

Unit-1

Introduction to operating system

* What is Operating System?

ans: An operating system is a software that manages all resources of a computer system both hardware and software, it is an interface between user and computer, it provides an environment in which the user can execute his/her program in a convenient manner.

* What if there is no OS?

ans ~~to~~ Bulky and complex app. [Hardware interaction code must be in app's code]

① Resource exploitation by 1 App.

② No memory protection.

* Functions of operating system.

① Access to the Computer hardware.

② Interface between the user and the computer hardware.

③ Resource management (memory, device, file, security, process).

④ Hides the complexity of the hardware.

⑤ Provide isolation and protection.

* Need of operating system

ans ① OS as a platform for application programs.

② Input/output Management

③ Multitasking

④ Memory management

⑤ Provides Security

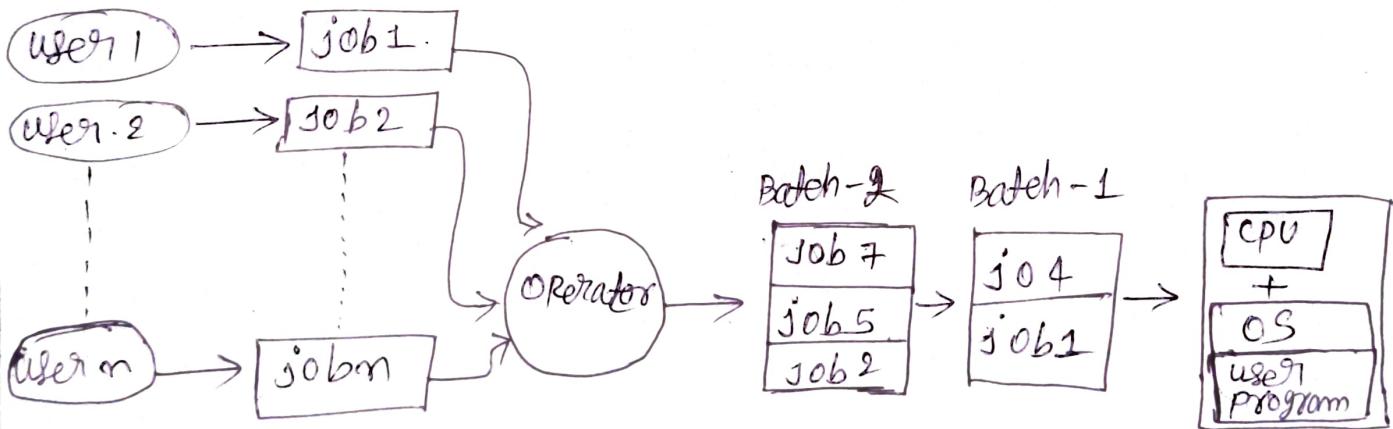
⑥ Acts as an interface between computer HW & SW. & user and computer.

* TYPES OF OPERATING SYSTEM

- i) Batch O.S ✅
- ii) Multi-programming O.S ✅
- iii) Multi-Processing O.S ✅ X
- iv) Multi-Tasking O.S ✅ X
- v) Time-sharing O.S ✅ X
- vi) Distributed O.S ✅
- vii) Real-Time O.S ✅
- viii) Parallel O.S ✅
- ix) Network O.S ✅ X

BMTDRP → TRPBMD

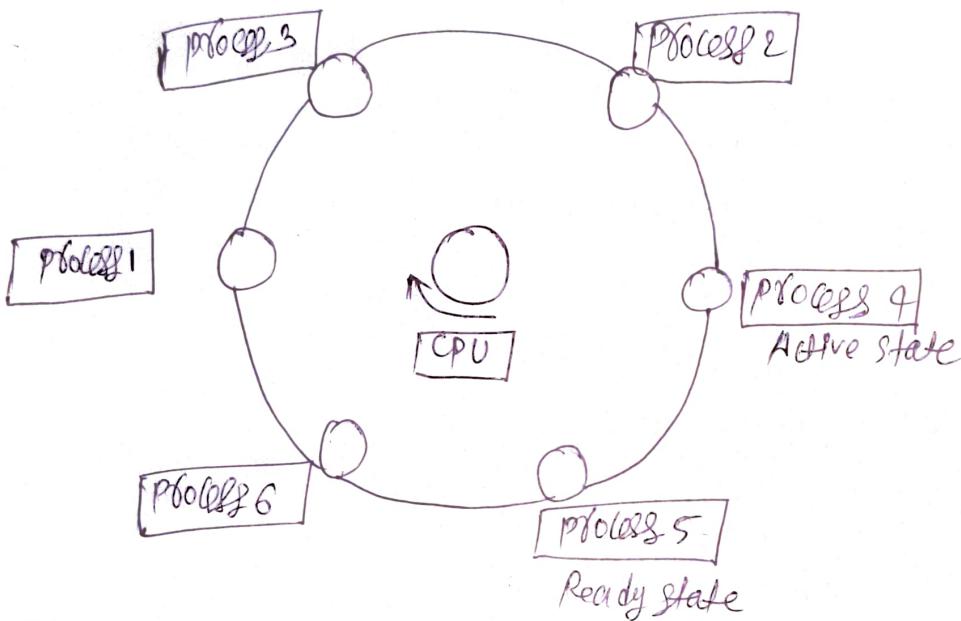
① Batch Operating system: Batch operating system is a type of operating system, it doesn't interact with the computer directly. There is an operator which takes similar jobs from the user having same requirement and groups them into batches. It is responsibility of operator to sort jobs with similar needs. Ex ⇒



- pros:
 - i) Multiple users can share the batch.
 - ii) Very less time take in batch system.
 - iii) Easy to manage large work in batch system.
- cons:
 - i) difficult to debug
 - ii) Sometimes costly
 - iii) Other jobs will have to wait for an unknown time if any job fails.

- Application:
- Bulk data processing
 - Media processing
 - Transaction processing

(i) Time-sharing operating system: Time sharing operating system is a ~~real time~~ operating system, in which each task is given some time to execute so that all the tasks work smoothly. This given time is known as time slice, time slot, or quantum. After this time interval is over OS switches over ~~the~~ to the next task. Ex =

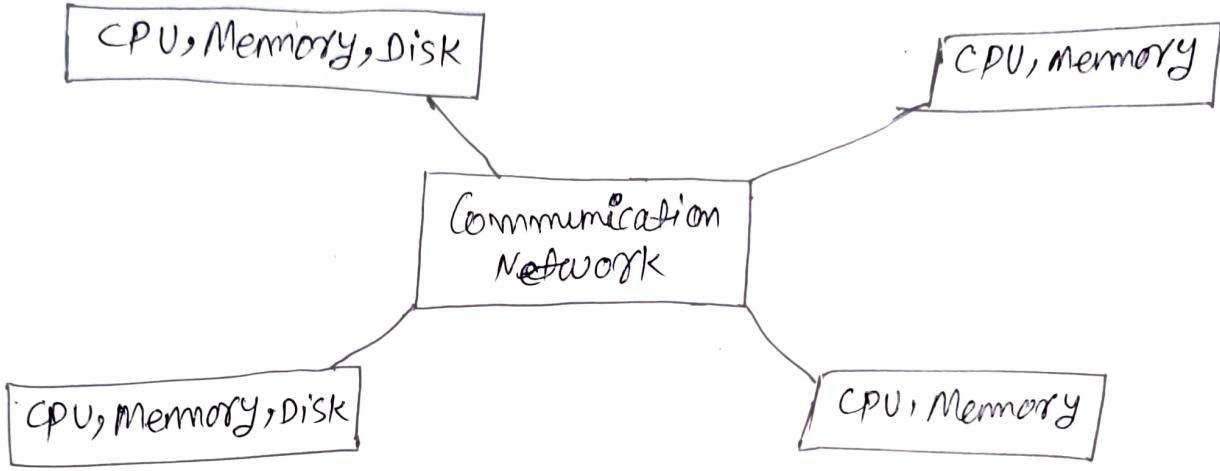


- i) Pros:
- each task gets equal opportunity.
 - each task gets equal time.
 - Multi-tasking can be performed.
 - CPU time can be reduced.

- Cons:
- Reliability problem.
 - Data communication problem.
 - Security problem.

- Application:
- Telecommunication system
 - Research and development
 - Resource optimization
 - Multi-user Environment

iii) **Distributed operating system:** Distributed operating system is a type of operating system, in which we have various systems and all those systems have their own CPU, main memory, secondary memory and resources. These systems are connected to each other using a shared communication network. Each system can perform its own task individually. Ex:-



Pros: i) Resource sharing

ii) better price and performance ratio.

iii) takes less time & higher throughput.

Cons: i) Security problem.

ii) Highly complex.

Application: i) High - performance computing

ii) cloud computing

iv) **Real-time operating system:** Real-time operating system is a operating system in which operating system respond quickly. Real time operating system used in missile system, we in hospital, ATC (air traffic control). this os provides maximum efforts to its task with quick response. Ex:-

i) Provide a quick response

ii) Used in organization (NASA & ISRO)

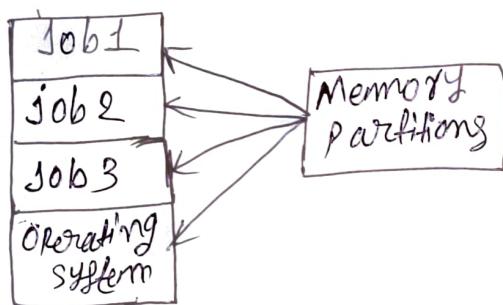
iii) We in ATC

- Cons:
- ① Too costly.
 - ② Complexity.
 - ③ Limited hardware support.

Application : Telecommunication

- ② Medical Devices
- ③ Aerospace and defense
- ④ Automatic system

⑤ Multiprogramming operating system : Multiprogramming operating system is a operating system which allows multiple programs to run on a computer. it's ~~is~~ aims to maximize the CPU utilization by efficiently switching between different programs as they execute.



Prob:

- ① Increased CPU utilization
- ② Resources utilization

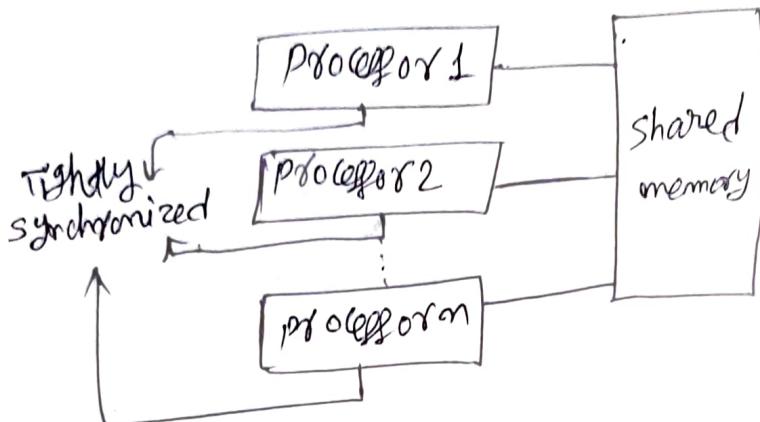
- Cons:
- ① Complexity
 - ② Difficulty in debugging

Application :

- ① Transaction processing
- ② Batch processing
- ③ Scientific computing

(vi) ~~poor~~

vi) Parallel operating system: parallel operating system is an operating system, parallel operating system is designed to speed up the execution of programs by dividing the program into multiple fragments and processing these fragments at the same time.



- PROS:
- ① Increased performance
 - ② High throughput

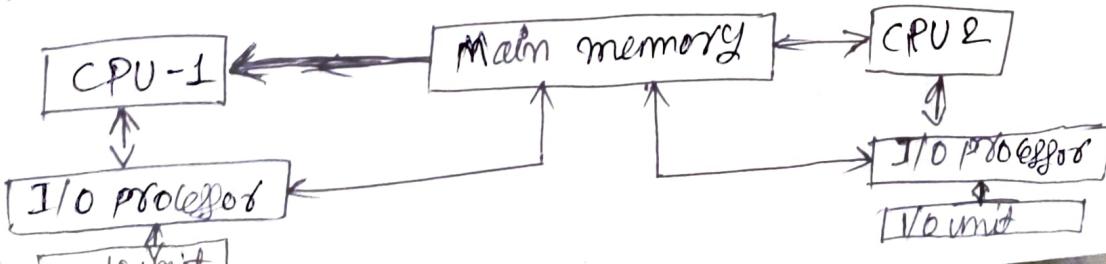
CONS:

- ① Complexity
- ② too costly

APPLICATION:

- ① High - Performance computing
- ② Database Management
- ③ Machine learning & AI

vii) Multiprocessing OS: Multi-processing operating system is a operating system that use more than one CPU to improve performance. Multiple processors work parallelly in multi-processing operating system. The main aim of the multi-processing operating system is to increase the speed of execution of the system. Ex → LINUX, UNIX



~~prob~~: failure of one processor does not affect the functioning of other processors.

