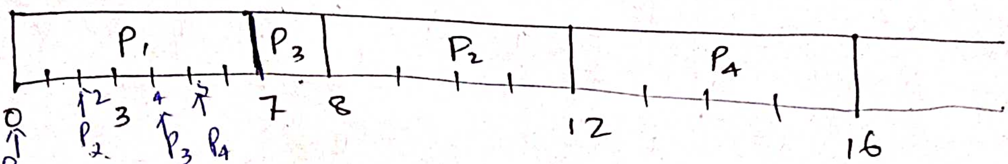
2019

Example of Non-preemptive SJF

Process	Arrival Time	Burst Time
P_1	0.0	~~7~~ 5
P_2	2.0	4 / 2
P_3	4.0	1
P_4	5.0	4

- SJF (non-preemptive)

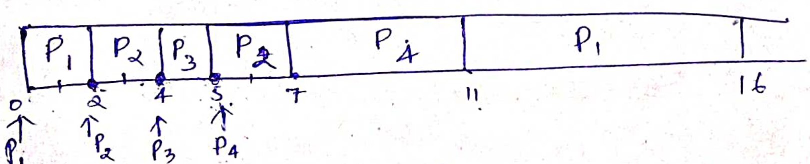


- Average waiting time = (0 + 6+3+7)/4 = 4
- TA = WT + CPU Burst time
- $TA\,P_1 = 0 + 7 = 7$
  $TA\,P_2 = 6 + 4 = 10$
  $TA\,P_3 = 3 + 1 = 4$
  $TA\,P_4 = 7 + 4 = 11$

  Avg. TA = (7 + 10 + 4 + 11)/4
  = 32/4 = 8 (in mili second)

SRTF

- Gantt chart



$P_1 - 5$
$P_2 - 2$
$P_4 - 4$

$WT(P_1) = 0 + 9 = 9$
$WT(P_2) = 0 + (3-4) = 1$
$WT(P_3) = 0$
$WT(P_4) = 7 - 5 = 2$

Avg. WT = (9+1+0+2)/4 = 3

$TA(P_1) = 9+7 = 16$

$TA(P_2) = 1+4 = 5$

$TA(P_3) = 0+1 = 1$

$TA(P_4) = 2+4 = 6$

$$\text{Avg. } TA = \frac{16+5+1+6}{4} = \frac{28}{4} = 7.$$

- Determining Length of Next CPU Burst

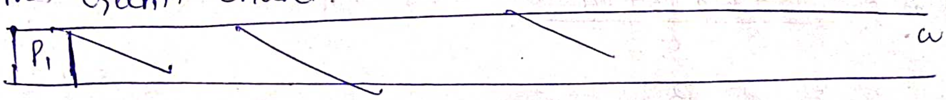
- Priority scheduling

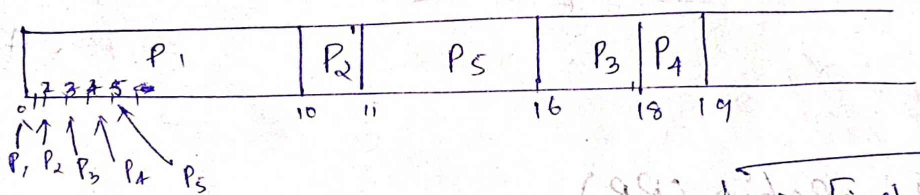
Process	Burst Time	Priority	AT
P_1	10	3	0
✓ P_2	1	1	2
P_3	2	4	3
P_4	1	5	4
✓ P_5	5	2	5

- Gantt Chart

After including Arrival Time (AT):

(Non- preemptive) Gantt chart:





WT(P_2) = 0
WT (P_5) = 1
WT P(1) = 6
WT (P3) = 16
WT (P4) = 18

Avg = $\frac{0+1+6+16+18}{5}$
= $\frac{41}{5}$ = 8.2

→ First process is allowed to complete and then others are completed one by one on the basis of their priority.

$WT(P_1) = 0$

$WT(P_2) = 10 - 2 = 8$

$WT(P_3) = 16 - 3 = 13$

$WT(P_4) = 18 - 4 = 14$

$WT(P_5) = 11 - 5 = 6$

Avg WT = $\frac{0+8+13+14+6}{5} = \frac{41}{5}$ = 8.2 mili sec.

$TA(P_1) = 0 + 10 = 10$

$TA(P_2) = 1 + 8 = 9$

$TA(P_3) = 2 + 13 = 15$

$TA(P_4) = 1 + 14 = 15$

$TA(P_5) = 5 + 6 = 11$

Avg TA = $\frac{10+9+15+15+11}{5}$
= $\frac{60}{5}$
= 12 mili sec.

22/9

Priority Scheduling

Preemtive: → Once process starts, if higher priority process joins the ready queque, it is interupted and higher priority process is completed first and so on.

P_1	P_2	P_1	P_5	P_1	P_3	P_4

0 2 3 5 10 16 18 19

↑ P_1

$WT(P_1) = 6$

$WT(P_2) = 0$

$WT(P_3) = 16-3 = 13$

$WT(P_4) = 18-4 = 14$

$WT(P_5) = 0$

WT BT

$TA(P_1) = 6+10 = 16$

$TA(P_2) = 0+1 = 1$

$TA(P_3) = 2+13 = 15$

$TA(P_4) = 1+14 = 15$

$TA(P_4) = 5+0 = 5$

$$\text{Avg WA} = \frac{6+0+13+14+0}{5} = \frac{33}{5} = 6.6 \text{ ms}$$

$$\text{Avg TA} = \frac{16+1+15+15+5}{5} = \frac{52}{5} = 10.4 \text{ ms}$$

Round Robin (RR)

- Each process gets a small unit of CPU time (time quantum), usually 10-100 miliseconds. After this time has elapsed, the process is preemted and added to the end of the ready queque.
- If there are n processes in the ready queque and the time quantum is q, then each process gets 1/n of the CPU time in chunks of at most q time units at once. No process waits more than

$(n-1)q$ time units.

