

# LAB REPORT

**NAME: ANURAN CHAKRABORTY**

**ROLL NO.: 001610501020 UG-III SECTION: A1**

## **PROBLEM STATEMENT**

Write a program to simulate Least Recently Used (LRU) page replacement algorithm for the following page reference string: 9, 10, 11, 7, 12, 8, 7, 6, 12, 5, 4, 3, 10, 11, 12, 4, 5, 6, 9, 4, 5.

Consider (i) 4 frames and (ii) 5 frames. Compare the results.

## **SOLUTION**

For the each memory reference *struct mem\_elem* has been defined which stores the page number of the memory element. Then an array of memory elements have been defined denoting the maximum number of frames and a *count* to store the number of pages currently in the frame.

```
//struct to maintain page number and priority
typedef struct mem_elem
{
    int pageno;
} mem;
mem arr[5];
int count=0;
```

- **void insert(mem ele, int pos, int max\_frames)**

This function takes a memory element *ele* and inserts it in the frame if it does not exist else it reorders the pages in the frames. Here we check if the number of pages already in memory is the maximum number of frames. If so, then LRU page replacement policy is followed and if the new page is not in memory then the least recently used page is replaced with the new page. If the frames are not all occupied then the page is just inserted at the end.

```
// Function to insert a new page in the array
void insert(mem ele, int pos, int max_frames)
{
    int i;
    if(count<max_frames)
    {
        arr[count++]=ele;
    }
    else
    {
        if(pos==-1)
            pos=0;
```

```

        for(i=pos;i<count-1;i++)
            arr[i]=arr[i+1];
        arr[count-1]=ele;
    }
}

```

- **int exists(int page\_no)**

This function checks if a page is already in memory. If the page is in memory it returns its position else it returns -1.

```

//Function to check if page exists
int exists(int page_no)
{
    int i;
    for(i=0;i<count;i++)
    {
        if(page_no==arr[i].pageno)
            return i;
    }
    return -1;
}

```

- **void printarr()**

This function prints the pages in the frames.

```

//Function to print the array
void printArr()
{
    int i=0;
    for(;i<count;i++)
        printf("%d ",arr[i].pageno);
    printf("\n");
}

```

- **int main()**

The main driver function to call the above functions. It consists of an array of pages referenced and the above functions are called with different parameters for max\_frames.

```

int main()
{
    //Define array of pages
    int parr[]={9, 10, 11, 7, 12, 8, 7, 6, 12, 5, 4, 3, 10, 11, 12, 4, 5,
6, 9, 4, 5};
}

```

```

int i,j;

for(j=4;j<=5;j++)
{
    pgfaults=0;
    printf("=====\n");
    count=0;
    printf("For %d frames\n",j);
    for(i=0;i<21;i++)
    {
        int pos=exists(parr[i]);
        mem temp={parr[i]};
        printf("Page %d accessed \t",parr[i]);
        insert(temp,pos,j);
        printArr();
    }
    printf("=====\n");
    printf("Page faults %d\n",pgfaults);
}

return 0;
}

```

```

/media/anuran/WORK/JUCSE WORK/3rd Year 1st Sem/Operating Systems/Assignment3 [anuran]
anuran:/media/anuran/WORK/JUCSE WORK/3rd Year 1st Sem/Operating Systems/Assignment3 (master M) $ gcc q3.c
anuran:/media/anuran/WORK/JUCSE WORK/3rd Year 1st Sem/Operating Systems/Assignment3 (master M) $ ./a.out
=====
For 4 frames
Page 9 accessed      9
Page 10 accessed     9 10
Page 11 accessed     9 10 11
Page 7 accessed      9 10 11 7
Page 12 accessed     10 11 7 12
Page 8 accessed      11 7 12 8
Page 7 accessed      11 12 8 7
Page 6 accessed      12 8 7 6
Page 12 accessed     8 7 6 12
Page 5 accessed      7 6 12 5
Page 4 accessed      6 12 5 4
Page 3 accessed      12 5 4 3
Page 10 accessed     5 4 3 10
Page 11 accessed     4 3 10 11
Page 12 accessed     3 10 11 12
Page 4 accessed      10 11 12 4
Page 5 accessed      11 12 4 5
Page 6 accessed      12 4 5 6
Page 9 accessed      4 5 6 9
Page 4 accessed      5 6 9 4
Page 5 accessed      6 9 4 5
=====
Page faults 17
=====
For 5 frames
Page 9 accessed      9
Page 10 accessed     9 10
Page 11 accessed     9 10 11
Page 7 accessed      9 10 11 7
Page 12 accessed     10 11 7 12
Page 8 accessed      10 11 7 12 8
Page 7 accessed      10 11 12 8 7
Page 6 accessed      11 12 8 7 6
Page 12 accessed     11 8 7 6 12
Page 5 accessed      8 7 6 12 5
Page 4 accessed      7 6 12 5 4
Page 3 accessed      6 12 5 4 3
Page 10 accessed     12 5 4 3 10
Page 11 accessed     5 4 3 10 11
Page 12 accessed     4 3 10 11 12
Page 4 accessed      3 10 11 12 4
Page 5 accessed      10 11 12 4 5
Page 6 accessed      11 12 4 5 6
Page 9 accessed      12 4 5 6 9
Page 4 accessed      12 5 6 9 4
Page 5 accessed      12 6 9 4 5
=====
Page faults 16
anuran:/media/anuran/WORK/JUCSE WORK/3rd Year 1st Sem/Operating Systems/Assignment3 (master M) $

```