OS

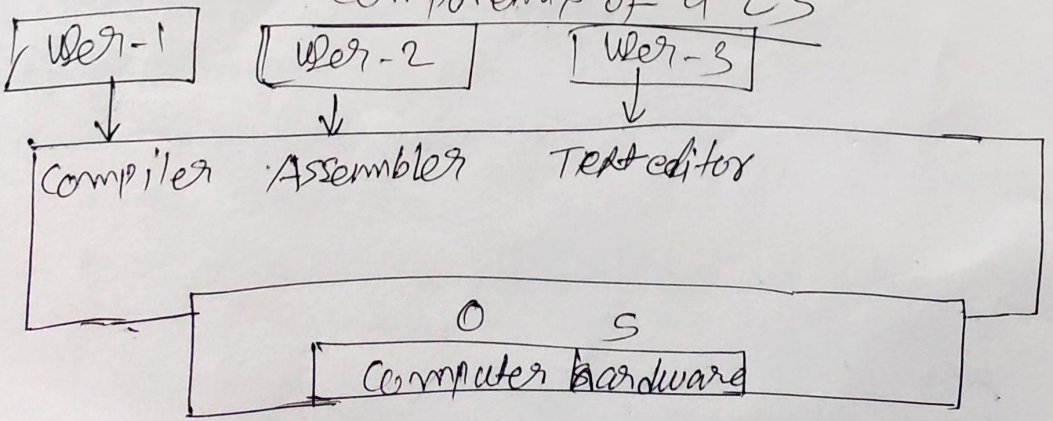
Need of OS

- Manages all the devices.
- Provides interface b/w user and the system.
- Manages the memory allocation and deallocation.
- schedules the computational process.
- Manages the files and disks.
- Protect the system and data
- Helps resources allocation and management
- shares of resources in terms of time, space and CPU to achieve optimal utilization of resources.

Objectives of OS

- Convenience: An Operating system makes a Computer more convenient to use.
- Efficiency: Allows the computers system resources to be used in efficient manner.
- Ability to Evolve: An OS should be constructed in such a way as to permit efficitive development.

Components of a CS



* 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

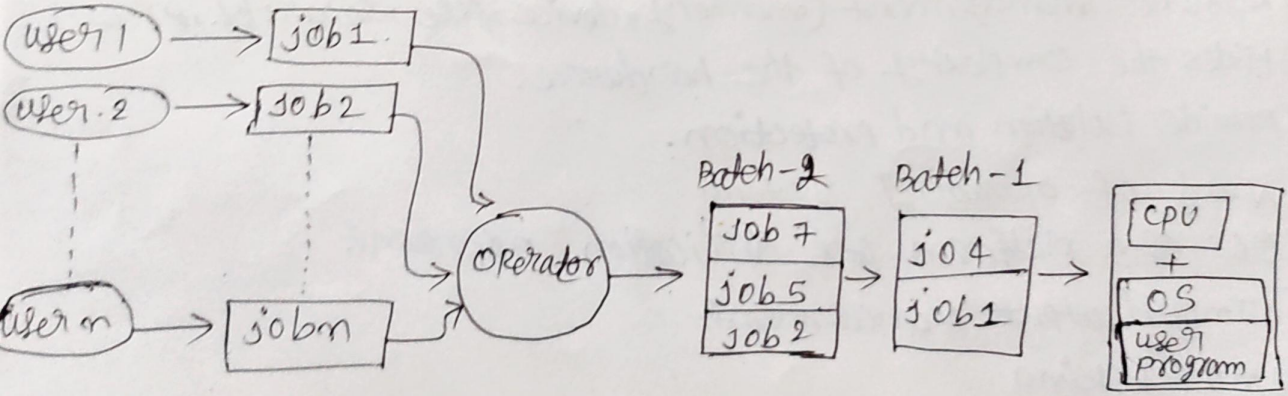
(5) provides Security

(6) Acts as an Interface between computer H/w & S/w. & user and computer.