- Performance :
  - ~ q large ⇒ FIFO
  - ~ q small ⇒ q must be large with respect to context switch otherwise overhead is too high.

- Example ~~with~~ of RR with Time Quantum = 20

Process	Burst Time
P_1	53 / 33 / 13 ✓
P_2	17 ✓
P_3	68 / 48 / 28 / 8
P_4	24 / 4 ✓

- The Gantt Chart is :

P_1	P_2	P_3	P_4	P_1	P_3	P_4	P_1	P_3	P_3

0 20 37 57 77 97 117 121 134 154 162

- Typically, higher average turnaround than SJF, but better response.

$WT(P_1) = 57 + 24 = 81$

$WT(P_2) = 20$

$WT(P_3) = \cancel{30}\ 40 + 37 + 17$
$= 94$

$WT(P_4) = 57 + 40 = 97$

$TA(P_1) = 81 + 53 = 134$

$TA(P_2) = 20 + 17 = 37$

$TA(P_3) = 94 + 68 = 162$

$TA(P_4) = 97 + 24 = 121$

$$\text{Avg WT} = \frac{81 + 20 + 94 + 97}{4} = \frac{292}{4} = \cancel{53.75}\ ms = 73\ ms$$

$$\text{Avg TA} = \frac{134 + 37 + 162 + 121}{4} = \frac{454}{4} = 113.5\ ms$$

P_1 53
P_2 17 ✓
P_3 68
P_4 24 ✓

If Time Quantum = 10 ms

P_1	P_2	P_3	P_4	P_1	P_2	P_3	P_4	P_1	P_3	P_4	P_1	P_3	P_1	P_3	P_1	P_3	P_3

0 10 20 30 40 50 57 67 77 87 97 101 111 121 131 141 144 154 162

7

$WT(P_1) = 30 + 27 + 14 + 10 + 10 = 91$

$WT(P_2) = 10 + 30 = 40$

$WT(P_3) = 20 + 27 + 20 + 14 + 10 + 3 = 94$

$WT(P_4) = 30 + 27 + 20 = 77$

$$\text{Avg WT} = \frac{91 + 40 + 94 + 77}{4} = \frac{302}{4} = 75.5 \text{ ms}$$

$TA(P_1) = 91 + 53 = 144$

$TA(P_2) = 40 + 17 = 57$

$TA(P_3) = 94 + 68 = 162$

$TA(P_4) = 77 + 24 = 101$

$$\text{Avg TA} = \frac{144 + 57 + 162 + 101}{4}$$

$$= \frac{464}{4}$$

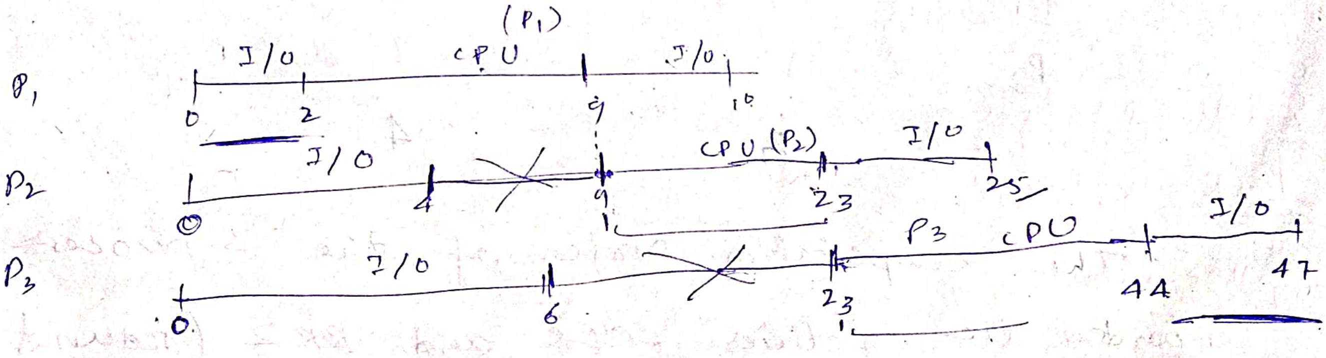
$$= \underline{116} \text{ ms.}$$

26/9 ~~Multilevel Queue~~

- Consider three process, all arriving at time zero, with total execution time of 10, 20 and 30 units, respectively. Each process spends the first 20% of execution time doing I/O, the next 70% of time doing computation, and the last 10% of time doing I/O again. The operating system uses a shortest remaining compute time first scheduling algorithm and schedules a new process either when the running process gets blocked on I/O or when the running process finishes its compute burst. Assume that all I/O operations can be overlapped as much as possible. Find the percentage of time does the CPU remain idle

Ans: 10.6 %.

Process	20% I/O	70% CPU	10% I/O	BT
P_1	2	7	1	10
P_2	4	14	2	20
P_3	6	21	3	30



CPU Idle time: 2 + 3 = 5 units

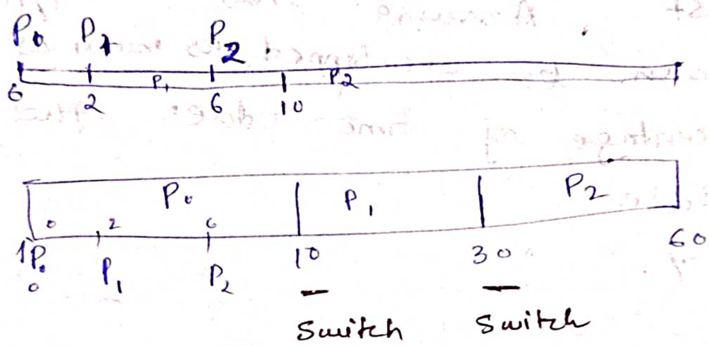
Total time: 47 units

% CPU ~~non~~ idle time = $\frac{5}{47} \times 100$ = 10.638 %

- Consider three intensive process, which require 10, 20, and 30 time units and arrive at times 0, 2 and 6 resp. How many context switches are needed if the operating systems implements a shortest remaining time first scheduling algorithm? Do not count the context switches at time zero and at the end.

