Assignment-2

1) S ← Binary Semaphore
↑
initialised value → 0

V → signal / up
P → wait / down

S | 0

operations → 5p, 7V, 10P

→ For 1st P s = 0, so block the process
2nd P, s=0, so block the process
Similarly for 5P block → 5 process

→ For next 7up / signal / V operation
5V ↳ 5 process unblock / more execution

For next
2V ↳ s = 1

→ For 10p
for First p → ass s = 1
make s=0 execute that process

For next 9P s = 0, so block the 9 process
So total 9 process blocked.

2)

T1
```
while (true) {
  wait(s3);
  printf("C");
  signal(s2);
}
```

T2
```
while (true) {
  wait(s1);
  printf("B");
  signal(s3);
}
```

T3
```
while (true) {
  wait(s2);
  printf("A");
  signal(s1);
}
```

Sequence → BCA BCA BCA
initialisation of semaphores is like $s_1 = 1, s_3 = 0, s_2 = 0$

3)
```
P1() {
  while (s1 == s2);
  // critical section
  s1 == s2;
}
```

```
P2() {
  while (s1 != s2);
  // critical selection
  s2 = ! s1;
}
```

S_1, S_2 are randomly assigned.

i) Mutual exclusion → It is satisfied because at any point of time suppose s1 ≠ s2 then P1() able to enter the critical section, so P1 in the CS. If P2 try to enter CS, as s1 ≠ s2 so busy wait. Similarly it is also applicable for p1 while p2 is in the CS.

Name:_____ Regd No.:_____

ii) Progress → Here progress is not satisfied suppose $s1 = 1$ and $s2 = 0$ ie. $s1 \neq s2$ if p1 is not interested to enter into the CS but P2 will not be able to enter as P1 have to enter CS and make $s1 = s2$, which is condition to enter CS for P2.

If one process is not interested in entering the critical section, it will not allow the other process to enter the critical section which is interested, so progress is not satisfied.

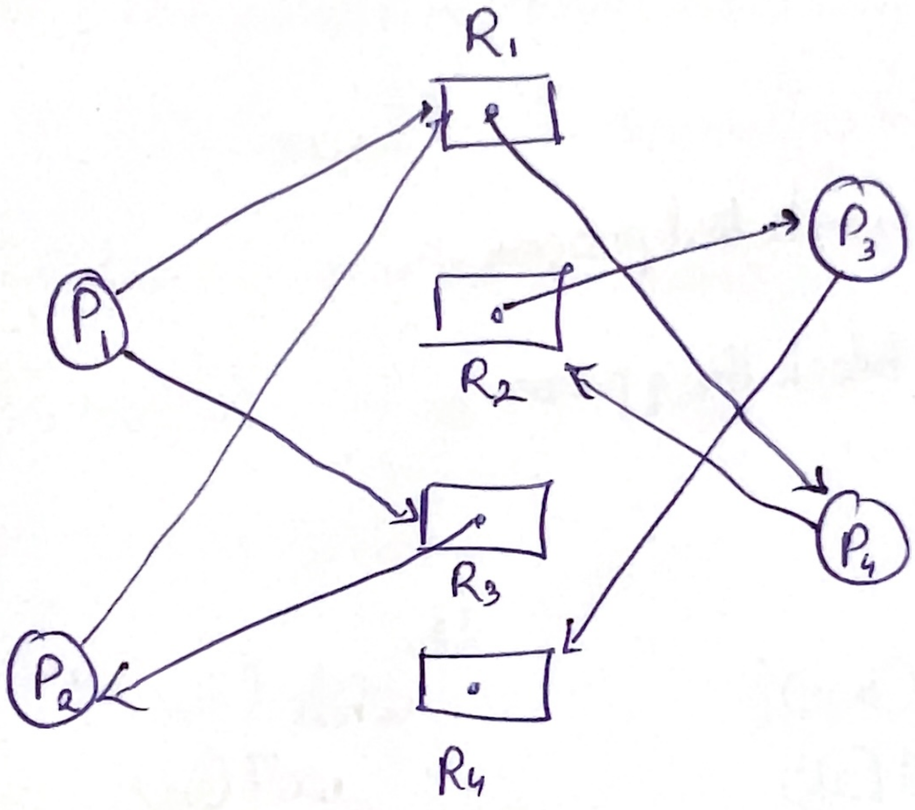
Hence, ans (a) Mutual exclusion but not progress.

