Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

ASSIGNMENT 6

Title: Page Replacement Algorithms
Problem Statement: Implement the C program for Page Replacement Algorithms: FCFS, LRU, and Optimal for frame size as minimum three.

//fifo_page_rep

```
//C Program to Implement the FIFO(First In First Out) Page replacement Algorithm
//Time Complexity = O(n)
//Space Complexity= O(no of frames + size of Page Table)

#include<stdio.h>
#include<stdbool.h>
#include<string.h>

struct PageTable
{
   int frame_no;
   bool valid;
};

//Function to check if referenced/asked page is already present in frame[] or not
//Returns true if page is already present else returns false
bool isPagePresent(struct PageTable PT[],int page,int n)
{
   if(PT[page].valid == 1)
      return true;
   return false;
}

//Function to update the page table
//Return Nothing
void updatePageTable(struct PageTable PT[],int page,int fr_no,int status)
{
   PT[page].valid=status;
   //if(status == 1 )
      PT[page].frame_no=fr_no;
}

//Function to print the frame contents
//Return nothing
void printFrameContents(int frame[],int no_of_frames)
{
   for(int i=0;i<no_of_frames;i++)
```

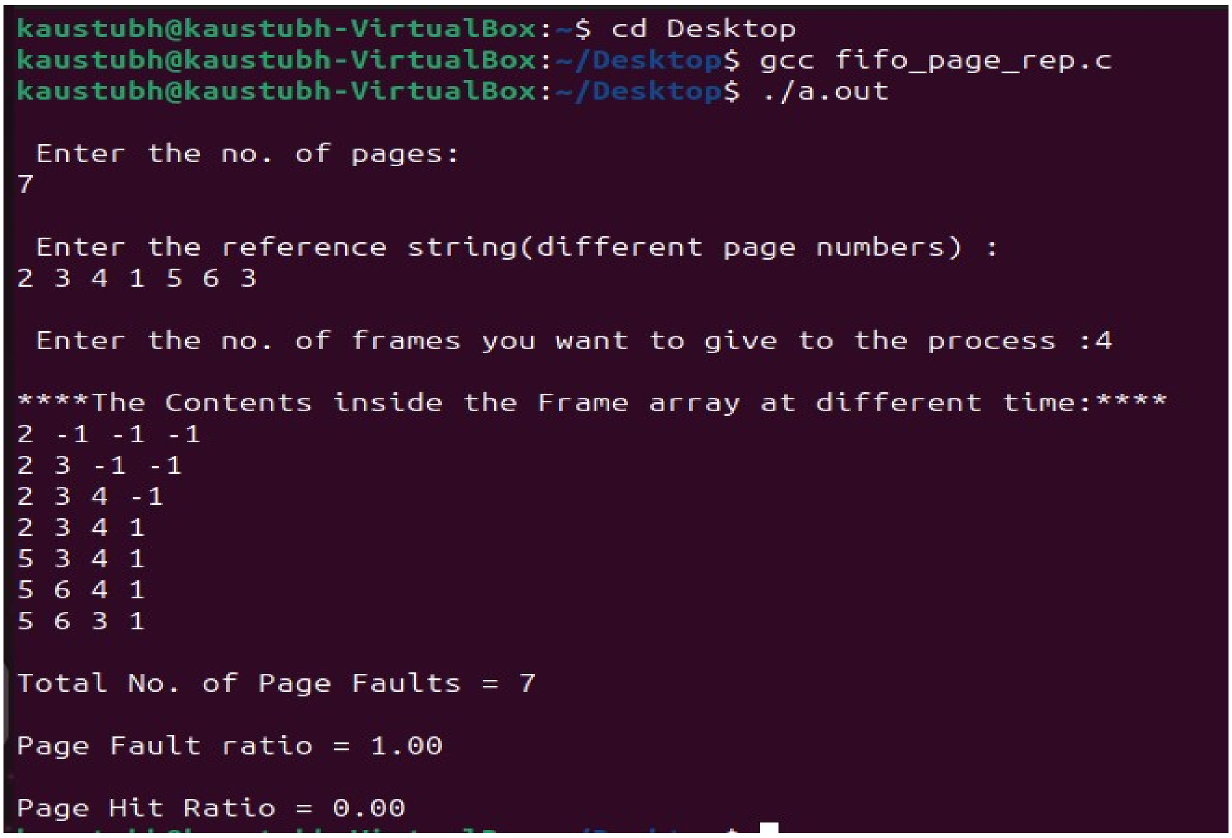
Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

```
    printf("%d ",frame[i]);
    printf("\n");
}

int main()
{
    int i,n,no_of_frames,page_fault=0,current=0;
    bool flag=false;
    printf("\n Enter the no. of pages:\n");
    scanf("%d",&n);
    //create reference string array
    int reference_string[n];
    printf("\n Enter the reference string(different page numbers) :\n");
    for(int i=0;i<n;i++)
     scanf("%d",&reference_string[i]);
    printf("\n Enter the no. of frames you want to give to the process :");
    scanf("%d",&no_of_frames);
    //create frame array to store the pages at different point of times
    int frame[no_of_frames];
    memset(frame,-1,no_of_frames*sizeof(int));
    struct PageTable PT[50] ; //asume page table can have entries for page 0 to 49
    for(int i=0;i<50;i++)
      PT[i].valid=0;

    printf("\n****The Contents inside the Frame array at different time:****\n");
    for(int i=0;i<n;i++)
    {
      //search the ith page in all allocated frames
      if( ! (isPagePresent(PT,reference_string[i],n)))
      {
         page_fault++;         // Increase the count of page fault
         if(flag==false && current < no_of_frames)
         {
                frame[current]=reference_string[i];
                printFrameContents(frame,no_of_frames);
                updatePageTable(PT,reference_string[i],current,1);
                current = current + 1;
                if(current == no_of_frames)
                {
                   current=0;
                   flag=true;  // so that we do not come to this if block again
                }

         }

         else //frame are full , APPLY FIFO
```

Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

```
  {
          //find the FIFO page (victim page) to replace;
          //The page pointed by current_head is FIFO page (victim page), so need to find it :)
          //mark that page as INVALID as in Page Table
          //set invalid frame no as -1 or anything ( as function needs this parameter),
             updatePageTable(PT,frame[current], -1 ,0);
             frame[current]=reference_string[i];
             printFrameContents(frame,no_of_frames);
             updatePageTable(PT,reference_string[i],current,1);
             current = ( current + 1)% no_of_frames;
        }
     } //end of outer if
    }   //end of for loop


  printf("\nTotal No. of Page Faults = %d\n",page_fault);
  printf("\nPage Fault ratio = %.2f\n",(float)page_fault/n);
  printf("\nPage Hit Ratio = %.2f\n",(float)(n- page_fault)/n);
  return 0;
}
```

Output:



```
kaustubh@kaustubh-VirtualBox:~$ cd Desktop
kaustubh@kaustubh-VirtualBox:~/Desktop$ gcc fifo_page_rep.c
kaustubh@kaustubh-VirtualBox:~/Desktop$ ./a.out

 Enter the no. of pages:
7

 Enter the reference string(different page numbers) :
2 3 4 1 5 6 3

 Enter the no. of frames you want to give to the process :4

****The Contents inside the Frame array at different time:****
2 -1 -1 -1
2 3 -1 -1
2 3 4 -1
2 3 4 1
5 3 4 1
5 6 4 1
5 6 3 1

Total No. of Page Faults = 7

Page Fault ratio = 1.00

Page Hit Ratio = 0.00
```

Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

//lru_page_rep

```
//C Program to Implement the LRU(Least Recently Used) Page replacement Algorithm

#include<stdio.h>
#include<stdbool.h>
#include<string.h>
#include<limits.h>

struct PageTable
{
    int frame_no;
    int last_time_of_access;
    bool valid;
};
//Function to check if referenced/asked page is already present in frame[] or not
//Returns true if page is already present else returns false
bool isPagePresent(struct PageTable PT[],int page)
{
    if(PT[page].valid == 1)
       return true;
    return false;
}

//Function to update the page table
//Return Nothing
void updatePageTable(struct PageTable PT[],int page,int fr_no,int status,int access_time)
{
    PT[page].valid=status;
    if(status == 1 )
    {
      PT[page].last_time_of_access =  access_time;
      PT[page].frame_no=fr_no;
    }
}

//Function to print the frame contents
//Return nothing
void printFrameContents(int frame[],int no_of_frames)
{
    for(int i=0;i<no_of_frames;i++)
      printf("%d ",frame[i]);
    printf("\n");
}

//Function to find the victim page index in frame[]
//Return that LRU page index using call by address
```

Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

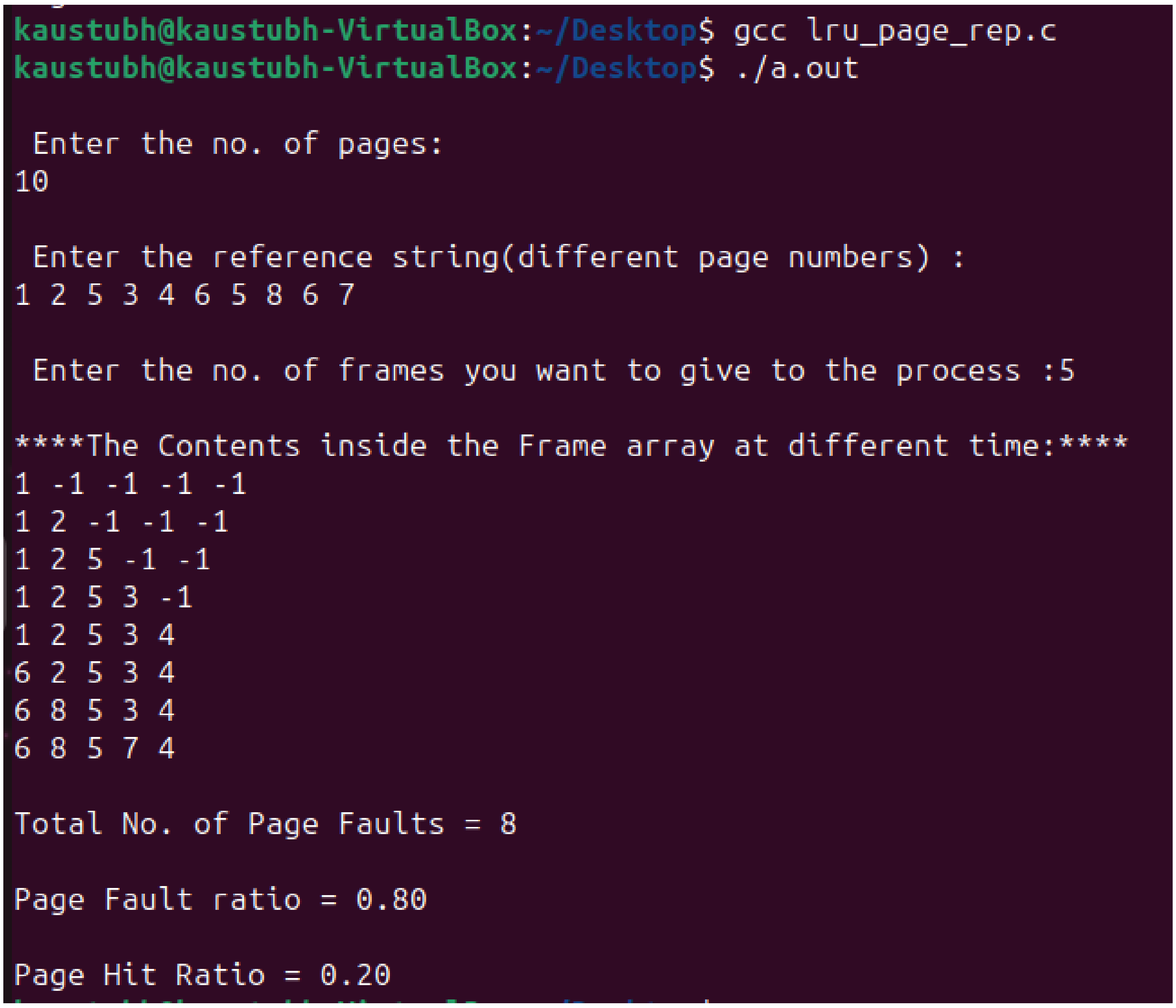
```
void searchLRUPage(struct PageTable PT[], int frame[], int no_of_frames, int *LRU_page_index)
{
    int min = INT_MAX;
    for(int i=0; i<no_of_frames;i++)
    {
       if(PT[frame[i]].last_time_of_access < min)
       {
           min = PT[frame[i]].last_time_of_access;
           *LRU_page_index = i;
       }
    }

}

int main()
{
    int i,n,no_of_frames,page_fault=0,current=0;
    bool flag=false;
    printf("\n Enter the no. of pages:\n");
    scanf("%d",&n);
    //create reference string array
    int reference_string[n];
    printf("\n Enter the reference string(different page numbers) :\n");
    for(int i=0;i<n;i++)
     scanf("%d",&reference_string[i]);
    printf("\n Enter the no. of frames you want to give to the process :");
    scanf("%d",&no_of_frames);
    //create frame array to store the pages at different point of times
    int frame[no_of_frames];
    memset(frame,-1,no_of_frames*sizeof(int));
    struct PageTable PT[50] ; //asume page table can have entries for page 0 to 49
    for(int i=0;i<50;i++)
      PT[i].valid=0;

    printf("\n****The Contents inside the Frame array at different time:****\n");
    for(int i=0;i<n;i++)
    {
      //search the ith page in all allocated frames
      if( ! (isPagePresent(PT,reference_string[i])))
      {
         page_fault++;      // Increase the count of page fault
         if(flag==false && current < no_of_frames)
         {

            frame[current]=reference_string[i];
            printFrameContents(frame,no_of_frames);
```

Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

```
            updatePageTable(PT,reference_string[i],current,1,i);

            current = current + 1;
            if(current == no_of_frames)
            {
               //current=0;
               flag=true;
            }

         }

         else //frame are full , APPLY LRU Algo
         {
            //search the LRU page( victim page) with the help of PT
            //mark that page as INVALID in Page Table
            int LRU_page_index;
            searchLRUPage(PT,frame,no_of_frames,&LRU_page_index);
            updatePageTable(PT,frame[LRU_page_index], -1 ,0,i);  //send invalid frame_no =-1

            frame[LRU_page_index]=reference_string[i];
            printFrameContents(frame,no_of_frames);
            //Update PT
            updatePageTable(PT,reference_string[i],LRU_page_index,1,i);
         }
      }
      //Update the Page Access time for reference_string[i]
      PT[reference_string[i]].last_time_of_access =  i;
    }  //end of for loop


   printf("\nTotal No. of Page Faults = %d\n",page_fault);
   printf("\nPage Fault ratio = %.2f\n",(float)page_fault/n);
   printf("\nPage Hit Ratio = %.2f\n",(float)(n- page_fault)/n);
   return 0;
}
```

Name: Kaustubh Gajanan Indulkar
Roll No: 35036 (25027)
Batch: B

Output:



```
kaustubh@kaustubh-VirtualBox:~/Desktop$ gcc lru_page_rep.c
kaustubh@kaustubh-VirtualBox:~/Desktop$ ./a.out

 Enter the no. of pages:
10

 Enter the reference string(different page numbers) :
1 2 5 3 4 6 5 8 6 7

 Enter the no. of frames you want to give to the process :5

****The Contents inside the Frame array at different time:****
1 -1 -1 -1 -1
1 2 -1 -1 -1
1 2 5 -1 -1
1 2 5 3 -1
1 2 5 3 4
6 2 5 3 4
6 8 5 3 4
6 8 5 7 4

Total No. of Page Faults = 8

Page Fault ratio = 0.80

Page Hit Ratio = 0.20
```