Ans: Only two context switches are needed P_0 to P_1 and P_1 to P_2.

	AT	CPU
P_1	0	10
P_2	2	20
P_3	6	30



- Consider 3 process P_1, P_2 and P_3 shown in the table.

Process	A T	Time Units Required
P_1	0	5
P_2	1	7
P_3	3	4

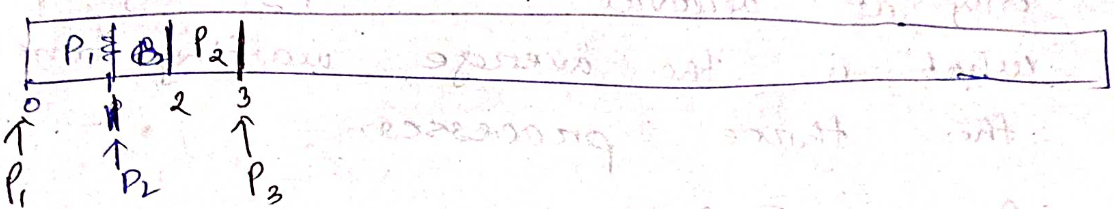
The computation order of the 3 process under the policies FCFS and RR2 (Round Robin scheduling with CPU quantum of 2 time units) and

Ans: FCFS: $P_1, P_2\ P_3$

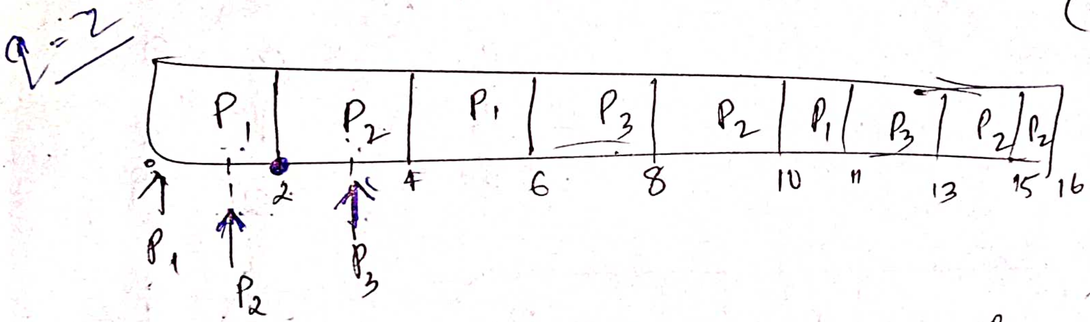
In RR_2 time slot of 2 units, Processes are assigned in following order:

$P_1, P_2, P_1, P_3, P_2, P_1, P_3, P_2, P_2$

RR2: P_1, P_3, P_2

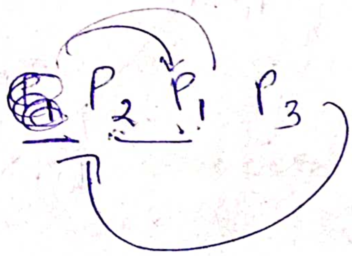


(n-1/q)



P_1 $P_3\ P_2\ P_1$

P_1 ✓ P_3 ✓



A: Consider the following table of arrival time and burst time for 3 processes P_0, P_1 and P_2.

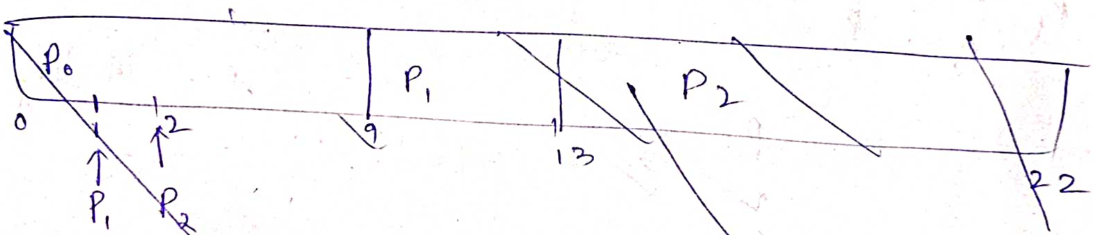
Process Arrival time and Burst time

P_0 : 0 ms 9 ms P_1 1 ms 4 ms P_2 2 ms 9 ms.

The pre-emptive shortest job first scheduling algorithm is used. Scheduling is carried out only at arrival or completion of process. What is the average waiting time for the three processes.

Ans: 5.0 ms.

Process	AT	BT
P_0	0	9 / 8
P_1	1	4 / 3
P_2	2	9

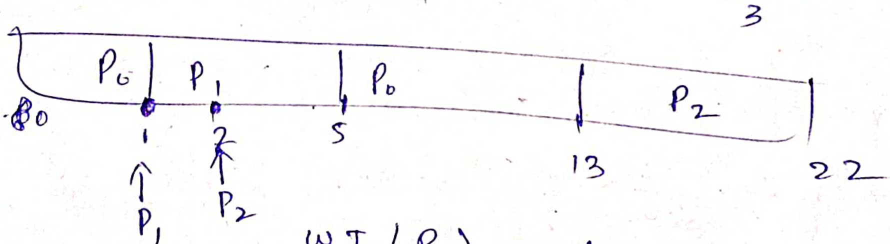


~~$WT(P_1) = 0$~~

~~$WT(P_2) = 8$~~

~~$WT\ P_3 = 11$~~

~~$Avg = \frac{0+8+11}{3} = \frac{19}{3}$~~



$WT(P_0) = 4$

$WT(P_1) = 0$

$WT(P_2) = 13 - 2 = 11$

$Avg\ WT = \frac{4+0+11}{3} = \frac{15}{3}$

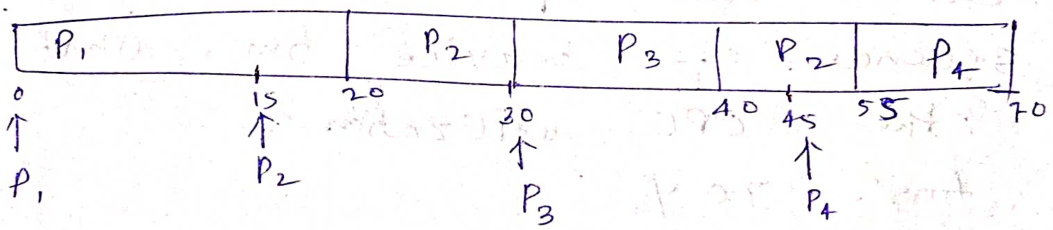
$= 5$

5. An operating system uses Shortest Remaining Time first (SRT) process scheduling algorithm. Consider the arrival and execution times for the following process.
Process Execution time, Arrival time
P1 : 20 0, P2 : 25 15, P3 : 10 30, P4 : 15 45. What is the total waiting time for process P2.

SRTF

Ans: 15 (P1, P2, P3, P2, P4)

Process	Ex. Time	AT
P_1	20	0
P_2	25	15
P_3	10	30
P_4	15	45



Total WT for P_2 = 5 + 10 = 15 ms

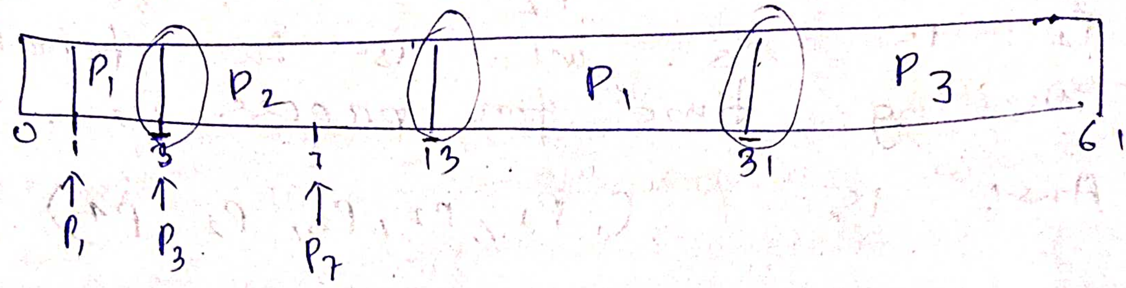
6. Consider three CPU intensive process P_1, P_2, P_3 which require 20, 10 and 30 units of time, arrive at times 1, 3. and 7 respectively. Suppose operating system is implementing Shortest Remaining Time first (preemptive scheduling) algorithm, then ____ context switches are required (Suppose context switch at the begining of Ready queue and at the end of Ready queue are ~~coat~~ not counted

Ans = 3 (P_1, P_2, P_1, P_3)

? RTF

Process	BT	AT
P_1	20	1
P_2	10	3
P_3	30	7

13
18
―
31



Ans:- 3 switches.

7. In a system using single processor, a new process arrives at the rate of six processes per minute and each such process requires seven seconds of service time. What is the CPU utilization?

Ans: 70 %.

60 s = 1 min ⟶ 6 process

1 process ⟶ 7 seconds

6 " ⟶ $6 \times 7 = 42$ sec

∴ CPU utilization $= \frac{42}{60} \times 100$

$= 70\%$.

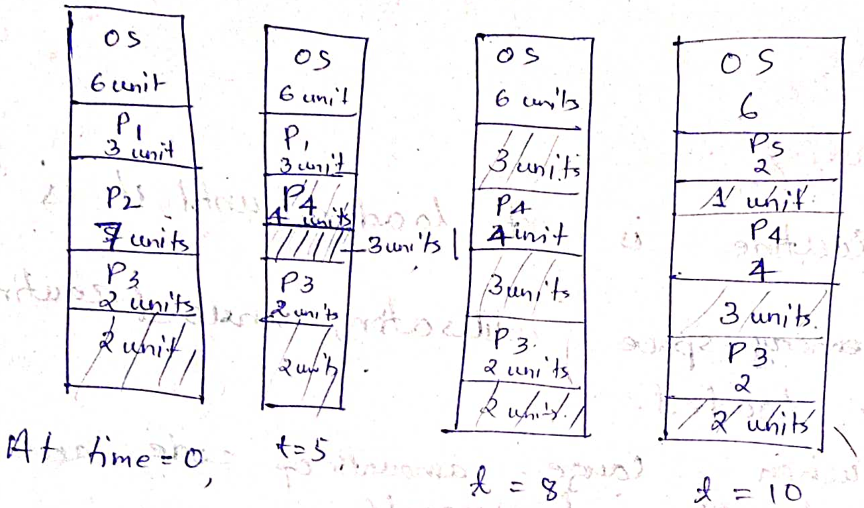
Overlays:

- Keep in .