(ii) parallelism, it allows to work ~~in~~ parallelly.

Contra: (i) More complex;

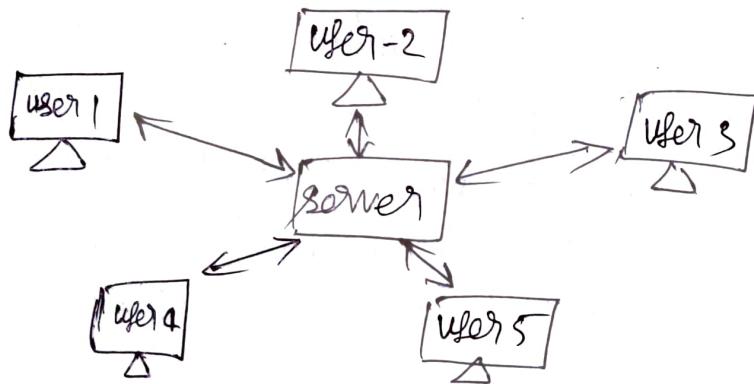
(ii) programming challenge.

Application: (i) High-performance computing

(ii) Multimedia processing

(iii) Scientific computing

(viii) Network Operating System: A Network operating system is a operating system, that connects multiple devices and computers on the network and allows them to share the resources.



Pros: (i) Highly stable due to ~~one~~ central server.
(ii) Remote access.

Contra: (i) Too cost

(ii) Regular updating and maintenance required

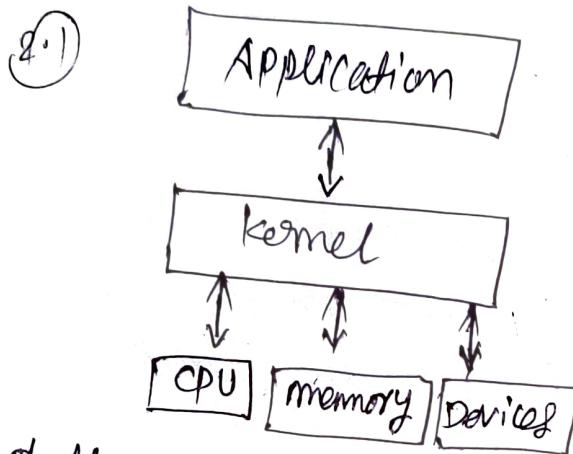
Application: (i) Manage user Account

(ii) Manage resource sharing

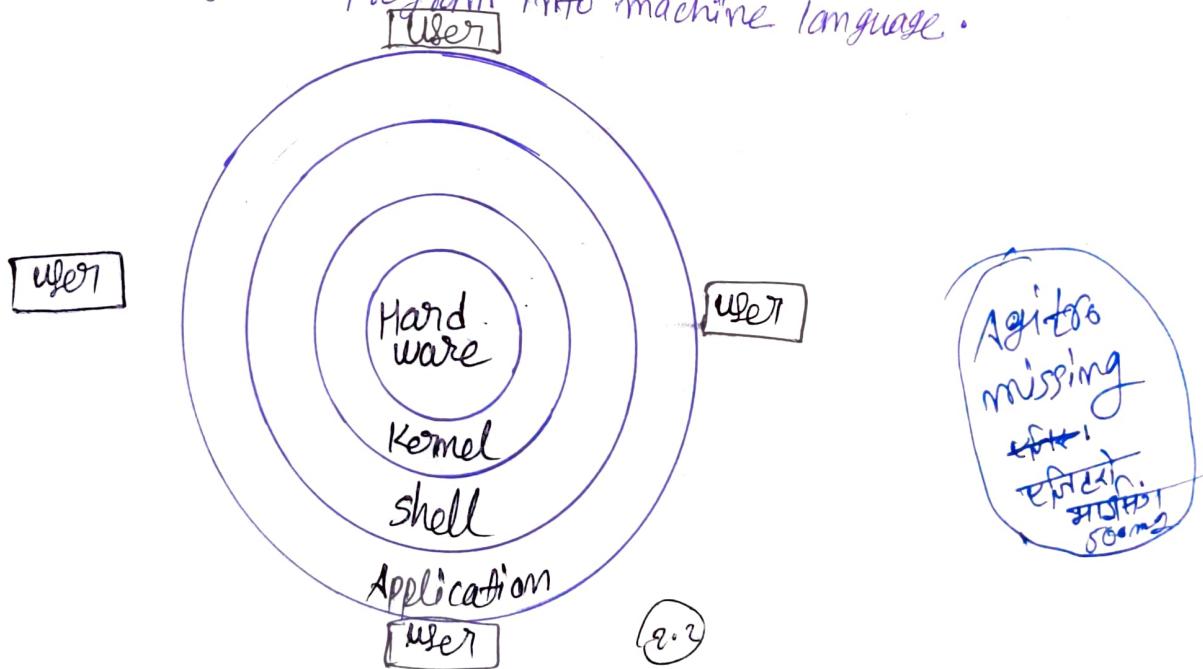
(iii) Remote access.

* Kernel and shell

Kernel: Kernel is the heart and core of the operating system. It acts as an interface between the application and data processing at hardware level. Kernel lies in the center of the operating system. It is responsible for memory management, scheduling and device drivers. It operates at a lower level than the shell and interacts directly with the hardware.



* Shell: Shell is the interface which takes input from users and sends instructions to the kernel. And shell also takes the output from kernel and send the result to user. It is also known as interpreter which is used to translate program into machine language.



Difference Between shell and Kernel

Shell

- i) It is the outer layer of O.S.
- ii) Shell provides interface between application and kernel.
- iii) Shell interact with user and translate language into machine language.
- iv) It's function to execute user commands, manage files, and provide user interface.
- v) It can be replaced or customized by the user.

Example:

Bash, powershell

Diagram 2.1

Kernel

- i) It is the inner layer of O.S.
- ii) Kernel provides interface b/w shell and hardware.
- iii) Kernel interact with hardware.
- iv) It's function is to manage memory, processes, device drives of O.S.
- v) It cannot be easily replaced or customized by user.

Example: Linux kernel, windows kernel

Diagram 2.2

* Operating system services: ①

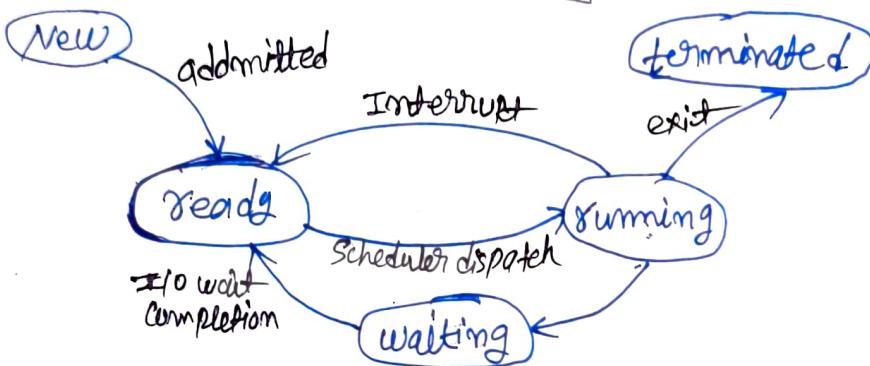
- ① program execution → load programs into memory and execute them.
- ② Input Output operations → handle communication b/w external device
- ③ File management → manage file on storage device → creation, deletion, update
- ④ Memory management → allocates & deallocates memory space of need by process
- ⑤ Process management → manage process → creation, scheduling, termination
- ⑥ Resource management → efficient allocation of resources such as CPU time, disk space
- ⑦ Time management → keep track of time scheduling tasks, manage clock
- ⑧ Error handling → detect and manage errors that may occur
- ⑨ User interface → provide user interface b/w computer and user.
such as graphical command line interface.

Unit - 2 → Process Management

(Q) What is a process? Describe the different states of a process with their detailed elaboration.

Process: A process is an instance of computer program that is being executed or a program in execution, a program that can be assigned to execute on a processor is called process.

States of a process.



- ① **New:** The process is being created but has not yet been admitted to the pool of executable processes.
- ② **Ready:** The process is waiting to be assigned to a processor. It is in main memory and it is prepared to execute as soon as the processor is available.
- ③ **Running:** The process is being executed on a processor.
- ④ **Waiting:** The process is waiting for some event like I/O operation, until this operation is not completed till then it is temporarily stopped.
- ⑤ **Terminated:** The process has finished execution. It may have terminated.

* **What is threads:** In an operating system, a thread is the smallest unit of execution within a process.

* **Process Control Block:** A process control block is a data structure used by operating system to store information about a process.

It includes details such as process state, program counter, registers, values, memory management information and scheduling related data.

* Process Scheduling : Process scheduling involves selecting a process from ready queue and allocating CPU time to it as well as deallocationing CPU time.

* Scheduling Algorithms:

Primitive Algo

- * ① Shortest Remaining Time First
- * ② Longest Remaining Time first
- * ③ Round Robin -
- * ④ Priority -

NON Primitive Algo

- ① First come first serve
 - ② Shortest job first
 - ③ Longest job first
 - ④ High est Response Ratio
- Next CHRW
- Non primitive & (FIFO) some times Priority (LRU) initializing

* Arrival time (AT): It is time at which the process becomes ready to run.

(1) ~~First Come First Serve~~:

* Burst time: The time requested by process to get executed on CPU.

* Completion (CT): The point of time at which the process complete its execution.

* Turn around Time (TAT): It is difference b/w CT and AT.

* Waiting Time: It is difference b/w TAT and Burst time.

$$WT = TAT - BT$$

* Response Time: The amount of time it takes for CPU to respond.



$$RT = CPU \text{ burst time} - AT$$

① first come first serve :

Find average Turn around time and average waiting time

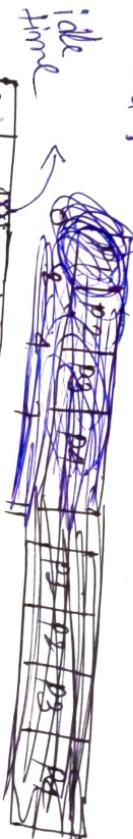
(3) unit

Process No	AT	BT	CT	TAT	WT
P1	0	2	2	2	0
P2	1	2	4	3	1
P3	5	3	8	3	0
P4	6	4	12	6	0

Avg TAT = ? TAT = CT - AT, CT = when process complete
Avg WT = ? WT = TAT - BT ,

Non pre-emptive

Gantt chart:



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

unit

Avg WT = $\frac{0+1+0+2}{4} = 0.75$ unit

Q2. Shortest job first: calculate average turn around time and average waiting time

Process No	AT	BT	CT	TAT	WT
P1	1	3	6	5	2
P2	2	4	10	8	4
P3	1	2	3	2	0
P4	4	4	14	10	6

$$\text{Avg TAT} = \frac{5+8+2+10}{4} = 6.25 \text{ unit}$$

$$\text{Avg WT} = \frac{2+0+0+6}{4} = 3 \text{ unit}$$

Gantt chart:

idle time	P3	P1	P2	P4
0 1 2 3 6 10 14				

③ Longest job first: calculate avg Turn around and avg waiting time.

Process	AT	BT	CT	WT	TAT
P1	1	2	3	0	2
P2	2	4	9	15	19
P3	3	6	9	0	6
P4	4	8	17	5	13

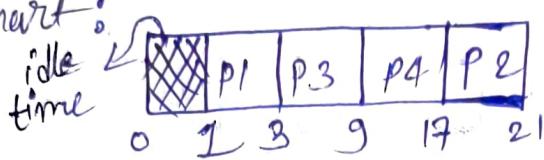
$$\text{Avg TAT} = \frac{2+19+6+13}{4}$$

$$\text{Avg TAT} = \frac{40}{4} = 10 \text{ unit}$$

$$\text{Avg WT} = \frac{0+15+0+5}{4}$$

$$\text{Avg WT} = \frac{20}{4} = 5 \text{ unit}$$

Gantt chart:



④ ~~Highest~~ Highest response ratio time: calculate average waiting time and average turnaround time!

