executed in the order they arrive in the ready queue.

→ Disadvantages :- Can lead to long average waiting times.

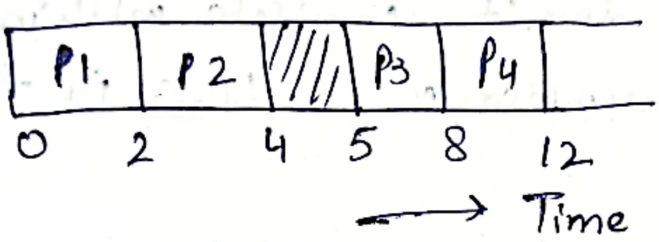
Example; FCFS:-

Criteria → Arrival Time
Mode → Non Preemptive

→ Consider the following processes P1, P2, P3, P4, .., arrival time Burst time.

Process No.	Arrival Time	Burst Time	Completion Time	TAT	WT	RT
P1	0	2	2	2	0	0
P2	1	2	4	3	1	1
P3	5	3	8	3	0	0
P4	6	4	12	6	2	2

Gantt Chart



Turn Around Time (TAT) = Completion time - Arrival time

Completion time can be calculated by Gantt chart. like Completion time for P1 process is 2 & for P2 process is 4 and so on.

$(TAT)_{P1} = 2 - 0 = 2$

$(TAT)_{P2} = 4 - 1 = 3$ ....

→ Waiting Time (WT) = TAT - Burst time

→ $(WT)_{P1} = 2 - 2 = 0$

→ Response time = Time at which CPU was supposed to be given to a process - Arrival time

→ Avg. TAT = $\frac{6+3+3+2}{4} = \frac{14}{4} = 3.5$.

ii) Round Robin scheduling :-

→ Preemptive CPU scheduling algorithm.
→ Each process is assigned a fixed time slice (quantum) to execute.
→ Processes are executed in a circular manner.
→ If a process doesn't complete within its time slice, it's preempted & put back in the ready queue.
→ Disadvantage :- May result in higher context switch overhead.

For Example :-
Consider a following processes P1, P2, P3, P4 & Arrival time & burst time.

Process No	Arrival Time	Burst time	Completion time	TAT	WT	RT
P1	0	5	12	12	7	
P2	1	4	11	10	6	
P3	2	2	6	4	2	
P4	4	1	9	5	4	

Given Time Quantum = 2
Criteria = Time Quantum
Mode = Preemptive

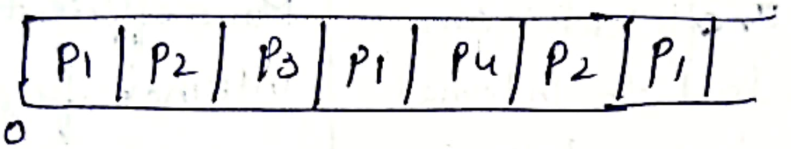
TAT = CT - AT
WT = TAT - BT

→ P1 will run for Time Quantum = 2 time units. then return to Ready State then P2 will run for TQ = 2 time unit then return to Ready state.

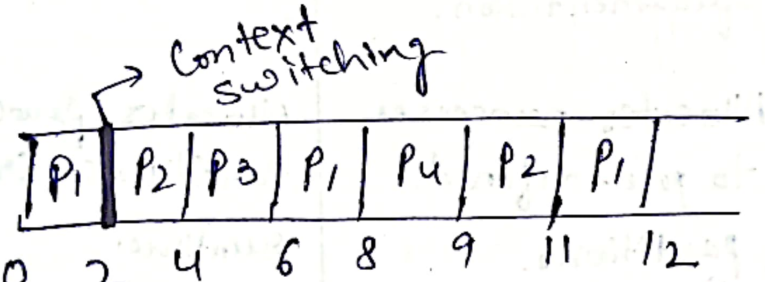
→ P1 will repeat this till its service gets completed. (Burst time).

→ Sequence of processes Ready queue.

Ready Queue



Gantt chart



→ Process P1 needs 5 time unit to complete its execution but in this algorithm as it is preemptive scheduling, there is Time Quantum of 2 time unit is there. i-e one process will maximum get 2 time unit for its execution

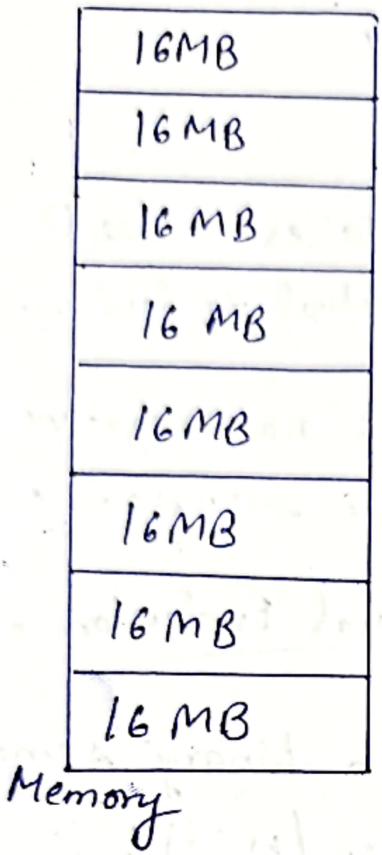
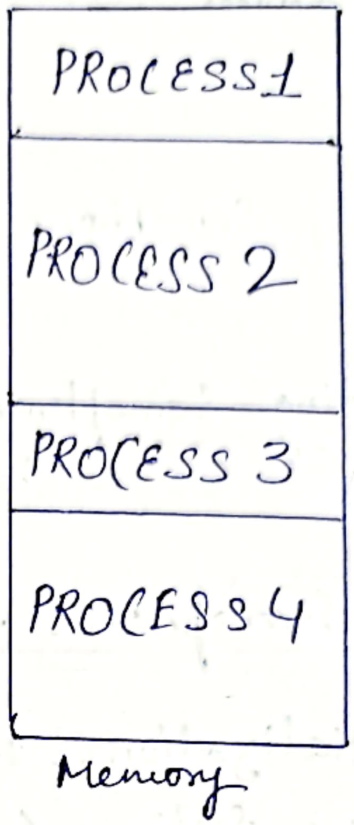
→ If the process is completed then next process will be there. But if like P1 process is left with 3 time unit after one time execution. It will be there in ready queue again as per round robin algorithm & completes its execution in this manner.

→ Likewise, all other processes will get executed.

Q4. Compare static & dynamic contiguous partitioned memory management schemes.

