Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 11

Course code: ICT- ICT-3110

Course title: Operating Systems Lab

Date of Performance:12-09-20

Date of Submission:27-09-20

Submitted by

Name: Nusrat Jahan Jui

ID:IT-18039

3rd year 1stsemester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Experiment no : 11

Experiment Name : Implementation of FIFO page replacement Algorithm.

Theory :

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. Page replacement happens when a requested page is not in memory (page fault) and a free page cannot be used to satisfy the allocation, either because there are none, or because the number of free pages is lower than some threshold.

Implementation :

1. Start the process

2. Declare the size with respect to page length

3. Check the need of replacement from the page to memory

4. Check the need of replacement from old page to new page in memory

5. Forma queue to hold all pages

6. Insert the page require memory into the queue

7. Check for bad replacement and page fault

8. Get the number of processes to be inserted

9. Display the values

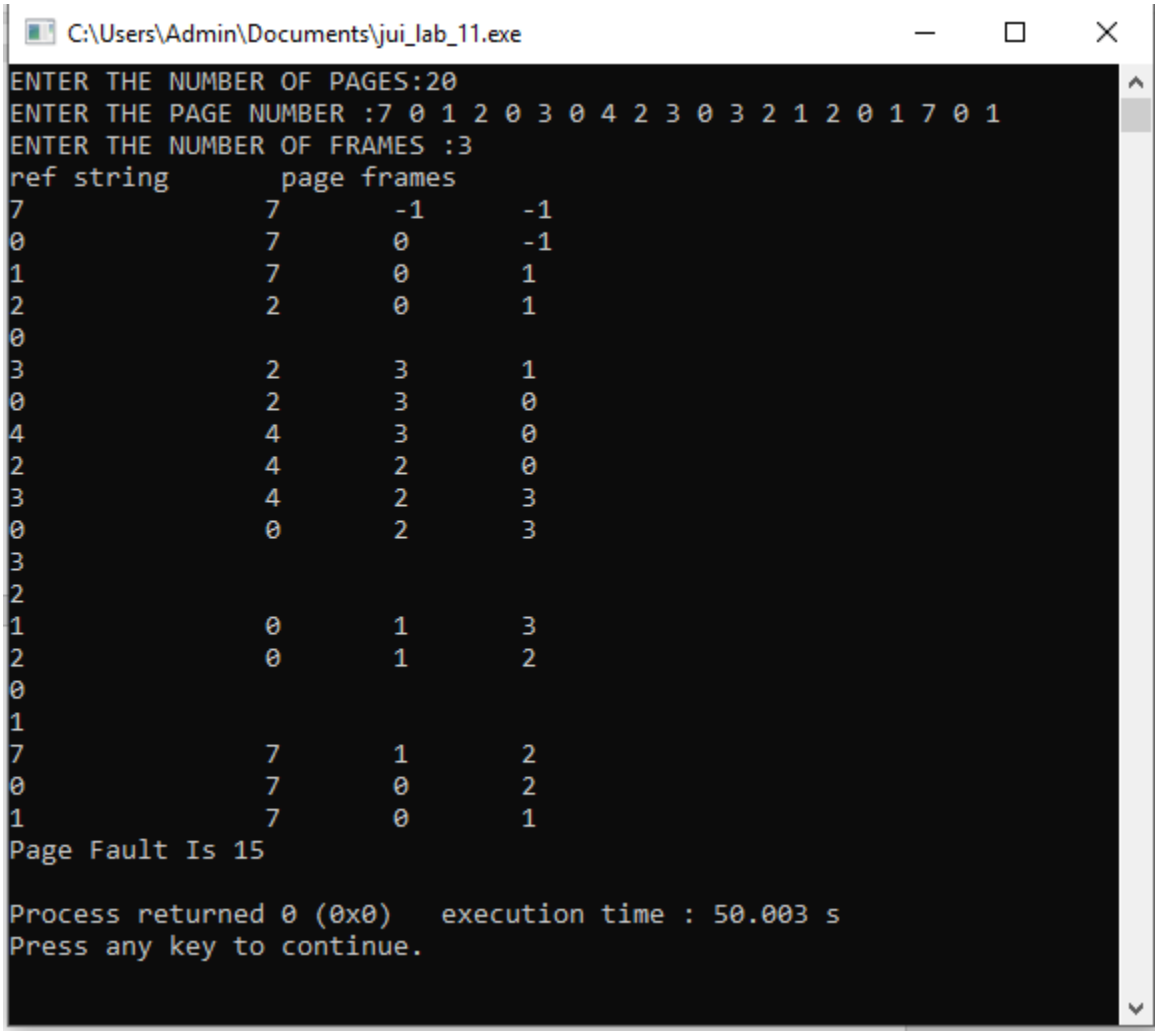
10. Stop the process

Working Process :

Code for FIFO page replacement Algorithm–

```
#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 :

The above algorithm has been implemented using C language. It is simple and easy to implement and understand. Initially all slots are empty, so when 7,0,1 came they are allocated to the empty slots->3page faults. Then 2 comes, it is not available in memory so it replaces the oldest page slot i.e 7->1 Page Fault. 0 is already in memory so 0 Page fault. Continuing the proccess we finally get page fault=15