1) Initial value of binary semaphore : S = 0

5P = -5 ; Thus S = -5

7V = +7 ; Thus S = -5+7 ⇒ 2

10P = -10; Thus S = 2-10 ⇒ S = -8

Thus, 8 processes are blocked.

2)

P1	P2	P3
S1 = 1	S2 = 0	S3 = 0
while (true){	while (true){	while (true){
semwait (S3);	semwait (S1);	semwait (S2);
printf ("C");	printf ("B");	printf ("A");
semsignal (S2);	semsignal (S3);	semsignal (S1);
}	}	}

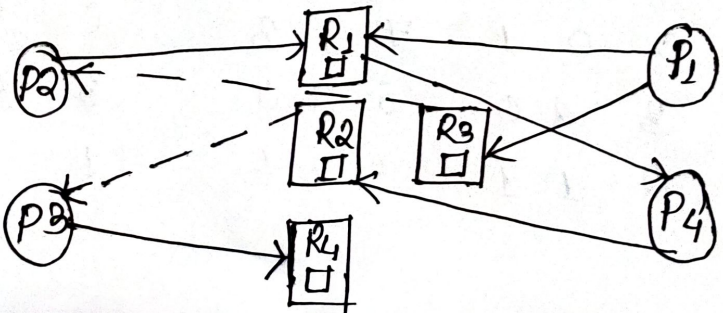
3) a) If S1 and S2 are different, P2 executes first but then, it can never go to P1() as after getting out of critical section, S2 != S1. Thus, P1 condition is never satisfied.

b) If S1 and S2 are different, P1() executes and can never go to P2() as S1 = S2 condition is fulfilled and P2() condition can never be satisfied.

∴ There is mutual exclusion but no progress.

4) 4 process : P1, P2, P3, P4

4 resources with 1 instances : R1, R2, R3, R4



P = {P1, P2, P3, P4} ; R = {R1, R2, R3, R4}

E = { P1 → R1 → P4 → R2 → R3 → R4,
P1 → R3 → P2 → R1 → P4 → R2 → P3 → P4 }

There is deadlock as each resource has only 1 instance.

The deadlock involves the process P1, P2, P4.

5) We use tape drives = 12

P0 : max = 4
allocated = 2 > Need = 4 - 2 = 2

P1 : max = 10
allocated = 5 > Need = 5

P2 : max = 9
allocated = 2 > Need = 7

Already allocated = 2 + 5 + 2 = 9

Free spaces = 12 - 9 = 3

Here, P0 : need < available. Thus, new available is 3 + 2 = 5, P1 : need = available, then P1 proceeds new available is 5 + 5 = 10, P3 : need < available. Thus executes new available = 10 + 2 = 12

6) a)

Process	Allocation			Max			Need		
	X	Y	Z	X	Y	Z	X	Y	Z
P0	0	0	1	7	4	3	0	0	2
P1	3	2	0	6	2	0	3	0	0
P2	2	1	1	3	3	3	1	2	2

b) Available (x, y, z) = (3, 2, 2)

P0 : Need (0, 0, 2) ≤ (3, 2, 2) ⇒ proceeds

New available = (3+0, 2+0, 2+1) = (3, 2, 3)

P1 : Need (3, 0, 0) ≤ (3, 2, 3)

New available = (3+3, 2+2, 3+0) = (6, 4, 3)

P2 : Need (1, 2, 2) ≤ (6, 4, 3)

New available = (8, 5, 4)

Safety sequence P0 → P1 → P2

c) If P0 : Need (0, 0, 4) ≤ (3, 2, 2) ⇒ False

can't be granted immediately.

7) First-fit :-

P1 (312 K) ⇒ 600K - 312K = 288K remains

P2 (517 K) ⇒ (700K - 517K) = 183 K remains

P3 (212 K) ⇒ (200 K - 212K) = not enough

P4 (526 K) ⇒ (183K - 526K) = not enough

Worst-fit :-

P1 = 700K - 312K = 388 K remain

P2 = 388 K - 517 K = ~~+8~~ not enough

P3 = 300K - 212 K = 88K remain

P4 = 600K - 526 K = 74 K remain

Best-fit :-

P1 = 700K - 31 K = 388K remain

P2 = 388K - 517 K = not enough

P3 = 288K - 212 K = 76 K

P4 = 76 K - 526 K = not enough

∴ ~~Bef~~ Best-fit is best

8) a) Logical memory = 128 bytes
Page size = 16 bytes
No. of pages = Logical memory / Page size
= 8
No. of bits needed to represent 8 is 3 bits

b) Physical memory = 2048 bytes
Page size = 16 bytes
No. of frames = 128 frame
No. of bits to represent 128 frames is 7 bits.