Q4) 4 process P1, P2, P3, P4

4 Resource

R1, R2, R3, R4 with single instance

→



→ P1 wait for R1, which is held by P4 so P1 → P4 (P1 dependent on P4)

→ P1 also wait for R3, which is held by P2 so, P1 → P2

→ P2 also wait for R1, also held by P4 P2 → P4

→ P4 wait for R2, which is held by P3. So, P4 → P3

→ P3 waits for R4 which is free. So P3 is not dependent on other process. So when P3 complete its execution. P2 is free.

→ As R2 is free. P4 now able to run, after completing R1, R_2 is free now.

→ Both P1 & P2 requesting for R1 but there is only one instance as P2 already hold R3 if P2 runs first and complete its execution then R1, R3 is free.

→ P1 now able to complete its execution as R1 and R3 is available.

→ No circular wait as at a point of time R1 is free not hold by P1 which will make a circular wait.

→ So system is not in deadlock.

5) System with 12 tape drives, 3 process P_0, P_1, P_2

Requirements → P0 → 4, P1 → 10, P2 → 9

At time to P0 hold 2, P1 hold 5, P2 hold 2

Total available = 12 − (2+5+2)

= 3

As need of P0 is less than available.

So, P0 able to run then available = 5

not P1's need is 5

So, P1 able to run, then available = 10

Now P2's need is 7 ≤ 10

P2 also able to run

So the resource allocation state is safe and the safe sequence is P0 then P1 and last P2.

If P2 will request for 1 more instances:

Its available = 3, requesting for 1 and request < need of P2 i.e 7

So suppose granted immediately.

Then, available is no = 3 − 1 = 2

Now needs are like $P_1 \rightarrow 2$, $P1 \rightarrow 5$, $P2 \rightarrow 7-1=6$

Safety Check,

if P1's need ≤ available

So granted then after available is 4

but P1's need 5 and P2's need 6 which is greater than available so system enter into unsafe mode. So request from P2 for 1 more tape drive cannot be granted immediately

8) i) Available Resource

	X Y Z
	3 2 2

	Allocation			MAX			Need		
	X	Y	Z	X	Y	Z	X	Y	Z
P_0	0	0	1	7	4	3	7	4	2
	3	2	0	6	2	0	3	0	0
	2	1	1	9	3	3	1	9	2

Available

X Y Z

3 2 2

+ 3 2 0

6 4 2

+ 2 1 1

8 5 3

0 0 1

8 5 4

a) Content of need matrix =

	x	y	z
P_0	7	4	2
P_1	3	0	0
P_2	1	2	2

b) P1's need < available

So P_1

Then

now P_2's need < Available

So P_2

now P_0's need < Available

So P_0

Yes system is in a safe state.

Safe sequence is P_1, P_2, P_0

c) If P_0 will request for 2 more instances of type 2 as request < available and if allocated then allocation ≤ max need So if granted the table is like this. ~~available~~

	Allocation	MAX	Need
	x y z	x y z	x y z
P_0	0 0 3	7 4 3	7 4 0
P_1	3 2 0	6 2 0	3 0 0
P_2	2 1 1	3 3 3	1 2 2

