# Practical No 12

**Aim** : Develop, debug and Execute a C program to simulate LFU page replacement algorithms

**Apparatus:** Mingw compiler for C/C++, and a text editor for developing C code file (Dev C++).

**Theory** :

**What is Page Replacement Algorithm?**

- In operating systems that use paging for memory management, page replacement algorithms 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.

**What is LFU Page Replacement?**

- LFU is short for Least Frequently Used page replacement Algorithm.
- In LFU algorithm the least frequently used page is removed and replaced with the new page
- LFU is one such page replacement policy in which the least frequently used pages are replaced.
- If the frequency of pages is the same, then the page that has arrived first is replaced first.

**What are the rules for LFU Page Replacement?**

1. **The Page with the smallest/least frequency will be replaced.**
2. **Whenever a page comes/arrives the frequency is increased.**
3. **Whenever a page leaves the memory its frequency is reset.**
4. **If the frequency is same, FIFO is used to decide which page to replace.**

**Example:**

Reference string: 7,0,1,2,0,3,0,4,2,3,0,3,2

No of frames: 3

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F3 | | | 1 | 1 | 1 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 |
| F2 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F1 | 7 | 7 | 7 | 2 | 2 | 2 | 2 | 4 | 4 | 3 | 3 | 3 | 3 |
| | * | * | * | * | HIT | * | HIT | * | * | * | HIT | HIT | HIT |

