OUTPUT:

```
mohamedinam@Mohamed-Inam-PC: ~
mohamedinam@Mohamed-Inam-PC:~$ clear

mohamedinam@Mohamed-Inam-PC:~$ gcc paging.c -o paging
mohamedinam@Mohamed-Inam-PC:~$ ./paging

MEMORY MANAGEMENT USING PAGING
Enter the Size of Physical memory: 16
Enter the size of Logical memory: 8
Enter the partition size: 2

The physical memory is divided into 8 no.of frames
The Logical memory is divided into 4 no.of pages
Enter the Frame number where page 0 must be placed: 5
Enter the Frame number where page 1 must be placed: 6
Enter the Frame number where page 2 must be placed: 7
Enter the Frame number where page 3 must be placed: 2

PAGE TABLE
PageAddress  FrameNo. PresenceBit
0               5               1
1               6               1
2               7               1
3               2               1

        FRAME TABLE
FrameAddress   PageNo

0               32555
1               32555
2               3
3               32555
4               32555
5               0
6               1
7               2
        Process to create the Physical Address
Enter the Base Address: 1000

Enter theLogical Address: 3

The Physical Address where the instruction present: 1013mohamedinam@Mohamed-Inam-PC:~
$
```

RESULT:

Thus the Memory management policy- Paging isimplemented successfully.

EX NO: 10	**PAGE REPLACEMENT ALGORITHMS**
DATE:	

A. FIFO PAGE REPLACEMENT ALGORITHM

AIM:

To implement page replacement algorithms FIFO (First In First Out).

ALGORITHM:

1: Create a queue to hold all pages in memory

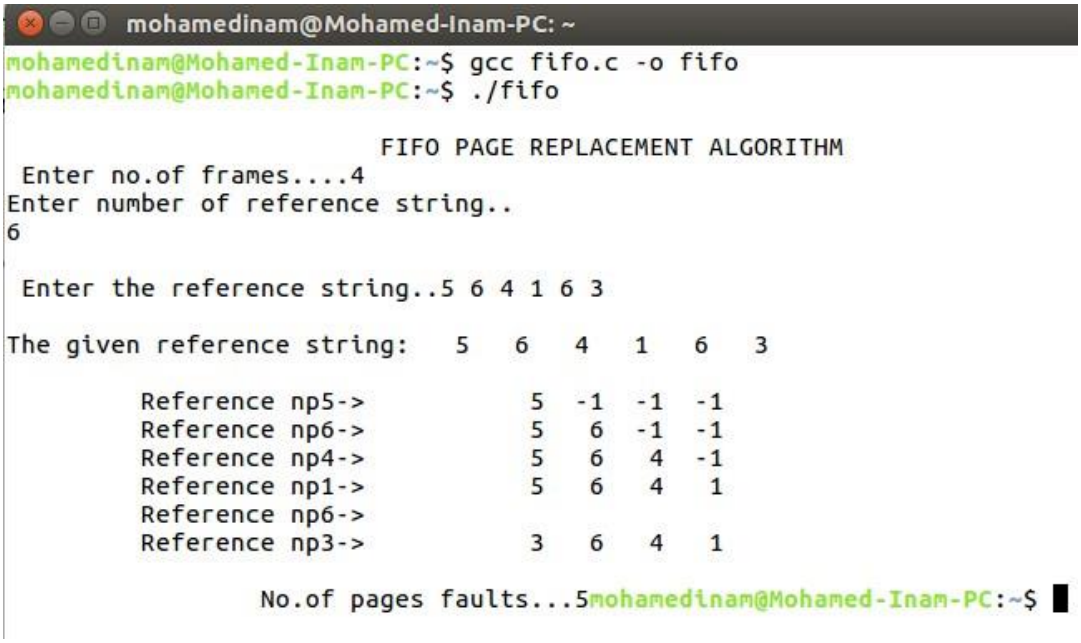
2: When the page is required replace the page at the head of the queue

3: Now the new page is inserted at the tail of the queue

PROGRAM:

```
#include<stdio.h>
int i,j,nof,nor,flag=0,ref[50],frm[50],pf=0,victim=-1; void
main()
{
printf("\n \t\t\t FIFO PAGE REPLACEMENT ALGORITHM");
printf("\n Enter no.of frames....");  scanf("%d",&nof);
printf("Enter number of reference string..\n");  scanf("%d",&nor);
printf("\n Enter the reference string..");
for(i=0;i<nor;i++)
scanf("%d",&ref[i]); printf("\nThe
given reference string:");
for(i=0;i<nor;i++)  printf("%4d",ref[i]);
for(i=1;i<=nof;i++)
 frm[i]=-1;
 printf("\n");
 for(i=0;i<nor;i++)
 {
   flag=0;
 printf("\n\t Reference np%d->\t",ref[i]);
   for(j=0;j<nof;j++)
   {
 if(frm[j]==ref[i])
 {
            flag=1;
             break;
 }}
     if(flag==0)
    {
  pf++;  victim++;
victim=victim%nof;
 frm[victim]=ref[i];
 for(j=0;j<nof;j++)
            printf("%4d",frm[j]);
   } }
printf("\n\n\t\t No.of pages faults...%d",pf);
}
```

OUTPUT:


mohamedinam@Mohamed-Inam-PC: ~
mohamedinam@Mohamed-Inam-PC:~$ gcc fifo.c -o fifo
mohamedinam@Mohamed-Inam-PC:~$ ./fifo
FIFO PAGE REPLACEMENT ALGORITHM
Enter no.of frames....4
Enter number of reference string..
6
Enter the reference string..5 6 4 1 6 3
The given reference string: 5 6 4 1 6 3
Reference np5-> 5 -1 -1 -1
Reference np6-> 5 6 -1 -1
Reference np4-> 5 6 4 -1
Reference np1-> 5 6 4 1
Reference np6->
Reference np3-> 3 6 4 1
No.of pages faults...5mohamedinam@Mohamed-Inam-PC:~$

RESULT:

Thus the FIFO page replacement algorithm is implemented successfully.