Available

3 2 0

+ 3 2 0

6 4 0

As P_1's need ≤ available So P_1

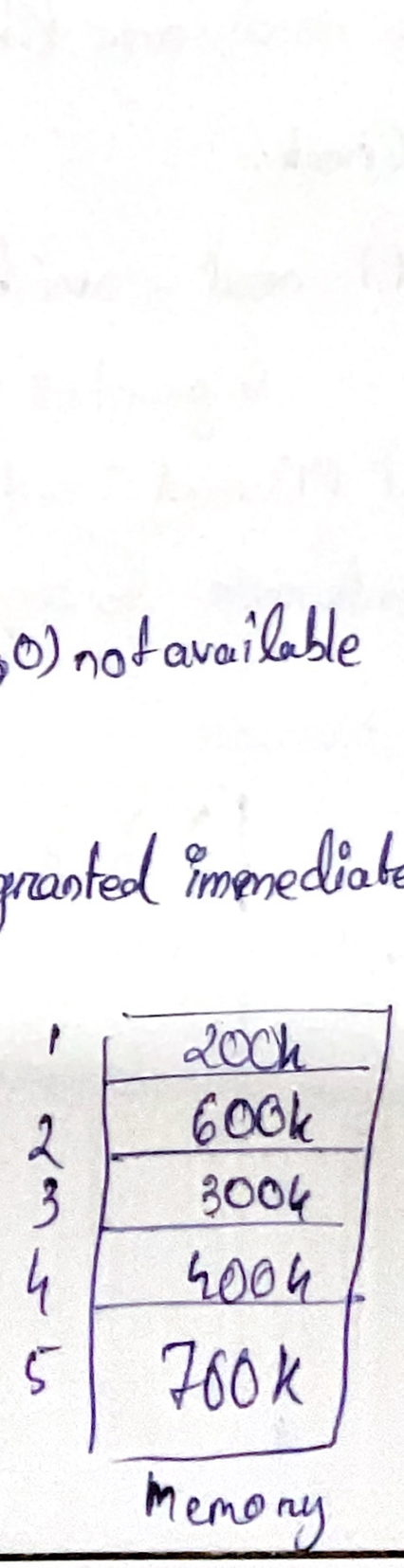
then P_2's need (1,2,2) but available (6,4,0) not available

P_0's need (7,4,0) but available (6,4,0)

So, system is in unsafe mode. So request can't be granted immediately

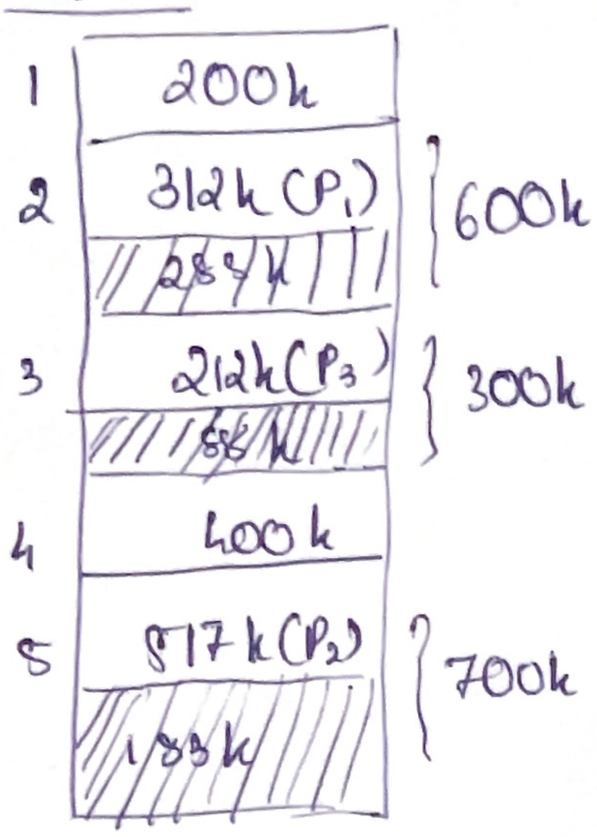
7) Process → 312k (P_1), 517k (P_2), 212k (P_3), 526k (P_4)

