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 (a) Bulky and complex app. [Hardware interaction code must be in app's code]

(b) Resource exploitation by 1 App.

(c) No memory protection.

* functions of Operating system.

(i) Access to the computer hardware.

(ii) interface between the user and the computer hardware.

(iii) Resource management (memory, device, file, security, process),

(iv) Hides the complexity of the hardware.

(v) 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 H/w & S/w & user and computer.

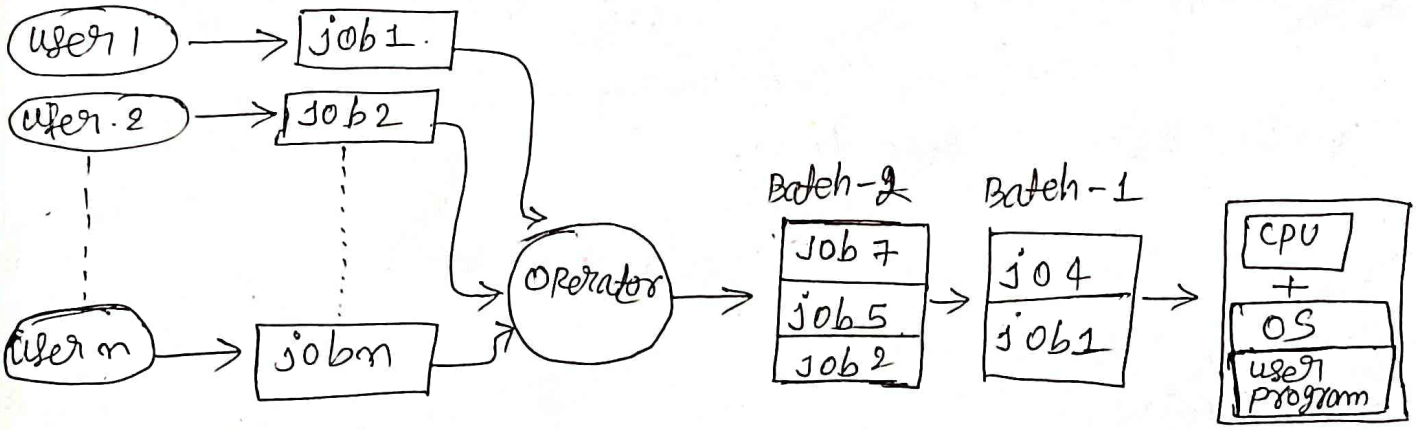
AP

* Types of operating system

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

BMTDRP → TRPBMD
DPTRMB

(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 a batches. It is responsibility of operator to sort jobs with similar needs. Ex ⇒

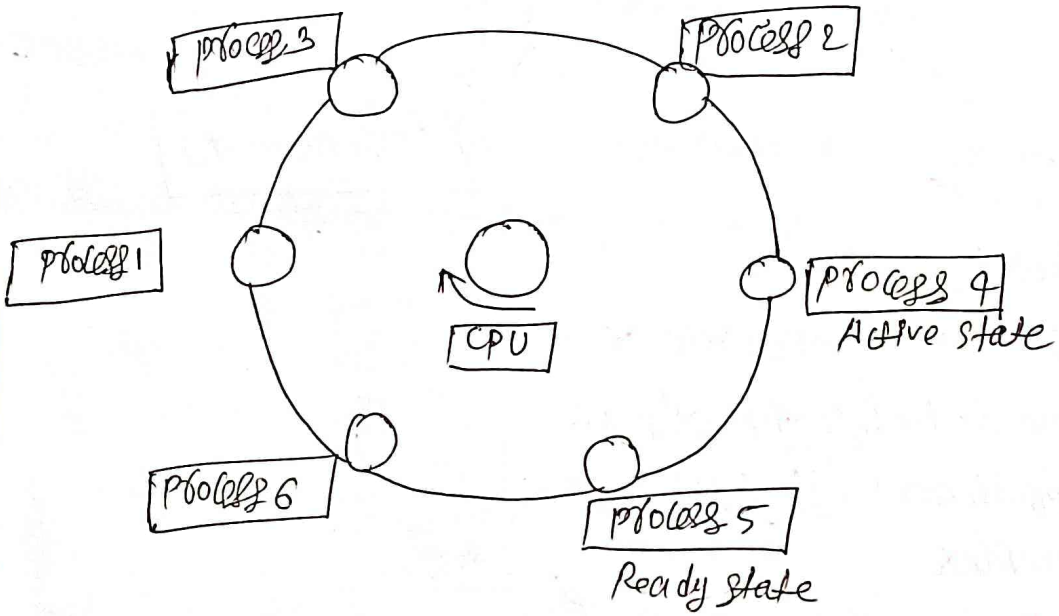


• pros: (i) Multiple users can share the batch.
(ii) very less time take in batch system.
(iii) Easy to manage large work in batch system.

• cons: (i) difficult to debug
(ii) Sometimes costly
(iii) Other jobs will have to wait for an unknown time if any job fails.

Application : (i) Bulk Data processing
(ii) Media processing
(iii) Transaction processing

(ii) Time-sharing operating system: Time sharing operating system is a 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 O.S switches over ~~the~~ to the next task. Ex=

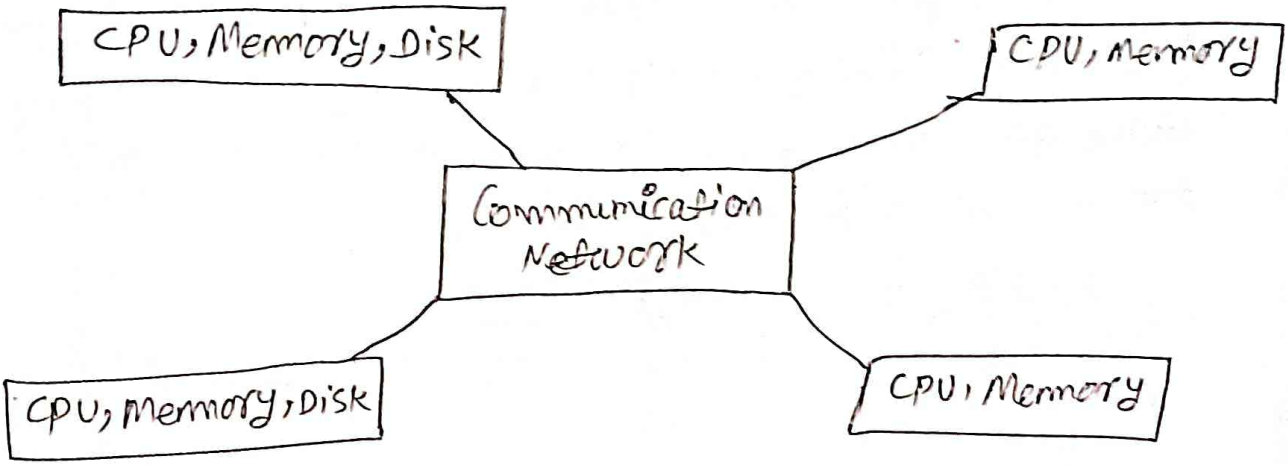


(i) pros: (i) each task gets equal opportunity.
(ii) each task gets equal time.
(iii) Multitasking can be performed.
(iv) CPU time can be reduced.

cons: (i) Reliability problem.
(ii) Data communication problem.
(iii) security problem.

Application: (i) Telecommunication system
(ii) Research and development
(iii) Resource optimization
(iv) Multi-user Environment

(iii) Distributed operating system: Distributed operating system is a type of operating system, in which we have various systems and all these 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. (iii) high cost
(ii) Highly complex.

Application: (1) High-performance computing
(2) cloud computing

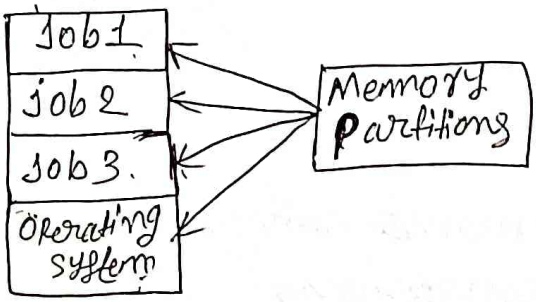
(iv) Real-time operating system: Real-time operating system is a operating system in which operating system respond quickly, Real time operating system used in missile system, use in hospital, ATC (air traffic control). this OS provides maximum efforts to its task with quick response. Ex ⇒

pros: (i) Provide a quick response
(ii) used in organization (Nasa & ISRO)
(iii) use in ATC

Cons: ① Too costly.
② Complexity.
③ Limited Hardware support.

Application: ① Telecommunication
② Medical Devices.
③ Aerospace and Defense.
④ Automatic systems.

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



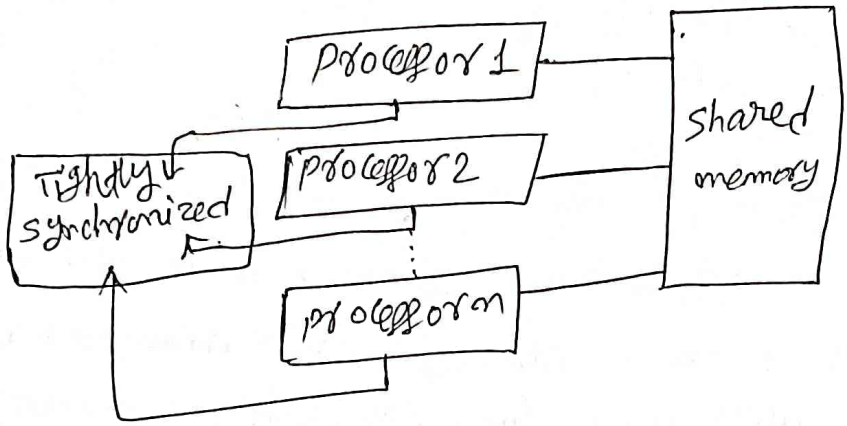
Pros: ① increased CPU utilization
② Resources utilization

Cons: ① Complexity
② Difficulty in debugging

Application: ① Transaction processing
② Batch processing
③ Scientific ~~computing~~ 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 programs into multiple fragments and processing these fragments at the same time.