Ans:-

Feature	Static Partitioned	Dynamic Partitioned
1.) Partition Size	Fixed size partitions	Variable size partitions
2.) Flexibility	Low flexibility; difficult to accommodate varying process sizes	High flexibility; partitions can adapt to process sizes.
3.) Memory Utilization	Inefficient; can lead to internal fragmentation.	More efficient; reduces internal fragmentation.
4.) Process Allocation	Allocates processes to pre-defined partitions.	Allocates processes to partitions created at runtime.
5.) Complexity	Simpler to implement	More complex; require dynamic allocation & deallocation.
6.) Fragmentation type	Internal fragmentation	External fragmentation.
7.) Process	Only one process can be placed in a partition	The process is allocated a chunk of free memory.
8.) Degree of multi-programming	Degree of multi-programming is less	Degree of multi-programming is higher.

Feature	Static Partitioning	Dynamic Partitioning
9.) Example		

Q5. What are semaphores? How can they be used to implement mutual exclusion?

Ans:- Semaphores :- Semaphores are synchronization Primitives used to manage concurrent processes by signaling.

→ They are used to enforce mutual exclusion, avoid race conditions, & implement synchronization between processes.

→ The process of using Semaphores provides two operations: wait (P) & signal (V).

→ Semaphores are used to implement critical sections, which are regions of code that must be executed by only one process at a time. By using semaphores, processes can coordinate access

to shared resources, such as shared memory or I/O devices.

Types :-

1.) Binary Semaphore :- Only takes values 0 & 1, used for mutual exclusion.

2.) Counting Semaphore :- Takes non-negative integer value used for resource counting.

Using Semaphores for Mutual Exclusion :-

1.) Initialization : Initialize a binary semaphore to 1.

2.) Entry Section (P operation / wait) :

→ A process wishing to enter the critical section performs a 'wait()' operation on the semaphore.

→ If the semaphore value is 1, it decrements to 0, allowing the process to enter.

→ If the semaphore value is 0, the process is blocked until the semaphore is available.

3.) Critical section : The process executes its critical section code.

4.) Exit Section (V operation / signal) :

→ After leaving the critical section, the process performs a 'signal()' operation on the semaphore.

→ This increments the semaphore value to 1, waking up one of the blocked processes (if any).

Q6. Explain Multi-programming & Time-sharing operating systems.

Ans:- Multi-programming O.S :- Allows multiple programs to reside in memory simultaneously & share the CPU.

→ Maximize CPU utilization by organizing jobs so that the CPU always has one to execute.

→ When a running program needs to wait (eg. for I/O), another program can use the CPU.

→ Increases CPU efficiency & system throughput.

→ Response time can also be reduced.

→ Disadvantages :- It provide an environment in which various systems resources are used efficiently, but they do not provide any user interation with the computer system.

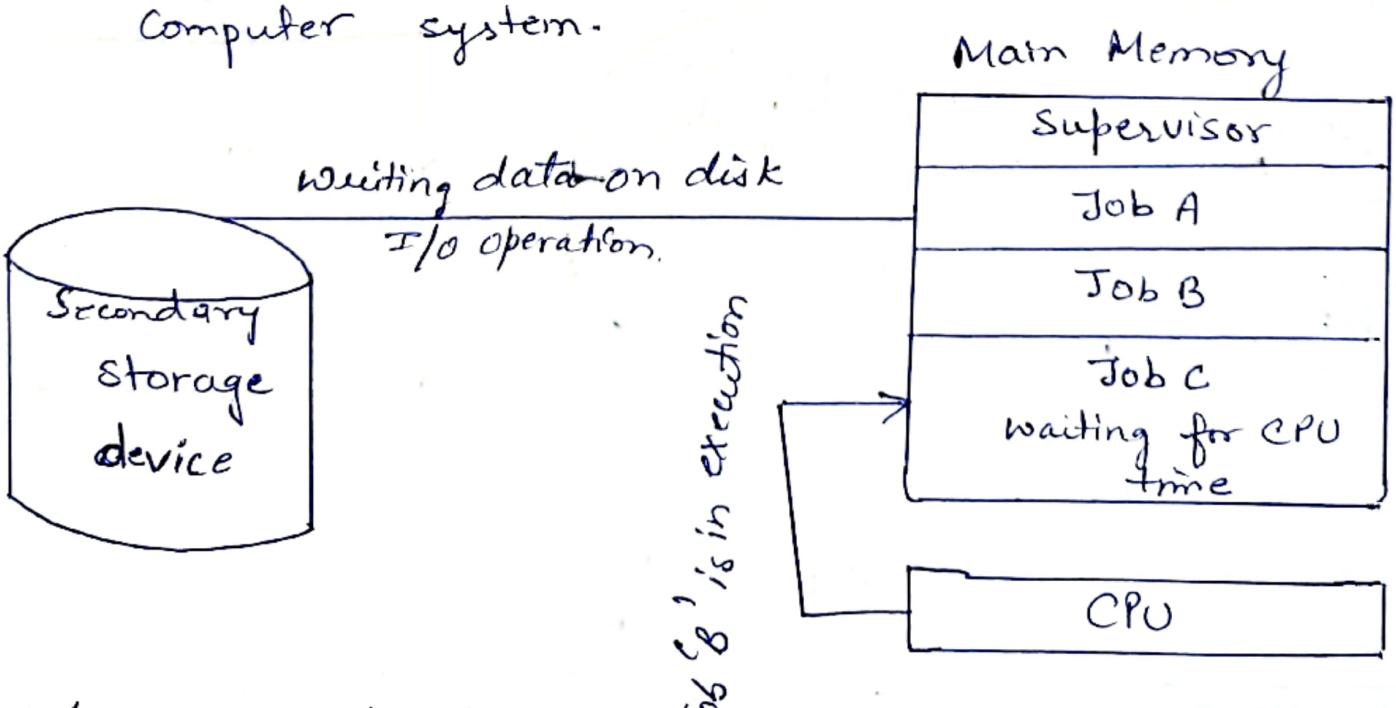


Fig:- Multiprogramming OS.

Time-Sharing OS:- An extension of multiprogramming OS where multiple users can interact with the system simultaneously.

→ Provide quick response times by allocating CPU time slices to each user or task.

→ CPU time is divided into small units (time slices), & each user gets a fair share of the CPU.

→ Enhances user interaction, supports multiple users, & provides immediate feedback.

→ These systems are also known as Multi-tasking systems.

→ **Drawbacks:-** Overhead due to Content switching & managing multiple users.

