**LAB – 6**

**AIM – Page Replacement algorithm (FIFO, LRU)**

SHRUTI MISHRA

21BCP110

**FIFO** – First In, First Out

**CODE:**

#include <stdio.h>

*int* main()

{

*int* referenceString[10], pageFaults = 0, m, n, s, pages, frames;

printf("\nEnter the number of Pages: ");

scanf("%d", &pages);

printf("\nEnter reference string values:\n");

for( m = 0; m < pages; m++)

{

   printf("Value No. [%d]:\t", m + 1);

   scanf("%d", &referenceString[m]);

}

printf("\n What are the total number of frames: ");

{

   scanf("%d", &frames);

}

*int* temp[frames];

for(m = 0; m < frames; m++)

{

  temp[m] = -1;

}

for(m = 0; m < pages; m++)

{

  s = 0;

  for(n = 0; n < frames; n++)

   {

      if(referenceString[m] == temp[n])

         {

            s++;

            pageFaults--;

         }

   }

   pageFaults++;

   if((pageFaults <= frames) && (s == 0))

      {

        temp[m] = referenceString[m];

      }

   else if(s == 0)

      {

        temp[(pageFaults - 1) % frames] = referenceString[m];

      }

      printf("\n");

      for(n = 0; n < frames; n++)

       {

         printf("%d\t", temp[n]);

       }

}

printf("\n");

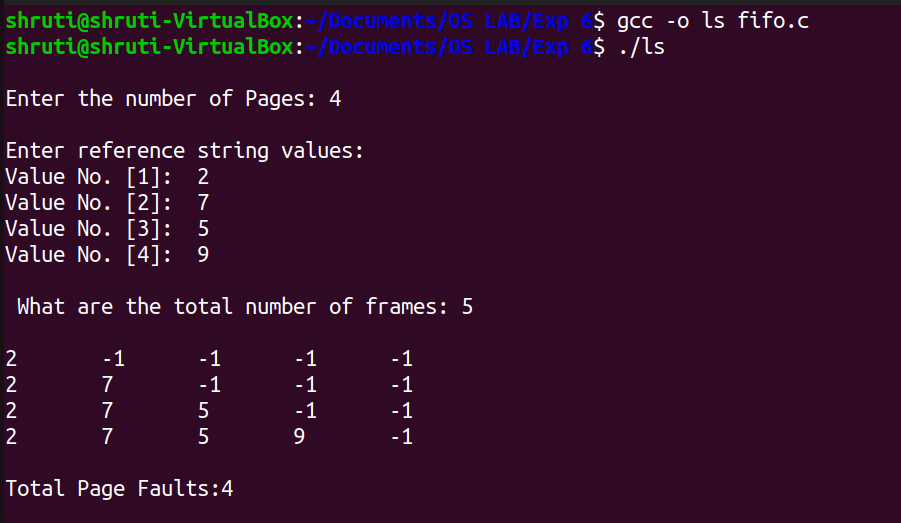
printf("\nTotal Page Faults:%d\n", pageFaults);

printf("\n");

return 0;

}

**OUTPUT**:



**LRU** – Least Recently Used

**CODE**

#include<stdio.h>

*int* findLRU(*int* *time*[], *int* *n*){

*int* i, minimum = *time*[0], pos = 0;

    for(i = 1; i < *n*; ++i){

        if(*time*[i] < minimum){

            minimum = *time*[i];

            pos = i;

        }

    }

    return pos;

}

*int* main()

{

*int* no\_of\_frames, no\_of\_pages, frames[10], pages[30], counter = 0, time[10], flag1, flag2, i, j, pos, faults = 0;

    printf("Enter number of frames: ");

    scanf("%d", &no\_of\_frames);

    printf("Enter number of pages: ");

    scanf("%d", &no\_of\_pages);

    printf("Enter reference string:\n");

    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]){

                counter++;

                time[j] = counter;

                   flag1 = flag2 = 1;

                   break;

               }

        }

        if(flag1 == 0){

            for(j = 0; j < no\_of\_frames; ++j){

                if(frames[j] == -1){

                    counter++;

                    faults++;

                    frames[j] = pages[i];

                    time[j] = counter;

                    flag2 = 1;

                    break;

                }

            }

        }

        if(flag2 == 0){

            pos = findLRU(time, no\_of\_frames);

            counter++;

            faults++;

            frames[pos] = pages[i];

            time[pos] = counter;

        }

        printf("\n");

        for(j = 0; j < no\_of\_frames; ++j){

            printf("%d\t", frames[j]);

        }

    }

    printf("\n\nTotal Page Faults = %d", faults);

    printf("\n");

    return 0;

}

**OUTPUT**:

