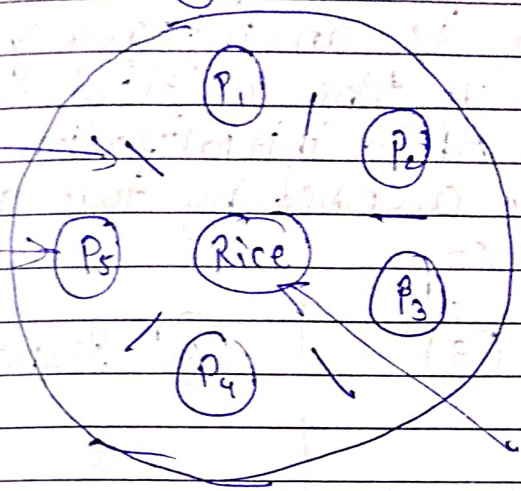
OS

Page No.

Dining Philosphers Problem.



Two Modes they work

1. Thinking
2. Eating

Shared resource

* Solution :

Maximum 2 person's can eat at the same time

Semaphores

* Solve Critical Section problem

** 3 criterias

Enter the C.S

P1 → CS

P2 X

1. Mutual Exclusion — one process at a time can enter the C.S
2. Progress ← P1 P2 P3 P4
3. Bounded wait

* Semaphore is an integer variable that solve the critical section problem. After initialization it can only be accessed by two atomic operations :-

(1) wait (S)	(2) Signal (S)
{	{
while (S<=0);	S = S+1;
S = S-1;	}
}	

* Solution of Critical Section Probem using Semaphores

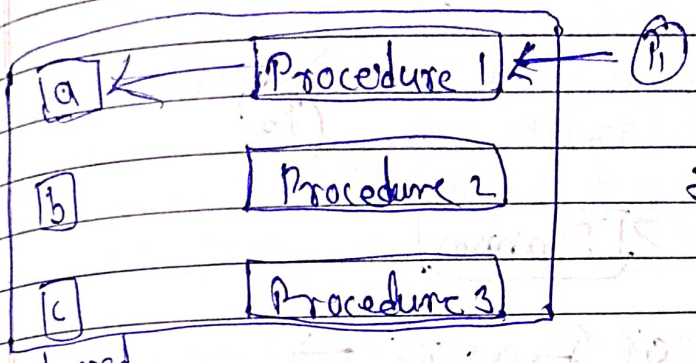
Let, $P_1, P_2, P_3, P_4 \ldots P_n$ are the process wants to go to the critical section

Types samphors
- Binary: 0, 1
- Counting: $-\infty$ to ∞

```
do
{
wait (S);
// critical section
signal (S);
// remainder section
} while (T)
```

Page No.
Date

Monitors



with the help of the procedure it can communicate with the shared data.

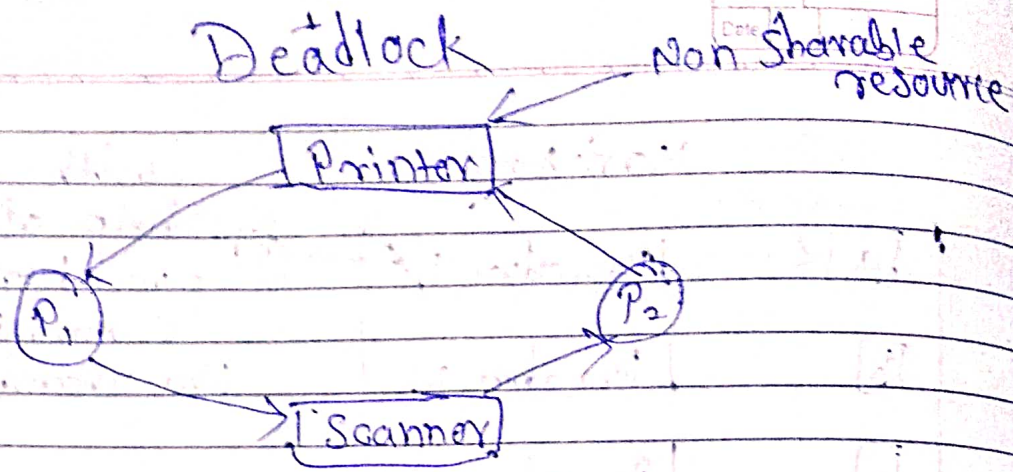
A monitor is a module that contains :-

- shared data
- procedures that operates on the shared data

→ Syntax of monitor:

```
monitor
{ condition variables;
  variables
  procedure1
  {
  }
procedure 2
  {
  }
```

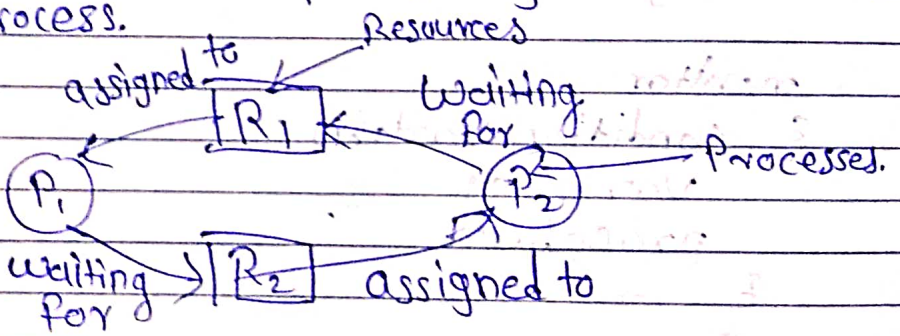
Deadlock



Defination of Deadlock →

Deddlock is a situation where a set of processes are blocked.

Because each process is holding a resoursce and waiting for another resource acquired by some other process.



** Four Necessary Conditions for deadlock to occure

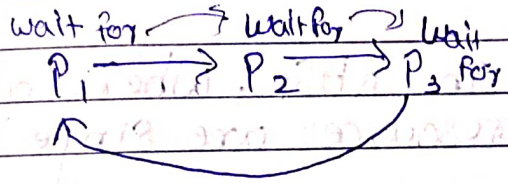
(1) Mutal Exclusion → one or more than one resource are non sharable (only one process can use it)

(2) Hold and wait → A process is holding at least one resource

and waiting for another resource.

(3) No Preemption :→ A resource cannot be taken from a process unless the process releass that resource.

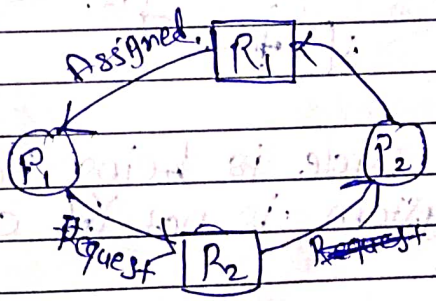
(4) Circular wait :→ A set of processes are waiting for each other in circular form:



*** Resource Allocation Graph (RAG)

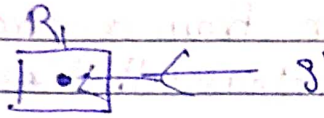
○ ← Process Denoted
□ ← Resource Denoted.

2 edges
(1) Request
(2) Assignment

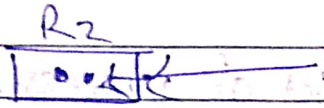


- Using RAG deadlock can be easily detected
- It is a Graph that represents the State of a System pictorially
- It has vertices and edges.

(1) Request edge : $P_i \rightarrow R_j$
(2) Assignment edge $R_j \rightarrow P_i$



Single instance

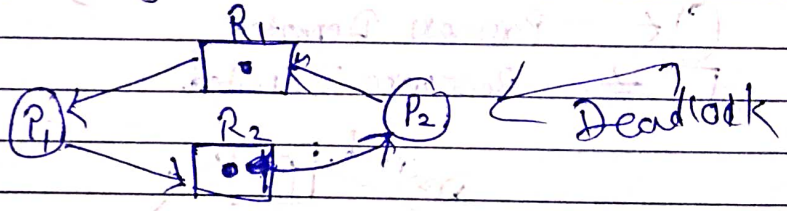


Multiple Instance

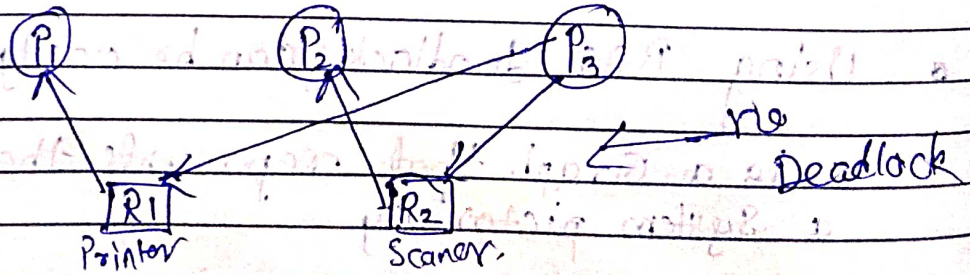
* Rules to detect deadlock using RAG

Rule-1 → In RAG, where all the resources are single instance

(a) If a cycle is being formed, then the system is in a deadlock state



(b) If no cycle is being formed, then the system is not in a ~~ded~~ deadlock state.



Banker's Algorithem.

(Deadlock Avoidance Algo)

Total : A = 10, B = 5, C = 7

Process	Allocation A B C	Max need A B C	Available A B C	Remaining need A B C
P_1	0 1 0	7 5 3	3 3 2	7 4 3 P_1
P_2	2 0 0	3 2 2	5 3 2	1 2 2 P_2
P_3	3 0 2	9 0 2	7 4 3	6 0 0 P_3
P_4	2 1 1	4 2 2	7 4 5	2 1 1 P_4
P_5	0 0 2	3 3 3	7 5 5	5 3 1 P_5
	7 2 5			

* Max − Allo = Remaining

$P_2 \rightarrow P_4 \rightarrow P_5 \rightarrow P_1 \rightarrow P_3$

Safe Sequence

Disk Scheduling Algorithem

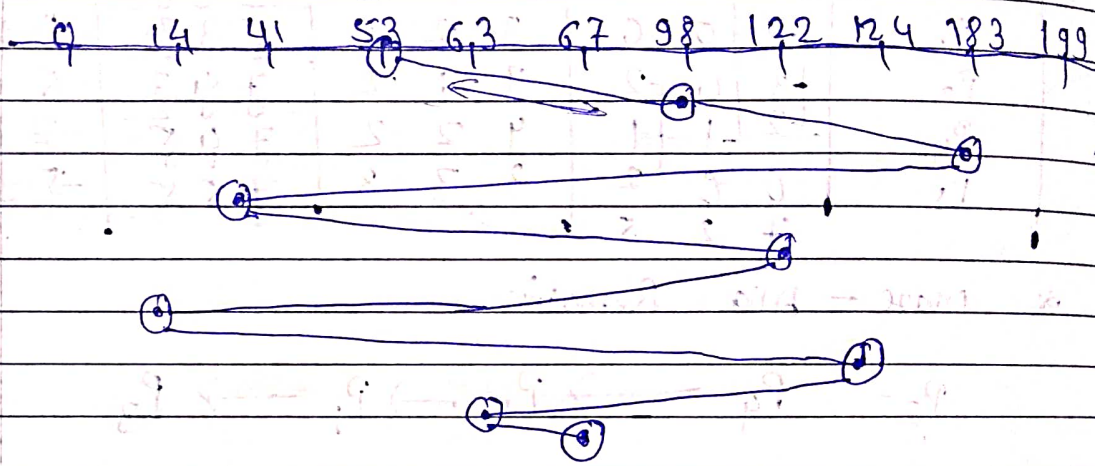
1. FCFS → Stand for first come first serve.
2. SSTF → Stand for short Seek Time first
3. SCAN → also known as elevator Algorithem
4. CSCAN → Cirular Scan.

Page No.
Date

FCFS : (First come First Serve)

Track request → 98, 183, 41, 122, 14, 124, 65, 67.

Current →
Read/write head is at → 53



Total track movement by Read/write head.

(98 − 53) = 45
+ (183 − 98) = 85
+ (183 − 41) = 142
+ (122 − 41) = 81
+ (122 − 14) = 108
+ (124 − 14) = 110
+ (124 − 65) = 59
+ (67 − 65) = 2

Ans ⇒ 632

Page No.
Date

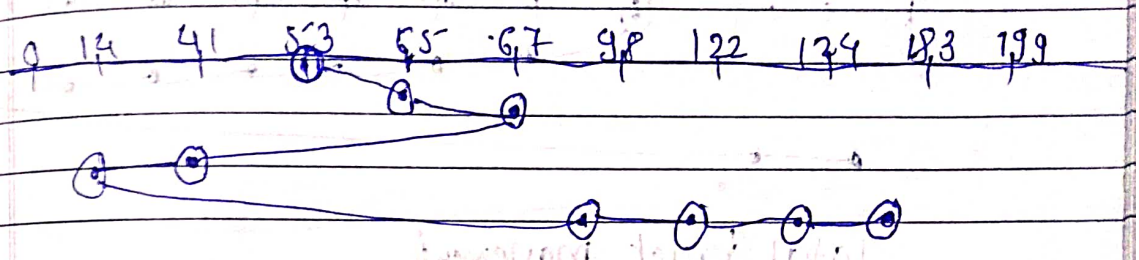
Nearest Track

SSTF (Short Seek Time First)

Track request → 98, 183, 41, 122, 14, 124, 65, 67.

current 7

Read/write head is at → 53



Total Track Movement

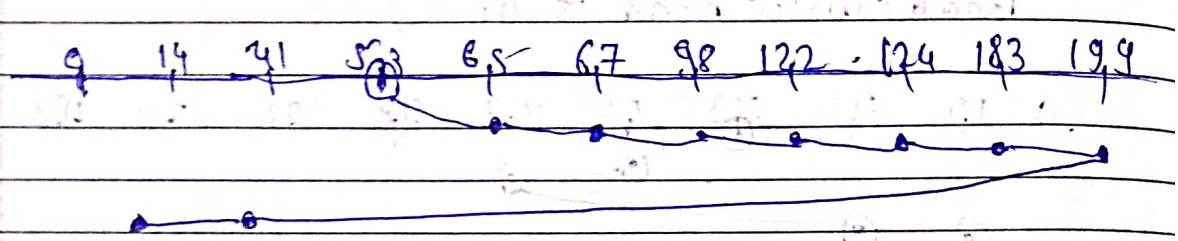
= (65 − 53) = 12
+ (67 − 65) = 2
+ (67 − 41) = 26
+ (41 − 14) = 27
+ (98 − 14) = 84
+ (122 − 98) = 24
+ (124 − 122) = 2
+ (183 − 124) = 59

Ans → 236

SCAN (ELEVATOR)

Track request → 98, 183, 41, 122, 14, 124, 65, 67

Read/write head is at → 53



Total Track movement

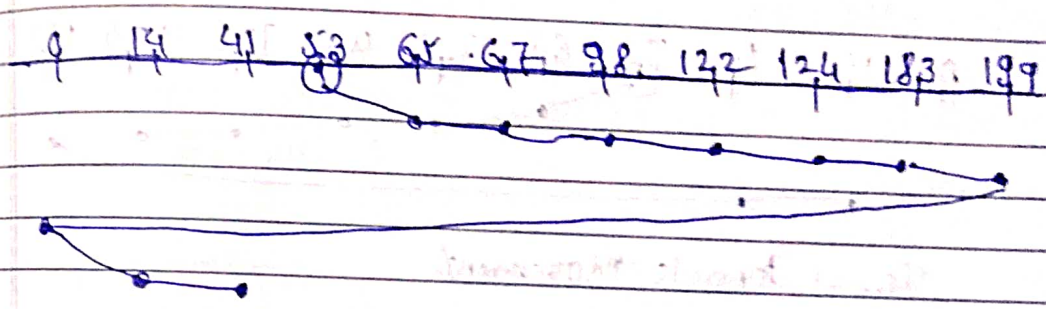
(65 − 53) = 12
+ (67 − 65) = 2
+ (98 − 67) = 31
+ (122 − 98) = 24
+ (124 − 122) = 2
+ (183 − 124) = 59
+ (199 − 183) = 16
+ (199 − 41) = 158
+ (41 − 14) = 27

Ans → 331

Circular SCAN (CSCAN)

Track request → 98, 183, 41, 122, 14, 124, 65, 67

Read / write head is at → 53



Total Track Movement

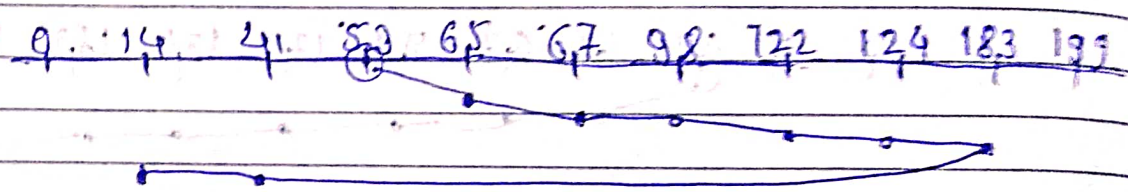
(65 - 53)
+ (67 - 65)
+ (98 - 67)
+ (122 - 98)
+ (124 - 122)
+ (183 - 124)
+ (199 - 183)
+ (199 - 0)
+ (14 - 0)
+ (41 - 14)

Ans → 386

LOOK

Track request → 98, 183, 41, 122, 14, 124, 65, 67

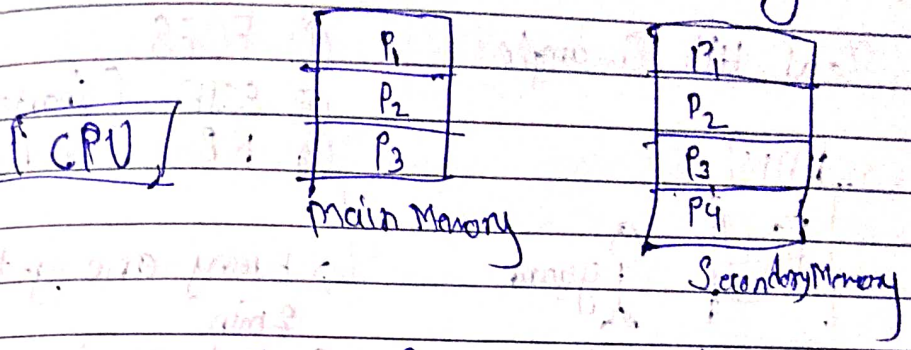
Read / write head is at → 53



Total track Movement

$$
\begin{aligned}
&(65-53)\\
+\ &(67-65)\\
+\ &(98-67)\\
+\ &(122-98)\\
+\ &(124-122)\\
+\ &(183-124)\\
+\ &(183-41)\\
+\ &(41-14)\\
&\hline
&299
\end{aligned}
$$

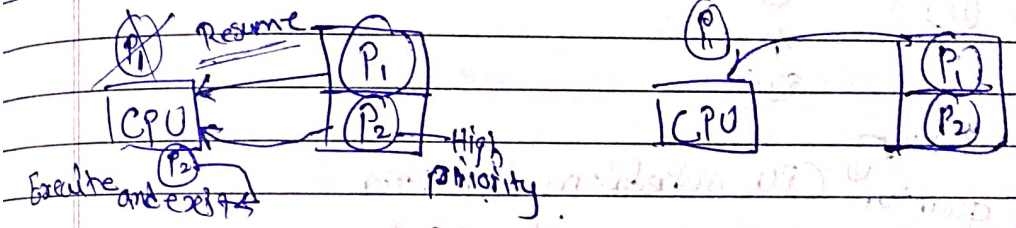
Intro to CPU Scheduling



Types of CPU scheduling

Pre-emptive Scheduling

non-preemptive scheduling



P2 preempt P1

In this a scheduler may preempt a low priority running process anytime when a high priority process enters into a ready state.

In this once a process enters the running state; it cannot be preempted until it completes its allocated time.

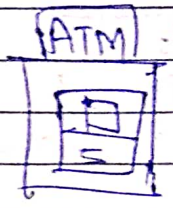
indefinet time ↔ Starvation problem

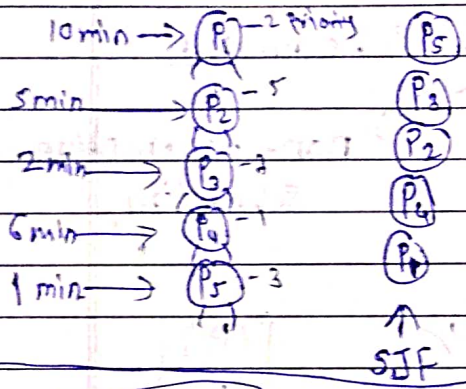
Page No.
Date

Rules

* Real life Example

(1) FCFS
(2) SJF (Shortes Job First)
(3) RR (Round Robin)
⋮
↳ Every one gets the 2 min
(4) Priority

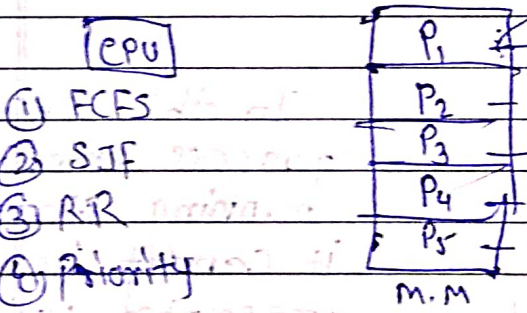




Every one 3 min ↓ Time quantum

CPU Scheduling Algorithms

(1) FCFS
(2) SJF
(3) RR
(4) Priority



(0.5)

The time required by the process for the execution is the Burst Time
(How many times execution)

↳ Every process has its own priority and the process which is having highest priority will get the CPU First

* * *

The Process have different times :-

① Arrival Time (AT) :- The Time when the process is arrived in the ready state.

② Burst Time (BT) :- The Time required by the process for its execution. (How

③ Completion Time (CT) :- The Time when the process completed its execcition. (How much time does it take)

④ Turnaround Time (TAT) :- It is ~~time~~ the time interval from the time of submission of a process to the time of the completion of of the process.

$$TAT = CT - AT$$

⑤ Waiting Time : The time difference between turnaround time and Burst time

OR

The time spent by a process waiting in the ready queue. For getting the CPU

$$WT = TAT - BT$$

⑥ Response Time :- The time difference b/w the 1st response and arrival time.

CPU Scheduling Algorithm

FCFS

- FCFS stands for First Come First Serve
- In this the process that arrives first, executed first.
- FCFS is non-preemptive in Nature
- In FCFS first process will get the CPU first.
- Other processes can get CPU only after the current process has fineshed its execution.
- Now suppose the first process has large burst time, and other processes hav less burst time, then the processes will have to wait more unnecessarily.
- This will result in more average waiting time.
- This effect is called as CONVOY EFFECT

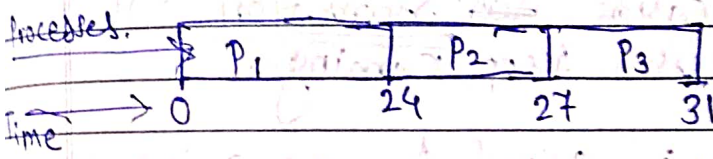
Page No.
Date

Example of FCFS

Process	Burst time
P_1	24
P_2	3
P_3	4

- Calculate Average waiting Time
- Calculate Turn Around Time.