* 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

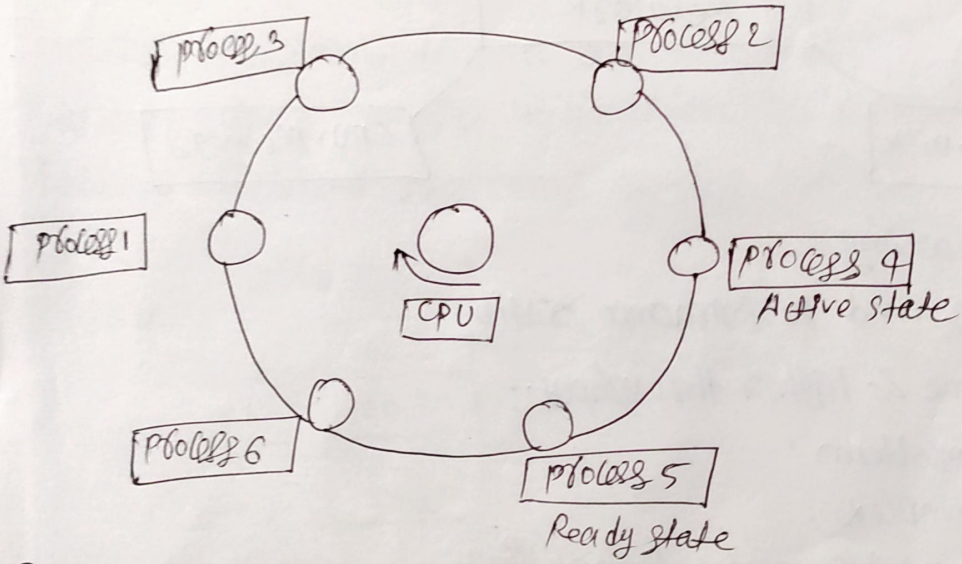
(1) Batch Operating system: Batch operating system is a type of operating system, it doesn't interact with the computer directly. There is an operator which takes similar jobs from the user having same requirement and groups them into batches. It is responsibility of operator to sort jobs with similar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

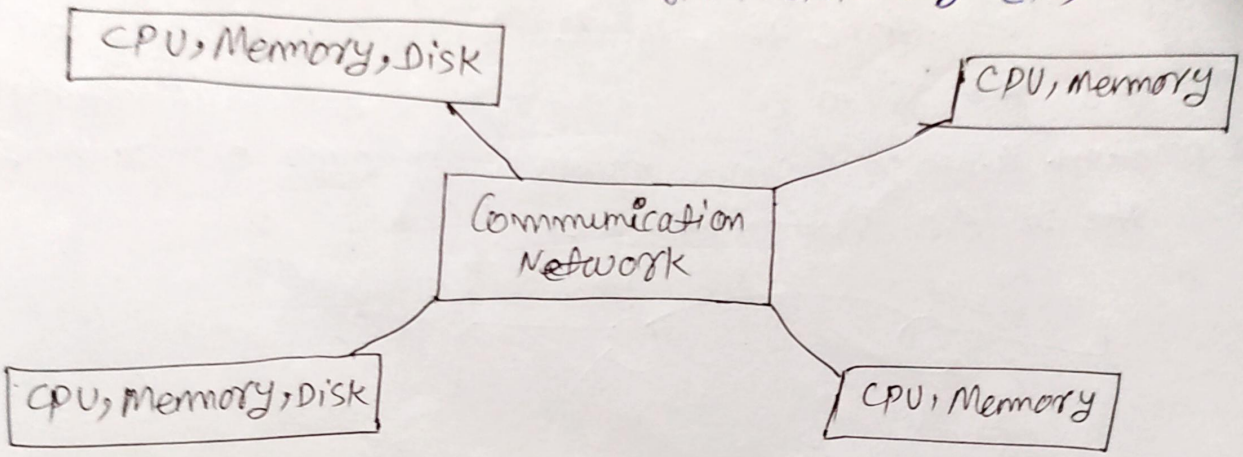


(i) pros: (i) each task gets equal opportunity.
(ii) each task gets equal time.
(iii) Multitasking can be performed.
(iv) CPU time can be reduced.

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.
(ii) Highly complex.