→ Data transmission rates are very high in comparison to other methods.

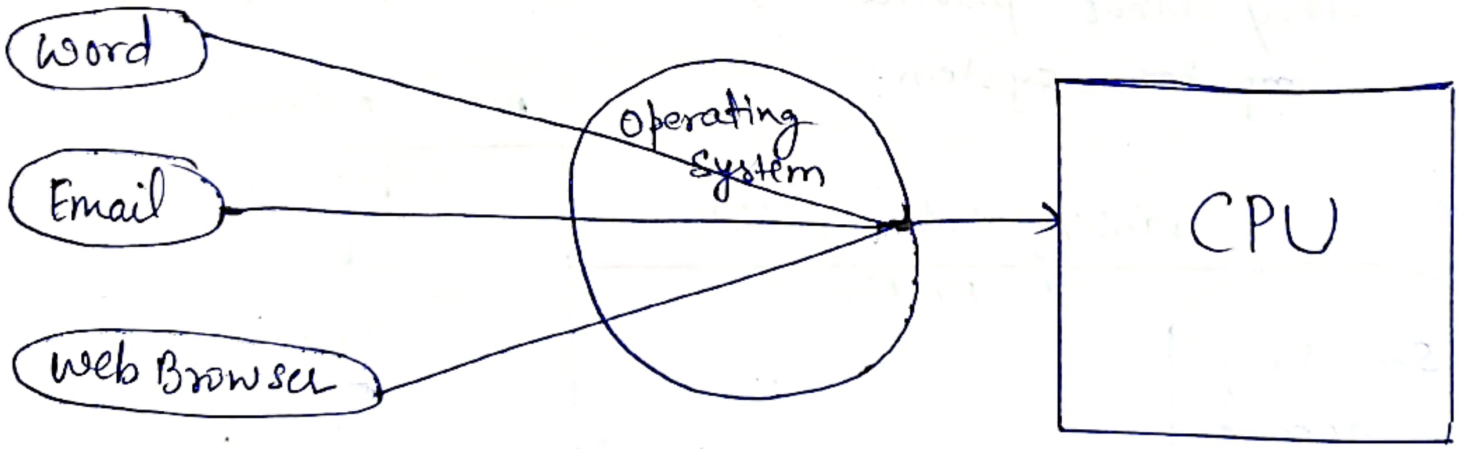


Fig:- Time-Sharing OS

Section – C

Q7 What is a deadlock? Explain necessary conditions for deadlock occurrence. Discuss any method used for deadlock avoidance with example.

Ans:- **Deadlock** :- Deadlock is a situation in Computing where two or more processes are unable to proceed because each is waiting for the other to release resources.

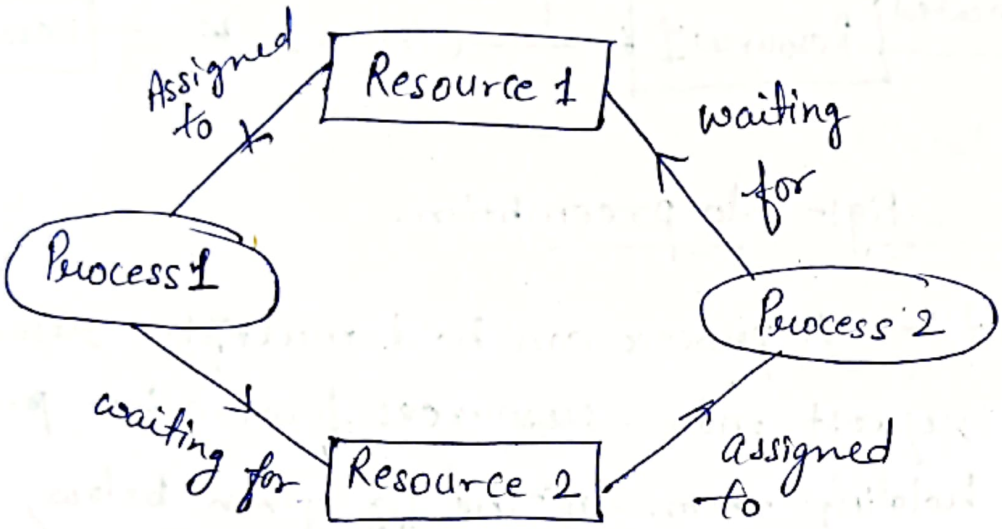


Fig:- Deadlock

Necessary Conditions for deadlock occurrence :-

1) **Mutual Exclusion** :- There should be a resource that can only be held by one process at a time. In the diagram below, there is a single instance of Resource 1 & it is held by Process 1 only.

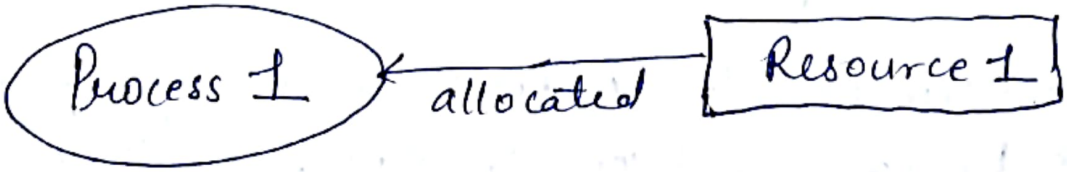


Fig:- Mutual exclusion.

2.) No preemption :- A resource cannot be preempted from a process by force. A process can only release a resource voluntarily. In the diagram below, Process 2 cannot preempt Resource 1 from Process 1. It will only be released when Process 1 relinquishes its voluntarily after its execution is complete.

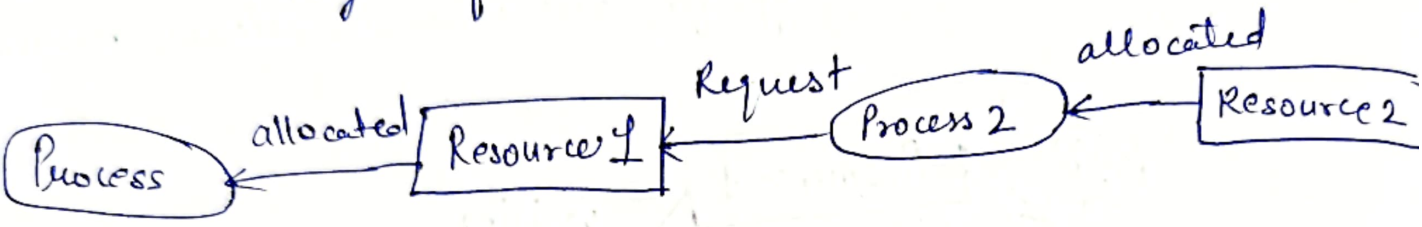


Fig :- No preemption.

