

**Lab report no. : 11**

**Lab report name :** Lab report on implementation of FIFO page replacement algorithm.

**Aim and objectives:** To know details about FIFO page replacement algorithm, To know how to reduce the number of page faults by this algorithm, To know how to used for cost flow assumption purposes ,implement it with a c program, to learn how to use this algorithm.

**Explanation:**

- i) **FIFO page replacement algorithm:** First-in, First-out (FIFO) page replacement algorithm is a low-overhead algorithm that requires little bookkeeping on the part of the operating system. This algorithm queue is maintained. The page which is assigned the frame first will be replaced first or The page which resides at the rare end of the queue will be replaced on the every page fault. 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.
  
- ii) **Implementation of FIFO page replacement algorithm in C Program:**

**Code:**

```
#include<stdio.h>
int main()
{
    int ref_string[10], page_faults = 0, m, n, s, pages, frames;
    printf("\nEnter Total Number of Pages: ");
    scanf("%d", &pages);
    printf("\nEnter values of Reference String:\n");
    for(m = 0; m < pages; m++)
```

```

{
    printf("Enter Value for page [%d]: ", m + 1);
    scanf("%d", &ref_string[m]);
}
printf("\nEnter Total Number of Frames:\t");
{
    scanf("%d", &frames);
}
int temp[frames];
for(m = 0; m < frames; m++)
{
    temp[m] = 4-1;
}
for(m = 0; m < pages; m++)
{
    s = 0;
    for(n = 0; n < frames; n++)
    {
        if(ref_string[m] == temp[n])
        {
            s++;
            page_faults--;
        }
    }
    page_faults++;
    if((page_faults <= frames) && (s == 0))
    {
        temp[m] = ref_string[m];
    }
    else if(s == 0)
    {
        temp[(page_faults - 1) % frames] = ref_string[m];
    }
    printf("\n");
    for(n = 0; n < frames; n++)
    {
        printf("%d\t", temp[n]);
    }
}

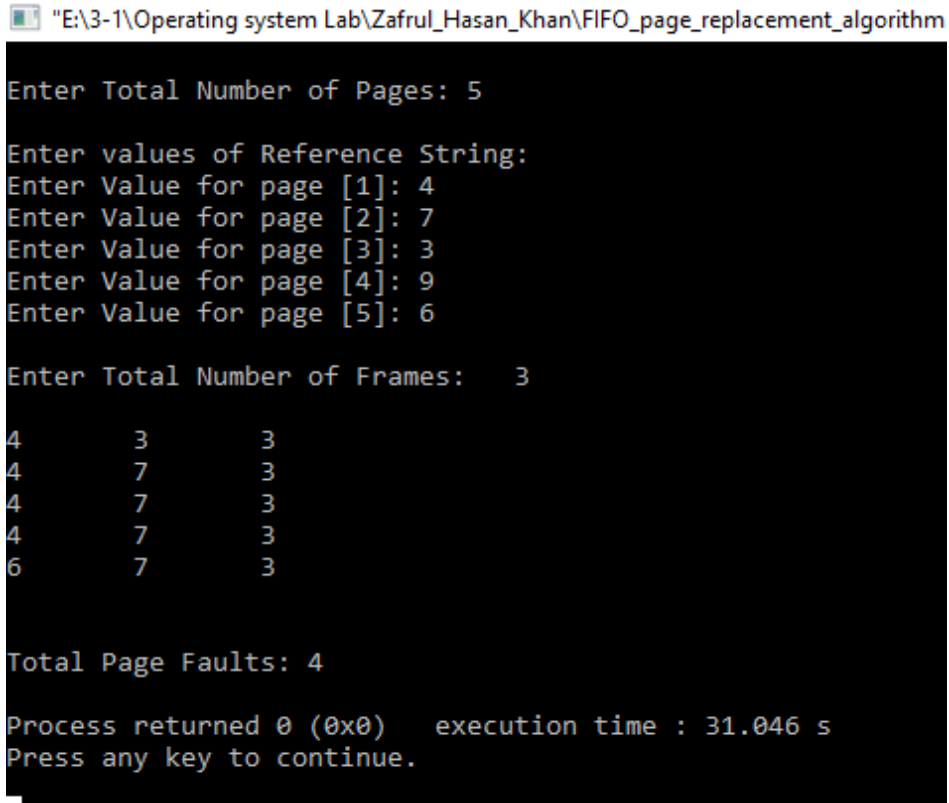
```

```

    }
    printf("\n\nTotal Page Faults: %d\n", page_faults);
    return 0;
}

```

### Output:



```

"E:\3-1\Operating system Lab\Zafrul_Hasan_Khan\FIFO_page_replacement_algorithm"
Enter Total Number of Pages: 5
Enter values of Reference String:
Enter Value for page [1]: 4
Enter Value for page [2]: 7
Enter Value for page [3]: 3
Enter Value for page [4]: 9
Enter Value for page [5]: 6
Enter Total Number of Frames: 3
4      3      3
4      7      3
4      7      3
4      7      3
6      7      3
Total Page Faults: 4
Process returned 0 (0x0)   execution time : 31.046 s
Press any key to continue.

```

**Conclusion:** From this lab , I have shown that FIFO page replacement algorithm is easy to implement. In this algorithm, the operating system keeps track of all pages in the memory in a queue. I also known that increasing the size of the memory increases the page fault rate in FIFO algorithm .