c) No. of pages = 8

d) No. of frames = 128

e) Physical address 20 falls within range of pages 2.
Corresponding frame number = 5
offset within page is 20 mod 16 = 4 (reminder)
Physical Address = (frame no × page size) + offset
= (5 × 16) + 4
= 84

9) Physical address = base + logical address

a) PA = 219 + 0 = 219

b) PA = 2300 + 10 = 2310

c) PA = 90 + 100 = 190

d) PA = 90 + 500 = 590

10) FIFO

Frame \ Request	3	8	2	3	9	1	6	3	8	9	3	6	2	1	3
5						1	1	1	1	1	1	1	1	1	1
4					9	9	9	9	9	9	9	9	2	2	2
3			2	2	2	2	2	2	8	8	8	8	8	8	8
2		8	8	8	8	8	8	3	3	3	3	3	3	3	3
1		3	3	3	3	3	6	6	6	6	6	6	6	6	6
Miss/ hit	M	M	M	H	M	M	M	M	H	H	H	M	H	H	

LRU

Frame \ Request	3	8	2	3	9	1	6	3	8	9	3	6	2	1	3
5						1	1	1	1	1	1	1	2	2	2
4					9	9	9	9	9	9	9	9	9	9	9
3			2	2	2	2	2	2	8	8	8	8	8	1	1
2		8	8	8	8	8	6	6	6	6	6	6	6	6	6
1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Miss/ hit	M	M	M	H	M	M	M	H	M	H	M	H	M	M	M

11) Total frame = 50
P1 needs 10 pages
P2 needs 90 pages

a) Equal scheme:-
frames / process = No. of frame allocated

$\Rightarrow$ $50/2 = \underline{25 \text{ frame}}$
both P1 & P2

b) Proportion of frames for P1 = $\left(\frac{10}{10+90}\right) * 5 = 5$ frame

P2 = $\left(\frac{90}{10+90}\right) * 5 = 45$ frame

Proportion of frame for Pi = $\frac{\text{Size of Pi}}{\text{Total size}} \times$ Total frame i)

12) a- FIFO :-

Average seek length ⇒ (85-120) + (240-85) + (164-240) + (275-164) + (150-275) + (360-150) + (225-360) + (140-225) + (330-140) + (45-330) = 1750/10

Seek time ⇒ 1750 * 4 ms = 7000 ms

b- SSTF : [Reorder based on shortest seek time]

Average seek length ⇒ (85-120) + (45-85) + (140-45) + (164-140) + (150-164) + (225-150) + (240-225) + (275-240) + (330-275) + (360-330) = ~~385~~ 112.5

Seek time ⇒ ~~385~~ 112.5 * 4 ms = ~~1540~~ 450 ms

c- SCAN: [Arrange requests based on the scanning direction]

Average seek length: (45-120) + (85-45) + (140-85) + (150-140) + (164-150) + (225-164) + (240-225) + (275-240) + (330-275) + (360-330) = ~~585~~ 43.5

Seek time ⇒ ~~585~~ 43.5 * 4 ms = ~~2340~~ 174 ms

d- CSCAN: Arrange requests based on the circular scanning direction.

Average seek length: (45 - 120) + (85 - 45) + (140 - 85) + (150 - 140) + (164 - 150) + (225 - 164) + (240 - 225) + (275 - 240) + (330 - 275) + (360 - 330) = ~~585~~ 43.5

Seek time = ~~585~~ 43.5 * 4 ms = ~~2340~~ 174 ms

e - LOOK: Arrange requests based on the scanning direction without going to end

Average seek length: (45 - 120) + (85 - 45) + (140 - 85) + (150 - 140) + (164 - 150) + (225 - 164) + (240 - 225) + (275 - 240) + (330 - 275) + (360 - 330) = ~~585~~ 43.5

Seek time = ~~585~~ 43.5 * 4 ms = ~~2340~~ 174 ms

f - CLOOK also has the same order, therefore

Average seek length = 43.5

Seek time = 174 ms