Gantt Chart → is a horizontal bar chart to demonstrate CPU Sched



Waiting Time

$P_1 = 0$
$P_2 = 24$
$P_3 = 27$

$= \frac{0+24+27}{3}$

$= \frac{51}{3}$

Average = 17

Turn Around time.

$P_1 = 24$
$P_2 = 27$
$P_3 = 31$

$= \frac{24+27+31}{3}$

$= \frac{82}{3}$

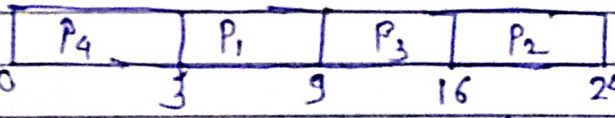
Average = 27

SJF

- SJF Stands for Shortest Job First
- In this the process that has the shortest burst time, executes first.
- SJF is both Preemptive and non-preemptive
- SJF may cause ~~for~~ Starvation ; if shorter processes keep coming.
- This problem is solved by ~~aging~~ aging

Example of SJF
Non-preemptive mode.

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3



Waiting Time

$P_4 = 0$
$P_1 = 3$
$P_3 = 9$
$P_2 = 16$

$\frac{0+3+9+16}{4} = 7$

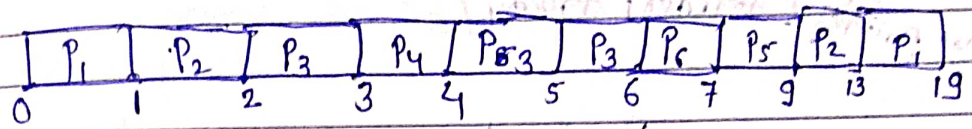
Turn Around Time

$P_4 = 3$
$P_1 = 9$
$P_3 = 16$
$P_2 = 24$

$\frac{3+9+16+24}{4} = 13$

SJF → Pre-emptive Mode

Process	Arrival Time	Burst Time
P_1	0	7 - 6
P_2	1	5 - 4
P_3	2	3 - 2 - 1
P_4	3	1 ✗
P_5	4	2 ✗
P_6	5	1 ✗



TAT = C.T − A.T

$P_1 = 19 - 0 = 19$
$P_2 = 13 - 1 = 12$
$P_3 = 6 - 2 = 4$
$P_4 = 4 - 3 = 1$
$P_5 = 9 - 4 = 5$
$P_6 = 7 - 5 = 2$

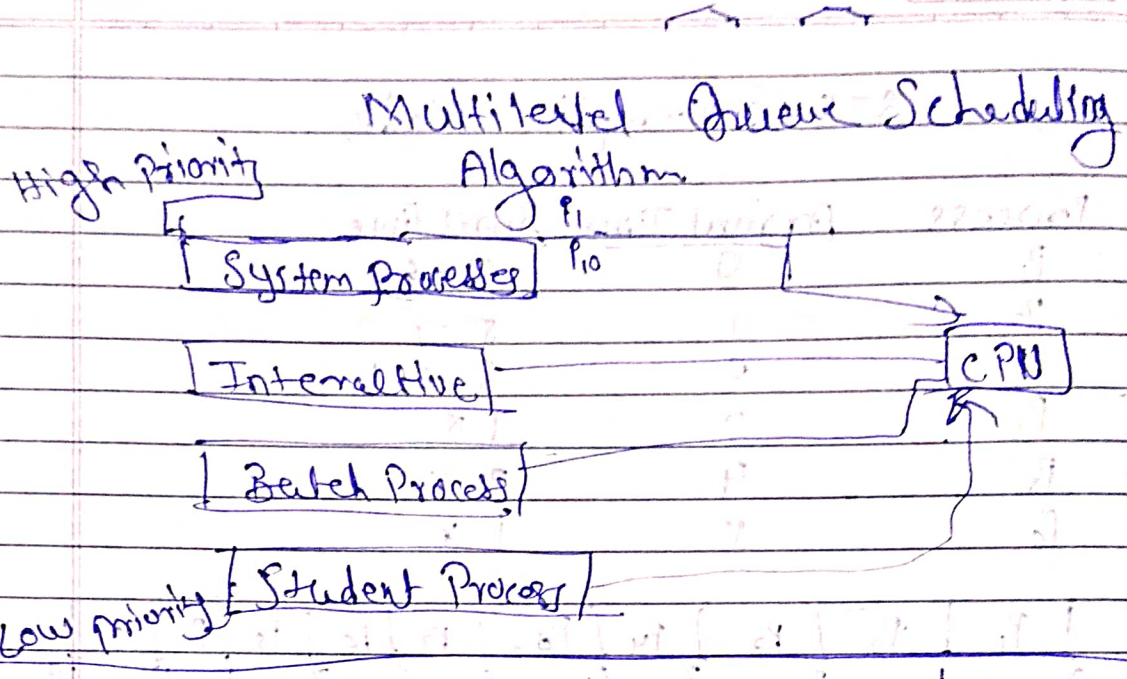
$$= \frac{19+12+4+1+5+2}{6}$$
$$= \frac{43}{6}$$
$$= \boxed{7.6}$$

Waiting Time = WT = TAT − BT

$P_1 = 19 - 7 = 12$
$P_2 = 12 - 5 = 7$
$P_3 = 4 - 3 = 1$
$P_4 = 1 - 1 = 0$
$P_5 = 5 - 2 = 3$
$P_6 = 2 - 1 = 1$

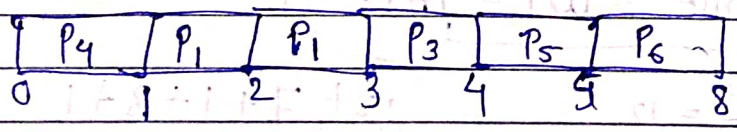
$$= \frac{12+7+1+3+1}{6}$$
$$= \frac{24}{6}$$
$$= \boxed{4}$$

Multilevel Queue Scheduling Algorithm



Priority CPU Scheduling Algo Preemptive.

