**Aim:** To find the number of page faults, hit-ratio by implementing the LRU Page

Replacement Algorithm.

**Algorithm:**

**Step 1.** Start the process

**Step 2.** Declare the page size

**Step 3.** Determine the number of pages to be inserted**.**

**Step 4.** Get the value.

**Step 5.** Declare the counter and stack value.

**Step 6.** Choose the least recently used page by the counter value.

**Step 7.** Stack them as per the selection.

**Step 8.** Display the values.

**Step 9.** Terminate the process.

**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);

float pagesss=no\_of\_pages;

printf("Enter 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]){

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);

float pagefaults=faults;

float miss=pagefaults/pagesss;

float hit=1-miss;

printf("\n\nHit ratio: %.2f",hit);

printf("\n\nMiss ratio: %.2f\n\n",miss);

return 0;

}

**Output(screenshots):**