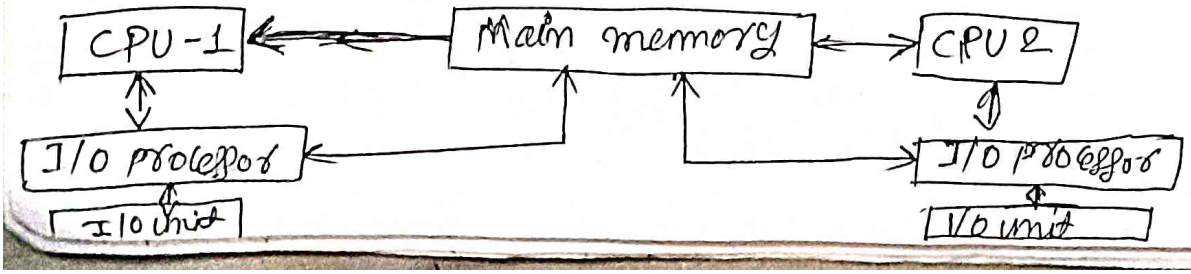


Pros: ① Increased performance ② Resource Sharing
② High throughput

Cons: ① Complexity
② too costly

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

(vii) *Multi-processing OS: Multi-processing operating system is a operating system that use more than one CPU to improve performance. Multiple processors work parallely 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



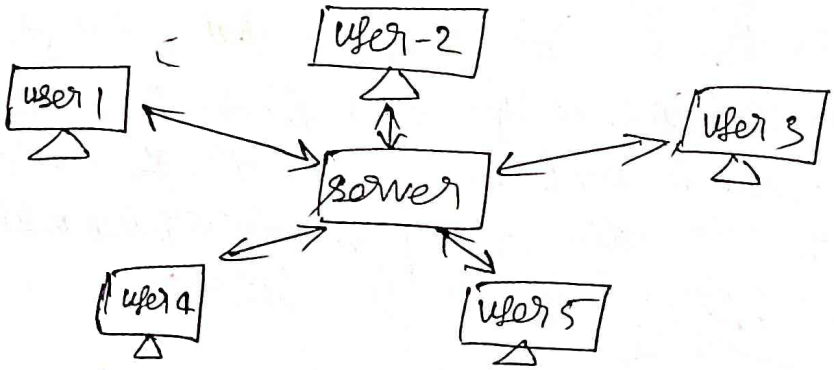
Pros: failure of one processor does not effect the functioning of other processors.

(ii) parallelism, it allows to work in parallely.

Cons: (1) More complexi
(2) Programming challenge.

Application: (1) High - Performance computing
(2) 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: (1) Highly stable due to central server.
(2) remote access.

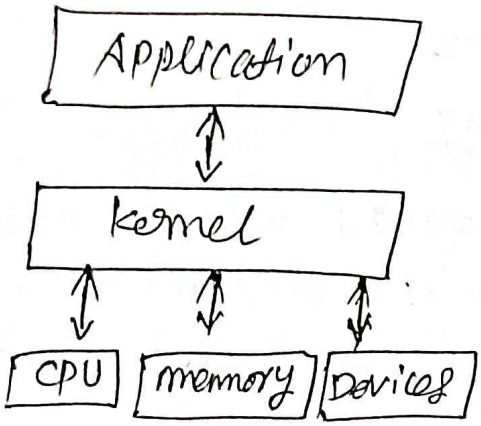
Cons: (1) Too cost
(ii) Regular updating and maintenance required

Application: (1) Manage user Acount
(2) Manage resource sharing
(3) Remote access.

Operating systems	pros	Cons	Application
① Distributed operating system	① Resource sharing ② high throughput ③ less time takes	① Highly complex ② High Cost	cloud computing
② parallel operating system	(i) High throughput ② increased performance ③ Resource sharing	① High complexity (ii) Too costly	Machine learning AI Data base Management system
③ Timesharing operating system	① Each tasks gets equal opportunity ② each tasks gets equal time	① Reliability problem ② Security problem	Telecommunication research & development
④ Real time operating system	(i) provides quick response (ii) Used in NASA, ISRO orgn (iv) use in ATC	① Too complex (ii) Too costly	① Medical Devices ② Aerospace and defense ③ Automatic system
⑤ multiprogramming operating system	① Increased cpu utilization ② Resource utilization	① complexity ② overhead to switching b/w diff programs	scientific computing
⑥ Batch operating system	① multiple users can share the ~~batch~~ job ② less time consume	① leads to starvation if any job gets fail ② difficult to debug	Transaction processing Media processing

* Kernel and shell

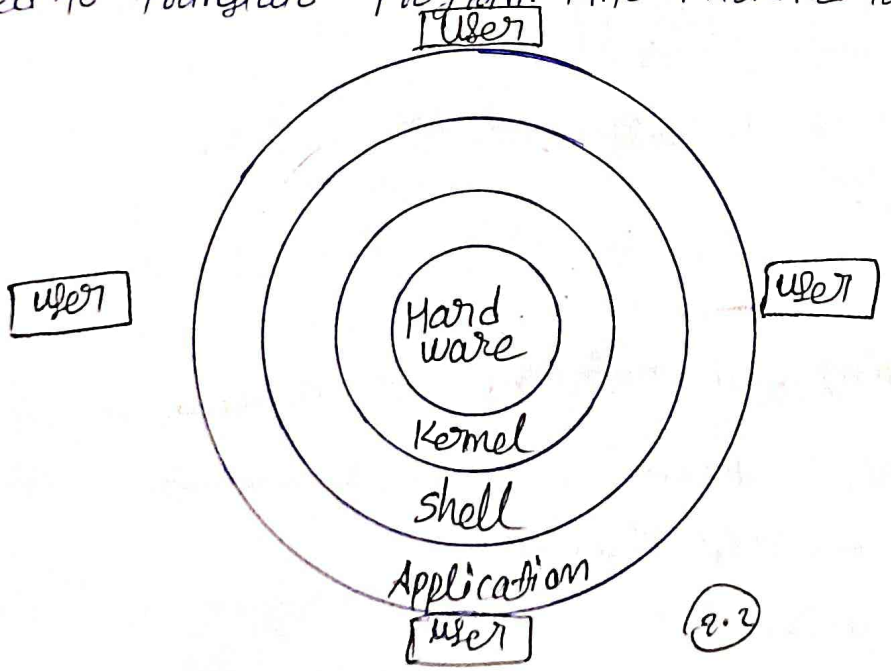
Kernel: Kernel is the heart and core of the operating system. It acts as an interface between the application and data processing at hardware level, kernel lies in the center of the operating system, It is responsible for memory management, process scheduling and device drivers. It operates at a lower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



8.1

between Application and kernel

* 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/output to output shell. It is also known as interpreter which is used to translate program into machine language.



8.2

Agitos missing

Difference between shell and kernel

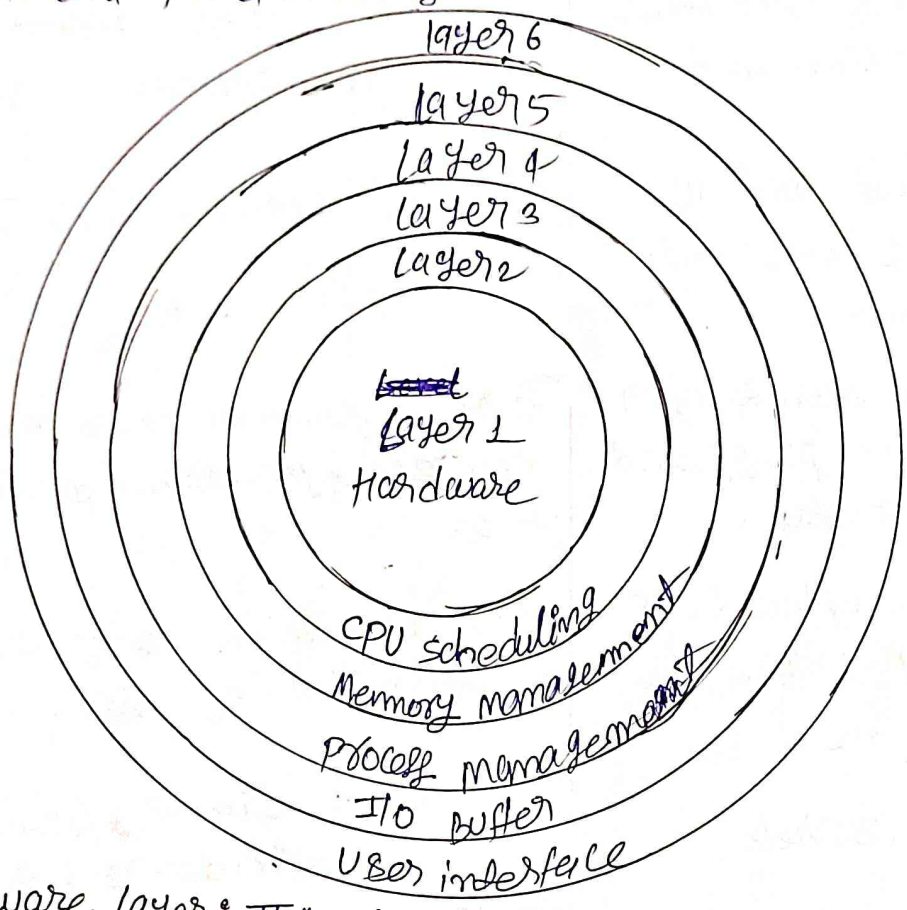
Shell	Kernel
(i) It is the outer layer of O.S.	(i) It is the inner layer of O.S
(ii) shell provides interface between application and kernel.	(ii) Kernel provides interface b/w shell and hardware.
(iii) Shell interact with user. and translate language into machine language.	(iii) Kernel interact with hardware
(iv) It's function to execute user commands, manage files, and provide user interface.	(iv) It's function is to manage memory, processes, device drives of OS.
(v) It can be replaced or customized by the user	(v) It cannot be easily replaced or customized by user.
(vi) Example: Bash, powershell	(vi) Example: Linux kernel, windows kernel
(vii) diagram 2.1	(vii) diagram 2.2

* Operating system services: (1)

(1) program execution → load programs into memory and execute them.
(2) Input/output operations → It handles comm b/w external device manage I/O device
(3) File management → manage files on storage device → creation, deletion, update
(4) Memory management → Allocate & deallocate memory space as need by prog
(5) Process management → manage process → creation, scheduling, termination
(6) Resource management → efficient allocate sys resources such as CPU time, disk space
(7) Time management → keeps track of time scheduling tasks, manage clocks
(8) Error handling → Detect and manage errors that may occur & pe.
(9) User interface → provide user interface b/w co and user.
such as graphical command line interface.

(7) Q. General structure of operating system / How layered architecture of operating system works and what are its various functions?

ans ⇒ Layered architecture of an operating system (OS) organizes its components into distinct layers, each responsible for different functionalities.



(i) Hardware layer: This is the lowest layer, interacting directly with the physical hardware components directly with the physical hardware components like the keyboard, mouse other peripherals devices such as CPU, Main memory, secondary memory (SSD)

(ii) CPU scheduling layer: This layer manages the execution of multiple processes sharing a single CPU. It decides which process should allocate the CPU and for how long. optimizing resource utilization and responsiveness.