Process	AT	Priority	B.T
P_1	1	5	4 -3-2
P_2	2	2	3
P_3	3	6	6-5
P_4	0	4	1
P_5	4	7	2
P_6	5	8	3



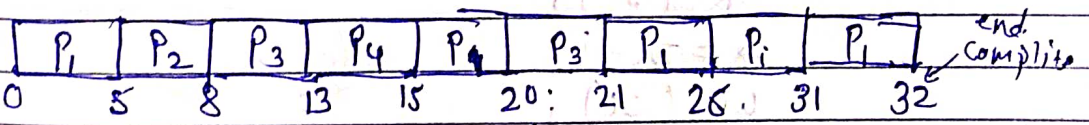
RR Round Robin

- RR stands for Round Robin Scheduling Algorithm.
- In this, each process is served by CPU for a fixed amount of time i.e. known as time quantum.
- RR is cyclic in nature, so there is no starvation.
- RR is preemptive is nature.

Example of RR.

Process	Burst Time.
P_1	21 - 16 - 11 + 6 - 1
P_2	36 X
P_3	6 - 1
P_4	2

Time quantum => 5



~~WT = TAT - BT~~

TAT = CT - AT

$P_1 = 32 - 0 = 32$

$P_2 = 8 - 0 = 8$

$P_3 = 21 - 0 = 21$

$P_4 = 15 - 0 = 15$

$\frac{32 + 8 + 21 + 15}{4}$

WT = TAT - BT

$P_1 = 32 - 21 = 11$

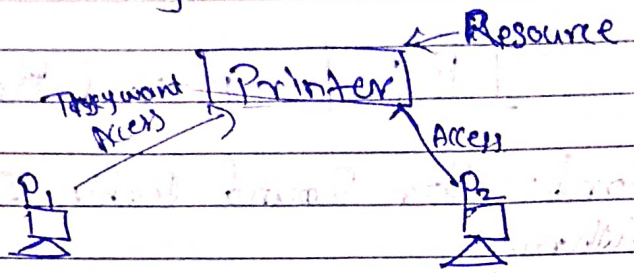
$P_2 = 8 - 3 = 5$

$P_3 = 21 - 6 = 15$

$P_4 = 15 - 2 = 13$

$\frac{11 + 5 + 15 + 13}{4} = \frac{44}{4} = 11$

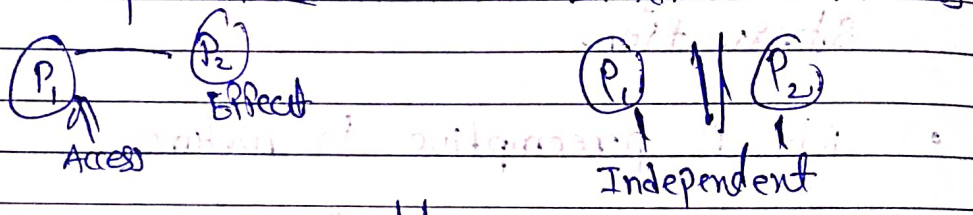
Synchronization in OS



They can not Access single printer

Two types of processes.

- Cooperative
- Interactive Process

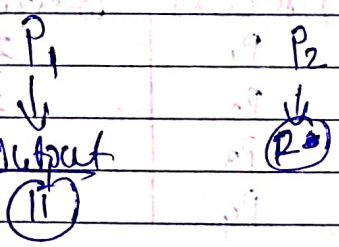


Race Condition in OS

R - Read
W - write

Shared variable → x = 10

```
Run()
{
  R(x);
  x = x+1;
  W(x);
}
```

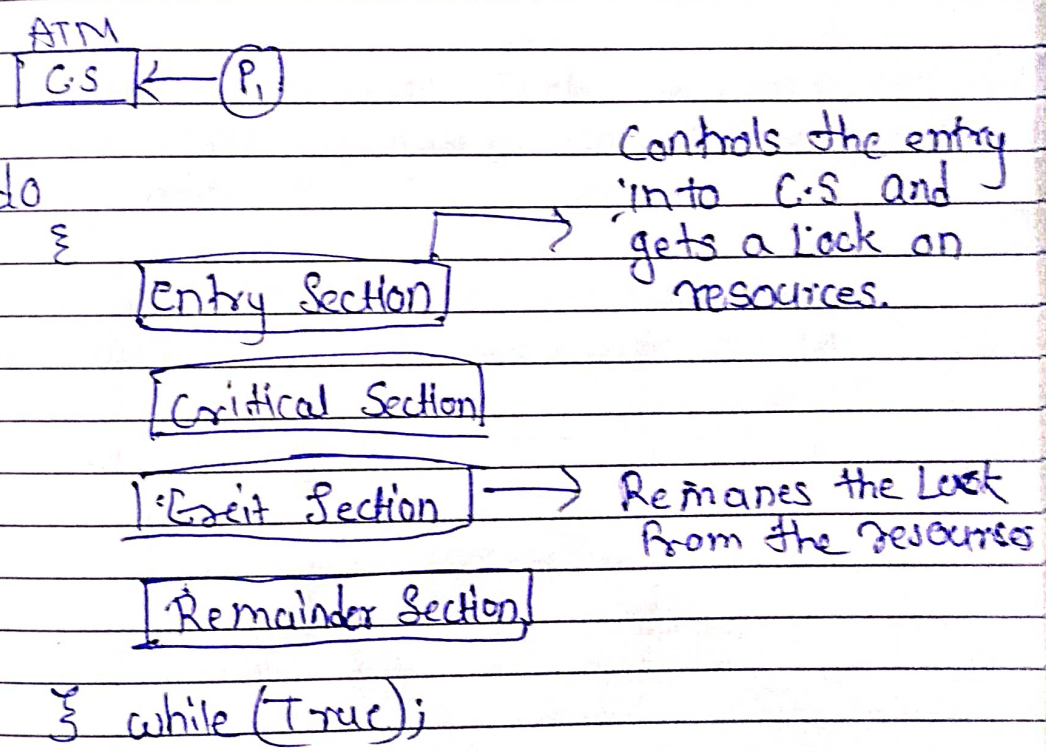


- When more than one processes are executing the same code or resource or any shared variable.
- In that condition there is a possibility

that the output as the value of that shared variable is wrong.