3.) Hold & wait :- A process can hold multiple resources & still request more resources from other processes which are holding them. In the fig given below, Process holds Resource 2 & Resource 3 & is requesting the Resource 1 which is held by Process 1.

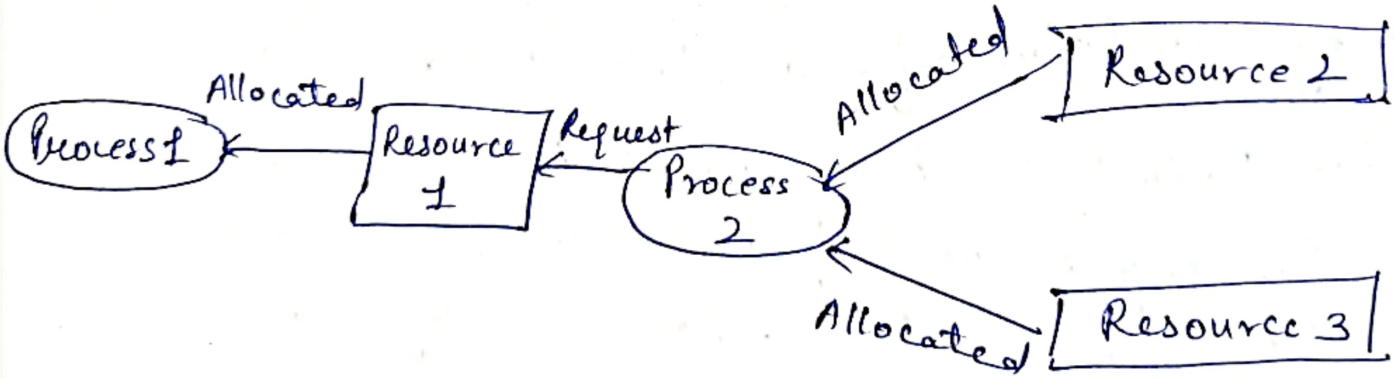


Fig :- Hold & wait

4.) Circular wait :- A process is waiting for the resource held by the second process, which is waiting for the resource held by the third process & so on, till

the last process is waiting for a resource held by the 1st process. This forms a circular chain. for example: Process 1 is allocated Resource 2 & it is requesting Resource 1. similarly, Process 2 is allocated Resource 1 & it is requesting Resource 2. This forms a circular wait loop.

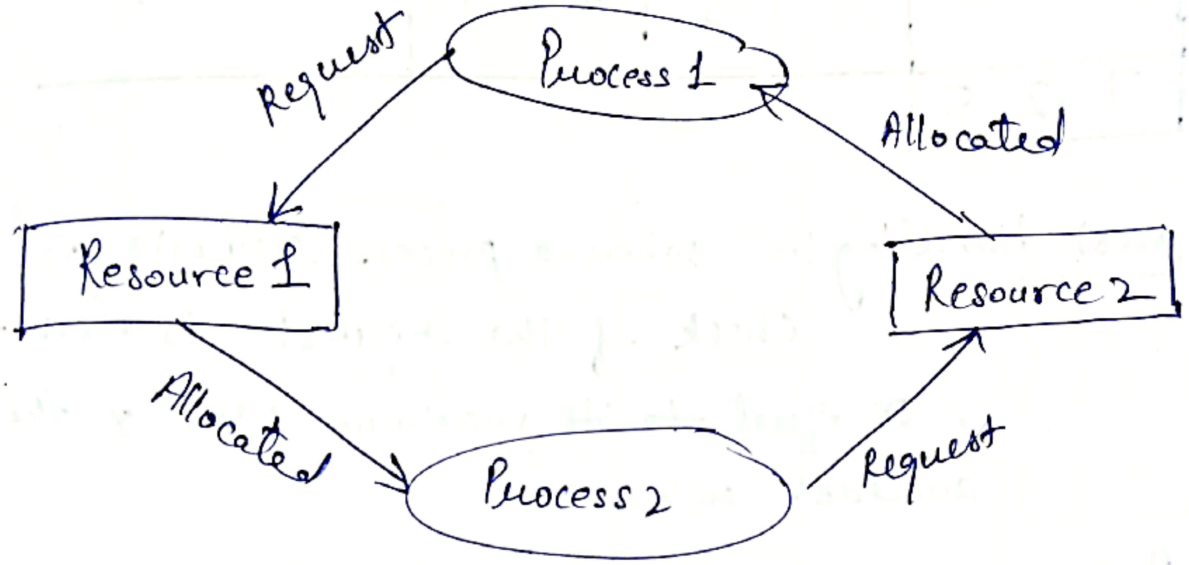


Fig:- Circular wait.

Method used for Deadlock avoidance

1) Banker's Algorithm :- Ensure a system remains in a safe state by allocating resources only if it doesn't lead to a deadlock.

Consider, Total A = 10, B = 5, C = 7 Resources &

Processes (P1, P2, P3, P4, P5), Allocation, Max Need.

Process No.	Allocation A B C	Max Need A B C	Available A B C	Remaining Need A B C
P1	0 1 0	7 5 3	3 3 2	7 4 3
P2	2 0 0	3 2 2	5 3 2	1 2 2
P3	3 0 2	9 0 2	7 4 3	6 0 0
P4	2 1 1	4 2 2	7 4 5	2 1 1
P5	0 0 2	5 3 3	7 5 5	5 3 1
	7 2 5			

→ Request handling :- when a process requests resource check if the request is less than or equal to its maximum need & the available resource.

→ Pretend Allocation :- Temporily allocate the requested resources to the process & update the 'Available', Allocation & Max need.

→ Perform a safety check to see if the system remains in a safe state after the allocation.

Q8. What do you mean by virtual memory? How it is implemented? Explain - Various techniques used to manage the virtual memory.

Ans:- **Virtual Memory :-** It is a memory management technique that creates an "illusion" of a large logical memory for users by using hardware & software to manage & extend the physical memory.

Advantages :- → Unlimited address space
→ Increased degree of multiprogramming
→ Reduced I/O.