~~202k~~



Name:_____ Regd No.:_____

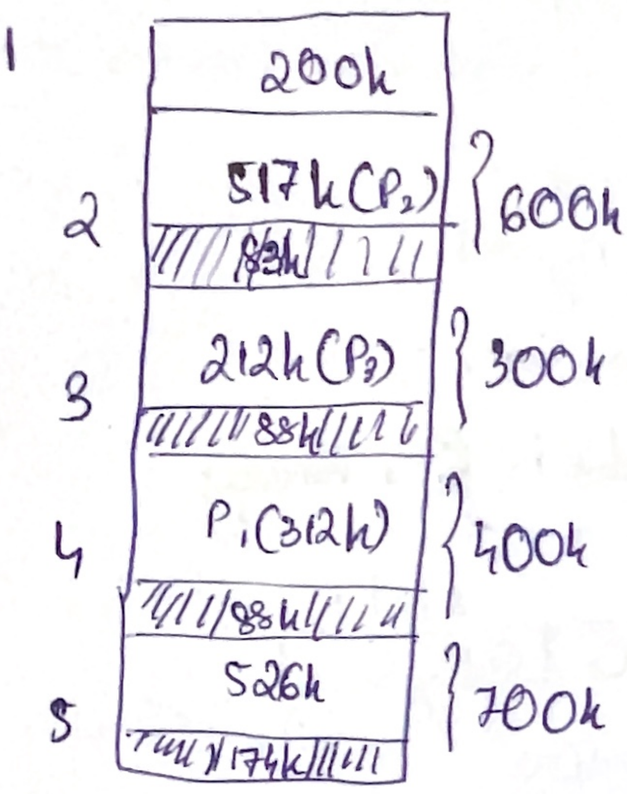
First Fit



→ Total Internal Fragmentation = 859k

P4 but total space is greater than process size

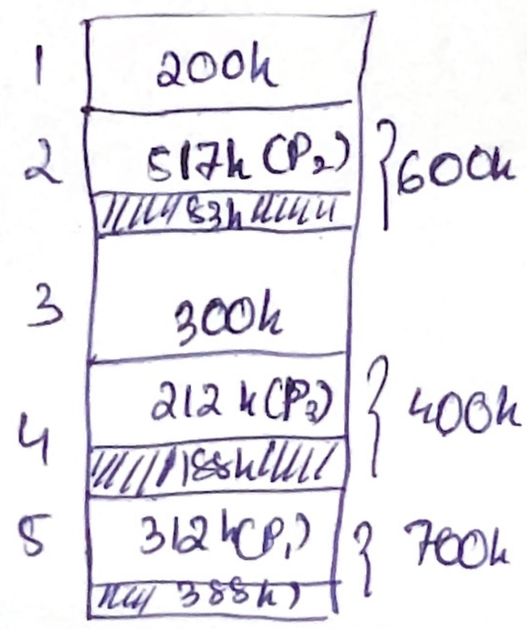
Best Fit →



Total IF = 83+88+88+174
= 433k

Most efficient use of memory → Best-Fit

Worst Fit



Total IF → 83+88+388 = 659k

P4 ✗

∴ Physical Memory = 2048 byte = 2^{11}

Logical Memory = 128 byte = 2^7

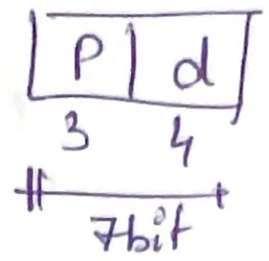
Page Size = 16 byte = 2^4

a) No. of bits required to represent logical address = 7bit.

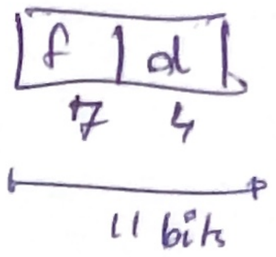
Name:_____ Regd No.:_____

b) No of bit required to represent physical address = 11 bit

No of page = $\frac{LAS}{ps} = \frac{2^7}{2^4} = 2^3 = 8$



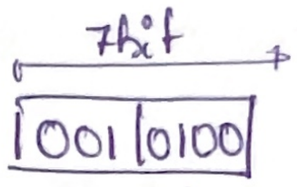
No. of frames = $\frac{PAS}{FS} = \frac{2^{11}}{2^4} = 2^7 = 128$



c) No. of entries in page table
↳ no. of pages = 8

d) Total no. of Frames = $2^7 = 128$.