First fit

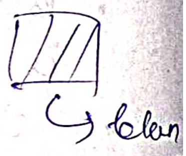
Table 4.1

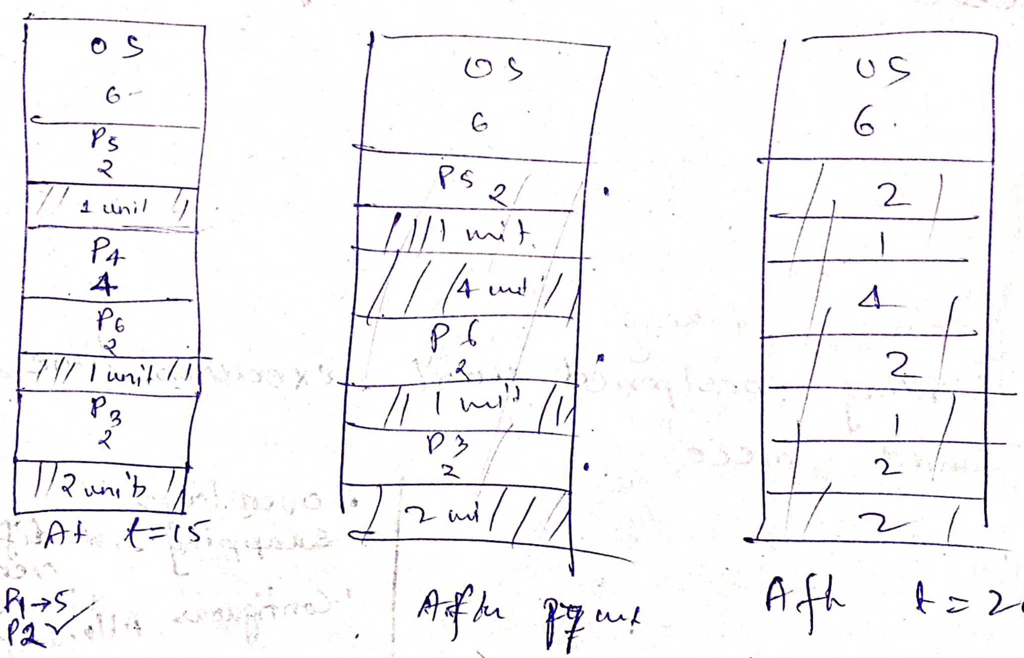
	P1	P2	P3	P4	P5	P6
AT	0	0	0	0	10	15
BT	8	5	20	12	10	5
Memory Req.	3 unit	7	2	4	2	2