Process	AT	BT	CT	WT	TAT
P1	0	3	3	0	3
P2	2	6	9	1	7
P3	4	4	13	5	9
P4	6	5	20	9	14
P5	8	2	15	5	17

$$P_3 \text{ Ratio} = \frac{5+4}{4} = \frac{9}{4} = 2.25$$

$$P_4 \gamma = \frac{3+5}{5} = \frac{8}{5} = 1.6$$

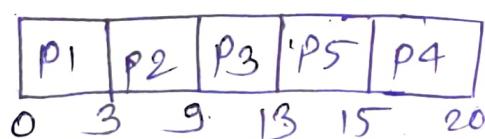
$$P_5 \gamma = \frac{1+2}{2} = \frac{3}{2} = 1.5$$

$$P_4 \gamma = \frac{7+5}{5} = \frac{12}{5} = 2.4$$

$$P_5 \gamma = \frac{5+2}{2} = \frac{7}{2} = 3.5$$

$$\text{Ratio} = \frac{\text{Waiting time} + \text{burst time}}{\text{burst time}}$$

Gantt chart



$$\text{Avg TAT} = \frac{3+7+9+14+7}{5} = \frac{40}{5} = 8 \text{ unit}$$

$$\text{Avg WT} = \frac{0+1+5+9+5}{5} = \frac{20}{5} = 4$$

primitive

① shortest remaining time first:

Process	AT	BT	CT	TAT	WT	RT
✓P1	0	5 8 2 10	9	9	4	0
✓P2	1	3 2 10	4	3	0	0
✓P3	2	4	13	11	7	7
✓P4	4	10	5	1	0	0

Grantt chart

P1	P2	P2	P2	P4	P1	P1	P1	P1	P3	P3	P3	P3
0	1	2	3	4	5	6	7	8	9	10	11	12

$$\text{Avg TAT} = \frac{9+3+11+1}{4} = \frac{24}{4} = 6 \text{ unit}$$

$$\text{Avg WT} = \frac{4+0+7+0}{4} = \frac{11}{4} = 2.75 \text{ unit}$$

$$\text{Avg RT} = \frac{0+0+7+0}{4} = \frac{7}{4} = 1.75 \text{ unit}$$

② longest remaining time first:

Process	AT	BT	CT	TAT	WT	RT
✓P1	0	2 10	9	9	7	0
✓P2	1	3 4 8 2 10	10	9	4	0
P3	4	3 2 10	11	7	4	0
P4	5	2 10	12	7	5	2

$$\text{Avg TAT} = \frac{9+9+7+7}{4} = \frac{32}{4} = 8 \text{ u}$$

$$\text{Avg WT} = \frac{7+4+4+5}{4} = \frac{20}{4} = 5 \text{ u}$$

$$\text{Avg RT} = \frac{0+0+0+2}{4} = \frac{2}{4} = 0.5 \text{ s}$$

Grantt chart

P1	P2	P2	P2	P3	P2	P3	P4	P1	P2	P3	P4
0	1	2	3	4	5	6	7	8	9	10	12

③ Round robin
process

③ Round Robin:

process	AT	BT	CT	TAT	WT	RT
P1	0	8	12	12	7	0
P2	1	4	11	10	6	1
P3	2	2	6	4	2	2
P4	4	2	9	5	4	4

Given
TQ = 2

$$\text{Avg TAT} = \frac{12+10+4+5}{4} = \frac{31}{4} = 7.75$$

Ready queue
o P1 P2 P3 P1 P4 P2 P1

$$\text{Avg WT} = \frac{7+6+4+4}{4} = \frac{19}{4} = 4.75$$

Running queue
o P1 P2 P3 P1 P4 P2 P1
0 2 4 6 8 9 11 12

$$\text{Avg RT} = \frac{0+1+2+4}{4} = \frac{7}{4} = 1.75$$

* No of context switching = 6

④ priority scheduling : Considering higher the no higher the priority.

priority	process	AT	BT	CT	TAT	WT	RT
10	P1	0	8	12	12	7	0
20	P2	1	4	8	7	3	0
30	P3	2	2	4	2	0	0
40	P4	4	2	5	1	0	0

Gantt chart

P1	P2	P3	P3	P4	P2	P1
0 1 2 3 4 5 8 12						

$$\text{Avg TAT} = \frac{12+7+2+1}{4} = \frac{22}{4} = 5.5$$

$$\text{Avg WT} = \frac{7+3+0+0}{4} = \frac{10}{4} = 2.5$$

* difference between process and thread

5

Process

- i) Process is heavy weight task.
- ii) System calls are performed in process.
- iii) Context switching is slow.
- iv) Different processes have different copy of data files and codes.
- v) Blocking a process will not block another.

Thread

- Thread is light weight task.
- No system call involved.
- Context switching is fast.
- It shares the same ~~copy~~ copy of code & data file.
- Blocking a thread will block another.

* difference b/w user level thread and kernel level thread

User level thread

- ① It is typically faster.
- ② Context switching is faster.
- ③ It is more efficient.
- ④ It is easy to manage.
- ⑤ Hardware support is needed.

kernel level thread

- It is slower.
- Context switching is slower.
- It is not so efficient.
- It is not easy to manage.
- No hardware support is ~~needed~~ needed.

* define context switching.

Context switching refers to the process of saving state of a running process so that it can be restored and loading another process and run it.

⑤ Mix-burst timer scheduling:

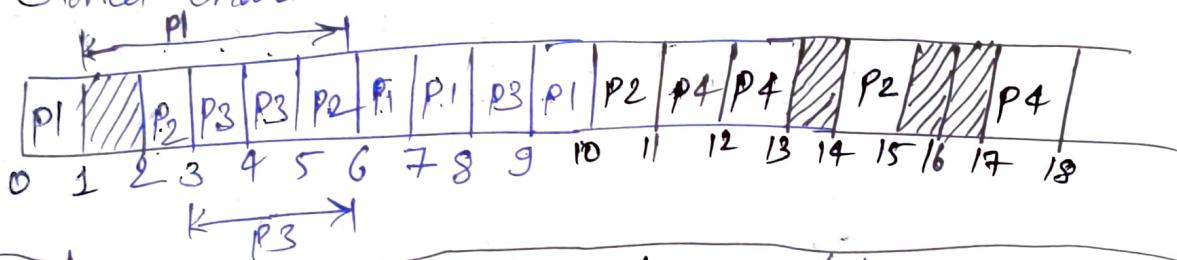
priority	process no	AT	CPU	I/O	CPU	CT	TAT	WT	RT
2	P1	0	1	5	3				
3	P2	2	8	2	3	1			
1	P3	3	0	2	3	1			
4	P4	3	2	4	1				

lower the no higher the priority

find the completion time of P1, P2, P3, P4

find the CPU idleness & CPU utilization time slots.

Gantt chart



Q.1 Non-primitive priority scheduling

Considering lower the no higher the priority

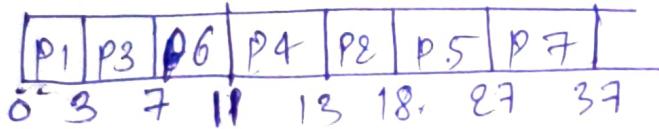
Process	Priority	AT	BT	CT	TAT	WT	RT
✓ P1	2	0	3	3	3	0	0
✓ P2	6	2	5	18	16	11	11
✓ P3	3	1	4	7	6	2	2
✓ P4	5	4	2	13	9	7	7
✓ P5	7	6	9	27	21	12	12
✓ P6	4	5	4	11	6	2	2
✓ P7	10	7	10	37	30	20	20

$$\text{Avg TAT} = 13$$

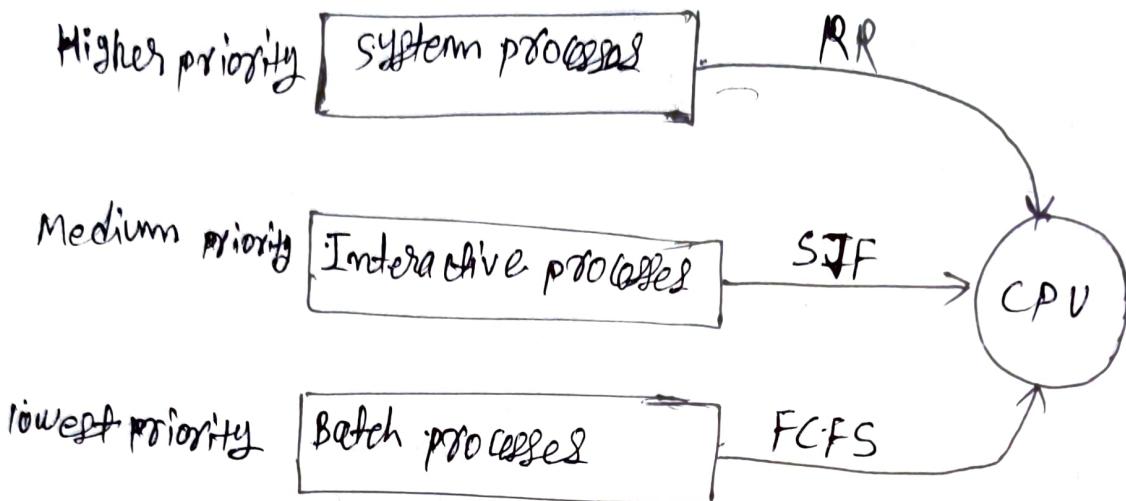
$$\text{Avg WT} = 7.71$$

$$\text{Avg RT} = 7.71$$

Gantt chart:



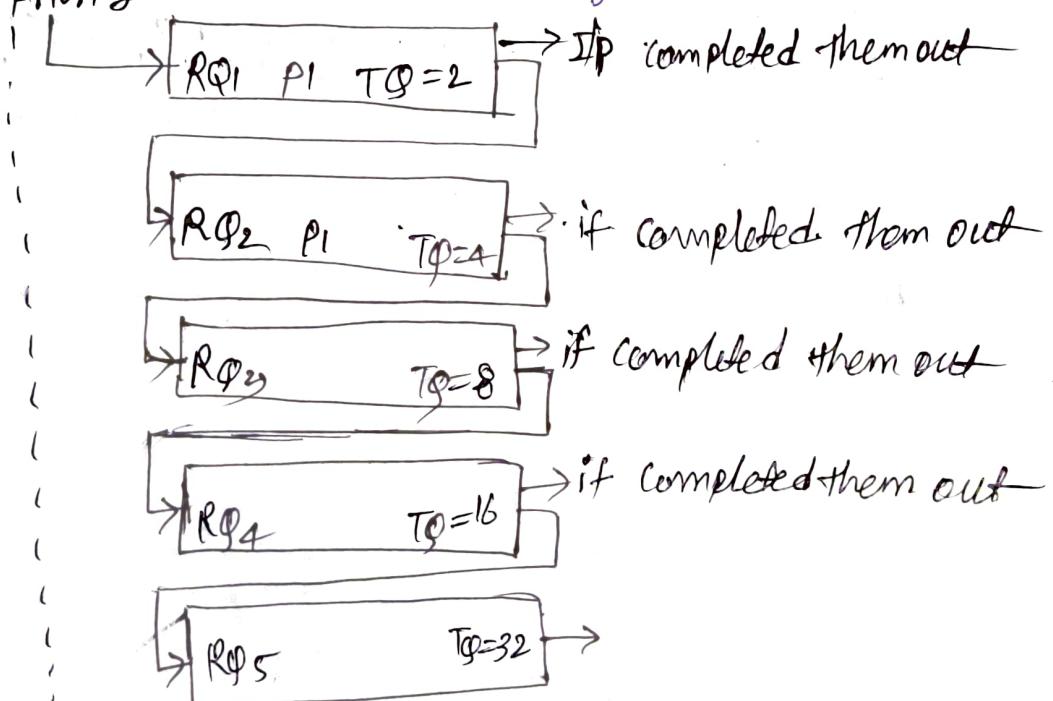
* Multi level queue scheduling:



the problem known as Starvation

Starvation over utilization of resources.

Multilevel feedback queue scheduling.

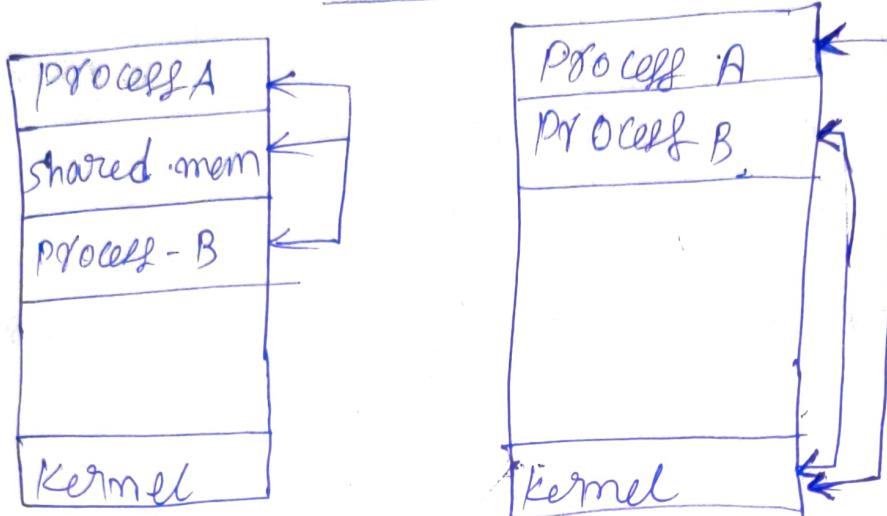


Highest priority

T -

Inter process communication : it is a communication system that allows processes to communicate one-another & sync ~~is~~ their actions.

S.M



* ~~Process synchronization~~: process can execute in two modes (i) serial mode (ii) parallel mode

(i) Serial mode: if one process completed then the next process will start. Ex → ATM

(ii) Parallel Mode: In parallel mode many processes runs in parallel. Ex → Net banking.

Process can be of two type

(i) Co-operating process (ii) Independent processes

(i) Independent process: In the independent execution of one doesn't affect other process. ex → Net banking

(ii) Co-operating process: In the co-operating process processes are dependent on each other. ex → IRCTC

Synchronization issue :

int shared = 5; 4 → race condition

Process P1

int x = shared;

x++; 8 6

sleep(1);

shared = x;

6

Process - P2

int y = shared;

y--; 8 4

sleep(1);

shared = y;

8 4

↓ time

in the sleep function, running process is passed for 1 time unit due to preemption.

A, B
Nc section

x, y
Nc section

Entry section

Entry section

Count ++;
C-S

Count--;
C-S

exit section

exit section

C-S : Critical Section.

N-C : None-critical Section.

critical section : It is the part of programs where shared resources are accessed by various processes. It is a common platform used by processes.

Condition for entry and exit section

Process synchronization

i) Mutual exclusion

ii) Progress

iii) Bounded wait

iv) No assumption related to H/W speed.

int shared = 5

process P1

int x = shared;

x++;

sleep(1);

shared = x;

process P2

int y = shared;

y--;

sleep(1);

shared = y;

(Entry) section

① Down (Semaphore s)

{
s.value = s.value - 1;
if (s.value < 0)

{ put process (PCB) in
suspended list;
sleep();

}

else {

return;

}

Exit section

Counting semaphore

UP (Semaphore s)

{
s.value = s.value + 1;
if (s.value ≤ 0)

{

Select a process from the
suspended list;

Wake up();

{

}

Entry Section → p() / down() / wait()

Exit Section → v() / up() / post / ReleaseU

*Semaphore: Semaphore is a method or tool used to prevent race condition. Semaphore is a integer variable used in mutually operative processes in order to achieve synchronization.

Chunhwan Kim

Semaphore of two type

① Counting Semaphore

(+∞, -∞) ② Binary semaphore, (1, 0)

1.1

Binary Semaphore

Up (Semaphore s)

{ if (s.value == 1)

{ s.value = 0;

{ else

{ block this process

and place in suspended

list, sleep();

}

}

}

}

1.2

Down (Semaphore s)

{ if (suspended list is empty)

{ s.value = 1;

{ else

{ select a process from

suspended list and

wake up();

}

}

}

- Q1) Interpret the role of process synchronization, critical section and mutual exclusion. How semaphores resolve the issue of process synchronization?
- Ans ⇒ Role of process synchronization
- : process synchronization coordinating the execution of processes to avoid race conditions, deadlocks and other synchronization issues.

- Role of critical section: Critical section ensure only one process access this section at a time, if concurrent access could lead to unpredictable results.
- Role of mutual exclusion: If a process is executing in the critical section ~~then~~ no other process is allowed to proceed execute in the critical section.

~~How to resolve Binary semaphore~~ Resolving synchronization issues: (1125) *

When a process wants to enter its critical section, it performs down operation. If the semaphore value 1, the process can proceed otherwise put process in the suspended list, sleep(), this ensures that only one process can be in the critical section at a time, preventing race conditions.

~~Calling 1-2~~

Down (Semaphore s)

{ if (s.value == 1)

{ s.value = 0;

3

.else

{ put the process in the suspended list;
sleep();

2g
3g

(Q) Demonstrate the four criteria required for the process synchronization, how two types of semaphores resolve the issue of process synchronization? Demonstrate through appropriate example.

Ans: Four criteria for process synchronization are:

① Mutual Exclusion: If a process is executing in critical section then no other process is allowed to execute in the critical section.

② Progress: If there is no any process in the critical section then Semaphore value have to progress to 1 otherwise no any process will be able to enter into critical section, while there is no any process into suspended list.

- ③ Bounded wait: There should exist bounded wait because it should not happen like that one process is executing for infinite no of time and another process is still waiting for its chance.
- ④ No Assumption of Related to hardware speed: No assumption is made about the speed of execution of the processes.

Types of Semaphore

① Binary semaphore ② Counting semaphore

① Counting semaphore: When a process wants to enter its critical section, it performs down operation if the semaphore value is greater than 0. Process can proceed to 0 them but if the semaphore value is less than that only one process can be in the critical section at a time preventing race conditions.

Down(Semaphore s)

```
{
    s.value = s.value - 1;
    if (s.value < 0)
        { put process in
         suspended list, sleep();
         }
    else
        return;
    }
```

Example :

Consider two processes sharing a printer. The counting semaphore ensures only one process prints at a time.

② Binary semaphore → calling

Example: Imagine a resource pool with ten identical resources. A counting semaphore with a count of 10 ensures no more than ten processes can access the resources concurrently.

1.125

(Q) Demonstrate the usage of stack, heap, data and code as a part of various sections in a program through appropriate example.

Ans → Example:

```
#include<iostream>
int globalvar = 10;
int main()
{
    int localvar = 20;
    int *dynamicvar = new int(30);
    std::cout<< "code section: This is the executable code";
    cout<< "data section: Global variable = " << globalvar << endl;
    cout<< "stack section: Local variable = " << localvar << endl;
    cout<< "heap section: " << "Dynamic variable = " << *dynamicvar << endl;
    delete dynamicvar;
    return 0;
}
```

Stack section: It manages local variable like 'localvar' within functions. It stores data at a contiguous location.

Heap section: Heap section dynamically allocated memory demonstrated with 'dynamicvar'. It stores data at random location.

Data section: Data section stores global variable like 'globalvar'.

Code section: Code section containing program's executable instructions.

(Q) Write a syntax for a. (a) free (b) cut in operating system.

free → Syntax: [REDACTED] free[OPTION]... [FILE]...

Cut command syntax → cut OPTION... [FILE]

(Q) Explain with relevant example that how an operating system provides services to both the user and to the programs.
Ans → detailed in PDF → PYQ 6.7

User Service

- User interface: GUI, CLI,
- File management: file, directories
create, delete, organize data.
- Security: user authentication, access control.

Example

- GUI: windows, macOS, Linux
- File Management: Explorer in Windows or Finder in macOS.
- Security: password protection, user permission in OS.

Program Services

- Memory Management: allocate deallocate memory, program.
- Process Management: processes resource allocation.
- I/O Service: Input or output operation.

Example

- Memory Management: allocate memory for running app... .
- Process management: Linux schedule processes to ensure efficient utilization of CPU resources.
- I/O services: windows, memory pointers, keyboards, other devices

(PYQ) Different Methods to handle deadlock Briefly explain.
Ans ⇒ ① Deadlock Ignorance ② Deadlock Prevention

③ Deadlock Avoidance ④ Deadlock Detection

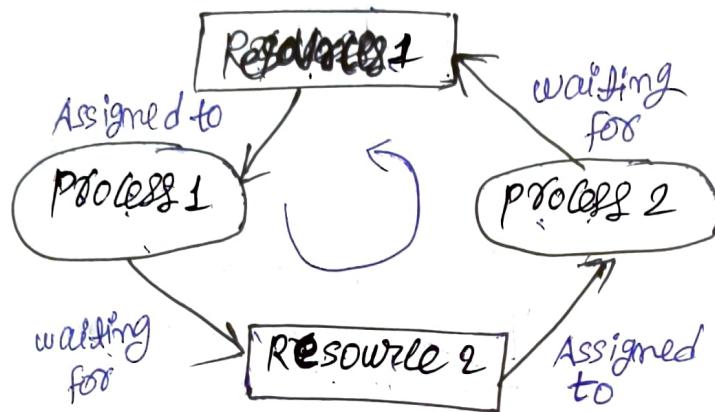
① Deadlock Ignorance: In deadlock ignorance method OS checks like the deadlock never occurs and completely ignores it even if the deadlock occurs.

② Deadlock Detection: In deadlock detection method OS periodically check whether deadlock occur in the system or not if it occurs then it applies some of the recovery methods to the system to get recover deadlock.

③ Deadlock Avoidance: Deadlock avoidance maintain some set of data and whenever a process requests for a resource, it checks if it will be safe to share resource or not if it does deadlock may occur then it avoid the deadlock.

Ques) What is the deadlock and what are the conditions to prevent it or describe necessary condition for a deadlock situation to occur. Brief about different

deadlock: A deadlock is a situation where a set of resources are blocked because each process is holding a resource and waiting for another resource acquired by some other process.



* Conditions for deadlock / To prevent deadlocks conditions are

- ① Mutual Exclusion
 - ② NO preemption
 - ③ Hold & wait
 - ④ circular wait
- ① Mutual Exclusion: only one process can use resources at a time.
- ② No preemption: A resource cannot be taken from a process unless the process releases the resources.
- ③ Hold & wait: A process is holding at least one resource and waiting for resources.
- ④ circular wait: Eliminate the possibility of circular wait resources.



④ Deadlock prevention: In deadlock prevention method OS prepares a solution by which there is no any possibility of deadlock occurring. They use like mutual exclusion, Hold & wait, No preemption, circular wait.

(P8) difference b/w deadlock and starvation

Ans \Rightarrow deadlock

- ① Deadlock is a situation where a process is blocked and waiting for some other resource that is held by some other waiting process.
- ② It is also called circular wait.
- ③ No progress ~~can~~ once it occurs.
- ④ It can be prevented by deadlock prevention.

starvation

- when a low priority process requests a resource but cannot get because a higher priority process is using that resource for a long time.
- ② It is also known as livelock.
- ③ Other processes can progress.
- ④ It can be prevented by using a proper scheduling algorithm.

(P9) Is deadlock state more critical than starvation justify.

Ans \Rightarrow Yes deadlock state more critical than starvation

because once deadlock occurs no any other process can progress but in case of starvation apart from victim process all other processes can progress that's why deadlock state more critical than starvation.

Q. diff b/w primitive and non primitive scheduling

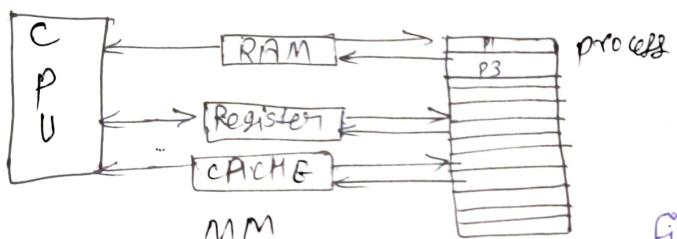
① CPU allocated to a process for a limited time.	The CPU allocated to a process until it terminates.
② process can be interrupted in between.	process cannot be interrupted until it terminates.
③ context switching.	No context switching
④ It is flexible	It is not flexible.
⑤ CPU utilization is higher.	CPU utilization is lower.

Banker's Algorithm

process	Allocation			Max Need			Available			Remaining		
	A	B	C	A	B	C	A	B	C	A	B	C
P1	0	1	0	7	5	3	3	3	2	7	4	3
P2	2	0	0	5	2	2	5	3	2	1	2	2
P3	3	0	2	4	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	28	12	11						

$$\begin{aligned}
 A &= 10 \\
 B &= 5 \\
 C &= 7
 \end{aligned}
 \quad \Rightarrow$$

Memory management



Memory Management Tech

- Configurable
- Non-Configurable
- fixed Partition
- variable Partition (Dynamic)
- Paging
- segmentation

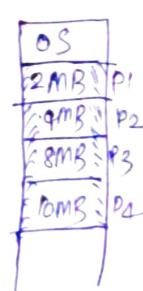
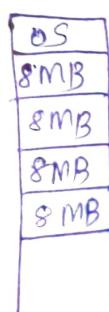
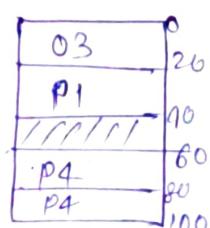
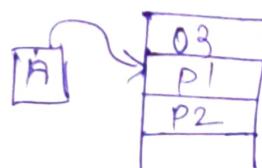
4MB

4MB

No. of process?

CPU utilization = 1 - K

K = I/O operation (To % e.g.)



Dynamic variable partitioning

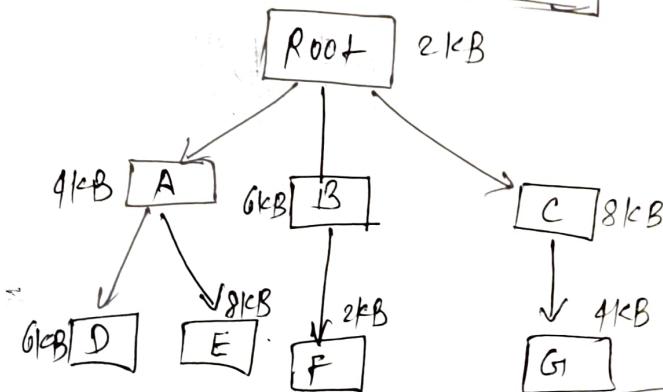
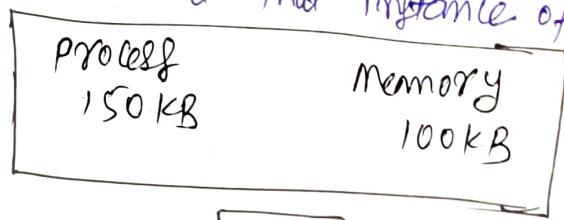
Static fixed
Partitioning

* **Contiguous Memory** : Internal fragmentation is the problem that is identified when the blank space left by 1% can't be used by another process on same time.

unit : Memory management

(PQ) Explain overlays, what is fragmentation and types and disadvantages, Internal and External fragmentation,

Ans \Rightarrow Overlay refers to a technique used to manage programs or processes. The overlay technique comes into picture when you have a computer with limited memory and you want to run a large process that size is greater than main memory then instead of loading the entire program into memory, OS divides it into smaller sections called overlays that is required at that instance of time.



$$\begin{aligned} \text{Root} + \text{A} + \text{D} &= 12 \text{ kB} \\ \text{Root} + \text{A} + \text{E} &= 14 \text{ kB} \\ \text{Root} + \text{B} + \text{F} &= 10 \text{ kB} \\ \text{Root} + \text{C} + \text{G} &= 19 \text{ kB} \end{aligned}$$

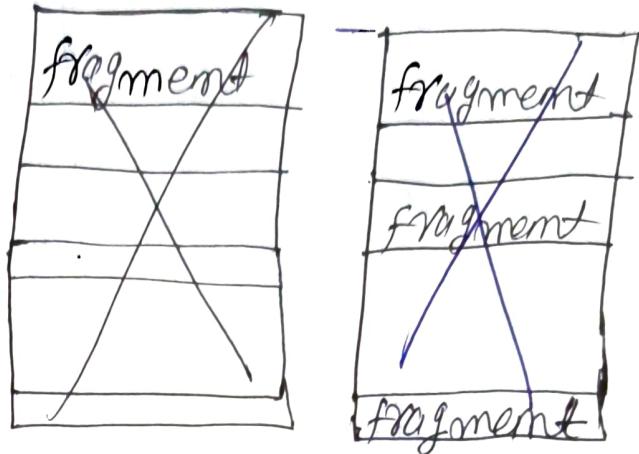
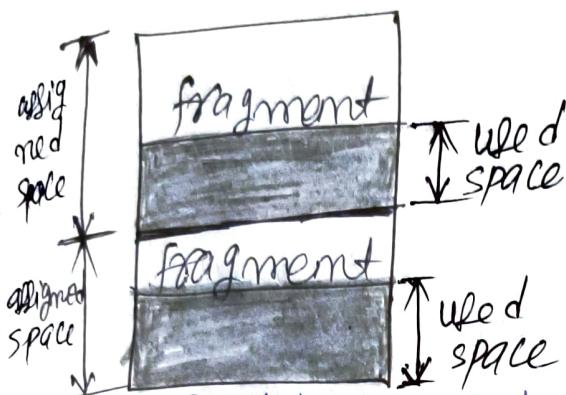
19 kB

There are two types of fragmentation:

- i) Internal fragmentation
- ii) External fragmentation

fragmentation: fragmentation refers to the phenomenon where memory divided into smaller, non-contiguous pieces over time.

1. Internal fragmentation: It occurs when fixed size memory block is allocated to the process. When memory assigned to the process that is slightly larger than the process size the free space is created that free space is called fragment and whole process is called internal fragmentation.



disadvant i) wasted memory, Reduced Efficiency, increased overhead

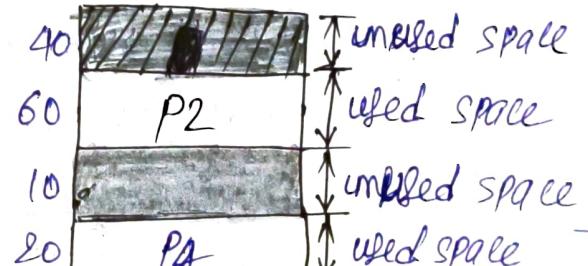
2. External fragmentation: It occurs when variable size memory space are allocated to processes dynamically. It happens when there is sufficient memory to allocate request process but cannot be allocated as it is in non-contiguous manner.

disadvant → Difficulty in memory allocation

i) performance degradation

40	P1
60	P2
10	P3
20	P4

Point A



Point B

Ans → 50

(PY) Elaborate the mapping and difference between logical and physical address.

ans ⇒ Memory Mapping is the process done by the OS to translate virtual memory address to physical address so, the program can run anytime when the OS loads it or it is required.

Physical address

i) It is also called real address.

ii) Physical address will not change.

iii) Computed by MMU.

iv) The user can indirectly access physical address but not directly.

v) User can never view physical address of program.

vi) Physical address is the set of physical addresses.

(PQ) What is virtual memory and how it is used and what are its advantages?

Ans → Virtual memory is a memory management technique used by operating system to efficiently utilize available physical memory (RAM) by temporarily transferring data b/w RAM and secondary memory. It allows programs to run as if they have more memory than physically available memory.

Virtual Memory

0	A
4K	B
8K	C
12K	D
16K	E

Physical main memory

0K	
4K	C
8K	
12K	
16K	B
20K	
24K	A



Secondary memory

How it's used:

- ① When a process needs more memory than that is greater than RAM, then virtual memory move some data from RAM to secondary memory.
- ② virtual memory frees up space in RAM and loads the required data from disk into RAM.

* Advantages of virtual memory

- ① Increased memory utilization.
- ② Increased efficiency.
- ③ Increased Multitasking.

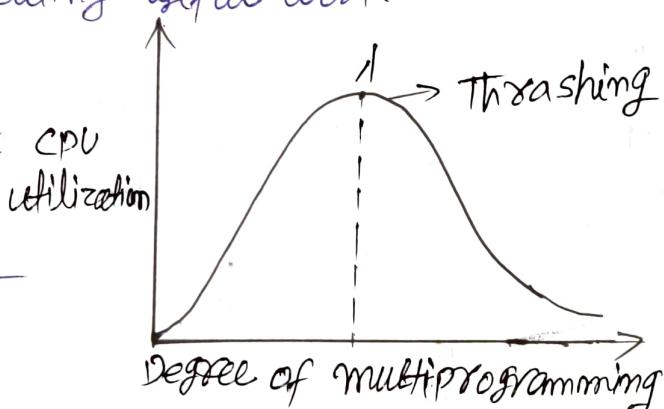
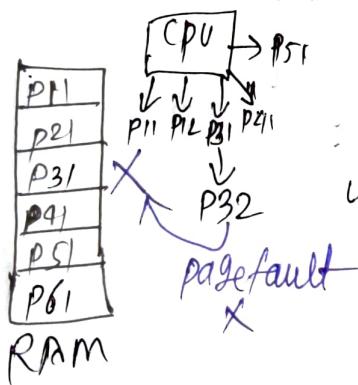
Need of virtual memory

- ① Increased memory capacity
- ② memory isolation
- ③ Demand paging
- ④ Memory management

(PQ) what is thrashing?

how is it controlled by OS.

Ans \Rightarrow Thrashing is a phenomenon that occurs in computer systems that occurs in computer systems when the system spends an excessive amount of time on page swapping rather than executing useful work.

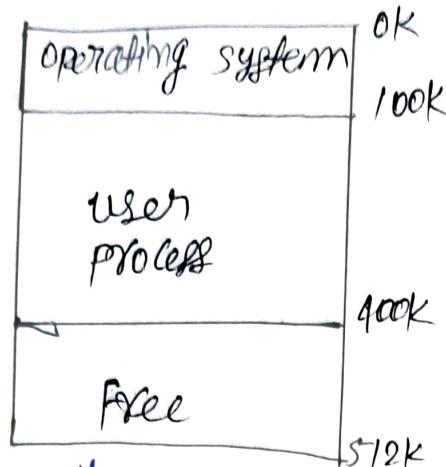


- ① Virtual memory
- ② Page replacement algorithm
- ③ I/O scheduling

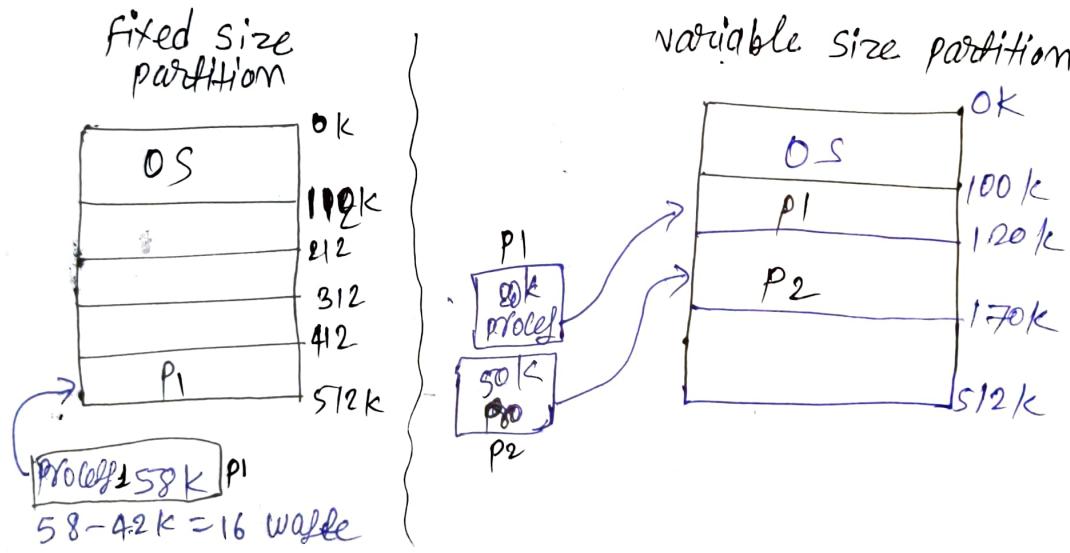
(PQ) Explain with diagram single partition allocation and multiple partition allocation.

Ans \Rightarrow Single-partition allocation refers to a memory management scheme where the entire memory space is divided into a single partition, which is then allocated to a single process. This means that each process occupies the entire memory space, which can lead to inefficiencies in memory utilization, especially when smaller processes are allocated more memory than they actually need.

Single partition Allocation



Multiple-partition allocation is a memory management technique where the memory is divided into multiple partitions, each capable of holding a single process. These partitions can vary in size and may be fixed or variable. When a process arrives, it is allocated to an appropriately size partition. This allows efficient memory utilization compared to single partition allocation.



(Q) Elaborate the file Management with detailed requirement and implementation issues of contiguous, linked and indexed allocation methods.

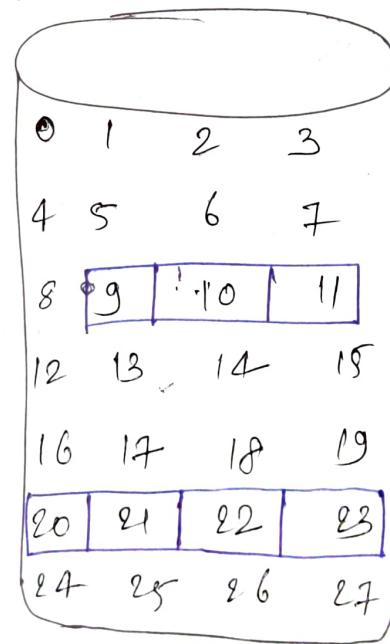
File management involves organizing and controlling access to data stored in files on a computer system. There are different allocation methods for storing files on disk, including contiguous, linked and indexed allocation.

① Contiguous Allocation:

Requirement: In contiguous allocation, each file occupies a contiguous block of disk space. The primary requirement is to find a large enough contiguous space to store the entire file.

Implementation Issues:

- Fragmentation: Contiguous allocation can lead to external fragmentation, where the disk space is fragmented ~~to~~ into small unusable blocks, making it challenging to find contiguous space for new files.
- External fragmentation: This occurs when there are many small free spaces scattered throughout the disk, making it difficult to allocate contiguous space for large file.
- Compaction: Compaction techniques may be required to reduce fragmentation by rearranging files and free space on the disk, but this can be time-consuming.

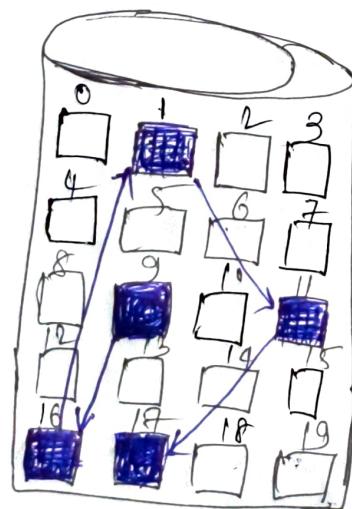


ii) Linked Allocation:

Requirement: Linked allocation allocates disk space for files using linked list. Each file has a linked list of disk blocks, where each block points to the next block in the file. The file system maintains a table of file allocation information, such as the starting block of each file.

Implementation issues:

- Overhead: In linked allocation overhead is to storing pointers for each block, which increases the size of the file allocation table.
- Random Access: Accessing file blocks randomly is slow because they are not stored next to each other, so you have to follow a ~~list~~ linked list to find what you need or requiring traversal of the linked list to access a specific block.
- Fragmentation: fragmentation is not an issue with linked allocation since files can be scattered across the disk but it may lead to slower access time due to disk head movement.



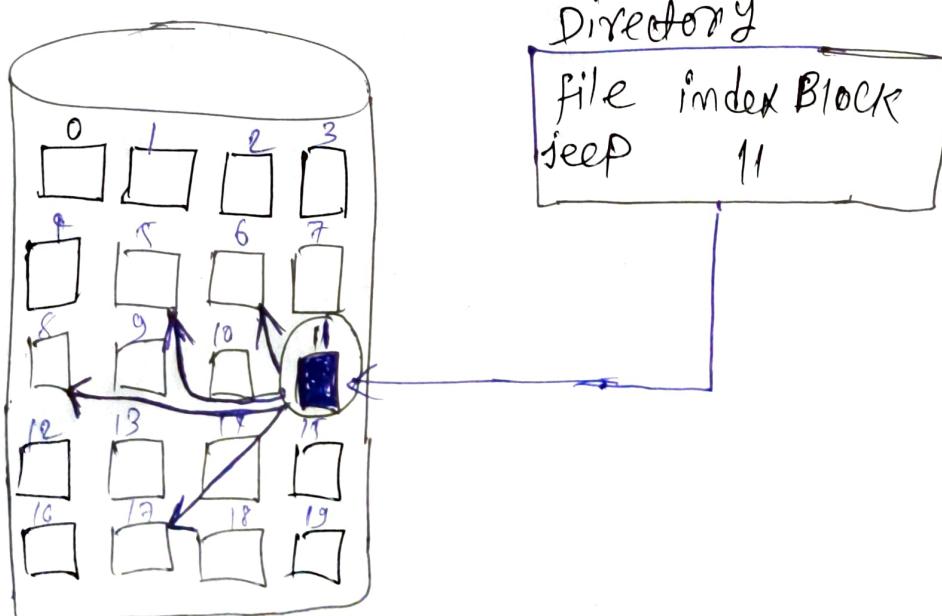
Directory			
File	start	end	
1eep	9	15	

Requirement:

(iii) **Indexed Allocation:** Indexed allocation uses a separate index block for each file, containing pointers to all the blocks of the file. The index block is kept in memory for fast access.

Implementation issues:

- **Index Block size:** The size of the index block determines the maximum size of a file that can be stored using indexed allocation. If the index block size is fixed, it may limit the maximum file size.
- **Index Block overhead:** Index allocation requires extra space to store index blocks for each file which consumes additional disk space.
- **Index Block Management:** Efficient management of index blocks is crucial, especially for large files to ensure fast access times and minimize disk I/O operations.



(PQ) why page size is always power of 2?

Ans → Page size is always power of 2 because paging is implemented by breaking up an address into a page and offset no.

It is most efficient to break the address into x-page bits and y-offset bits rather than doing by arithmetic operations.

Since each bit represents a power of 2, splitting an address into bits results in page size that is power of 2.

(PQ) difference b/w paging and segmentation (i) page table and segment table (ii) tightly coupled system and loosely coupled systems.

Paging

- i) Memory is divided into fixed-size blocks called pages, and processes are divided into blocks of the same size called frames.
- ii) For the paging OS is accountable.
- iii) It is faster than segmentation.
- iv) It leads to internal fragmentation.
- v) Paging comprises a page table that encloses the base address of every page.
- vi) Paging is invisible to the user.

Segmentation

Memory is divided into variable size blocks called segments.

- i) For segmentation compiler is accountable.
- ii) It is slower than paging.
- iv) It leads to external fragmentation.
- v) Segmentation also comprises a segment table which encloses ~~base~~ segment number and segment offset.
- vi) Segmentation is visible to the user.

Segment Table

Page Table

i) page table is a table maintained by the OS to translate logical page address into physical frame addresses.

ii) it divides logical address space into fixed-size blocks.

iii) it uses page number and offset to calculate physical address.

iv) invisible to the user.

v) it supports paging.

vi) prone to external fragmentation

vii) it contains a frame number and access control bits.

i) A table maintained by the OS to translate logical segment address into linear addresses.

ii) it divides logical address space into variable-size blocks.

iii) uses segment number and offset to calculate linear address.

iv) visible to the user.

v) doesn't necessarily support paging.

vi) prone to internal fragmentation.

vii) it contains base address register and access control bits.

Tightly Coupled System

i) systems where components are heavily dependent on each other.

ii) communication between components is direct.

iii) components are closely integrated on each other.

iv) changes or failures in one component can significantly impact others.

v) scalability may be limited due to tight integration.

vi) modification or update are difficult to implement.

Loosely Coupled System

systems where components have minimal dependency on each other. communication between components is indirect.

components are relatively ~~independently~~ integrated on each other. changes or failure in one component have minimal impact.

scalability is typically better due to the modular nature.

modification or update are easier to implement.

(PQ) List three major activities of an OS with regard to memory management.

i) Memory Allocation: The OS allocates memory to processes if they request it. It ensures that each process gets the required memory space without interfering with other processes.

ii) Memory protection: The OS enforces memory protection to prevent processes from accessing memory that doesn't belong to them, it sets up boundaries and access permissions for each process to ensure the security of the system.

iii) Memory deallocation: When a process terminates the OS deallocates the memory to make it available for other processes.

(PQ) Discuss the Belady's Anomaly and segmentation.

* Page Replacement algorithm (First In first out)

Reference string: 1, 2, 3, 4, 1, 2, 1, 5, 1, 2, 3, 4, 5
where frame size is 3

f3		3	3	8	2	2	2	2	4	4	4			
f2		2	2	x	1	1	1	1	x	3	3			
f1		1	2	x	4	4	x	5	5	5	5			
	*	*	*	*	*	*	*	*	HIT	HIT	*	HIT		
Ref	→	1	1	3	4	1	2	1	5	1	2	3	4	5

Hits = 3
Page fault = 9

Now, where frame size is 4

f4		4	4	4	4	4	3	3	3	3	3					
f3		3	3	3	3	3	x	2	2	2	2					
f2		2	2	2	x	2	x	1	1	1	x	5				
f1		1	1	1	x	1	x	5	5	5	x	4				
	*	*	*	*	*	*	*	*	*	*	*	*	*			
Ref	→	1	1	2	3	4	1	1	2	1	5	1	2	3	4	5

Hits = 2
Page fault = 10

Belady's Anomaly: From Above example when frame size is 3 then no of hits are 3 and no of page fault is 0 but when increasing the no of page frames result in increase in the no of page faults so this phenomenon is called Belady's Anomaly.

Segmentation: Segmentation is a memory management technique where the logical address space is divided into segments that can vary in size; it allows for more flexibility in memory allocation and protection, but it can lead to fragmentation.

Optimal page replacement algorithm: LRU

Consider the page reference sequence 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 1, 2, 1, 2, 0, 1, 7, 0, 1 with four page frames. Find number of page faults using optimal and Least recently used page replacement.

\Rightarrow Optimal Page replacement

$$\text{Page Hits : } 12 \quad \text{Hit ratio} = \frac{\text{No of hits}}{\text{No of Reference}} \times 100 = \frac{12}{20} \times 100 =$$

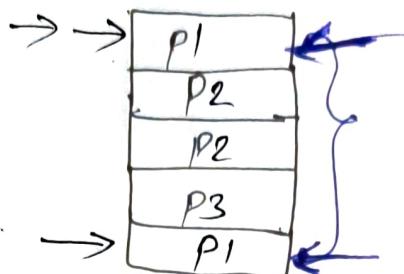
Page fault: 8 least recently used page Algo

		Least recently used																	
th	th	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	
PG	c	f4	1	1	1	1	1	4	4	4	4	4	4	1	1	1	1	1	1
m	c	f3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
c	f2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	f1	7	7	7	7	3	3	3	3	3	3	3	3	3	3	3	8	7	7

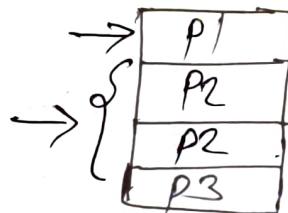
Referential string: 7 0 1 20 3 0 9 2 8 0 3 2 1 2 0 1 + 0
Page fault: 8
Hits: 12

My family PM-1
Q) Explain Local and Global page Replacement, Bad Block, File Layered Architecture and Protection Mechanisms.

Ans: Local page replacement: When a process needs a page which is not in the memory, it can bring in the new page and allocate it a frame from its own set of allocated frames only.



Global replacement: When a process needs a page which is not in the main memory, it can bring in the new page and allocate it a frame from the set of all frames, even if that frame is currently allocated to some other process; that is one process can take a frame from another.



Bad Blocks: Bad blocks refer to damaged or defective blocks on a storage device such as a hard drive or SSD. These blocks are unable to reliably store data and can lead to data loss or corruption if data is written to them.

File layered Architecture: File layered architecture refers to the organization of file systems into multiple layers, each layer responsible for different aspects of file management. Common layers include the user interface layer, device drivers layer. This architecture helps in modularizing and simplifying the design and implementation of file systems.