e) As logical address → 20 in binary

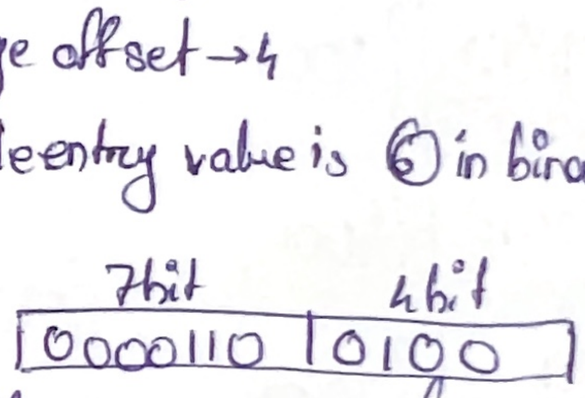


↳ in page table page no → 1

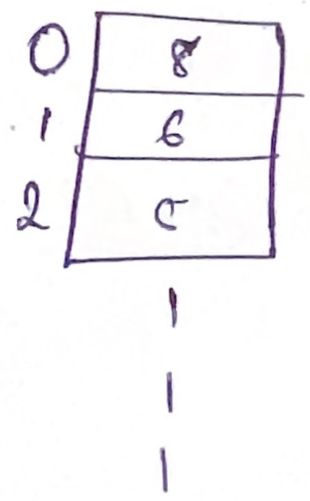
page offset → 4

As in the page table entry value is 6 in binary

so,



so, physical address = 100



Or Another method

page number = $\left\lfloor \frac{20}{16} \right\rfloor = 1 \rightarrow$ value 6

offset = 20 mod 16 = 4

PA = (Frame no x page size) + offset

= (6 x 16) + 4 = 100

g)

Segment	Base	Length
0	219	600
1	2300	100
2	90	110
3	1327	400
4	1950	50

a) 0, 430:
↑ ↳ 430 < 600. Yes so, PA = 219 + 430 = 649
Seg-0

b) 1, 10
seg1 ↳ 10 < 100; Yes so, 2300 + 10 = 2310 = PA

c) 2, 100 → 100 < 110. Yes so PA = 90 + 100 = 190
↑
Seg2

d) 2, 500 → 500 < 110 No, so TRAP Addressing Error
Seg2

Name:_____ Regd No.:_____

10) Ref string → (3; 8; 2; 3; 9; 1; 6; 3; 8; 9; 3; 6; 2; 1; 3)

$l = 15$ $n = 4$

FIFO

0	3	1	1	1	1	9	9	9	9					
1	8	8	6	6	6	2	2	2						
2	2	2	2	3	3	3	1	1						
3	9	9	9	9	8	8	8	3						

Total no of pagefault = 12 (8+4)

(optimal) <Replace that page which will no be used for the longest duration of time the future reference>

0	3	3	3	3	3									
1	8	8	8	8	8									
2	2	1	6	2	2									
3	9	9	9	9	1									

No. of page faults = 8

c) LRU (Least Recently used) → <Replace that page which has not been used for the longest duration of time in the past>

Ref string <3, 8, 2, 3, 9, 1, 6, 3, 8, 9, 3, 6, 2, 1, 3>

0	3	3	3	3	3	3	3							
1	8	1	1	1	9	9	1							
2	2	2	6	6	6	6	6							
3	9	9	9	8	8	2	2							

No. of page faults = 10 (6+4)

11) Total no. of frames → 50 , Total demand → 100

P1 needs 10
P2 needs 90

	Demand	Equal Allocation	Proportional Allocation ($\frac{\text{Demand}}{\text{Total frames}} \times$ frames)
P_1	10	25	$10/100 \times 50 = 5$
P_2	90	25	$90/100 \times 50 = 45$

a) P_1 get 25
P_2 — 25
Frames equal allocation

b) P_1 get 5
P_2 get — 45
in proportional allocation.

Name:_____ Regd No.:_____