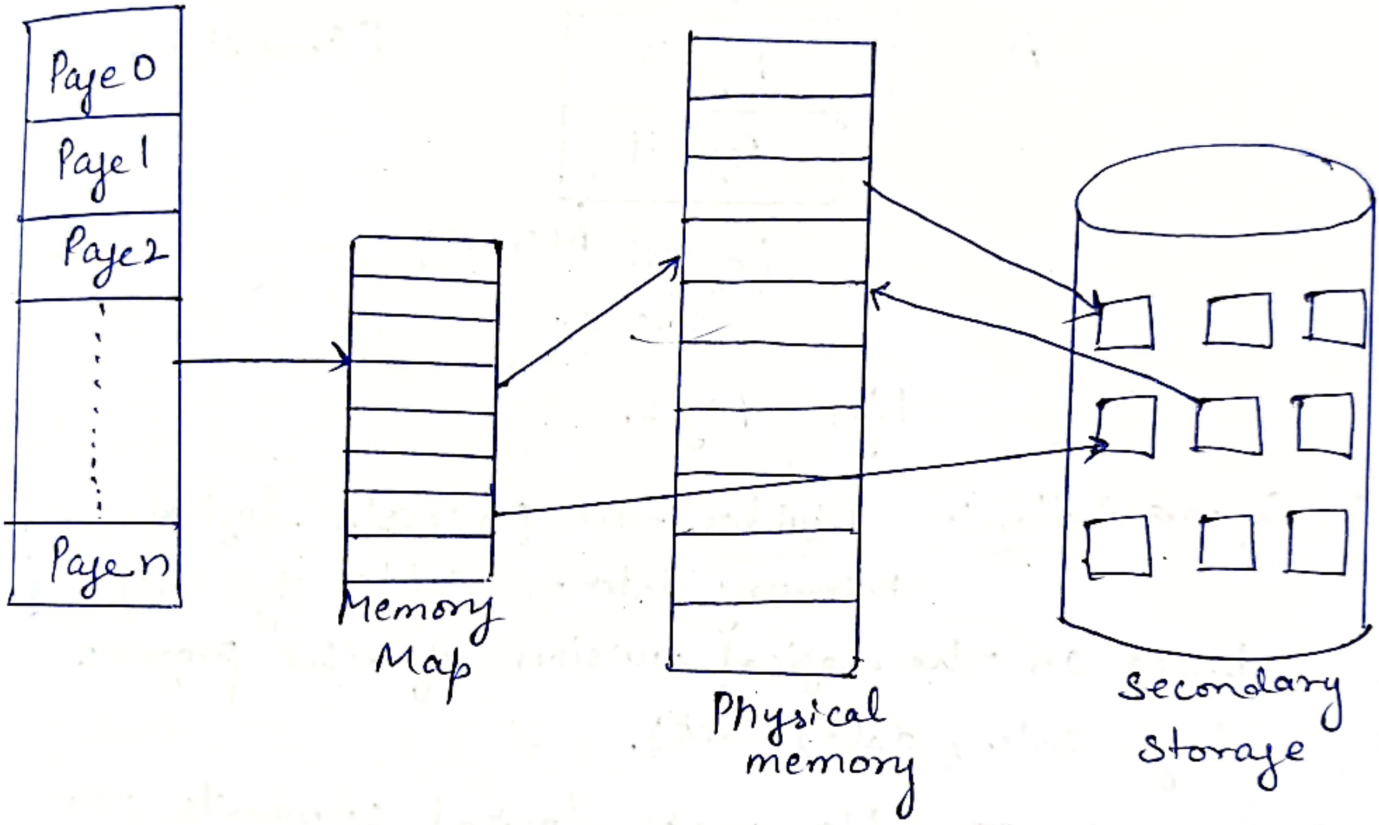


fig :- Virtual memory is larger than physical memory

Implementation Of Virtual Memory :-

1. **Paging :-** Divides the process's logical memory into fixed-size blocks called pages.

→ Divides physical memory into fixed sized blocks frames.

→ A page table maps logical pages to physical frames.
→ Provides efficient & flexible memory allocation, avoiding fragmentation.

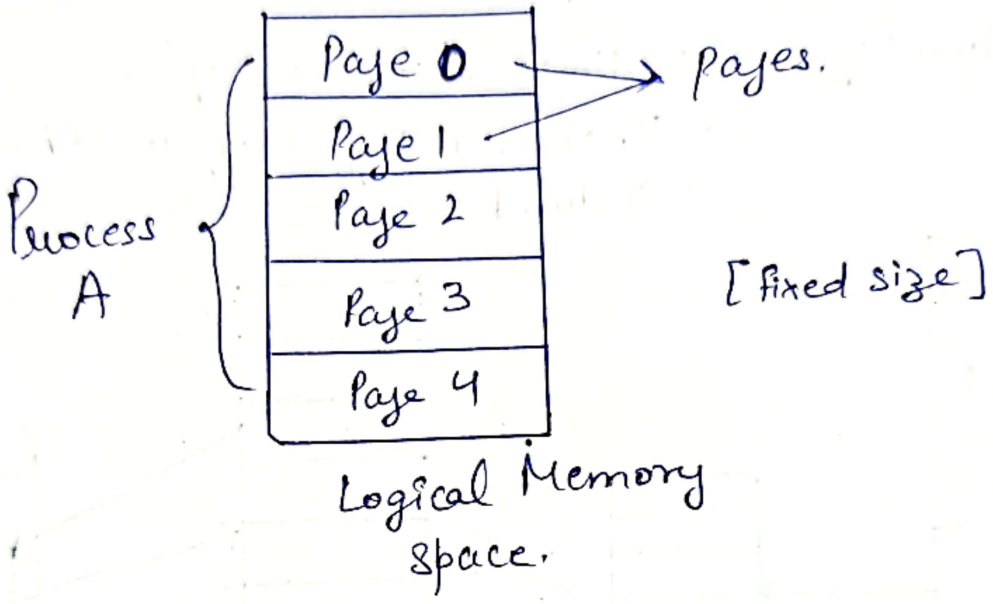


Fig:- Pages.

2) Segmentation:- Divides the process's logical memory into variable size segments based on the logical division of the program (eg, code, data, stacks).

→ A segment table maps logical segments to physical memory.
→ Provides better support for the programmer's views of memory but can suffer from external fragmentation.