Page fault (*): 8

Page hit (HIT): 5

Page fault ratio = No. of page fault / No. of reference string

= 8/13

= 61.54%

Page hit ratio = No. of page ratio / No. of reference string

= 5/13

= 38.46%

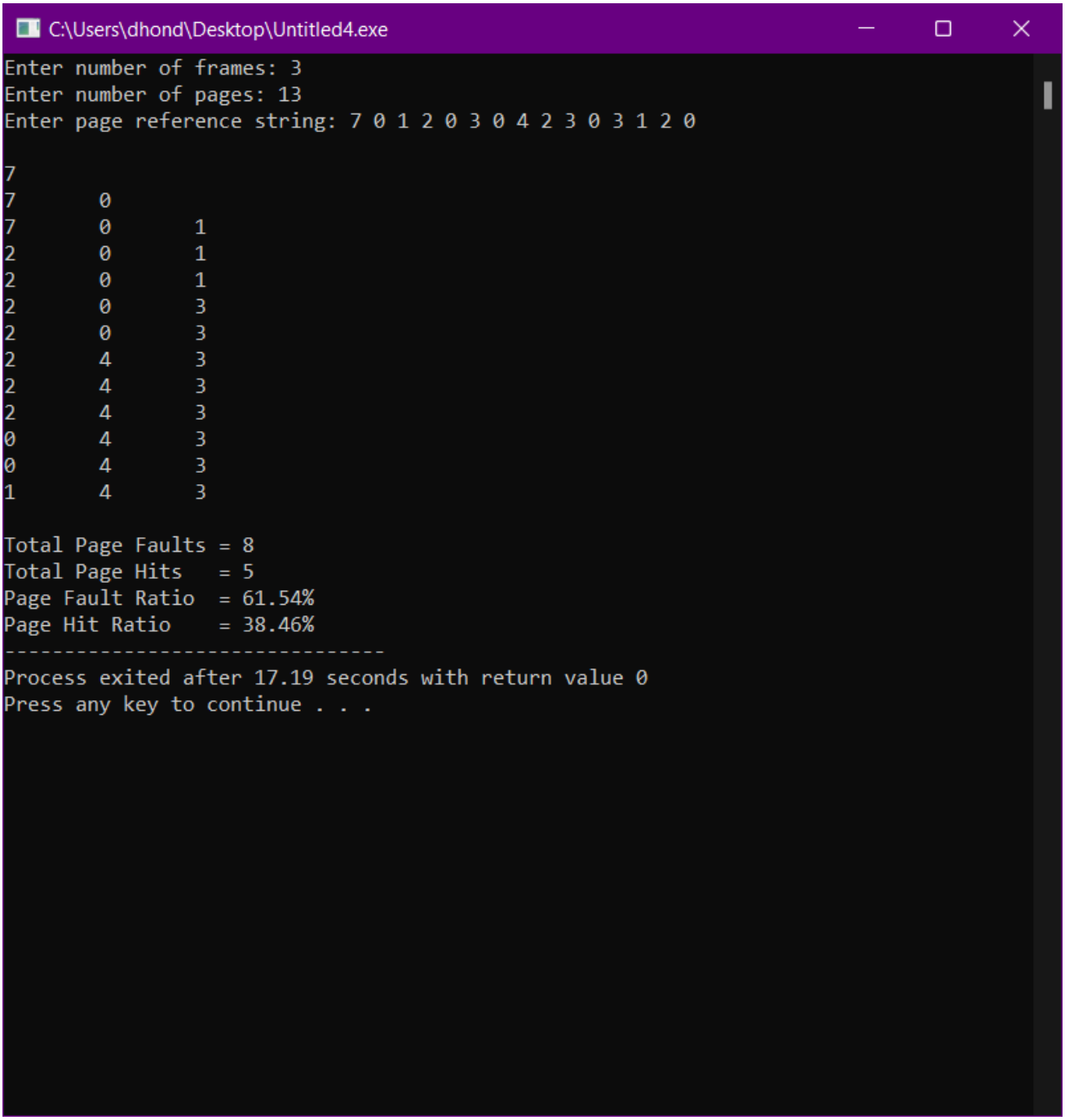
**Code:**

```
#include<stdio.h>

int main(){
    int no_of_frames, no_of_pages, frames[10], pages[30], temp[10], flag1, flag2,
flag3, i, j, k, pos, max, faults = 0;
      printf("Enter number of frames: ");
    scanf("%d", &no_of_frames);

    printf("Enter number of pages: ");
    scanf("%d", &no_of_pages);

    printf("Enter page reference string: ");

    for(i = 0; i < no_of_pages; ++i){
        scanf("%d", &pages[i]);
    }

    for(i = 0; i < no_of_frames; ++i){
        frames[i] = -1;
    }
```

```
for(i = 0; i < no_of_pages; ++i){
       flag1 = flag2 = 0;

       for(j = 0; j < no_of_frames; ++j){
           if(frames[j] == pages[i]){
                  flag1 = flag2 = 1;
                  break;
              }
       }

       if(flag1 == 0){
           for(j = 0; j < no_of_frames; ++j){
               if(frames[j] == -1){
                   faults++;
                   frames[j] = pages[i];
                   flag2 = 1;
                   break;
               }
           }
       }

       if(flag2 == 0){
               flag3 =0;

           for(j = 0; j < no_of_frames; ++j){
                   temp[j] = -1;

                   for(k = i + 1; k < no_of_pages; ++k){
                           if(frames[j] == pages[k]){
                                   temp[j] = k;
                                   break;
                           }
                   }
           }

           for(j = 0; j < no_of_frames; ++j){
                   if(temp[j] == -1){
                           pos = j;
                           flag3 = 1;
                           break;
                   }
           }

           if(flag3 ==0){
                   max = temp[0];
                   pos = 0;
```

```
            for(j = 1; j < no_of_frames; ++j){
                            if(temp[j] > max){
                                    max = temp[j];
                                    pos = j;
                            }
                    }
            }

                        frames[pos] = pages[i];
                        faults++;
        }

        printf("\n");

        for(j = 0; j < no_of_frames; ++j){
             if(frames[j] == -1)
             printf(" \t");
             else
            printf("%d\t", frames[j]);
        }
    }

    printf("\n\nTotal Page Faults = %d", faults);
      printf("\nTotal Page Hits   = %d",no_of_pages - faults);
      printf("\nPage Fault Ratio  = %.2f%%",faults/(float)no_of_pages * 100);
      printf("\nPage Hit Ratio    = %.2f%%",(no_of_pages -
faults)/(float)no_of_pages * 100);

    return 0;
}
```

**Output:**




Output 10.1

**Conclusion**:

Hence, by performing this practical I got to know about the LFU page replacement algorithm i.e. Least Frequently Used Page Replacement Algorithm. I also developed, debugged and executed a C program to simulate LFU page replacement algorithm.