EXPERIMENT NO: 9

AIM:- Write a program to implement various page replacement policies.

THEORY:-

In a operating systems that use paging for memory management, page replacement algorithm are needed to decide which page needed to be replaced when new page comes in. Whenever a new page is referred and not present in memory, page fault occurs and Operating System replaces one of the existing pages with newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce number of page faults.

Page Fault – A page fault is a type of interrupt, raised by the hardware when a running program accesses a memory page that is mapped into the virtual address space, but not loaded in physical memory.

Page Replacement Algorithms :

1. **First In First Out (FIFO) –**
   This is the simplest page replacement algorithm. In this algorithm, operating system keeps track of all pages in the memory in a queue, oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.

/*Program to implement FIFO page replacement algorithm*/

```
#include<stdio.h>
#include<conio.h>
main()
{
int i, j, k, f, pf=0, count=0, rs[25], m[10], n;
clrscr();
printf("\n Enter the length of reference string -- ");
scanf("%d",&n);
printf("\n Enter the reference string -- ");
for(i=0;i<n;i++)
scanf("%d",&rs[i]);
printf("\n Enter no. of frames -- ");
scanf("%d",&f);
for(i=0;i<f;i++)
m[i]=-1;
printf("\n The Page Replacement Process is -- \n");
for(i=0;i<n;i++)
{
for(k=0;k<f;k++)
```

```
{
if(m[k]==rs[i])
break;
}
if(k==f)
m[count++]=rs[i];
pf++;
}
for(j=0;j<f;j++)
printf("\t%d",m[j]);
if(k==f)
printf("\tPF No. %d",pf);
printf("\n");
if(count==f)
count=0;
}
printf("\n The number of Page Faults using FIFO are %d",pf);
getch();
}
```

OUTPUT:

```
user@user-H81M-S:~$ ./exp9-1

 Enter the length of reference string -- 5

 Enter the reference string -- 2
8
9
4
5

 Enter no. of frames -- 4

 The Page Replacement Process is --
        2       -1      -1      -1      PF No. 1
        2       8       -1      -1      PF No. 2
        2       8       9       -1      PF No. 3
        2       8       9       4       PF No. 4
        5       8       9       4       PF No. 5

 The number of Page Faults using FIFO are 5user@user
```

2. Least Recently Used (LRU)

In Least Recently Used (LRU) algorithm is a Greedy algorithm where the page to be replaced is least recently used. The idea is based on locality of reference, the least recently used page is not likely.

/*Program to implement LRU page replacement algorithm*/

```
#include<stdio.h>
#include<conio.h>
main()
{
int i, j , k, min, rs[25], m[10], count[10], flag[25], n, f, pf=0,
next=1; clrscr();
printf("Enter the length of reference string -- ");
scanf("%d",&n);
printf("Enter the reference string -- ");
for(i=0;i<n;i++)
{
scanf("%d",&rs[i]);
flag[i]=0;
}
printf("Enter the number of frames -- ");
scanf("%d",&f);
for(i=0;i<f;i++)
{
count[i]=0;
m[i]=-1;
}
printf("\nThe Page Replacement process is -- \n");
for(i=0;i<n;i++)
{
for(j=0;j<f;j++)
{
if(m[j]==rs[i])
{
flag[i]=1;
count[j]=next;
next++;
}
}
if(flag[i]==0)
{
if(i<f)
{
m[i]=rs[i]; count[i]=next;
next++;
}
else
min=0;
for(j=1;j<f;j++)
```

```
if(count[min] > count[j])
min=j;
m[min]=rs[i];
count[min]=next;
next++;
}
pf++;
}
for(j=0;j<f;j++)
printf("%d\t", m[j]);
if(flag[i]==0)
printf("PF No. -- %d" , pf);
printf("\n");
}
printf("\nThe number of page faults using LRU are %d",pf);
getch();
}
```

OUTPUT:

```
user@user-H81M-S:~$ ./exp9_2
Enter the length of reference string -- 5
Enter the reference string -- 5
7
8
2
3
Enter the number of frames -- 5

The Page Replacement process is --
5       -1      -1      -1      -1      PF No. -- 1
5       7       -1      -1      -1      PF No. -- 2
5       7       8       -1      -1      PF No. -- 3
5       7       8       2       -1      PF No. -- 4
5       7       8       2       3       PF No. -- 5

The number of page faults using LRU are 5user@user-H81M-S
```

CONCLUSION: -Thus we have studied and implemented page replacement algorithms.

SIGN AND REMARK

R1 (3 Marks)	R2 (3 Marks)	R3 (3 Marks)	R4 (3 Mark)	R5 (3 Mark)	Total (15 Marks)	Signature