Techniques Used to Manage Virtual Memory:-

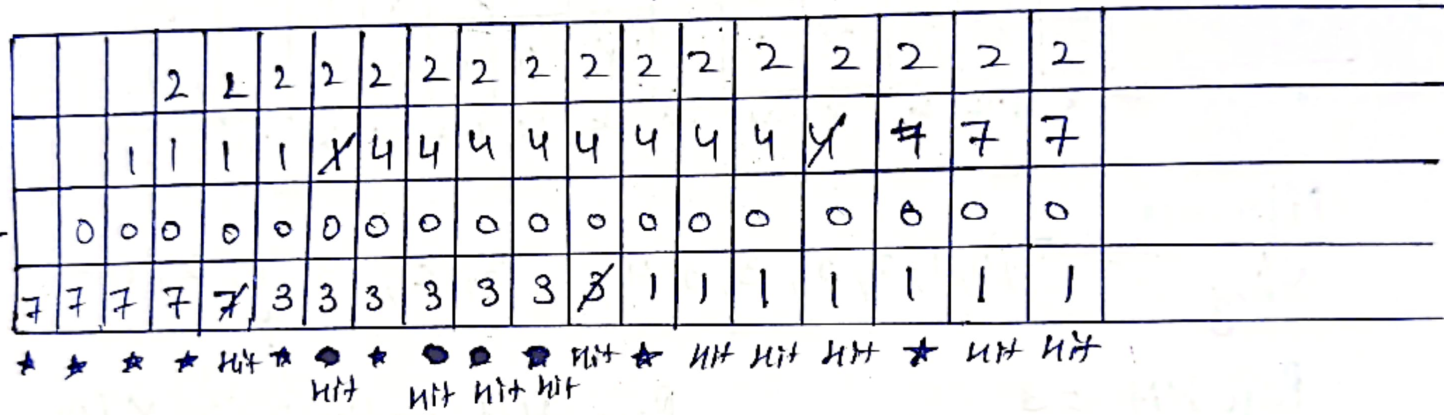
1) Demand Paging:- Pages are loaded into memory only when they are needed.

→ Reduces the initial loading time & memory usage.
→ Involves page replacement strategies when memory is full.

2) Page Replacement Algorithms :-

i) Optimal Page Replacement (Replace the page which is not required in longest dimension of time in future.

Consider 4 frames. F1, F2, F3 & F4.
& the reference string.



Reference String :- 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0,

→ Page fault :- Occurs when a requested page is not in memory, requiring it to be loaded from disk.

⇒ 8

→ Page Hit :- Occurs when a requested page is already in memory, allowing immediate access.

=) 12

Page Hit Ratio :- $\frac{12}{20} \times 100^{5} = 60\%$.

Page Fault Ratio :- $\frac{8}{20} \times 100^{5} = 40\%$.

2.) FIFO (first In first Out) :-

F3			1	1	1	1	0	0	0	3	3	3	3	2	2
F2		0	0	0	0	3	3	3	2	2	2	2	1	1	1
F1	7	7	7	2	2	2	2	4	4	4	0	0	0	0	0
	*	*	*	*	Hit	*	*	*	*	*	*	Hit	*	*	Hit

Reference string :- 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 1, 2, 0

Page Hit = 3

Page Fault :- 12

Page Hit Ratio = $\frac{3}{15} \times 100$ = 20%.

Page fault Ratio = $\frac{12}{15} \times 100$ = 80%.

3.) ~~boost~~ Thrashing :- Occurs when a system spends more time swapping pages in & out of memory than executing processes.

→ Can be managed by adjusting the degree of multiprogramming or using working set models

Q9. What do you mean by disk scheduling? Explain in detail various disk scheduling algorithms with the help of suitable example.

Ans:- **Disk scheduling**:- Disk scheduling refers to the method by which the OS decides the order in which disk I/O requests are serviced to improve efficiency & performance.

→ Goal of this algorithms is to minimize the seek time.

→ **Seek Time**:- Time taken to reach upto desired track.

→ Hard disk is a collection of platters.

Platter
↓
Surface
↓
Track
↓
Sector

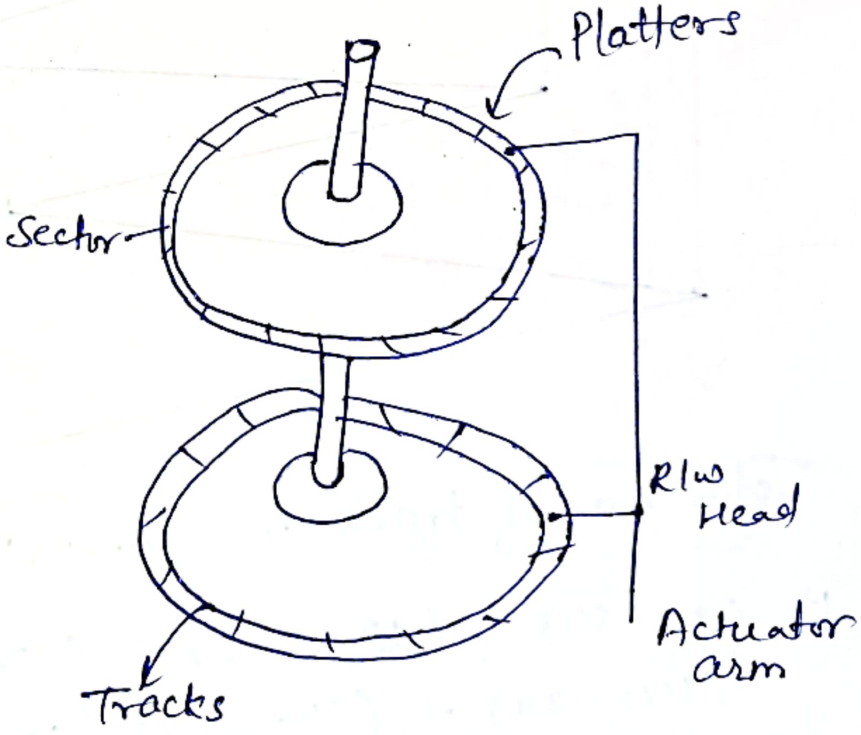


Fig:- Disk

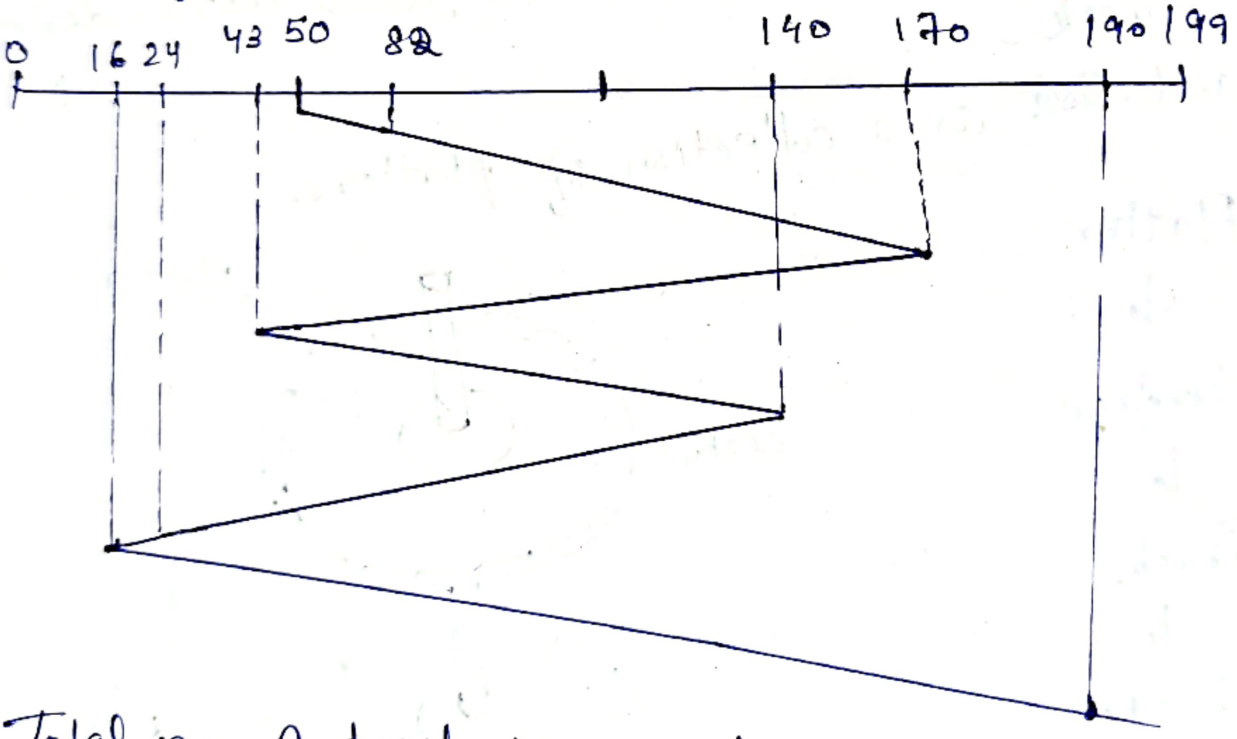
Various Disk scheduling Algos:-

1) FCFS :- First Come First Served :-

→ Processes requests in the order they arrive.

→ Simple but can lead to long wait times.

Ques- A disk contains 200 tracks = 0-199
Request queue contains track No. 82, 170, 43, 140, 24, 16, 190 respectively.
Current position of RW Head = 50
Calculate ~~total~~ no. of tracks movement by RW.



Total no. of track movements :-

1) (82-50) + (170-82) + (170-43) + (140-43) + (140-24) + (24-16) + (190-16)

2) (170-50) + (170-43) + (140-43) + (140-16) + (190-16) = 642.

2) Shortest Seek Time first (SSTF):-

→ select the request with the shortest seek time from the current head position,

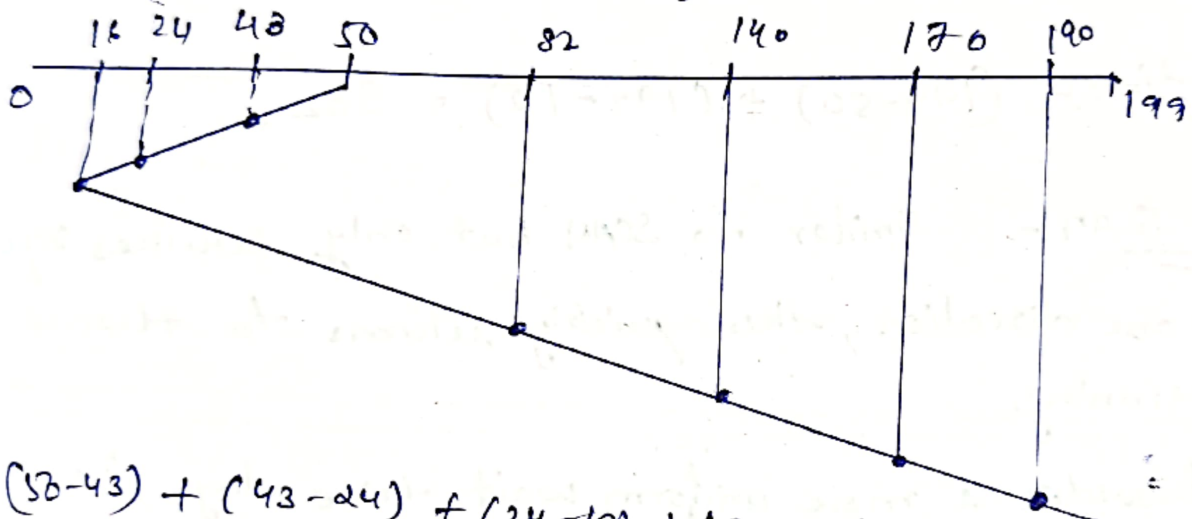
→ Reduces overall seek time but may cause starvation of distant requests.

Example:- Consider, a track = 0-199.

Requesting queue track no. 82, 170, 43, 140, 24, 16, 190 :

Current position of R/W Head = 50.

Calculate total no. of track movement.



1) (50-43) + (43-24) + (24-16) + (82-16) + (140-82) + (170-140) + (190-170).

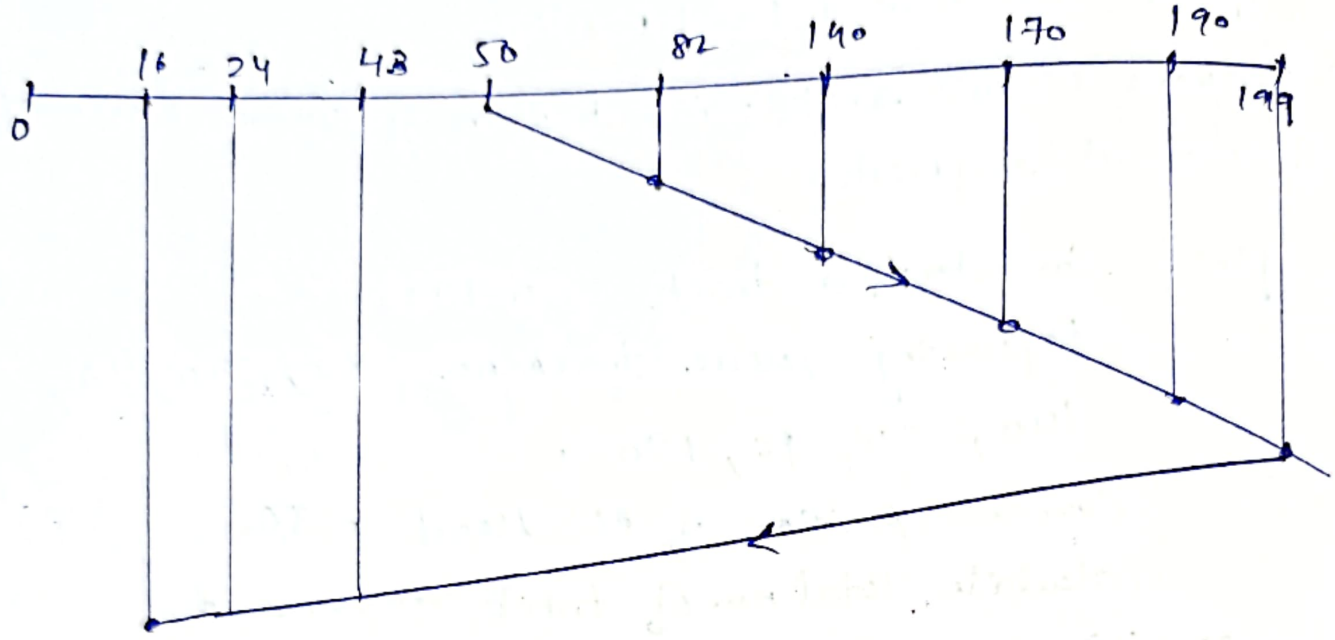
2.) (50-16) + (190-16) = 208

3.) SCAN Algorithm:- Disk arm moves in one direction servicing requests unit it reaches the end, then reverse direction.

→ Provides a more uniform wait time compared to SSTF.

Example:- Track = 0-199, current position = 50

Requesting queue track: 82, 170, 43, 140, 24, 16, 190.



Seek time:- (199-50) + (199-16) = 332.

4.) C-SCAN:- Similar to SCAN but only services request in one direction, then quickly returns to the beginning.

→ Provides a more uniform wait time by treating the disk as a circular list.

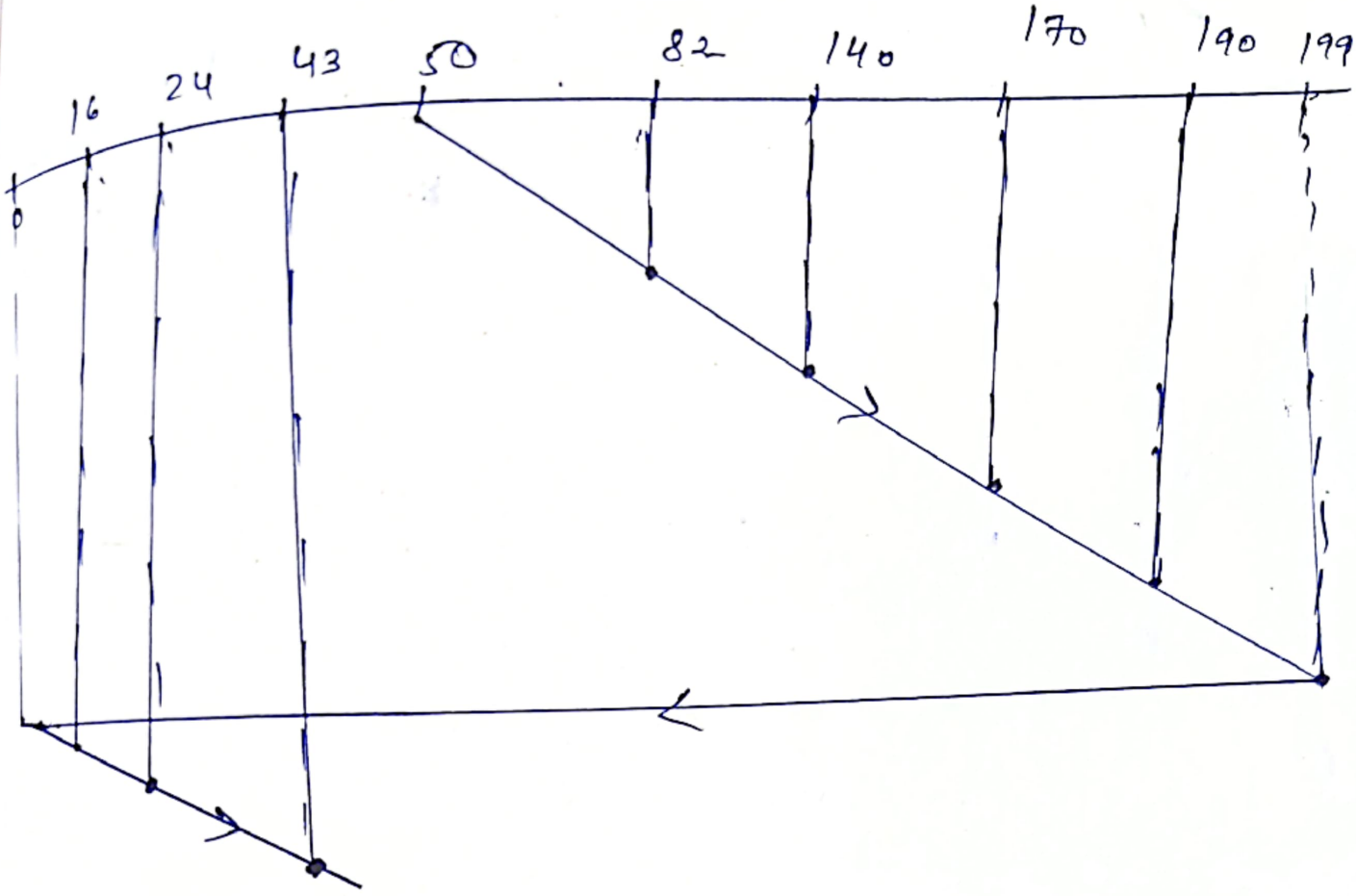
Example; Consider, Track = 0-199

current position = 50

Requesting queue track: 82, 170, 43, 140, 24, 16, 190.

Calculate total track movement (seek time).

→ Direction is towards larger value.



Seek time:- (199-50) + (199-0) + (43-0) = 391

Hence, Various disk scheduling Algorithms are discussed.