(iii) Memory management layer: Responsible for managing the system's memory resources, including RAM, it allocates memory space to processes, keeps track of memory usage, and handles memory fragmentation.

(iv) Process management layer: It manages processes and. It handles process synchronization, inter-process communications and provides mechanisms for processes to interact with each other.

(v) I/O Buffer Layer: It handles input and output operations between the OS and external devices like keyboard, mouse and other peripheral devices. It provides buffering and caching mechanisms to optimise I/O performance and handles data transfer between devices.

(vi) User interface layer: This is the topmost layer that interact directly with users, it provides users to an interface to interact using graphical user interfaces, command line interface and using other components.

(PYQ) Some computer systems do not provide a privileged mode of operation in hardware. Is it possible to construct a secure operating system for these computer systems? justify your answer.

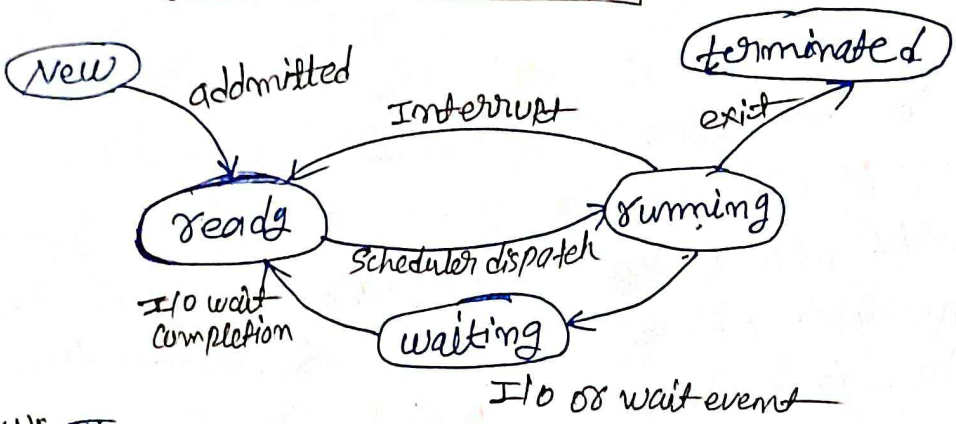
ans ⇒ Yes, it's possible to construct a secure operating system for computer systems that lack hardware privileged mode but it would rely heavily on software-based security mechanisms. However achieving the same level of security as hardware supported privileged mode can be challenging and might require additional layers of protection and hard software design to prevent unauthorized access and ensure system integrity.

Unit - 2 → process management

(PYQ) 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 process 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, register value, 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 deallocating CPU time.

* Scheduling Algorithms:

Primitive Algo	NON primitive Algo
① Shortest Remaining Time first	① First come first serve
② Longest Remaining Time first	② Shortest job first
③ Round Robin	③ Longest job first
④ priority	④ Highest Response Ratio Next (HRRN)

Non primitive ~~at~~ (WT) (RT) some of stats priority (2:1) Initializing

* Arrival time (AT): It is time at which the process enter the ready queue.

~~① First come first serve~~:

* Burst time (BT): 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.

TAT = CT - 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 response.

~~RT = R - AT~~

RT = CPU first time - AT

NON- Preemptive

① First come first serve :

Find average Turn around time and average waiting time

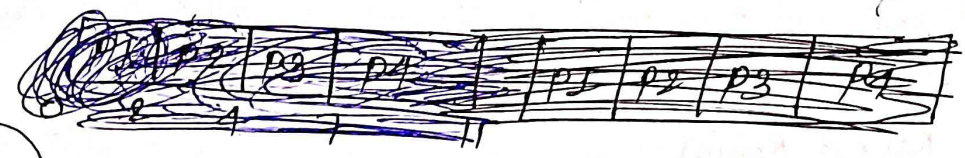
Process No	AT	BT	CT	TAT	WT
P1	0	2	2	2	0
P2	1	2	4	3	1
P3	5	3	8	3	0
P4	6	4	12	6	2

avg TAT = ? TAT = C.T - AT, CT = when process complete

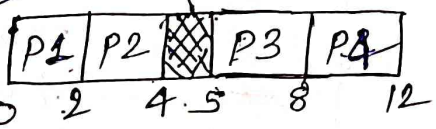
avg WT = ? WT = TAT - BT,

NON Premitive

Gantt chart :



idle time



$$\text{Avg TAT} = \frac{2+3+3+6}{4} = \frac{14}{4} = 3.5 \text{ unit}, \quad \text{Avg WT} = \frac{0+1+0+2}{4}$$

$$\text{Avg WT} = \frac{3}{4} = 0.75 \text{ unit}$$

---

② Shortest Job first : Calculate average turn around time and avg waiting time.

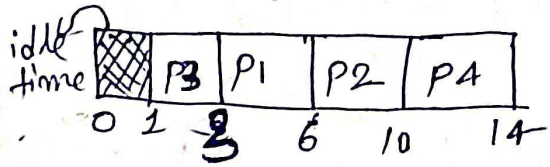
PROCESS NO	AT	BT	CT	TAT	WT
✓ P1	1	3	6	5	2
P2	2	4	10	8	4
✓ P3	1	2	3	2	0
P4	4	4	14	10	6

$$\text{avg TAT} = \frac{5+8+2+10}{4}$$

$$\text{avg TAT} = \frac{25}{4} = 6.25 \text{ unit}$$

$$\text{avg WT} = \frac{12}{4} = 3 \text{ unit}$$

Gantt chart :



AT ? → BT → FCFS

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	21	15	19
P3	3	6	9	0	6
P4	4	8	17	5	13

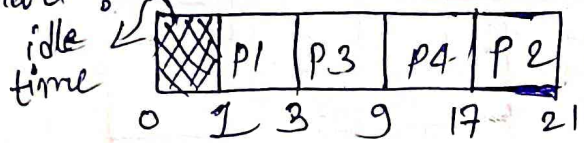
$$\text{Avg TAT} = \frac{2+19+6+13}{4}$$

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

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

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

Gantt chart:



FCFS → BT → FCFS

④ 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	07

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

P3 ratio $= \frac{5+4}{4} = \frac{9}{4} = 2.25$

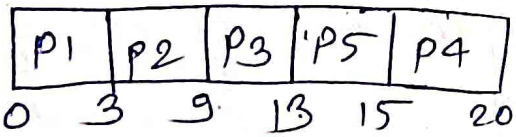
P4 $r = \frac{3+5}{5} = \frac{8}{5} = 1.6$

P5 $r = \frac{1+2}{2} = \frac{3}{2} = 1.5$

P4 $r = \frac{7+5}{5} = \frac{12}{5} = 2.4$

P5 $r = \frac{5+2}{2} = \frac{7}{2} = 3.5$

Gantt chart



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

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

Primitive

(1) Shortest remaining time first:

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

Gantt 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 13

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

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

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

(2) Longest remaining time first:

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

$$\text{Avg TAT} = \frac{9+9+7+7}{4} = \frac{18+14}{4} = \frac{32}{4} = 8U$$

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

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

Gantt chart

P1	P2	P2	P2	P3	P2	P3	P4	P1	P2	P3	P4

0 1 2 3 4 5 6 7 8 9 10 11 12

③ Round Robin:

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

Given TQ = 2

Ready queue: ~~P1~~ ~~P2~~ ~~P3~~ ~~P1~~ ~~P4~~ ~~P2~~ P1

0

Running queue: P1 | P2 | P3 | P1 | P4 | P2 | P1

0 2 4 6 8 9 11 12

* No of Context switching = 6

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

Avg WT = $\frac{7+6+2+4}{4} = \frac{19}{4} = 4.75$

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

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

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

Gantt Chart

P1	P2	P3	P3	P4	P2	P1

0 1 2 3 4 5 8 12

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

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

* difference between process and thread

Process	Thread
(i) A process is an instance of program that is being executed.	(i) thread is a smallest unit of execution within a process.
(ii) Context switching is slow.	Context switching is faster.
(iii) A process is a heavy weight task.	Thread is a light weight task.
(iv) Process control block (PCB) control process state.	Thread control block (TCB) control thread state.
(v) Example: Running browser Application.	Example: different tabs within a browser.
(vi) It is more secure because it donot share any resources	It is less secure because it shares resources.

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

User level Thread	kernel level thread
(1) It is typically faster.	It is slower.
(2) Context switching is faster.	Context switching is slower.
(3) It is more efficient..	It is not so efficient.
(4) It is easy to manage	It is not easy to manage.
(5) Hardware support is needed.	No hardware support is needed.

* define Context switching.

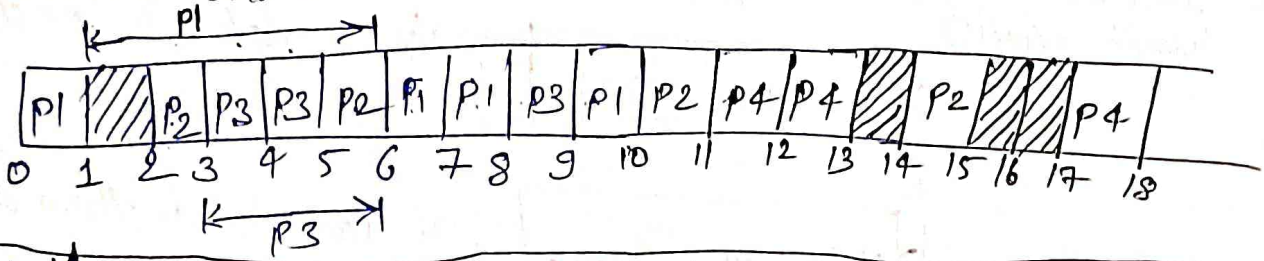
Context switching refers to the process of saving state of a running process so that it can be restored and loading another process and run it.