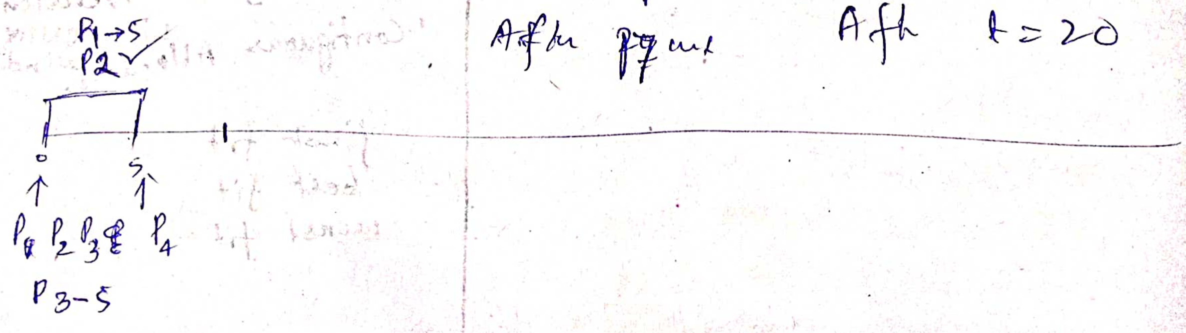


Given, OS Require 6 me unit

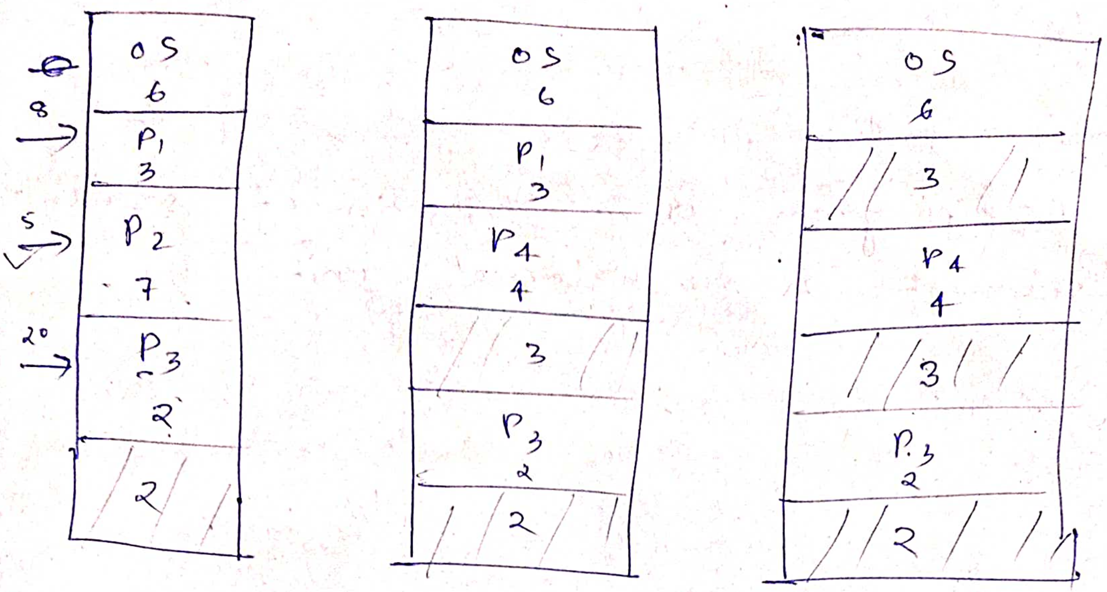
→ Total memory available → 20 un



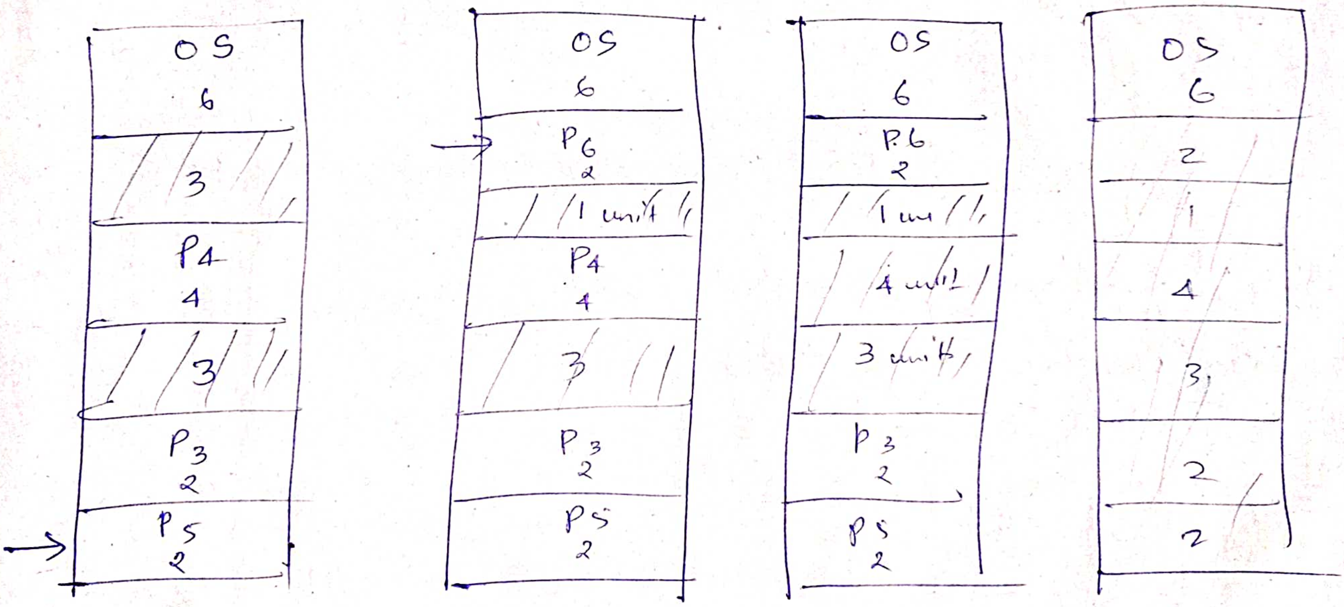




• Best Fit



At t = 0   After t = 5   After t = 8



After t = 10   After t = 15   After t = 17   After t = 20

• Paging :

• Address Translation Scheme.

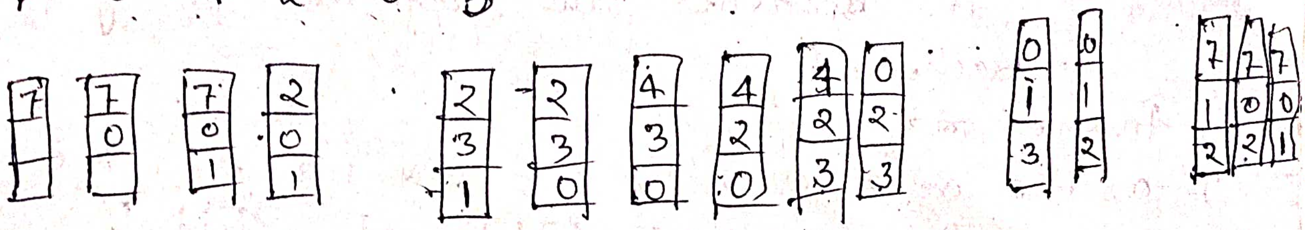
• Inverted Page Table
• Segmentation

11/10

- FIFO Page Replacement

→ Frame size = 3

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

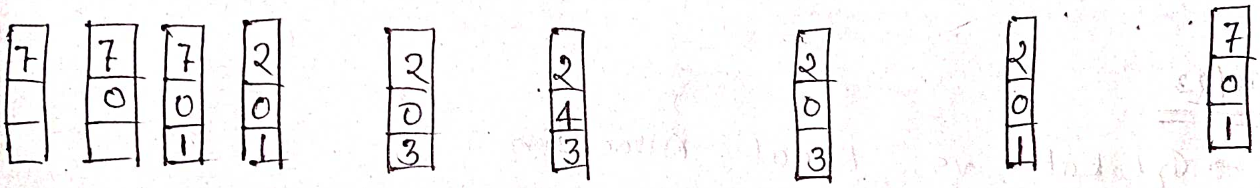


No. of page fault = 15

- Optimal Page Replacement :→

⇒ Frame size : 3

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

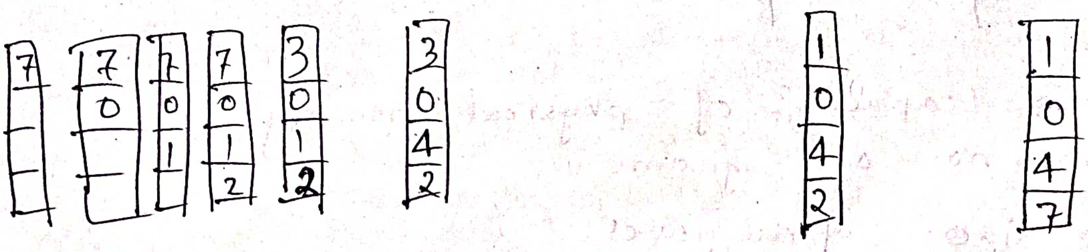


No. of page fault = 9.

