Dated: 06/07/2021

Experiment No. – 08 (Page Replacement Algorithms)

Objective:

Consider the following page reference string:
7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0, 1.

Assuming demand paging with three frames, how many page faults would occur for the following replacement algorithms?

- LRU replacement
- FIFO replacement
- Optimal replacement

Programs:

```
#include<stdio.h>

int minimum(int arr[], int n)
{
   int i, m = 0;
   for(i=0; i<n; i++)
   {
      if(arr[i] < arr[m])
         m = i;
   }
   return m;
}

int maximum(int arr[], int n)
{
   int i, m = 0;
   for(i=0; i<n; i++)
   {
      if(arr[i] > arr[m])
         m = i;
   }
   return m;
}

void FIFO(int str[], int frames_no, int total_pages)
{
   int i, j, flag, last_in = -1, page_fault = 0, frames[frames_no];
   for(i=0; i<total_pages; i++)
   {
      flag = 0;
      for(j=0; j<frames_no; j++)
      {
         if(str[i]==frames[j])
            flag = 1;
      }
      if(flag == 0)
```

```
        {
            last_in = (last_in+1)%frames_no;
            frames[last_in] = str[i];
            page_fault++;
        }
    }
    printf("*****************************(FIFO)*****************************\n");
    printf("Final Frames :\n");
    for(i=0; i<frames_no; i++)
        printf("%d\t",frames[i]);

    printf("No of Page faults : %d\n",page_fault);
}

void LRU(int str[], int frames_no, int total_pages)
{
    int i, frames[frames_no], page_faults = 0, full = 0, age[frames_no], j, flag, leastRecent;
    for(i=0; i<total_pages; i++)
    {
        if(full < frames_no)
        {
            frames[full] = str[i];
            age[full] = i;
            full++;
            page_faults++;
        }
        else
        {
            flag = 0;
            for(j=0; j<frames_no; j++)
            {
                if(str[i]==frames[j])
                    {
                        flag = 1;
                        age[j] = i;
                    }
            }
            if(flag == 0)
            {
                leastRecent = minimum(age, frames_no);
                frames[leastRecent] = str[i];
                age[leastRecent] = i;
                page_faults++;
            }
        }
    }
    printf("\n*****************************(LRU)*****************************\n");
    printf("Final Frames :\n");
    for(i=0; i<frames_no; i++)
        printf("%d\t",frames[i]);
    printf("No of Page faults : %d\n",page_faults);
}
```

```
void Optimal(int str[], int frames_no, int total_pages)
{
   int i, frames[frames_no], page_faults = 0, full = 0, j, flag, freq[frames_no], optimal, k;
   for(i=0; i<total_pages; i++)
   {
      if(full < frames_no)
      {
         frames[full] = str[i];
         full++;
         page_faults++;
      }
      else
      {
         flag = 0;
         for(j=0; j<frames_no; j++)
         {
            if(str[i]==frames[j])
               {
                  flag = 1;
               }
         }
         if(flag == 0)
         {
            for(j=0; j<frames_no; j++)
            {
               int f = 0;
               for(k=i; k<total_pages; k++)
               {
                  if(str[k] == frames[j])
                     {
                        freq[j] = k;
                        f = 1;
                        break;
                     }
               }
               if(f==0)
                  freq[j] = 100;
            }
            optimal = maximum(freq, frames_no);
            frames[optimal] = str[i];
            page_faults++;
         }
      }
   }
   printf("\n*****************************(Optimal)*****************************\n");
   printf("Final Frames :\n");
   for(i=0; i<frames_no; i++)
      printf("%d\t",frames[i]);

   printf("No of Page faults : %d\n",page_faults);
}
```

```
int main(void)
{
   int page[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1};
   int total_pages = 20;
   int frames_no = 3;
   FIFO(page, frames_no, total_pages);
   LRU(page, frames_no, total_pages);
   Optimal(page, frames_no, total_pages);
}
```

Output:

```
*************************(FIFO)*************************
Final Frames :
7       1       0       No of Page faults : 14

*************************(LRU)**************************
Final Frames :
1       0       7       No of Page faults : 12

************************(Optimal)************************
Final Frames :
7       0       1       No of Page faults : 9

Process returned 0 (0x0)   execution time : 0.025 s
Press any key to continue.
```