Name : Abubaker Attique

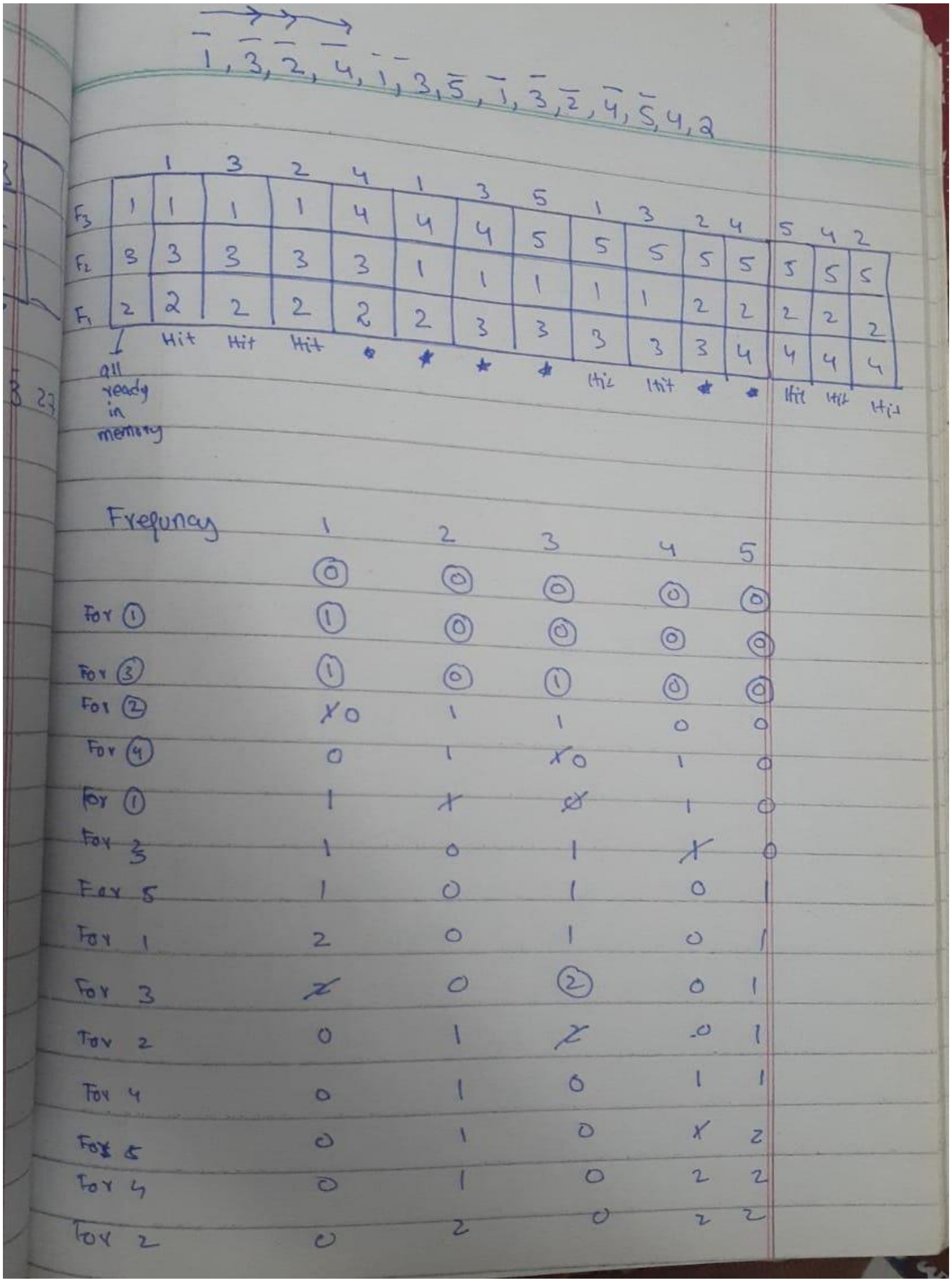
Roll : P20-0560

Section : 5-A

Assignment 2

Question : 1

If most frequently used page replacement algorithm is used, show how pages will be replaced upon each new page reference.