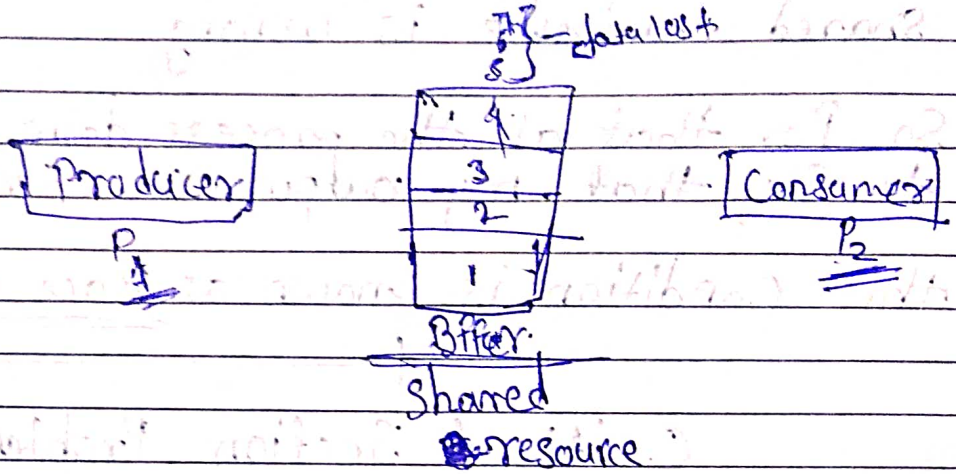
- So for that all the process doing race to say that my output is correct.
- this condition is known as Race Condition

Critical Section Problem in OS

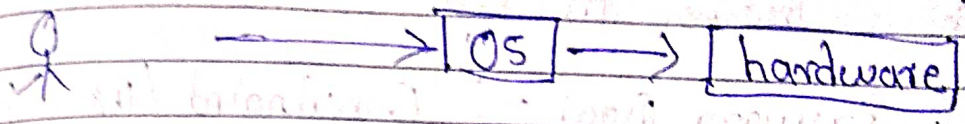


Classical Problem of Synchronization

Producer - Consumer / Bounded Buffer Problem



OS



user knows only high level language

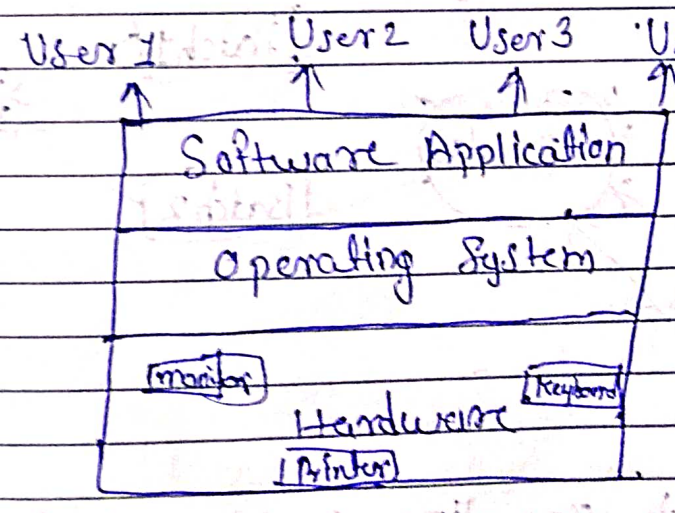
hlw knows only machine language

* OS Definition

Operating System in an intermediate between the user and the hardware.

Software application such as word, ms-excel or any user application can't Communicate directly with the hardware.

So we need a software interface between them and that ~~inf~~ interface is an Os.



** Goals of OS.

(1) Primary Goal → Convinaint / User Friendly

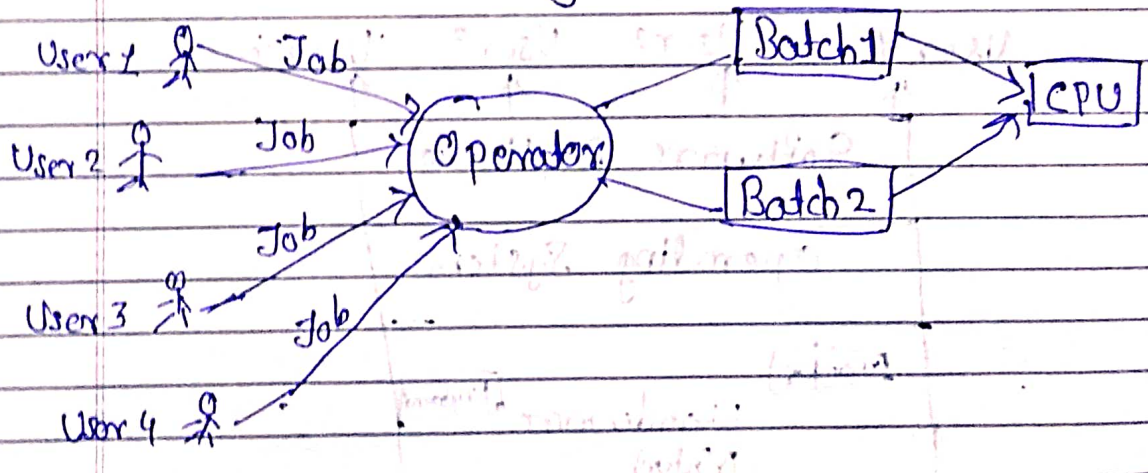
(2) Secondary Goal → Efficiency.