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

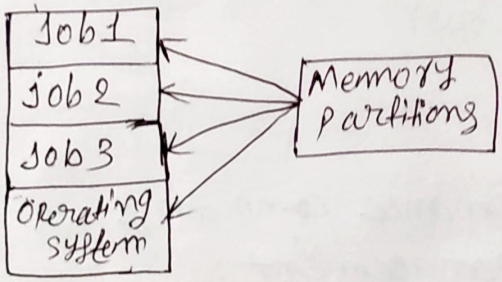
(iv) Real-time operating system: Real-time operating system is a operating system in which operating system respond quickly, Real time operating system used in missile system, use in hospital, ATC (air traffic control). this OS provides maximum efforts to its task with quick response. Ex ⇒

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 system

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



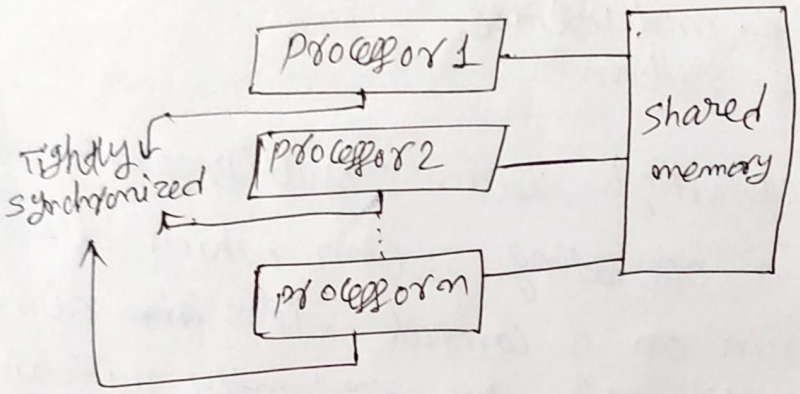
Pros: ① increased CPU utilization
② Resources utilization

Cons: ① Complexity
② Difficulty in debugging

Application: ① Transaction processing
② Batch processing
③ Scientific Computing

(vi) Poor.

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

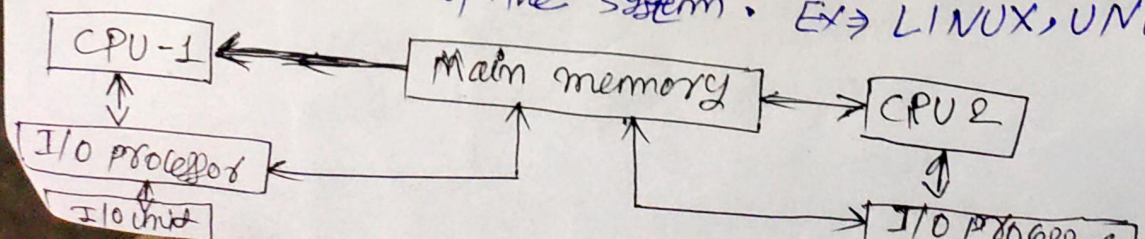


Pros: ① Increased performance
② High throughput

Cons: ① Complexity
② too costly

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

(vii) * 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



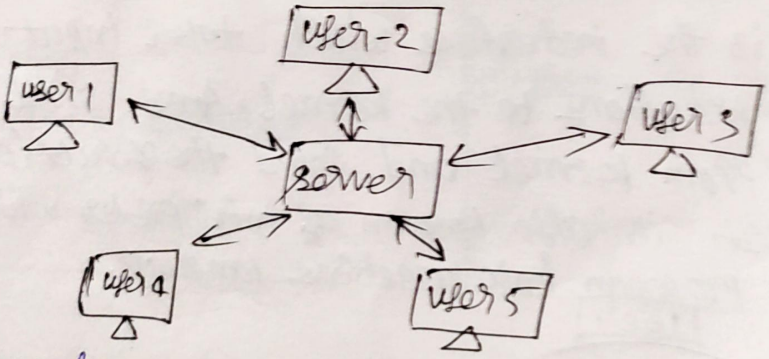
Pros: Failure of one processor does not affect the functioning of other processors.

(ii) Parallelism, it allows to work in parallelly.

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

Application: (1) High - Performance computing
(2) Multimedia processing
(iii) Scientific Computing

(iii) 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.

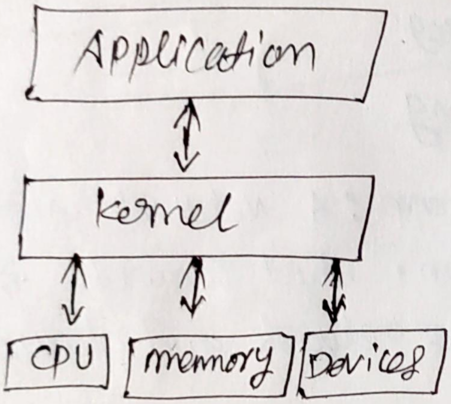
Cons: (1) Too Cost
(ii) Regular updating and maintenance required

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

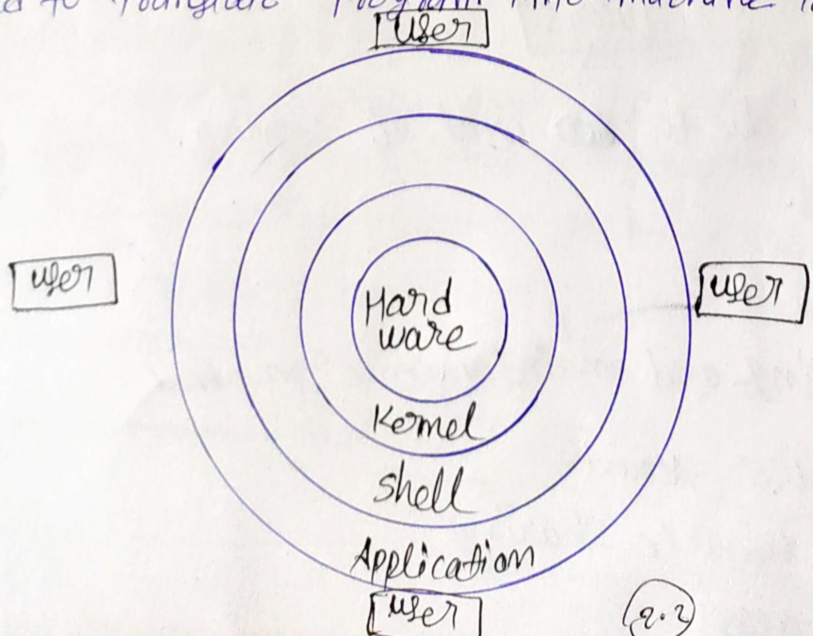
* Kernel and shell

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



* 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.



2.2

Agitoro missing
~~एरिटरो~~
एजिटरो मासिंग
500m2

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)

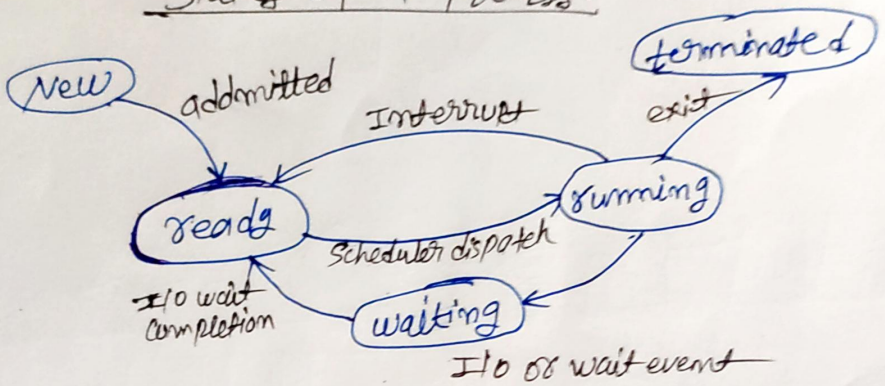
(1) program execution → load programs into memory and execute them.
(2) Input/output operations → It handles comm b/w external device means I/O device
(3) File management → manage file on storage device → creation, deletion, update
(4) Memory management → Allocate & deallocate memory space as need by 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

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 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 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 ≠ WT, RT same

* 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.

~~$RT = R - AT$~~

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

① First come first serve :

Find average Turn around time and average waiting time

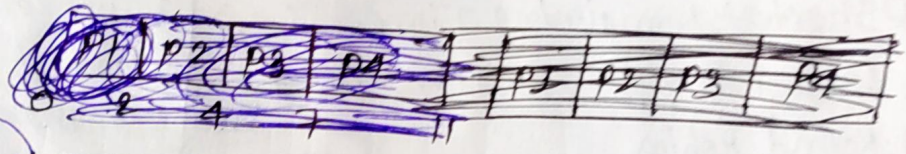
Process No	AT	BT	CT	TAT	WT
P1	0	2	2	2	0
P2	1	2	4	3	1
P3	5	3	8	3	0
P4	6	4	12	6	2

avg TAT = ? TAT = CT - AT, CT = when process complete

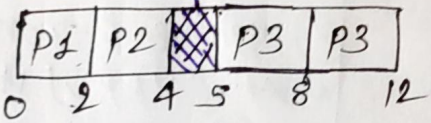
avg WT = ? WT = TAT - BT,

NON Premitive

Gantt chart :



idle time



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

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

Q2. Shortest Job first : Calculate average turn around time and avg waiting time.

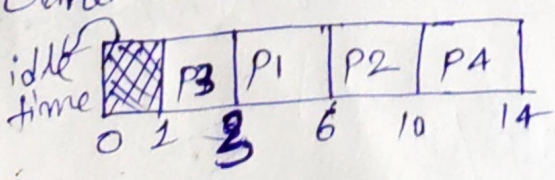
Process No	AT	BT	CT	TAT	WT
✓ P1	1	3	6	5	2
P2	2	4	10	8	4
✓ P3	1	2	3	2	0
P4	4	4	14	10	6

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

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

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

Gantt chart :



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

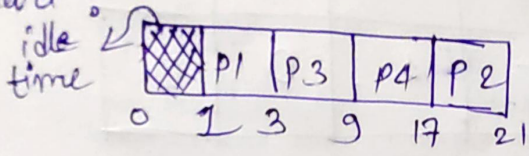
Avg TAT = $\frac{2+19+6+13}{4}$

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

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

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

Gantt chart:



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	7

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

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

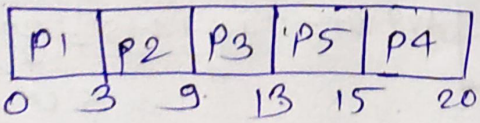
P5 r = $\frac{1+2}{2} = \frac{3}{2} = 1.5$

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

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

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

Gantt chart



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

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

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

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

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

② Longest remaining time first :

Process	AT	BT	CT	TAT	WT	RT
P1	0	7 6 5 4 3 2 1 0	9	9	7	0
P2	1	8 7 6 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

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

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

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

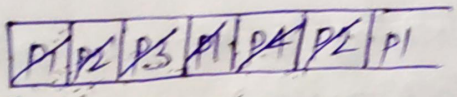
0 1 2 3 4 5 6 7 8 9 10 11 12

③ Round Robin:

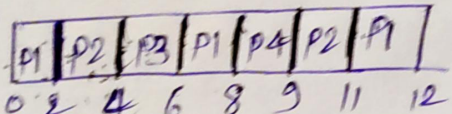
Process	AT	BT	CT	TAT	WT	RT
P1	0	~~8~~ ~~10~~	12	12	7	0
P2	1	~~4~~ ~~0~~	11	10	6	1
P3	2	~~2~~ 0	6	4	2	2
P4	4	~~1~~ 0	9	5	4	4

Given TQ = 2

Ready queue



Running queue



~~*~~ No of context switching = 6

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

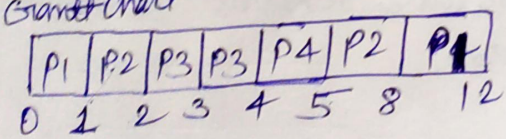
Avg WT = $\frac{7+6+2+4}{4} = \frac{19}{4} = 4.75$

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

④ priority scheduling:

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

~~Gantt~~ chart



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) process is heavy weight task.	Thread is light weight task.
(ii) system calls are performs in process	No system call involve.
(iii) Context switching is slow.	Context switching is fast.
(iv) different process have different copy of data files and codes.	It shares the same copy of code & data file.
(v) Blocking a process will not block another	Blocking a thread will block another

* 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.

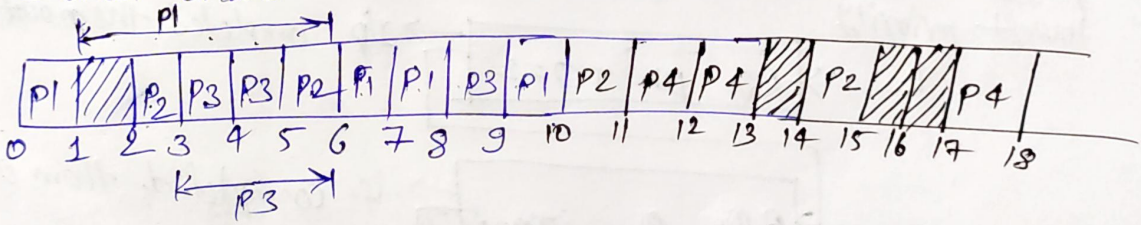
define Context switching.

⑤ Mix-burst time priority scheduling:

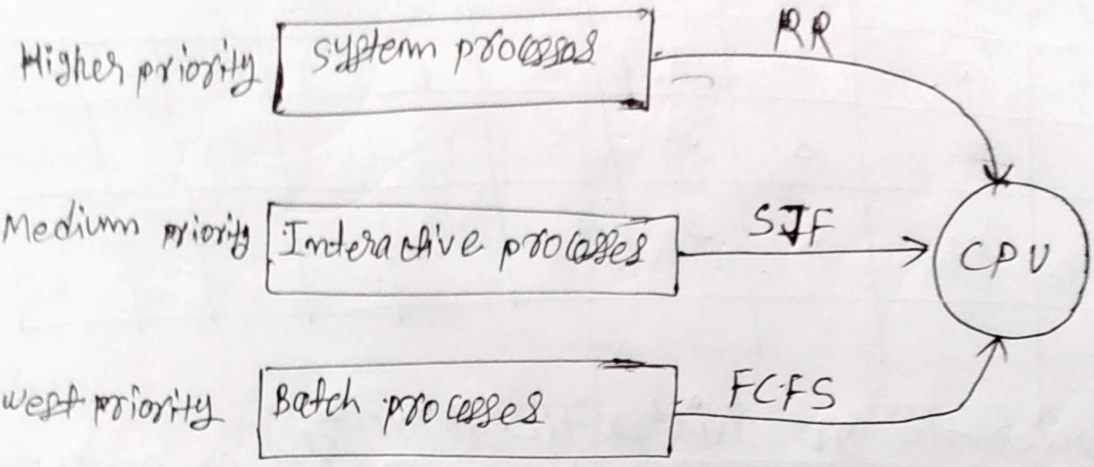
priority	process No	AT	CPU	IO	CPU	CT	TAT	WT	RT
2	P1	0	1	5	3				
3	P2	2	2	3	1				
1	P3	3	2	3	1				
4	P4	3	2	4	1				