↳ It will find out the page which will not be used for the longest period of time.

⇒ Frame size = 4

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1



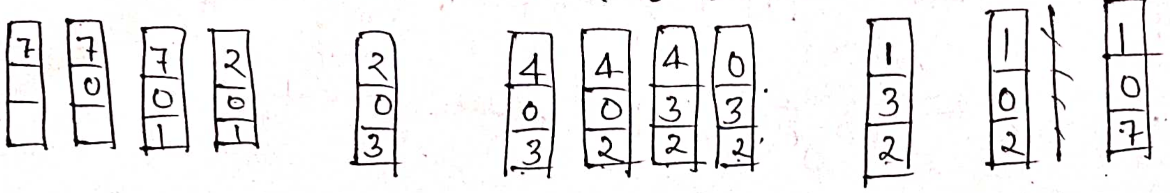
Page fault = 8.

- Least Recently Used (LRU)

↳ it searches the reference string backward.

frame size = 3

7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1



No. of page fault = 12

13/10/23

→ Global vs Local Allocation
→ Thrashing:

- If the capacity of physical memory is m unit and the size of each ~~size~~ page is p then the no. of frames in the physical memory will be f.

$$f = \frac{m}{p}.$$

m → capacity of physical memory
f → no. of frame
p → size of each pages

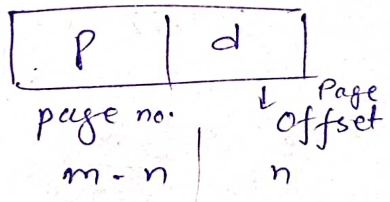
↳ No. of frames in physical memory and no. of pages in logical memory are identical

→ Both m & p are usually an integer, power of 2.
Thus resulting in f also being an integer.

→ The page size is similar to the frame size is defined by the hardware & it varies acc. to the architecture of the computer system.

→ For the convenience of mapping, page sizes are usually an integer power of 2.

→ If the size of logical address page is 2^m, and page size is 2^n unit.
Then the high order m−n units of logical address designate the page no. and the n low order units designate the page offset. Thus, the logical address is defined as



where p is the index no. in page table and d is the offset value.

Q. Consider a logical address page of 8 pages of 1024 words, each map onto a physical memory of 32 frames.
Then ① how many bits are in the logical address.
② how many bits are in physical address.

Soln: → Let No. of bits in physical mem. be m.

∴ Size of physical memory $= 2^m$

No. of pages $= 8 = 2^3$

Size of each page $= 1024$ words.

No. of frames $= 32 = 2^5$.

Size of each frame = size of page

$= 1024$ words

$= 2^{10}$

(2) $f = \frac{m}{p}$

$\Rightarrow 2^5 = \frac{2^m}{2^{10}}$

$\Rightarrow 2^m = 2^{15}$

$\Rightarrow m = \underline{15}$.

(1)

No. of page = $\frac{\text{size of logical memory}}{\text{size of page/frame}}$

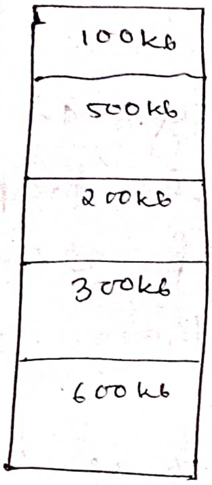
$\Rightarrow 2^3 = \frac{2^n}{2^{10}}$

$\Rightarrow 2^n = 2^{13}$

$\Rightarrow n = 13$

17/10/23

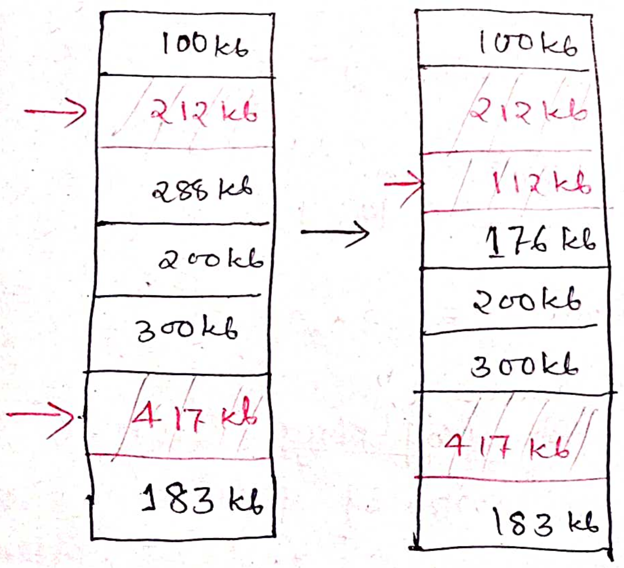
Q.1. Given 5 memory partition of 100 kb, 500 kb, 200 kb, 300 kb and 600 kb in order, how would the first fit, best fit and worst fit algorithm plays, processes 212 kb, 417 kb, 112 kb and 426 (in order). Which ~~ma~~ algo makes the most efficient use of memory?