- protection mechanisms: There are various mechanisms to protect files.
- i) (ACLs) Access control lists: ACLs define who can access a file and what permissions they have (Read, write, execute).
 - ii) File ownership: It associates a user or group with a file, influencing access permissions based on predefined rules.
 - iii) Encryption: Searches the contents of a file using a key making it unreadable without proper decryption.
 - iv) checksums: Generate a mathematical value representing the file's content. Any changes to the data will alter the checksum, allowing for data integrity verification.

Q Share the need and significance of paging concept.

Paging is a memory management technique that is used to retrieve processes from secondary storage to main memory.

Need of ~~virtual~~ paging

- Eliminates contiguous allocation: Traditional memory allocation required contiguous blocks of memory for a process. This can be inefficient, as it may lead to memory fragmentation even if there's enough free space. Paging divides memory into fixed-size blocks (pages). This allows for non-contiguous allocation, making better use of available memory.
- supports virtual memory: Paging is a foundation for virtual memory. Virtual memory allows processes to use more memory than physically available by swapping pages between RAM and secondary storage.
- simplifies memory management: Paging simplifies memory allocation for the OS. The OS only needs to keep track of which pages are in memory and where they are located in secondary storage. This simplifies memory management compared to keeping track of

Protection: Paging provides a level of memory protection. Each process has its own page table, which maps virtual addresses to physical addresses. This restricts processes from accessing memory that doesn't belong to them.

Significance of paging

- Efficient memory utilization:
 - Reduces fragmentation: By dividing memory into fixed-size pages, paging eliminates external fragmentation.
- Enable virtual memory:
 - Run large programs: It allows processes to use more memory than physically available by swapping pages to secondary storage and bringing them back when needed.
 - Supports Multitasking: With virtual memory, multiple processes can be loaded concurrently, enabling smooth multitasking.
- Security and Process Isolation:
- Memory Protection: Paging provides a level of memory protection. Each process has its own page table, which maps virtual addresses to physical addresses. This restricts processes from accessing memory that doesn't belong to them.

- ### Process of Paging in case of nesting of page tables
- i) Virtual Address Translation: When a process access memory address to a physical address. The CPU first translates this virtual address to a physical address using page table.
 - ii) Page Table Structure: In nesting Paging there are multiple levels of page tables. Each level translates part of the virtual address to a physical address.
 - iii) Guest Page Table: The guest page table translates the virtual address used by the guest ~~host~~ operating system.
 - iv) Page Fault Handling: If a page table entry is not present in memory it results in page fault. It handles this by loading the required page table.

Unit - 7

case studies

Q Elaborate the major features of window, unix and linux operating system.

→ windows:

- **User-friendly Interface:** Windows operating systems are known for their graphical user-friendly graphical interface (GUI), making them more accessible for the wide range of users.
- **Wide Software Availability:** Windows has the most extensive software library compared to Unix and Linux. This includes popular productivity tools, games and creative applications.
- **Cost:** Windows is a proprietary operating system, meaning you need to buy a license to use it legally.
- **Security:** ~~While~~ Windows can still be more ~~more~~ vulnerable to malware compared to Unix and Linux due to its widespread use.

→ Unix

- **(CLI) Command Line Interface:** Unix primarily relies on text-based commands for interaction. This requires some technical knowledge to navigate and use effectively.
- **Multitasking:** Unix has been around for a long time and well-established for multitasking and handling multiple users efficiently.
- **Cost:** Unix can be licensed software, with costs varying depending on the specific version. There are also free versions available for development purposes.
- **Security:** Unix is known for its robust security features and user permissions system, making it a popular choice for servers and mission-critical applications.

→ Linux:

- **Open-source:** Linux is a free and open-source operating system. This source code is publicly available, allowing for customization and community development.
- **Customizability:** Due to its open-source nature, Linux offers a high degree of customization. Users can modify the system to their specific needs and preferences.
- **Cost:** Since it's open-source, Linux itself is free to use and modify. However, some Linux distributions (versions) might have paid support options.
- **Security:** Similar to Unix, Linux is known for its strong security features.

Q. Categorise the major difference among windows, unix, linux operating systems in details.

Features	Windows	Unix	Linux
Kernel	Monolithic NT Kernel	Unix ; BSD kernel	Monolithic
User-interface	Graphical User Interface (GUI)	Command-line interface	Command line and Graphical user interface
File system	NTFS, FAT32, exFAT	UFS, ZFS	(ext4, XFS, Btrfs) various
License	Proprietary	BSD, GNU, GPL	Open source
Hardware support	Broad range of hardware support	Limited hardware support	Broad range of hardware support
Command line	Command prompt	Terminal	Terminal (Bash)
System Administration	GUI-based tools	Command-line tools	Command-line and GUI tools
Networking	Integrated networking tools	Basic networking tools	Comprehensive networking tools

Security	vulnerable to malware and viruses	Relatively secure	Relatively secure
Customization	Limited customization options	Highly customizable	Highly customizable
Package Management	MSI, EXE, Windows Store	Package managers	Package managers
Cost	Typically requires	Often free (some commercial variants)	Often free (open-source)

Unit - 5 ⇒ Secondary storage

(PQ) What is disk scheduling? Explain, FCF and SCAN disk scheduling algorithm, Advantages & Disadvantages of disk scheduling. Disk scheduling is done by operating system to schedule I/O requests arriving for the disk. Disk scheduling also known as I/O scheduling. It's aims to minimize the seek time and optimize disk performance.

- * **Seek time:** The time taken to reach up to desired track where the data is to be read or write.
- * **Rotational latency:** Rotational latency is the time taken by the desired sector of the disk to rotate into a position so that it can access the read/write heads.
- * **Transfer time:** Transfer time is the time to transfer the data.
- * **Disk Access Time:**

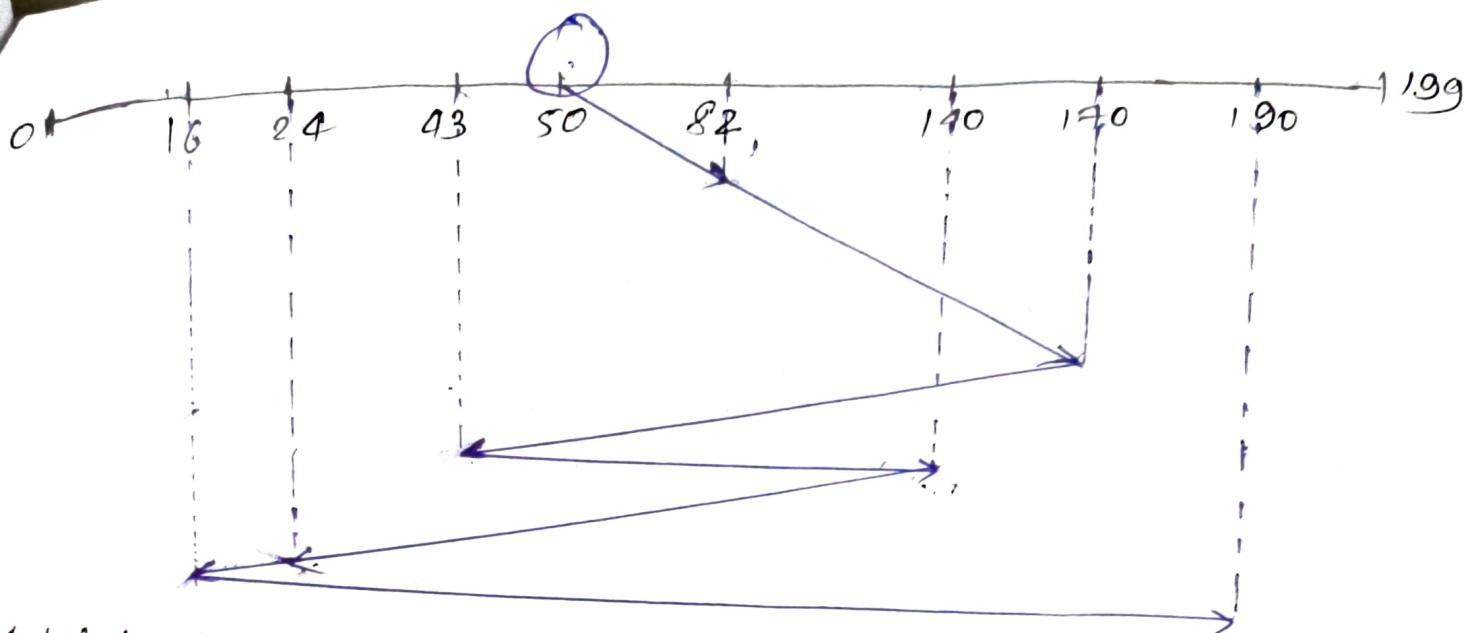
$$\text{Disk Access Time} = \text{Seek Time} + \text{Rotational Latency} + \text{Transfer Time}$$

$$\text{Total Seek Time} = \text{Total Head Movement} \times \text{Seek Time}$$

- * There are several disk algorithms,
- ① FCFS ② SSTF ③ SCAN ④ C-SCAN ⑤ LOOK
⑥ GLOOK

i) **First come first serve:** FCFS is the simplest of all disk scheduling algorithms, in FCFS the requests are addressed in first come first serve basis in the disk queue.
 Ex: A disk contains 200 tracks (0-199). Request contains position of Read/write head = 50, calculate total no of tracks movement by Read/write head.

Given ⇒ tracks no: 82 170 43 190 24 16 190
 head = 50
 movement of tracks = ?



So, total head movement = $(82 - 50) + (170 - 82) + (170 - 43) + (140 - 43) + (140 - 24) + (24 - 16) + (190 - 16) = 32 + 88 + 127 + 97 + 116 + 8 + 174 = 642$

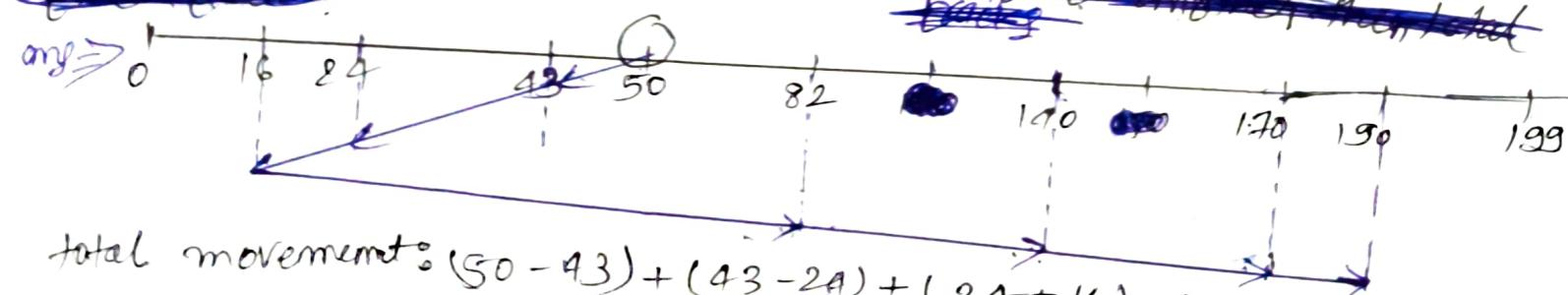
Advantages of FCFS: • Every request gets a fair chance
• No indefinite postponement

Disadvantage: • It does not optimize seek time

• May not provide the best possible service

② SSTFC shortest seek time first: In SSTF, requests having the shortest seek time are executed first. SSTF is an improvement over FCFS as it decreases the seek time.
Ex ⇒ A disk contains 200 tracks (0-199). Request queue containing tracks no 82, 170, 43, 140, 24, 16, 190 respectively. Current position of R/W head = 50. Calculate total no of tracks moved by R/W head using shortest seek time first.

~~total seek time = time taken to move from one track to another track~~



total movements: $(50 - 43) + (43 - 24) + (24 - 16) + (16 - 82) + (82 - 190) + (190 - 170) + (170 - 140) = 7 + 19 + 8 + 66 + 58 + 30 + 20 = 208$

* Advantages of shortest seek Time First:

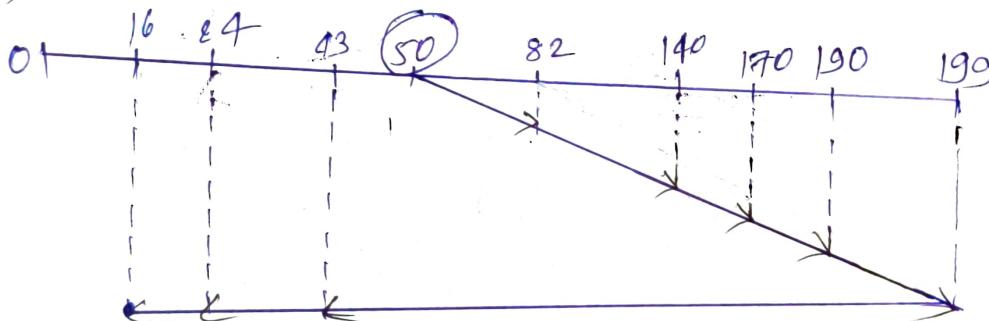
- The average Response Time decreases
- Throughput increases

Disadvantages

- overhead to calculate seek time
- leads to starvation

*③ SCAN: In SCAN algorithm the disk arm moves in a particular direction till end of track and after reaching the end of track, it reverse its direction and again service the request. This algorithm works as an elevator and is hence also known as an elevator algorithm.

Ex: A disk containing 200 tracks (0-199) Request queue containing track no. 82, 170, 43, 140, 24, 16, 190 respectively current position of R/W head = 50, calculate total no of tracks movement by R/W head using SCAN? Direction is towards large value?



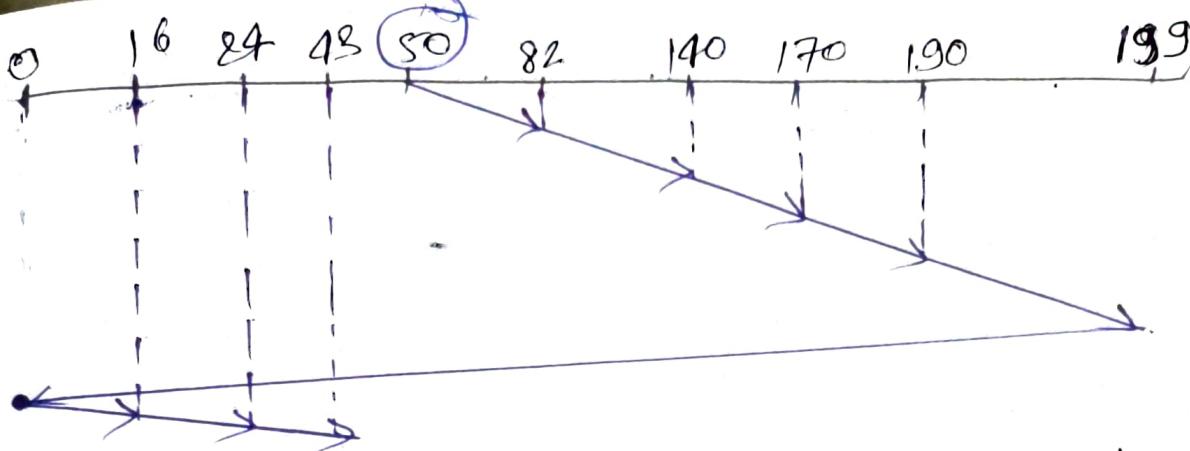
$$\text{total movement} = (82 - 50) + (140 - 82) + (170 - 140) + (190 - 170) + (199 - 190) + (199 - 43) + (43 - 24) + (24 - 16) = 332$$

• Advantage: • High throughput.

• low variance of response time

• Disadvantage: • leads to starvation • High Disk head movement

*④ C-SCAN: Ex → A disk containing 200 tracks (0-199) Request queue containing track no 82, 170, 43, 140, 24, 16, 190 respectively Current position of R/W head = 50, calculate total no of tracks movement by R/W head using C-SCAN? (Direction is towards large value)

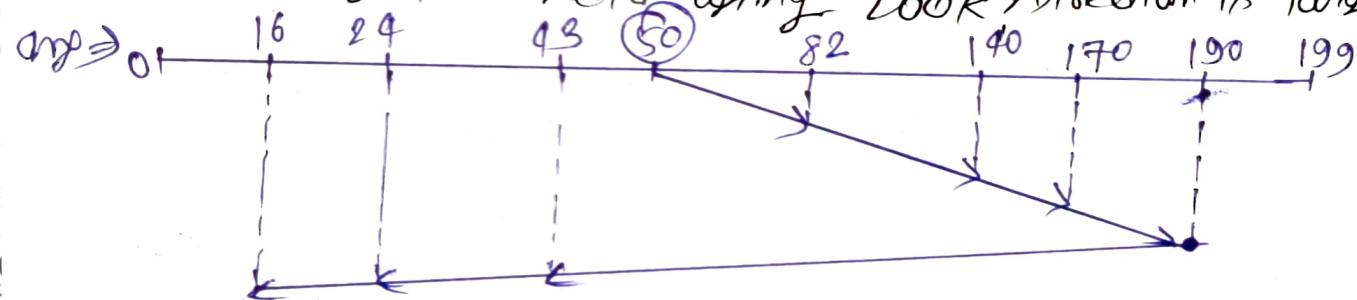


total movement : $(199 - 50) + (199 - 0) + (43 - 0) = 391$

Advantage: It provides more uniform wait time compared to SCAN.
Reduce Head movement.
Prevention of starvation

Disadvantage: increased response time compared to SCAN

⑤ **LOOK:** A disk containing 200 tracks (0-199) requests contain tracks no 82, 170, 43, 140, 24, 16, 190 respectively. Current position of R/W head = 50 calculate total no of tracks movement by R/W head using LOOK, direction is towards larger value.

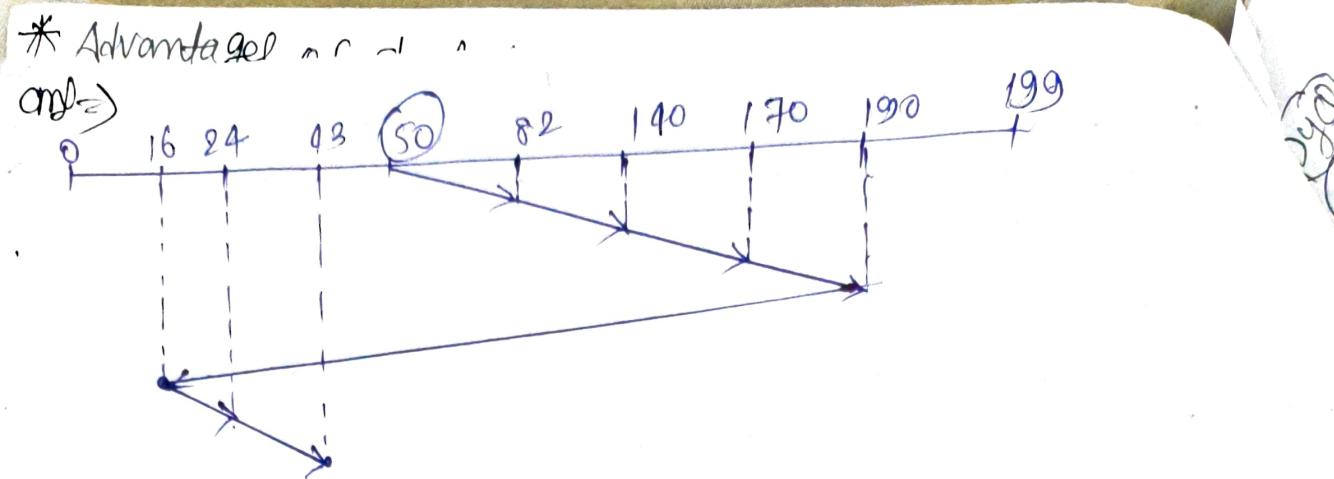


total movement : $(190 - 50) + (190 - 16) = 314$

Advantage: Reduce Head movement than SCAN.

Disadvantage: Potential for starvation chance

⑥ **c-LOOK:** A disk contains 200 tracks. Requests are of R/W head = 50, calculate total no of tracks movement by head using c-LOOK? Direction is towards large value.



$$\text{Total movement} = (190 - 50) + (190 - 16) + (43 - 16) = 391$$

Advantage :

- prevention of starvation.
- Reduce Head movement.

Disadvantage : • Increased response Time than LOOK.

(Q) why rotational latency is not considered in disk scheduling?

\Rightarrow Most disk scheduling does not consider rotational frequency because, in a most modern system, the actual physical location of blocks ~~are~~ are not available. it is very large compared to seek time. Rotational latency depends on the rotational speed of the spindle.

File Management

- (i) Explain the following
i) File types ii) file operation
iii) file attributes

~~ans~~ File: A file is a named collection of data that is stored on a storage device, such as a hard drive or SSD, files can contain various types of data, including text, images, programs and more.

File types:

- Regular files: ~~These, contain data, such as text files, images, executables etc.~~
- Directories: ~~Containers for other files and directories, used for organizing files in a hierarchical structure.~~

Types of file:
i) Text file: plain text, readable with any text editor (e.g., .txt, .cpp, .html)
ii) Executable file: contain instructions for the CPU to execute programs (Ex → .exe on windows, .app on macOS)
iii) Document file: created by specific software (Ex → .docx, .pdf).

- iv) Image file: ~~Created by specific software~~ store digital pictures (ex: .jpg, .png, .bmp)
v) Audio file: contain sound data (.mp3, .wav, .ogg)
vi) Video file: combine images and audio for moving pictures (ex → .mp4, .mkv)
vii) Archive file: compress multiple files into a single package (ex → .zip, .rar)

- * File operations:
 - i) Create: Establish a new file for storing data.
 - ii) Read: Access the content of a file.
 - iii) Write: Modify the content of a file or create a new one.
 - iv) Delete: Remove a file from the storage device.
 - v) Rename: Change the name of a file.
 - vi) Copy: Create a duplicate copy of a file.
 - vii) Move: Change the location of a file within the storage device.
 - viii) Open: Prepare a file for use by a specific program.
 - ix) Close: Release resources associated with a file after use.

File Attributes:

- Name: The user assigned filename (eg: Mydocument.txt)
- Extension: The part after the period in the filename, indicating the file type (eg: .txt)
- Size: The amount of storage space occupied by the file, measured in bytes, MB, GB or TB.
- Location: The directory structure where the file resides (Ex → C:\Documents\Myfile\mydocument.txt)
- Creation Date and Time: When a file was first created.
- Modification Date and Time: When the file content was last changed.
- Type: Tells type of the file whether it is .txt, .app, .png, .jpg, .mp4.

(Q) What are the different access methods of files? How are they implemented?

(Ans) There are three main file access method in OS:

- ① Sequential Access: This method reads or writes data in linear, one record at a time, starting from the beginning of the file. Imagine reading a book - you typically do from page

page 1 to end.

2. Direct Access : This method allows jumping directly to any specific location in the file to read or write data. It's like flipping to a particular page in a book with an index. Direct access is faster for retrieving specific data points within a large file.
3. Indexed Sequential Access : This method combines both sequential and direct access. It maintains an index like a book's table of contents that maps data records to their storage location. This allows for both faster retrieval of specific data using the index and sequential processing of the file.

Ques

Q) List three major activities of an operating system with regard to memory management.

Sol: ~~Ques~~ three major activities of an os.

i) Allocation : This involves assigning portions of memory to processes that request them. The OS needs to decide which memory space.

ii) Deallocation : This involves reclaiming the memory that is no longer in used by a process. When a process terminates OS must reclaim those segments to make them available for other processes.

iii) Memory Protection : The OS ensures that processes do not interface with each other, preventing unauthorized access.

Memory locations that mechanisms enforce a class control policies to prevent processes from accessing memory locations that they do not own or not authorized.

(PQ) Calling PQ 2.1 (before this page)

(PQ) what are the points to be consider in file system design?

Explain Unlinked list allocation & index allocation in detail.

There are several key points to consider when designing a file system

- ① **File structure:** This refers how files and directories are organized
- ② **File operation:** The system should efficiently support basic operations like creating, reading, writing, deleting and renaming files.

- ③ **Memory allocation:** This refers to how the file system allocates memory to files.

- ④ **Security:** features like encryption to protect sensitive data

- ⑤ **Error handling and recovery:** The ability to handle errors that might occur during file operations and recover data if possible.

- ⑥ **Scalability:** The ability to large data and growing number of files efficiently.

Calling → PQ 2.1