⑤ Mix-burst Priority scheduling:

priority	process No	AT	CPU	IO	CPU	CT	TAT	WT	RT
2	P1	0	1	5	3				
3	P2	2	3	3	1				
1	P3	3	2	5	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



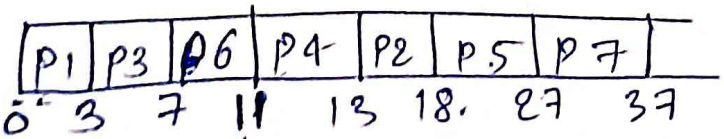
Q.1 ★ Non-primitive priority scheduling
Calling
Considering ~~lower the no higher the priority~~

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

Avg TAT = 13
Avg WT = 7.71
Avg RT = 7.71

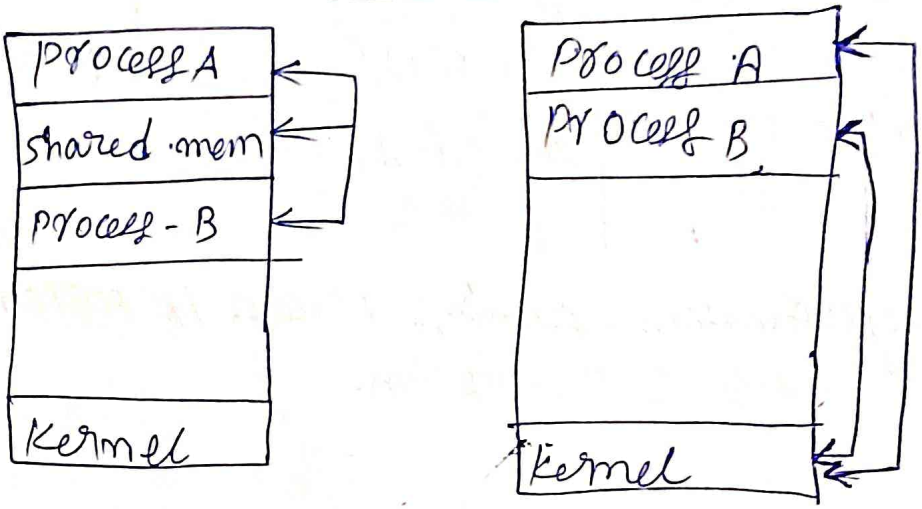
1 2 3 4
AT → Priority → BT → FCFS

Gannt chart:



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

S.M



* Inter process communication: Inter process communication is a communication system that allows process to communicate with each-other and synchronize their actions.

* process synchronozition: 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) Coperating process (ii) Independent processes

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

(ii) Coperating process: In the coperating process processes are dependent on each-other processes. Ex→ IRCTC

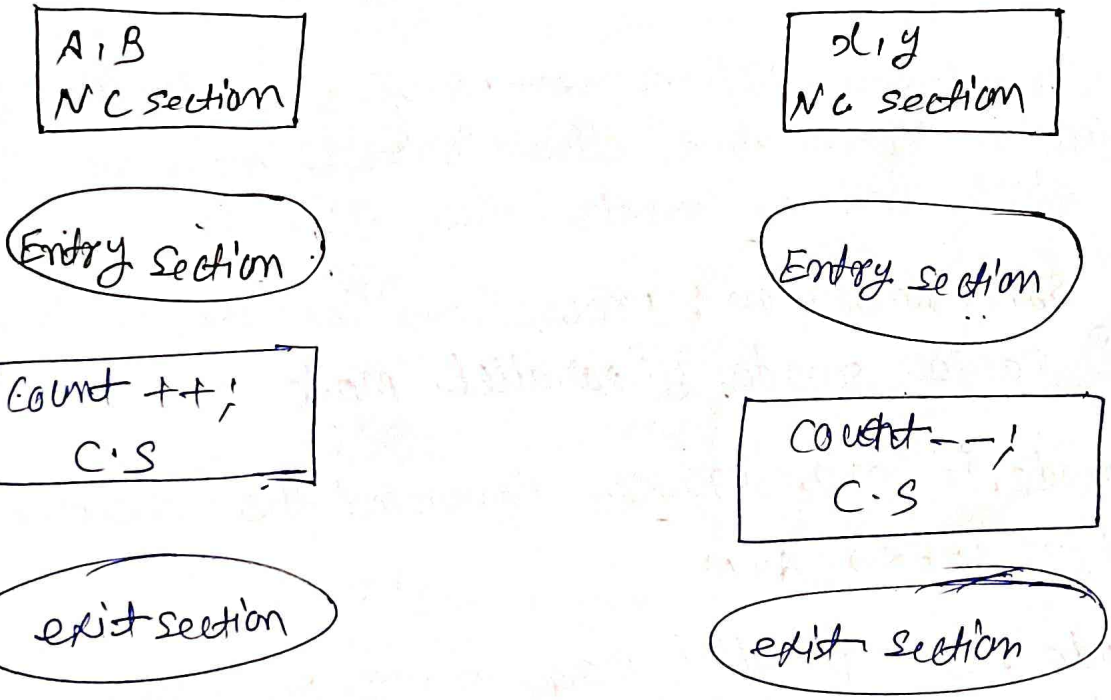
Synchronization issue :

int shared = 5 6 4 → race condition

```
Process → P1                    Process - P2
int x = shared;                 int y = shared;
x++; 5 6                        y--; 5 4
sleep(1);                       sleep(1);
shared = x;                     shared = y;
    6                               4
```

↓ time

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



C.S : Critical section.
N.C : None - critical section.

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

Condition for entry and exit section & Process synchronization

(i) Mutual exclusion
(ii) progress
(iii) Bounded -weight wait
(iv) No assumption related to H/w speed.

```
int shared = 5

process P1                    | process P2
int x = shared;               | int y = shared;
x++;                          | y--;
sleep(1);                     | sleep(1);
shared = x;                   | shared = y;
```

(1.1)

Entry section

```
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 / Release()

*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

Semaphore of two type

(1.1) (1) Counting Semaphore (+∞, -∞)

(2) Binary Semaphore. (0,1) (1.2)

(1.2) Binary Semaphore

```
Down (semaphore s)
{ if (s.value == 1)
  { s.value = 0;
  }
  else
  {
  Block this process
  and place in suspended
  list, sleep();
  }
}
```

```
.up (semaphore s)
{ if (suspended list is empty)
  { s.value = 1;
  }
  else {
  select a process fraum
  suspended list and
    wake up();
  }
}
```

(Q5) 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 accesses this section at a time, as concurrent access could lead to unpredictable results.

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

How ~~semaphore~~ Binary semaphore Resolving synchronization issues: (11/25) *

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 to 2~~

```
Down (semaphore S)
{ if (s.value == 1)
   { s.value = 0;
   } return;
   else
   { put the process in the suspended list;
      sleep();
   }
}
```

(pyq) Demonstrate the four criterias 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 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 bound wait because it should not happen like that one process is executing CS 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
Solving synchronization issue

(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 critical section but if the semaphore value is less than 0 then put process in the suspended list, sleep(), this ensure that only one process can be in the critical section at a time preventing race conditions. for CS

```
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. for BS

Example: Consider two processes sharing a camera the binary semaphore ensures only one process can use camera 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

(pg8) Demonstrate the usage of stack, heap, data and code as a part of various sections in a process through appropriate example.