* Functions of OS

(1) Memory Management
(2) Process —11—
(3) I/O Device —11—
(4) File —11—
(5) Network —11—
(6) Security. —11—

Types of OS

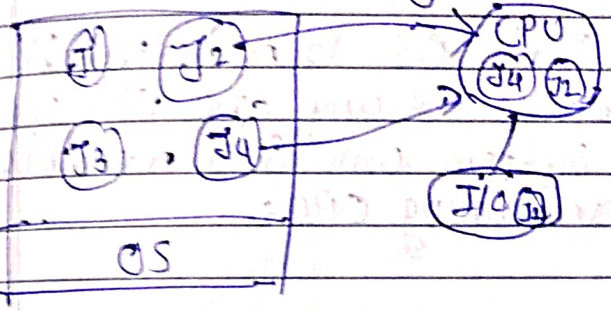
(1) Batch Operating System.



- In Batch Operating System, Similar kinds of Jobs are grouped together in a batch and loaded in the main memory.

- The responsibility of the O.S is to execute each job in a batch.
- Every job requires two times (1) CPU time (2) I/O time to execute.
- disadvantage efficeincy and throughput will decrease.

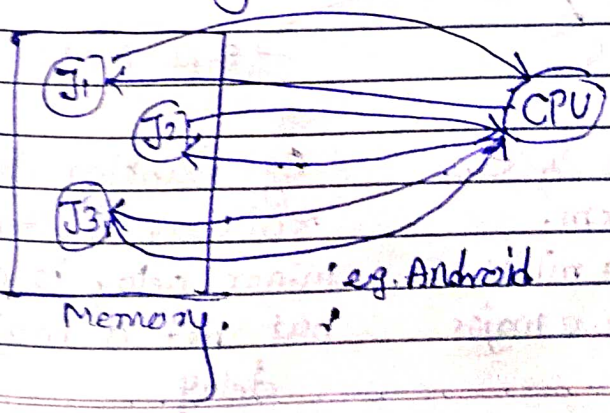
(2) Multiprogramming OS



- Multiprogramming means multiple Jobs are loaded in main memory.
- But in this, if in between any job is in CPU and requires I/O operation, it will leave CPU

- Advantage — CPU utilization and throughput will increase.

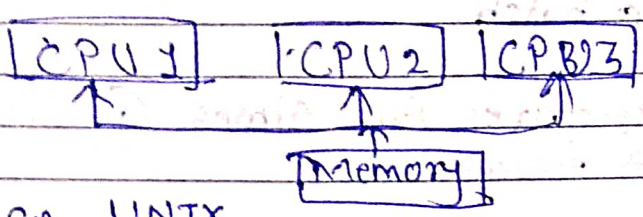
(3) Multitasking OS



- In multitasking operating system job will be executed in time sharing mode.

eg. Android

(4) Multiprocessor O.S



e.g. UNIX

- Multiprocessor means multiple CPUs. This is also called Parallel System.

Advantages → This OS is more reliable because if any one CPU will fail, the system will still run with the help of remaining CPU's.

(5) Real Time OS

The System which are time bounded are called as Real time OS.

Time bound means there should not be any delay while executing the job.

RTOS is Two Types

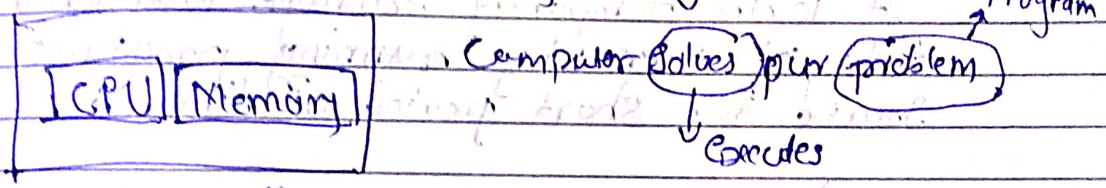
Hard RTOS → Ex - Satellite System, missile System. In this even a minor delay lead to a major loss.

Soft RTOS → eg. Banking Sector, railway System. Minor delay is accepted but not a major delay.

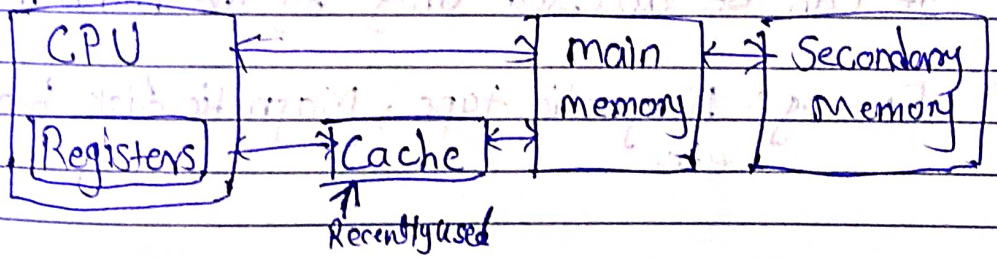
⑥ Embedded OS

- This is a type of Operating System that is embedded and specifically configured for a certain hardware configuration.
- This is mostly used in Micro wave, Washing Machine etc.

Memory Mangement



* 3 Criterias.
① Size ② Access Time ③ Cost



* Registers :- It is a temporary Storage area built in a CPU. Access time of Register is below 1ns, and Registers have lowest capacity i.e. few KB of words.

* **Cache Memory** :- It is a high speed memory. The purpose of cache memory is to store those programs that are repeatedly used or likely to be used in the near-future.

* **Main Memory** :- It is also known as Primary memory or RAM of physical Memory. The CPU can execute only the data that is present in the main memory. It is the temporary storage location where data of currently running programs are stored for short period of time.

* **Secondary Memory** :-
  - It allows user to store data that can be easily retrieved.
  - This memory cannot be directly used.
  - It can be accessed only by the main memory.
  - For e.g:- Magnetic tape, Magnetic disk, hard disk etc.