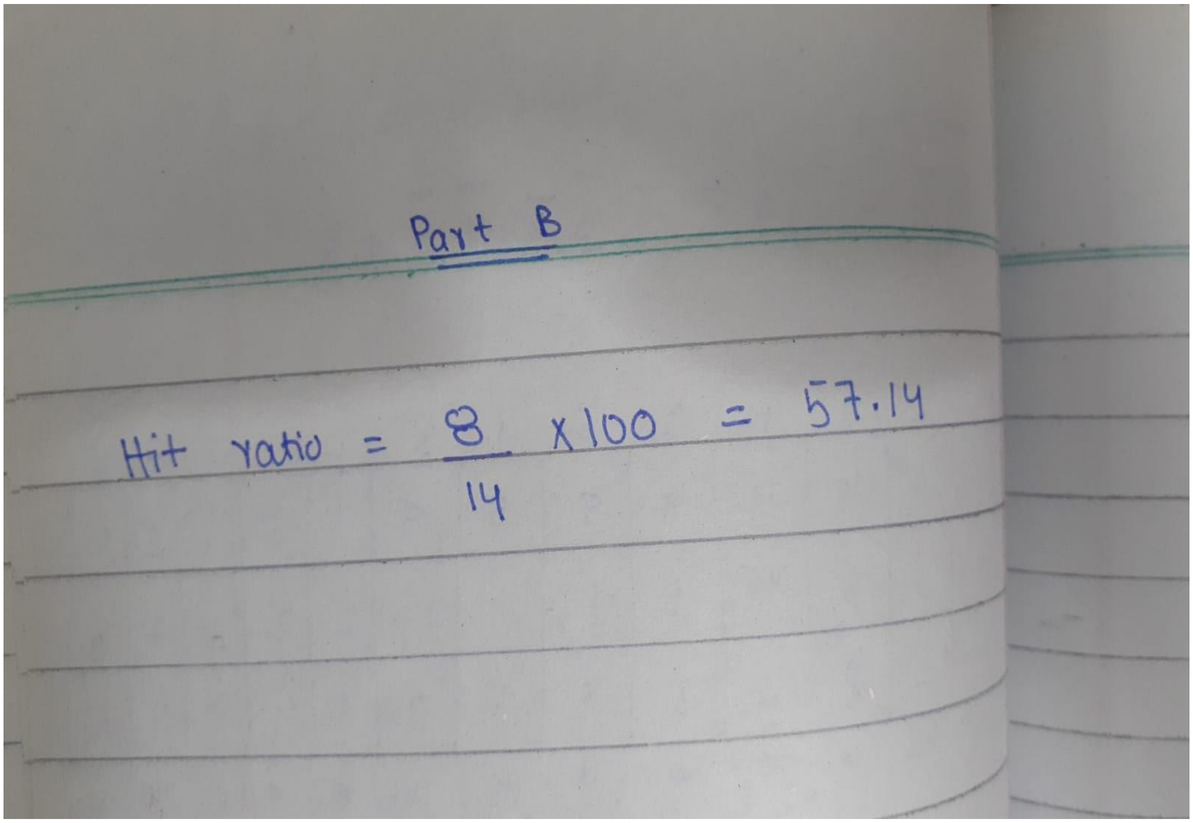


$\bar{1}, \bar{3}, \bar{2}, \bar{4}, \bar{1}, \bar{3}, \bar{5}, \bar{1}, \bar{3}, \bar{2}, \bar{4}, \bar{5}, 4, 2$

		1	3	2	4	1	3	5	1	3	2	4	5	4	2
F_3	1	1	1	1	4	4	4	5	5	5	5	5	5	5	5
F_2	3	3	3	3	3	1	1	1	1	1	2	2	2	2	2
F_1	2	2	2	2	2	2	3	3	3	3	3	4	4	4	4
	all ready in memory	Hit	Hit	Hit	*	*	*	*	Hit	Hit	*	*	Hit	Hit	Hit

Frequnay	1	2	3	4	5
	⓪	⓪	⓪	⓪	⓪
For ①	①	⓪	⓪	⓪	⓪
For ③	①	⓪	①	⓪	⓪
For ②	X0	1	1	0	0
For ④	0	1	X0	1	0
For ①	1	X	~~0~~	1	0
For 3	1	0	1	X	0
For 5	1	0	1	0	1
For 1	2	0	1	0	1
For 3	~~2~~	0	②	0	1
For 2	0	1	~~2~~	0	1
For 4	0	1	0	1	1
For 5	0	1	0	X	2
For 4	0	1	0	2	2
For 2	0	2	0	2	2

Part B:



Part B

Hit ratio = $\frac{8}{14} \times 100 = 57.14$

Question 2:

Part A

Will the concurrent execution of both of the above process result in an infinite wait?

Answer : Yes, the current execution of the both of the process result in an infinite wait, because there is deadlock in the about process. It can not be futher executed.

Part B

Make changes in the above code so the two processes never enter into an infinite waiting state without violating the mutually exclusive entry to the critical region.

Answer:

void P1 ( )	void P2 ( )
{	{
while (true)	while (true)
{	{
While(turn!=0)	While(turn!=1)
{	{
wait(S);	wait(W);
wait(W);	wait(S);
critical region;	critical region;
signal (S);	signal (W);
signal (W);	signal (S);
turn=1;	turn=0;
}	}
}	}

Explanation : In the above code every process will wait. From this you will not lead to deadlock.