ans → Example:

```
#include<iostream>
int globalvar = 10;
int main()
{ int Local var = 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 contigious location..

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

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

Code section: code section contains programs executable instructions.

(pg9) Write a syntax for a. (a) tree (b) cut in operating system.

tree → Syntax: [redacted] tree [OPTION]... [FILE]...

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

(PYQ) Explain with relevant example that how An operating system provides services to both the users and to the programs.

ans → detailed in pdf → PYQ 6.7

user service	program services
• 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.	• Memory Management: allocate deallocate memory, program. • Process Management: processes resource allocation. • I/O Sorvice: 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 manage printers, keyboards, other devices

(PYQ) Different Methods to handle deadlock Briefly explain.

ans ⇒ ① Deadlock Ignorance ② Deadlock prevention
③ Deadlock Avoidance ④ Deadlock detection

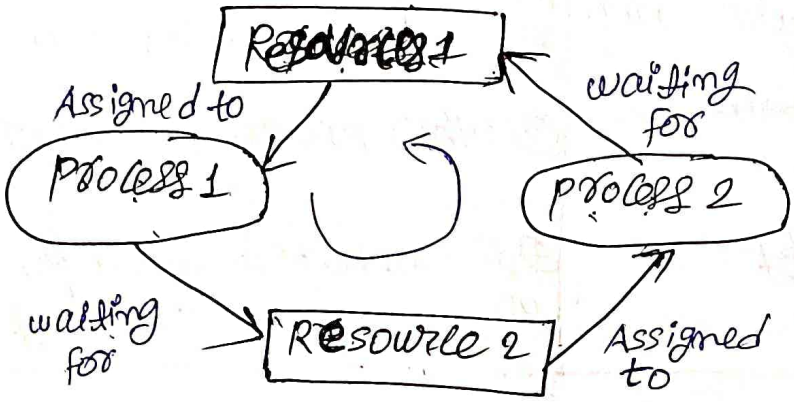
① Deadlock ignorance: In deadlock ignorance method OS acts like the deadlock never occurs and completely ignores it even if the deadlock occurs.

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 check it will be safe to share resource or not if it deadlock may occur then it avoid the deadlock.

Q5) what is the deadlock and what are the conditions to prevent it or Describe necessary condition for a deadlock situation to arise:

deadlock: A deadlock is a situation where a set of resources are blocked because each process is holding a resources 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 release 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.

Q6)

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

(Q23) difference b/w deadlock and starvation

ans ⇒

deadlock	starvation
① Deadlock is a situation where a process is blocked and waiting for some other resource that is held by some other waiting process.	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 called circular wait.	② It is also known as livelock.
③ No progress ~~can~~ once it occurs.	③ Other processes can progress.
④ It can be prevented by deadlock prevention.	④ It can be prevented by using a proper scheduling algorithm

(Q24) Is deadlock state more critical than starvation justify.

ans ⇒ Yes deadlock state more critical than starvation because once deadlock occur no anyother processes can progress but in case of starvation apart from victim process all other process can progress that's why deadlock state more critical than starvation.

Q. diff b/w primitive and non primitive scheduling

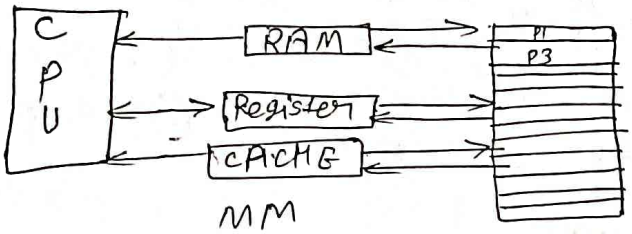
primitive scheduling	non primitive scheduling
① cpu allocated to a process for a limited time.	The cpu allocated to a process until it terminate.
② process can be interrupted in between.	process cannot be interrupted until it terminate.
③ context switching.	No context switching
④ It is flexible	It is not flexible
⑤ CPU utilization is higher.	cpu utilization is lower.
⑥ If this running task can be interrupted and moved to the ready queue by the scheduler to allow another task to run. (R/R)	In this once task starts its execution it run till completion Ex → FCFS

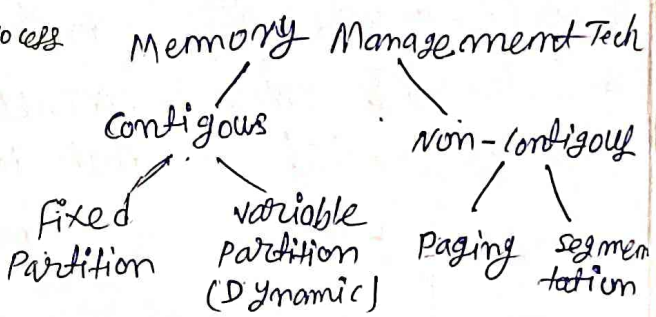
Banker's Algorithm

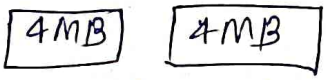
Process	Allocation			Max Need			Available			Remaining		
	A	B	C	A	B	C	A	B	C	A	B	C
P1	0	1	0	7	5	3	3	3	2	7	4	3
P2	2	0	0	3	2	2	5	3	2	1	2	2
P3	3	0	2	4	0	2	7	4	3	6	0	0
P4	2	1	1	4	2	2	7	4	5	2	1	1
P5	0	0	2	5	3	3	7	5	5	5	3	1
	7	2	5	28	12	12						

A = 10
B = 5
C = 7

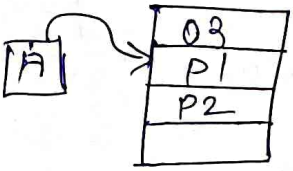
Memory management

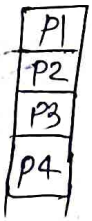


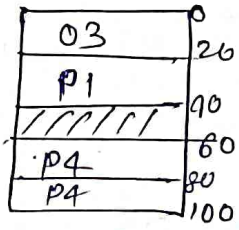


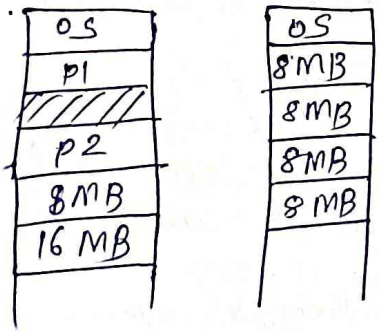


No of process ?
CPU utilization = 1 - K
K = I/O operation (70% eg)

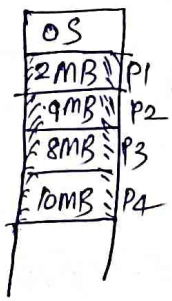








Static Fixed Partitioning



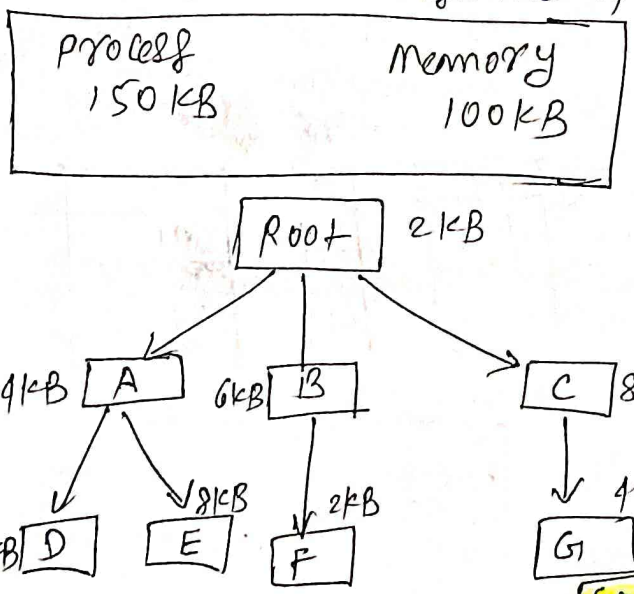
Dynamic variable partitioning

* Contigous 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

(PYQ) Explain overlays, what is fragmentation and diff type and disadvantage Internal and External fragmentation,

ans ⇒ 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. and loading only the necessary portions in main memory that is requird at that instance of time.



Root + A + D = 12KB
Root + A + E = 14KB
Root + B + F = 10KB
Root + C + G = 14KB

14KB

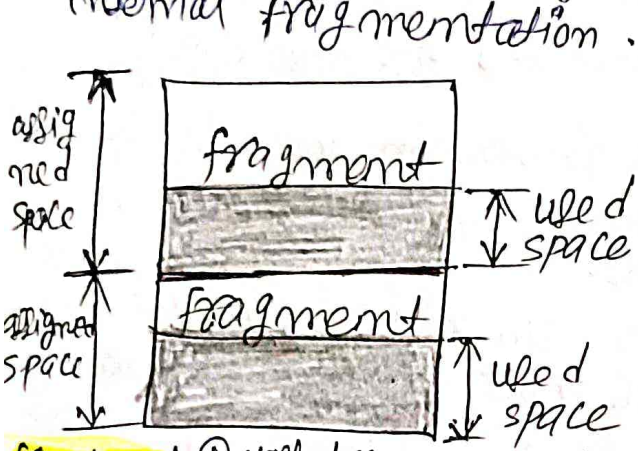
fragmentation : fragmentation refers to the phenomenon where memory divided into smaller, non-contigious pieces over time. There are two main types.

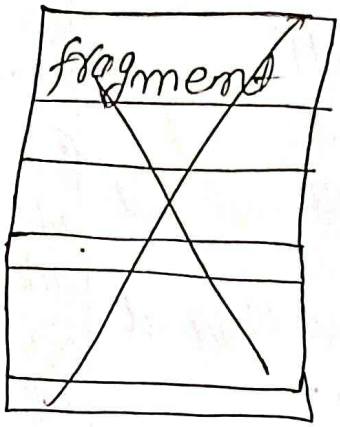
There are two types of fragmentation

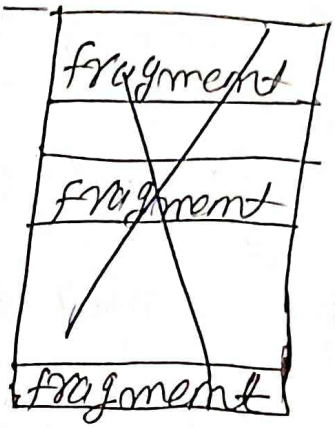
(i) Internal fragmentation

(ii) External fragmentation

1. Internal fragmentation: It occurs when fixed size memory block is allocated to the processes. 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.





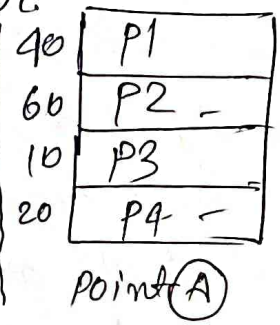


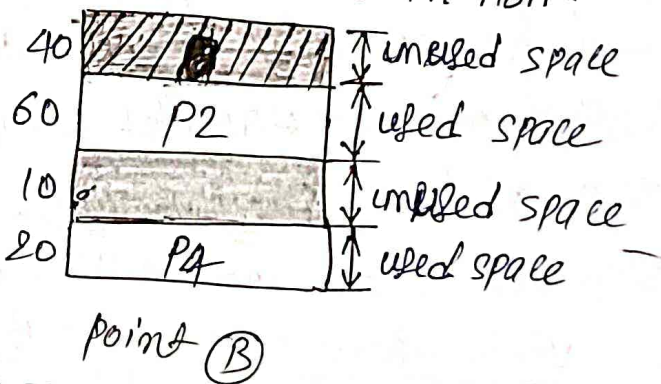
disadvant: (i) wasted memory, Reduced Efficiency, increased overhead

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 can't be allocated as it is in non-Contiguous manner.

disadvant → Difficulty in memory allocation

(ii) performance degradation





Pros → 50

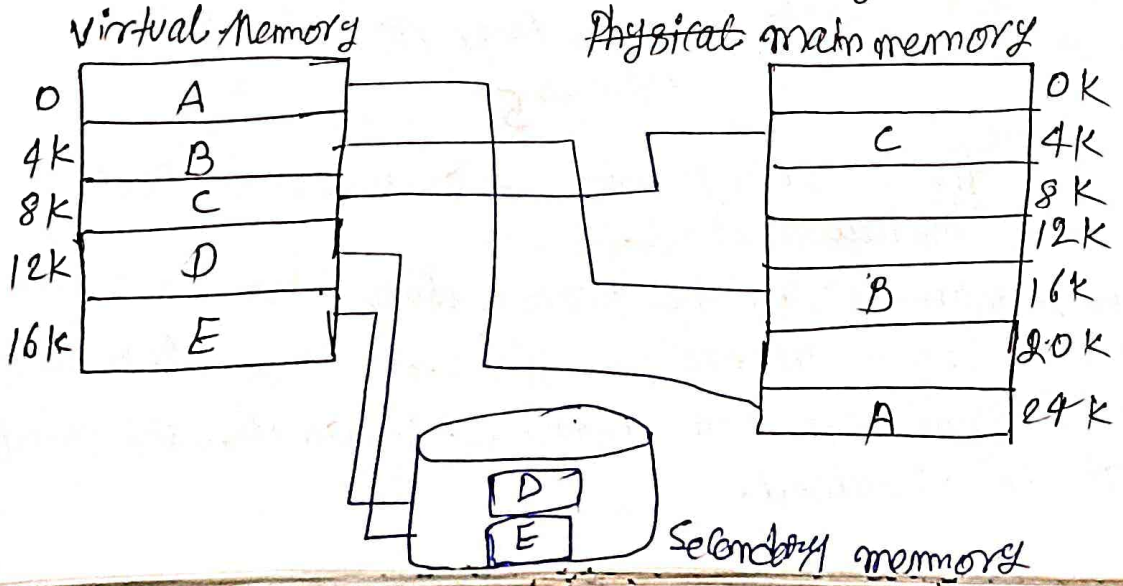
(PYQ) Elaborate the mapping and differemlee between logical and Physical address.

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 as it is required.

Physical address	Logical address
(i) It is also called real address.	(i) It is also called virtual address.
(ii) Physical address will not change.	(ii) Logical address can be changed.