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

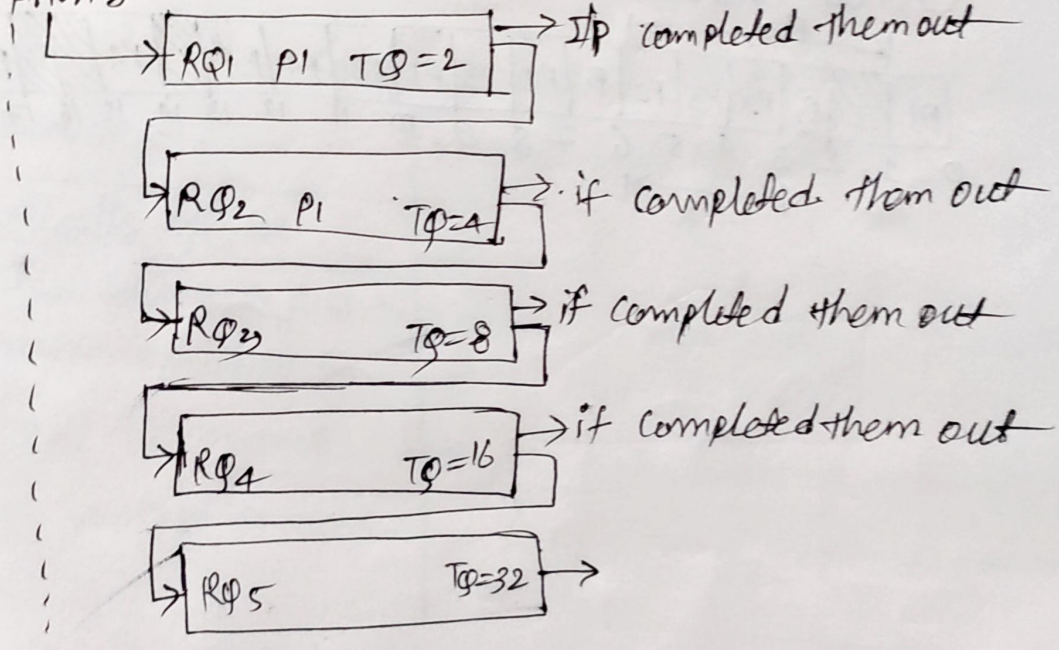


* Multi level queue scheduling:



the problem known as starvation
starvation over utilization of resources.

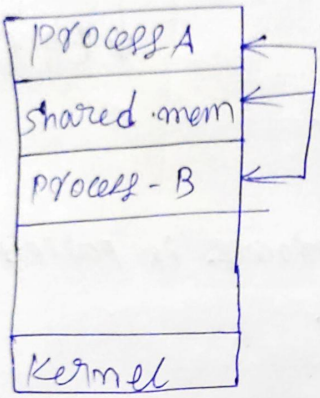
* Multilevel feedback Queue scheduling.

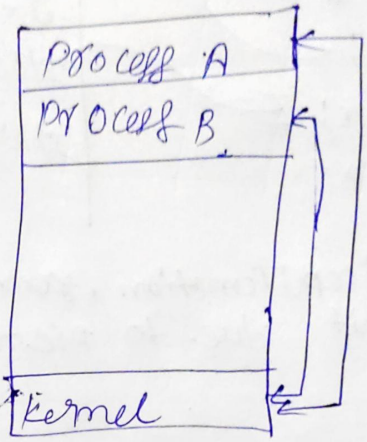


OS

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

S.M





* 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

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

(i) 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	5

↓ time

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

*

A, B	x, y
N C section	N C 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 program where shared resources are accessed by various ~~resources~~ process. It is a common platform used by processes

OS

condition for entry and exist section.

(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();                      | sleep();
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

1.1 Semaphore of two type

① Counting Semaphore $(+\infty, -\infty)$

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 from
  suspended list and
    wakeup();
  }
}
```

PYQ 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 the critical section then no other process is allowed to execute in the critical section.

OS

How ~~semaphore~~ Binary semaphore Resolving synchronization issues: (1.125) *

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;
   }
   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 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 bound 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
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.

```
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 binary 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

(9) Demonstrate the usage of stack, heap, data and code of 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 section";
  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 a random location.

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

Code section: code section contains program's executable instructions.

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

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

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

(PY7) 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

- 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 manage printers, keyboards, other devices

(PY8) 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.

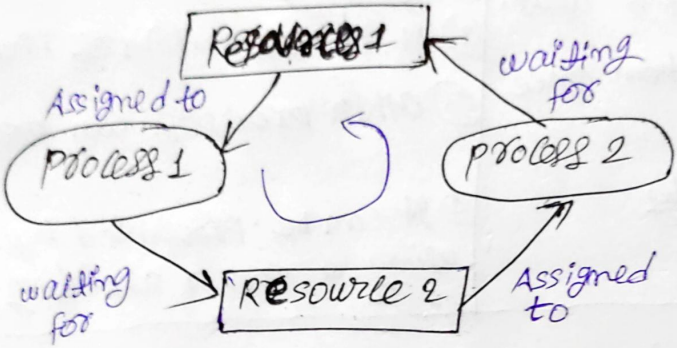
② Deadlock detection: In deadlock detection method OS periodically check whether deadlock occur in the system or not if it occurs then it applies some of the recovery methods to the system to get recover deadlock.

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 detect deadlock may occur then it avoid the deadlock.

OS

(Yes) 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.

(Yes)

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.

(Q8) 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 occurs 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

(Q9) 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 processes can progress that's why deadlock state more critical than starvation.