**Number of the Experiment :** 11

**Name of the Experiment :** Implementation of FIFO page replacement Algorithm using C program.

**Date of Performance :**

**Date of Submission :**

**Aim and Objectives :** To learn about FIFO page replacement algorithm and implement it with a C program.

**Theory : First In First Out (FIFO) \_**  
This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the 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.

**Source Code :**

#include<stdio.h>

int main()

{

int i,j,n,a[50],frame[10],no,k,avail,count=0;

printf("\n ENTER THE NUMBER OF PAGES:\n");

scanf("%d",&n);

printf("\n ENTER THE PAGE NUMBER :\n");

for(i=1; i<=n; i++)

scanf("%d",&a[i]);

printf("\n ENTER THE NUMBER OF FRAMES :");

scanf("%d",&no);

for(i=0; i<no; i++)

frame[i]= -1;

j=0;

printf("\tref string\t page frames\n");

for(i=1; i<=n; i++)

{

printf("%d\t\t",a[i]);

avail=0;

for(k=0; k<no; k++)

if(frame[k]==a[i])

avail=1;

if (avail==0)

{

frame[j]=a[i];

j=(j+1)%no;

count++;

for(k=0; k<no; k++)

printf("%d\t",frame[k]);

}

printf("\n");

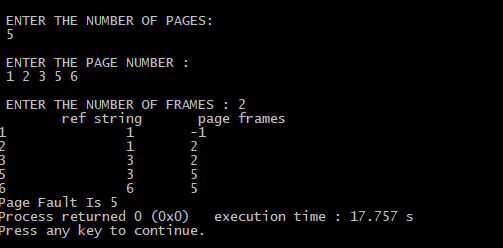
}

printf("Page Fault Is %d",count);

return 0;

}

**Output :**



**Discussion :** In 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.