(iii) Computed by MMU.	(iii) Generated by the CPU.
(iv) The user can indirectly access physical address but not directly.	(iv) The user can use the logical address to access the physical address.
(v) User can never view physical address of program.	(v) User can ~~never~~ view the logical address of a program.
(vi) Physical address is the set of Physical addresses.	(vi) Logical address is the set of logical addresses.

(PYQ) 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.



How it's used:

(i) When a process needs more memory ~~than~~ that is greater than RAM, then virtual memory move some data from RAM to secondary memory.

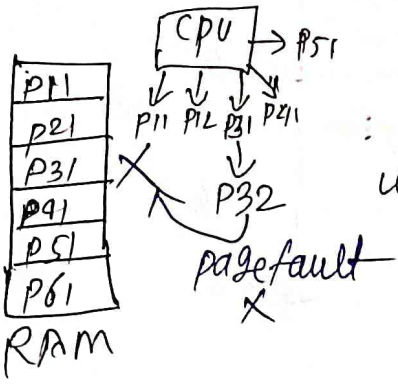
(ii) virtual memory frees up space in RAM and loads the required data from disk into RAM.

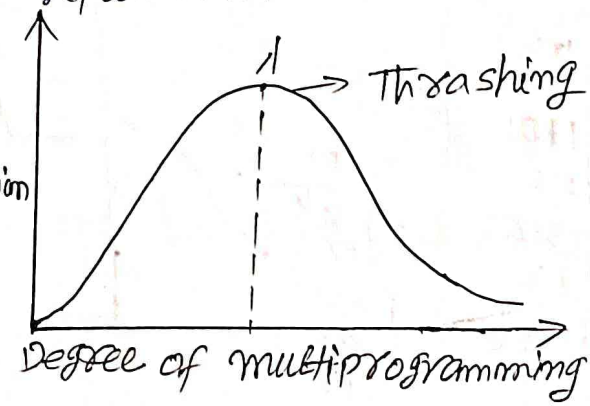
* Advantages of virtual memory
(i) Increased memory utilization
(ii) Increased efficiency.
(iii) Increased Multitasking.

* Need of virtual memory
(i) Increased memory capacity
(ii) memory isolation
(iii) Demand paging
(iv) Memory management.

PYQ) what is thrashing how is it controlled by OS.

ans⇒ 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 ~~exeut~~ executing useful work.



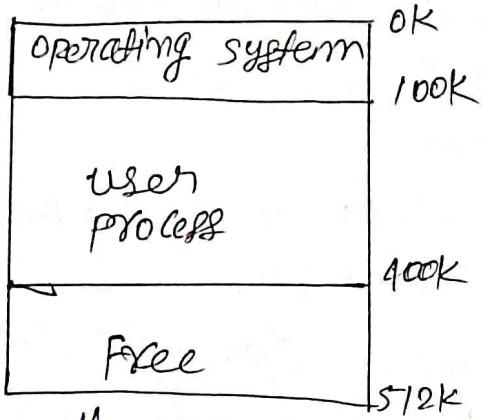


(i) virtual memory
(ii) Page replacement algorithm
(iii) I/O scheduling

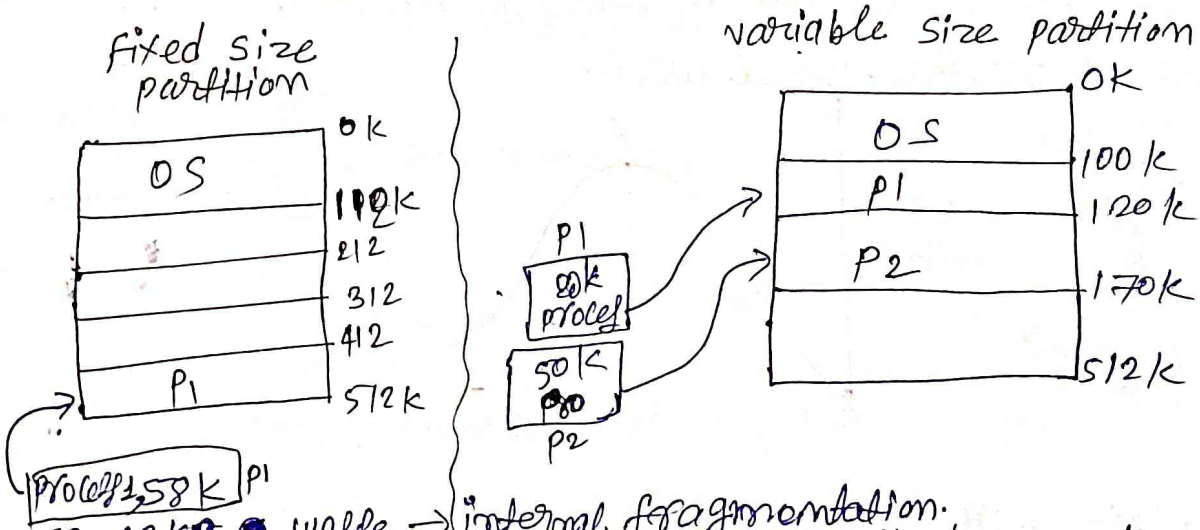
PYQ) Explain with diagram single partition allocation and multiple partition allocation.

ans⇒ 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 ~~more memory~~ 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.



100-58=42K wasted → internal fragmentation.

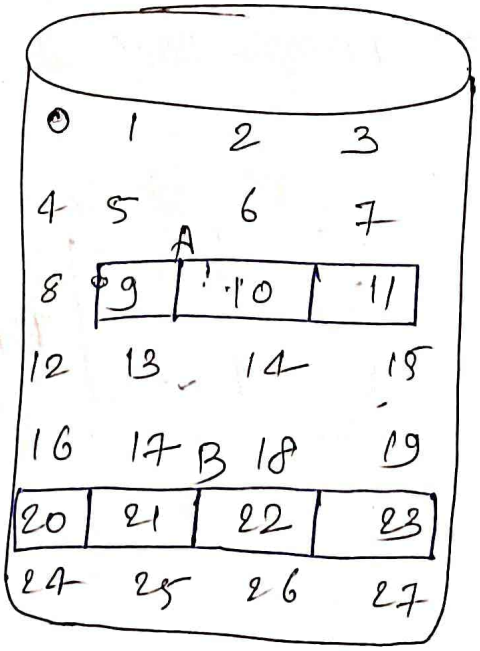
@fm pyq 2.1 what are the three methods for allocating disk space

pyq Elaborate the file Management with detailed requirement and implementation issues of contiguous, Linked and Indexed allocation methods

ans⇒ 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 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.



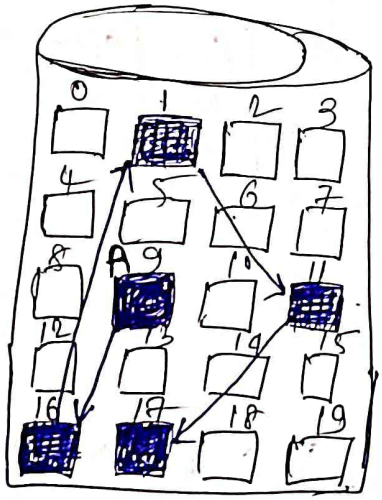
File	Size
A	3
B	4

(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

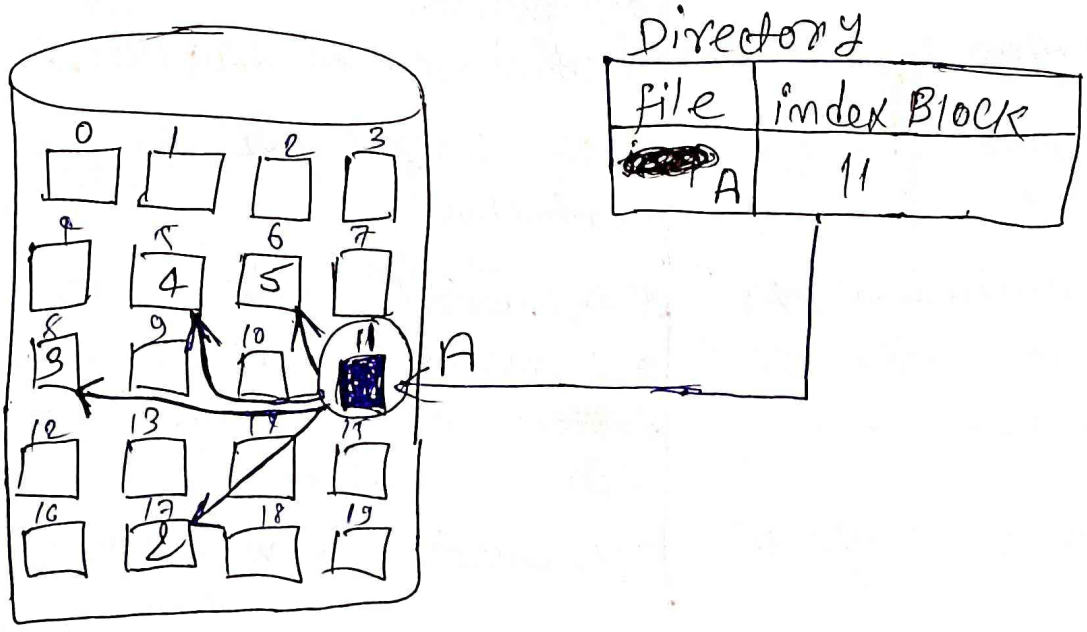
file	start	end
A	9	19

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 the size.
- Index Block overhead: Index allocation requires extra space to store index blocks for each file which consuming 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.



(PYQ) 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 and y offset bits, rather than doing by arithmetic operations.

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

(PYQ) difference blw paging and segmentation (b) page table and segment table (c) tightly coupled system and loosely coupled systems.

Paging	Segmentation
(i) Memory is divided into fixed-size blocks called pages, and processes are divided into blocks of the same size called frames.	Memory is divided into variable size blocks called segments
(ii) for the paging os is accountable	(ii) for segmentation compiler is accountable.
(iii) It is faster than segmentation	(iii) It is slower than paging.
(iv) It leads to internal fragmentation.	(iv) It leads to external fragmentation.
(v) paging comprises a page table that encloses the base address of every page.	(v) segmentation also comprises a segment table which encloses segment number and segment offset
(vi) paging is invisible to the user	segmentation is visible to the user

Page table	Segment table
(i) page table is a table maintained by the OS to translate logical page address into physical frame addresses.	(i) A table maintained by the OS to translate logical segment address into linear addresses.
(ii) It divides logical address space into fixed-size blocks.	(ii) It divides logical address space into variable-size blocks.
(iii) It uses page number and offset to calculate physical address	(iii) Uses segment number and offset to calculate linear address.
(iv) Invisible to the user.	(iv) Visible to the user.
(v) It supports paging.	(v) Doesn't necessarily support paging.
(vi) Prone to external fragmentation	(vi) prone to internal fragmentation.
(vii) It contains a frame number and access control bits	(vii) It containg base address register and access control bits.

tightly coupled system	loosely coupled systems
(i) Systems where components are heavily depends on each other.	Systems where components are minimal dependency on each other.
(ii) Communication between components is direct.	Communication between components is indirect.
(iii) Components are closely integrated on each other.	Components are relatively integrated on each other
(iv) changes or failures in one component can significantly impact others	changes or failure in one component have minimal impact
(v) Scalability may be limited due to tight integration.	Scalability is typically better due to the modular nature.
(vi) Modification or updates are difficult to implement	Modification or updates are easier to implement.

Q5) List three major activities of an OS with regard to memory management.

(i) Memory Allocation: The OS allocates memory to process as 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.

Q6) Discuss the Belady's Anomaly and segmentation.

* Page Replacement algorithm (First In First Out)

Reference string: 1,2,3,4,1,2,5,1,2,3,4,5

where frame size is 3

f3			3	3	3	2	2	2	2	2	4	4			
f2		2	2	2	1	1	1	1	1	3	3	3			
f1	1	1	1	4	4	4	5	5	5	5	5	5			
	*	*	*	*	*	*	*	Hit	Hit	*		Hit			

Ref → 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Hits = 3

Page fault = 9

Now, where frame size is 4

f4				4	4	4	4	4	2	3	3	3			
f3			3	3	3	3	3	3	2	2	2	2			
f2		2	2	2	2	2	2	1	1	1	1	5			
f1	1	1	1	1	1	1	5	5	5	5	4	4			
	*	*	*	*	Hit	Hit	*	*	*	*	*	*			

Ref → 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Hits : 2

Page fault : 10

Belady's Anomaly : From Above example when frame size was 3 then no of Hits are 3 and no of Page fault are 9 but when increasing the no of page frames result in of 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 segment that can vary in size; it allows for more flexibility in memory allocation and protection, but it can lead to fragmentation.

PYQ * Optimal page replacement algorithm & LRU

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

ans⇒ Optimal Page replacement

f4				2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2					
f3			1	1	1	1	1	4	4	4	4	4	4	1	1	1	1	1	1	1					
f2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
f1	7	7	7	7	7	3	3	3	3	3	3	3	3	3	3	3	3	7	7	7					
	*	*	*	*	Hit	*	Hit	*	Hit	Hit	Hit	Hit	Hit	*	Hit	Hit	Hit	*	Hit	Hit					

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

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

page fault : 8

Least recently used page Algo

F4				2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2					
f3			1	1	1	1	1	4	4	4	4	4	4	1	1	1	1	1	1	1					
f2		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
f1	7	7	7	7	7	3	3	3	3	3	3	3	3	3	3	3	3	7	7	7					
	*	*	*	*	Hit	*	Hit	*	Hit	Hit	Hit	Hit	Hit	*	Hit	Hit	Hit	*	Hit	Hit					

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

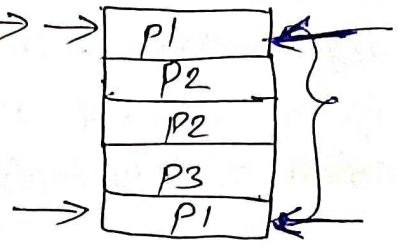
Page fault : 8

Hits : 12

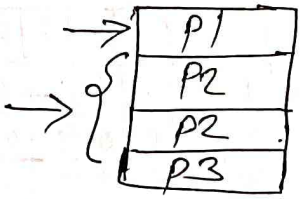
MY fem(PM.1)

PYQ Explain Local and Global page Replacement, Bad Blocks, 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.



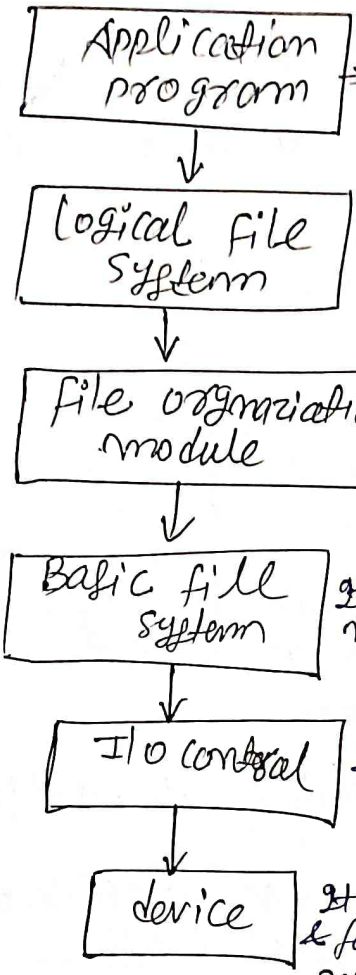
Bad Blocks: Bad blocks refers to damaged or defective blocks on a storage device such as a hard drive or SSD. These blocks are unable to reliably store data and can lead to data loss or corruption if data is written to them.

Boot Block: The boot block is also known as boot sector on a storage device such as a hard disk or SSD. This block contains the necessary machine code to initiate the booting process of a computer system.

~~design and implementation of File Systems.~~

Boot Block करना है।

* File layered Architecture:



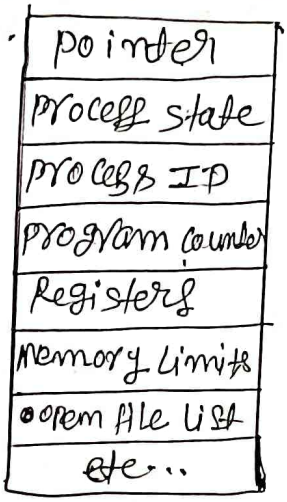
* Protection mechanism: The protection mechanism provide the facility of the controlled access by limiting the type of access to the file.

type of access are

(i) Read : Read a file
(ii) write : writting a file
(iii) Execute : Executing written file
(iv) Append : writting the new info to the existing data
(v) Delete : Deleting the file
(vi) List : List the name and attribute of the file.

PYQ Explain process controll block

A process control Block (PCB) is data structure in operating system that stores information about a process. It typically includes details such as process state, program counter, registers, process ID.



(i) pointer: PCB contains a pointer to the next PCB in the list

(ii) process state: This indicates current state of the process whether it is running, ready, blocked state.

(iii) process ID: A unique ID is assigned to each process by the operating system that uniquely identify the process.

(iv) program counter: It holds the address of the next instructions to be executed for the process.

(v) Registers: PCB stores the contents of CPU registers for the process. Registers hold data temporarily during execution and saving their state.

(vi) Memory Limits: It contains information about the memory allocated to the process. such as the base address and the limit of the address space.

(vii) Open file list: PCB maintains a list of files opened by the process along with their current status and pointers to file descriptions.

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 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 allow 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 keep 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:
  - Runs 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 it uses a virtual 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 Tables: The guest page tables translate the virtual address used by the guest operating system.

(iv) page fault Handling: if a page table entry is not present in memory it results in page fault. OS 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: window is a proprietary operating system, meaning you need to buy a license to use it legally.
- Security: ~~while~~ windows can still be more ~~vul~~ 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.
- ~~Multi~~ 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 version available for development purpose.
- Security: Unix is known for its robust security features and user permissions system, making it a popular choice for servers and mission-critical application.

→ 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: Duto to its open-source nature, Linux offers a high degree of customization. Users can modify the system to their specific needs and prefrences.
- 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.

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

Features	windows	Unix	Linux
Kernel	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)

(Pyq) Difference b/w hard real-time systems and soft real-time systems

Hard Real-Time system	Soft real-time systems
Systems where tasks must be completed within a strict deadline.	Systems where meeting deadline is important but occasional misses are tolerable.
(i) Often more complex due to strict deadline.	(ii) Generally less complex compared to hard real time sys
(iii) It can handle small and medium sized data files	(iii) It can handle large data files
(iv) Examples: Aircraft control system nuclear reactors	(iv) Examples: online gaming video streaming

(Pyq) Why API's need to be used rather than system call?

API (Application programming interface) provide a higher level of abstraction and flexibility compared to system calls. offering several Advantages

(i) Portability: APIs abstract the underlying system details, allowing developers to write code that can run on different OS without modification

(ii) Easy to use: APIs provide a more user friendly interface with well defined functions and parameters.

(iii) Safety and security: APIs often include built in error handling and parameter validation mechanisms

(iv) Abstract of complexity: APIs abstract the complexities of system internals.

Unit-5 ⇒ Secondary storage

(PYQ) what is disk scheduling ? Explain FCFS and SCAN disk scheduling algorithm, Advn, disadv

ans ⇒ Disk scheduling is done by operating system to schedule I/O requests arriving for the disk. Disk scheduling also known as I/O scheduling. its aims to minimize the seek time and optimize disk performance.

* Seek time : The time taken to reach the disk arm to the 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.~~ The time required by the read/write head to rotate to the requested sector from the current position is called RL

* Transfer time : Transfer time is the time to transfer the data

* Disk Access Time :

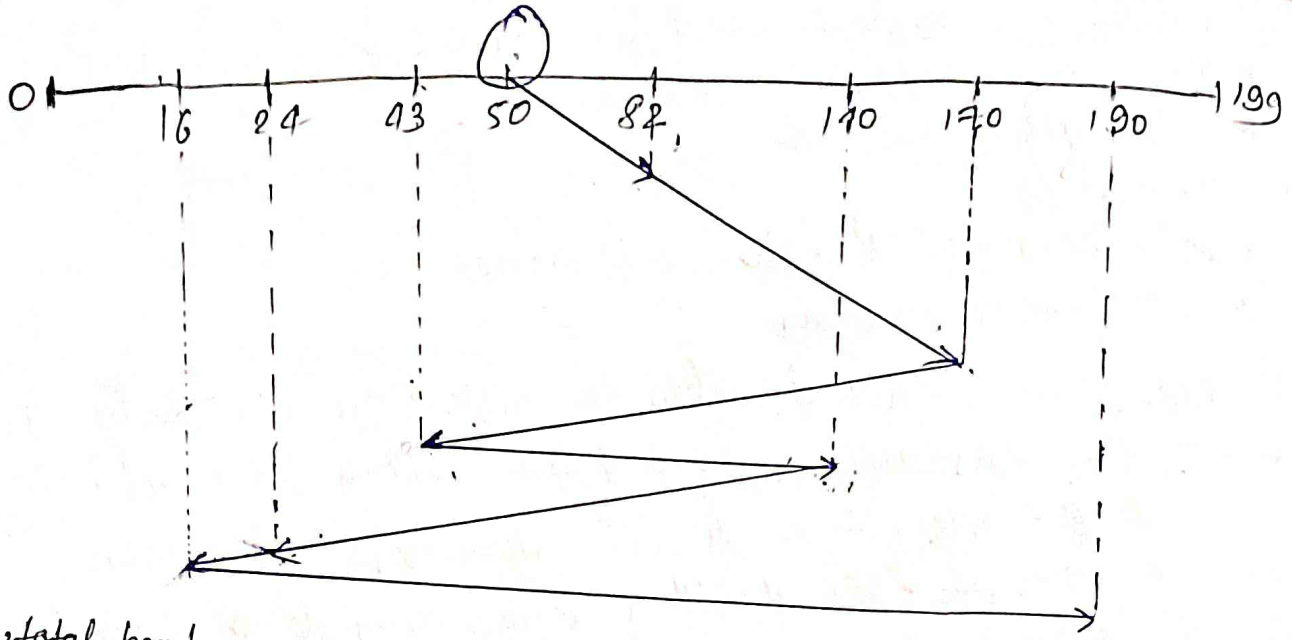
Disk Access Time = See Time + Rotational + Transfer Time
Total Seek Time = Total head Movement

* There are several Disk Algorithm,
  (i) FCFS (ii) SSTF (iii) SCAN (iv) C-SCAN (v) LOOK (vi) C-LOOK

(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 tracks no 82, 170, 43, 140, 24, 16, 190 respectively current position of Read/write head = 50, calculate total no of tracks movement by Read/write head.

Given ⇒ tracks no : 82 170 43 140 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

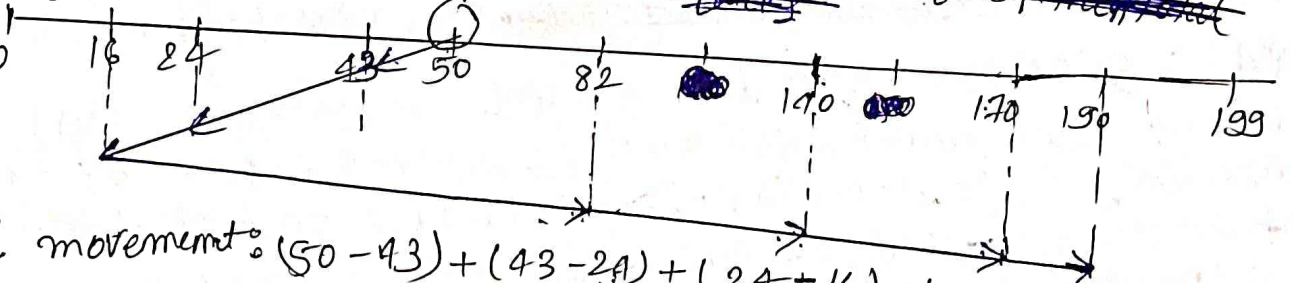
Advantages of FCFS: • Every request gets a fair chance
• No indefinite postponement
• No starvation.

Disadvantage: • It does not optimize seek time
• May not provide the best possible service

② SSTF (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 contains 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 shortest seek time first.

ans ⇒



total movement: (50 − 43) + (43 − 24) + (24 − 16) + (82 − 16) + (140 − 82) + (170 − 140) + (190 − 170) = 7 + 19 + 8 + 66 + 58 + 30 + 20

= 208.

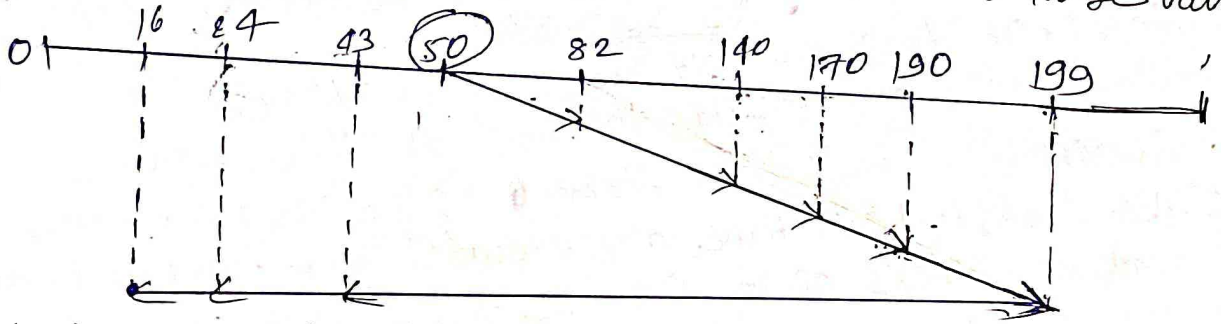
* Advantages of Shortest seek Time First:
  - The average Response Time decreases
  - Throughput increases

Disadvantage
  - 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 containg 200 tracks (0-199) Request queue containg 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?

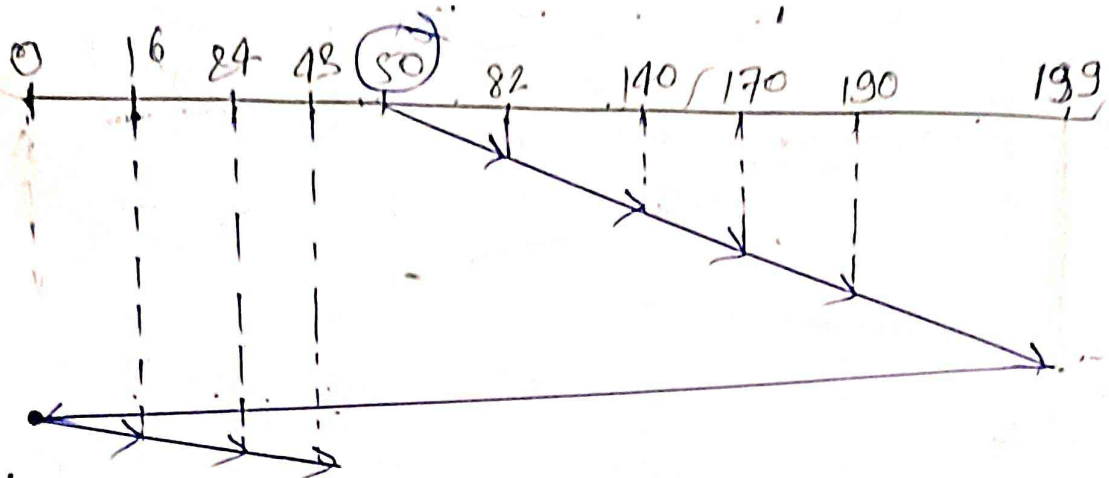
ans →



total movement: (82-50)+(140-82)+(170-140)+(190-170)
+(199-190)+(199-43)+(43-24)+(24-16) = 332

- Advantage: • High throughput • prevention of starvation
  • low variance of response time
- Disadvantage: • High Disk head movement

*(4) C-SCAN: Ex → A disk containg 200 tracks (0-199) Request queue containg 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)

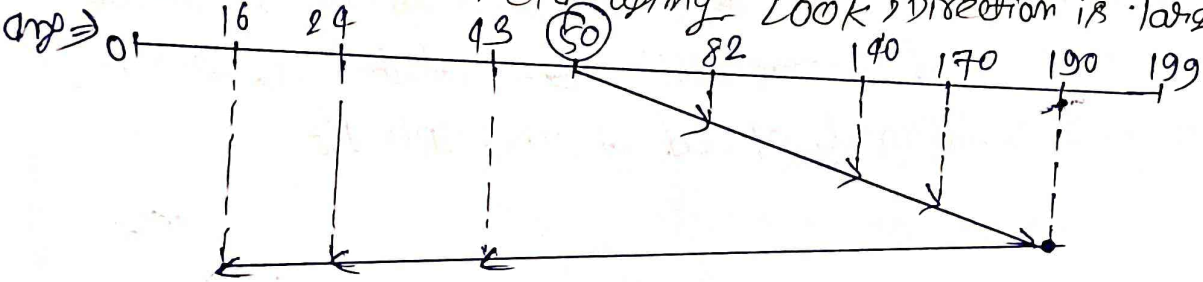


total movement: $(199-50)+(199-0)+(43-0) = 391$

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 times (0-199) request contain 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 LOOK, Direction is larger value)

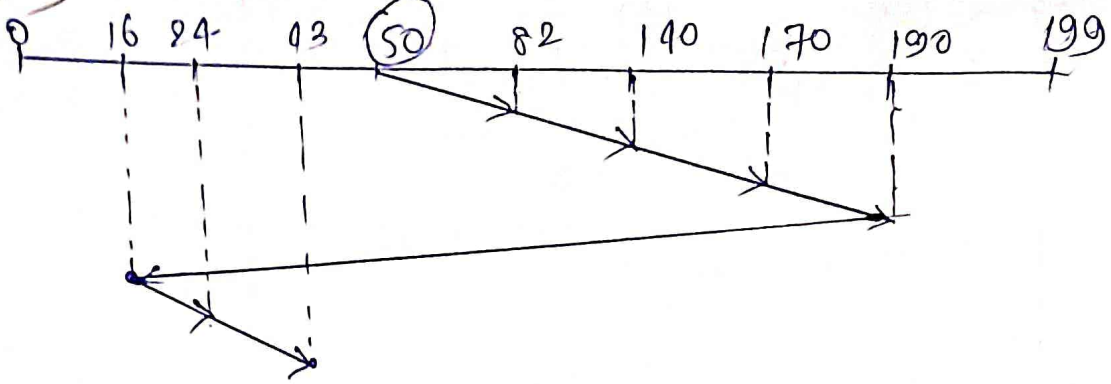


total movement: $(190-50)+(190-16) = 314$

Advantage: • Reduce Head movement than SCAN.
• Disadvantage: ~~potential~~ chances for starvation

⑥ C-LOOK: A disk contains 200 tracks. Request queue containing track 82, 170, 43, 140, 24, 16, 190 current position of R/W head = 50, calculate total no of tracks movement by head using C-LOOK? Direction is towards large value.

ans:->



Total movement = $(190-50)+(190-16)+(43-16) = 341$

Advantage : • prevention of starvation.
• Reduce Head movement.

Disadvantage : • Increased response Time than LOOK.

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

ans⇒ Most disk scheduling does not confider rotational frequency because, in a most modern system, the actual physical location of blocks are not available. ~~it is very hard to estimate the time~~ Rotational latency depends on the rotational speed of the spindle.

File Management

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

ans⇒ File: A file is 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 file 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 files: contain instructions for the CPU to execute programs (Ex→ .exe on window, .app on macOs)

(iii) Document file: created by specific software (Ex→ .docx, .xlsx, .pdf).

(iv) Image files: ~~created by specific~~ 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 files: 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\Myfiles\mydocument.txt")
- Creation Data and Time: When a file was first created.
- Modification Data and time: When the files content was last changed.
- Type: Tells type of the file whether it is .txt, .cpp, .png, .jpg, .mp4.

(Q5) What are the different access method 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 begining of the file. Imagine reading a book - you typically go 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 maintain 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.

2M

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

Soln: 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 process gets how much memory and where in the 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 interfere with each other's, preventing unauthorized access

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

(pyq) Calling pyq 2.1 (Before this page)

(pyq) what are the points to be consider in file system design? Explain Linked list allocation & index allocation in detail.

Ans: 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 ~~storing~~ memory to ~~data~~ 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 handle large data and growing number of files efficiently.

Calling → pyq 2.1

FS, FO, MA, S, EH, S