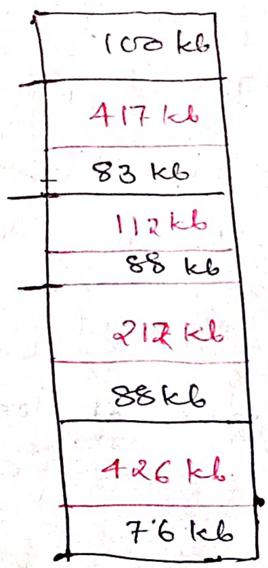
Processes

212 kb
417 kb
112 kb
426 kb

First fit:-



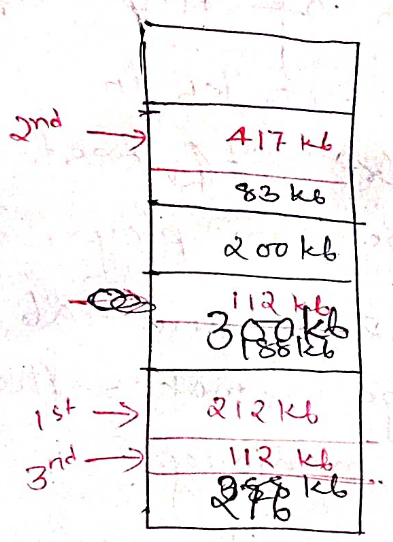
Best Fit



Worst fit

100
500
200
300
600

212 kb
417 kb
112 kb
426 kb



388
112
276

Q. Consider a logical address page of 64 pages of 1024 words each, map onto a physical memory of 32 frame

① How many bits are there in the logical address?

② How many bits are there in the physical address?

Soln:→

No. of pages $= 64 = 2^6$

Size of each page $= 1024 = 2^{10}$

" " " frame $= 1024 = 2^{10}$

No. of frame $= 32 = 2^5$

① Let size of logical mem $= 2^n$

No. of page $= \dfrac{\text{size of logical mem}}{\text{size of frame}}$

$\Rightarrow 2^6 = \dfrac{2^n}{2^{10}}$

$\Rightarrow 2^n = 2^{10+6} = 2^{16}$

$\Rightarrow n = 16$

(2) Let size of physical mem = 2^m.

$$f = \frac{m}{P}$$

$$\Rightarrow 2^5 = \frac{2^m}{2^{10}}$$

$$\Rightarrow 2^m = 2^{10+5} = 2^{15}$$

$$\Rightarrow m = 15$$

Q. How many pages will be required to convert a logical address of 18 bits into ~~a~~ physical address during mapping if the size of each page is 1k.

Sol^n! ⇒ No. of page = $\frac{\text{Size of logical mem}}{\text{Size of address}}$

$$= \frac{2^{18}}{2^{10}}$$

$$= 2^8$$

Q. Consider a paging system ~~of~~ with the page stored in memory?

(i) If a mem. refereance takes 200 ns, how long does a page mem. referance take?

TLB

Translation Lookaside Buffer.

(ii) If we add TLB and 75% of the all page table reference are found in TLB's what is effective mem. reference time (assume that finding a page table entry takes 0 time, if entry is there)

h → hit ratio t_1 → TLB access time
t_2 → memory " "

effective access time = hit ratio(t_1+t_2) + $(1-h)$
$(t_1+t_2+t_2)$

$= h\ (t_1+t_2) + (1-h)\ (t_1+2t_2)$

(ii) hit ratio = 75% = 0.75

$t_1 = 0$

$t_2 = 200$ ns

$T_{eff} = 0.75(0+200) + (1-0.75)(0+2\times200)$

$= 150 + 0.25 \times 400$

$= 150 + 100$

$= 250$ ns.

(i) 200

Q. TLB access time = 200 ns

hit ratio = 80%.

mem. access time = 100 ns

Eff access time.

$T_{eff} = .8(200+100) + 0.2(200+2\times100)$

$= 240 + 80$

$= 320$ ns

FC

SS
2
SC
:

19/10

Disk Structure

• Disk Scheduling :-

① FCFS → First Come, First Serve
② SSTF → Shortest Seek Time First
③ SCAN →
↳ 1st it goes to one end and then it get reversed to another end (starting from particular point in between)
1st towards small request
↳ Also called elevator algorithm.

⑤ C-SCAN → moves to one end, returns directly at starting, again it moves
↳ 1st towards last request

⑥ C-LOOK → it is like C-scan, it first moves towards the last request (not the end) then directly goes to the smallest request (not the begining) then then again moves towards.

Q. Queue: 98, 183, 37, 122, 14, 124, 65, 67

• Disk Head movement (start at 53) 640

FCFS / (98 - 53) + (183-98) + (183-37) + (122-37)
+ (122-14) + (124-14) + (124-65) + (67-65)
= 45 + 85 + 146 + 85 + 108 + 110 + 59 + 2
= 640

SSTF 236
SCAN 208

SCAN: (53-14) + (37-14) + (65-37) + (67-65) +
(98-67) + (122-98) + (124-122) + (183-124)
= 39 + 23 + 28 + 2 + 31 + 24 + 2 + 59
= 208

SSTF: (65-53) + (67-65) + (65-37) + (37-14) + (98-14) + (122-98) + (124-122) (183-124)
= 236 12 + 2 + 30 + 23 + 84 + 24 + 2 + 59

20/11

Resource Allocation Graph

- Banker's Algorithm

Example

	Allocation			Max			Available		
	A	B	C	A	B	C	A	B	C
× P_0	0	1	0	7	5	3	3	3	2
P_1	2	0	0	3	2	2			
× P_2	3	0	2	9	0	2			
P_3	2	1	1	2	2	2			
P_4	0	0	2	4	3	3			

Need = Max − Allocation

Need	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

Work = Work + Allocation

Work = [3 3 2]

For P_0 :

finish[0] = false
need $\nleq$ work

For P_1 :

need $\leq$ work
work = work + allocation
= [3 3 2] + [2 0 0]
= [5 3 2]
finish[1] = true.

P_2 :

need $\nleq$ work
finish[2] = false
[6 0 0] $\nleq$ [5 3 2]

P_3 :

~~work~~ =
[0 1 1] $\leq$ [5 3 2]
work = [5 3 2] + [2 1 1]
= [7 4 3]

For P_4 :

[4 3 1] $\leq$ [7 4 3]
work = [7 4 3] + [0 0 2]
= [7 4 5]

Safe Sequence
< P_1 P_3 P_4 P_0 P_2 >
or
< P_1 P_3 P_4 P_2 P_0 >
any one is correct