**Name:** Adwait S Purao

**UID:** 2021300101

**Batch:** B2

**Branch:** Computer Engineering

**Experiment No.** 7

**Page replacement algorithms:**

1. **First In First out.**

**Code :**

#include<stdio.h>

void fifo(int string[20],int n,int size)

{

int frames[n];

for (int i=0;i<n;i++)

frames[i]=-1;

int index=-1;

int page\_miss=0;

int page\_hits=0;

for (int i=0;i<size;i++)

{

int symbol=string[i];

int flag=0;

for(int j=0;j<n;j++)

{

if (symbol==frames[j])

{

flag=1;

break;

}

}

if (flag==1)

{

printf("\nSymbol: %d Frame: ",symbol);

for (int j=0;j<n;j++)

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

page\_hits+=1;

}

else

{

index=(index+1)%n;

frames[index]=symbol;

page\_miss+=1;

printf("\nSymbol: %d Frame: ",symbol);

for (int j=0;j<n;j++)

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

}

}

printf("\nPage hits: %d",page\_hits);

printf("\nPage misses: %d",page\_miss);

}

int main(void)

{

int n;

printf("Enter the size of string\n");

scanf("%d",&n);

int string[n];

printf("Enter the string\n");

for(int i=0;i<n;i++){

scanf("%d",&string[i]);

}

int nf;

printf("Enter the number of frames\n");

scanf("%d",&nf);

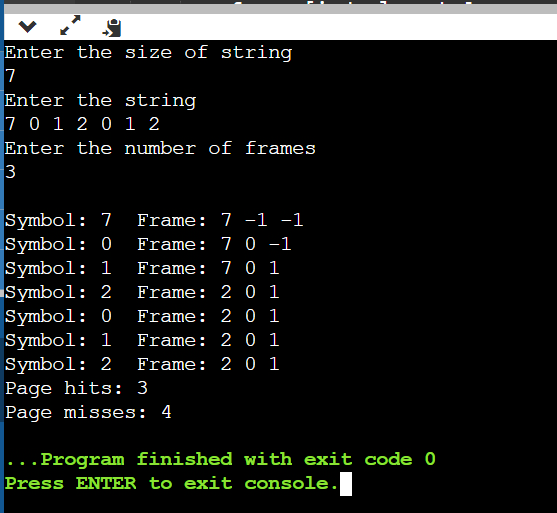
int size=sizeof(string)/sizeof(int);

fifo(string,nf,size);

return 0;

}

**Output:**

****

1. **Optimal Page Replacement.**

**Code:**

#include<stdio.h>

int search(int key, int frame\_items[], int frame\_occupied)

{

for (int i = 0; i < frame\_occupied; i++)

if (frame\_items[i] == key)

return 1;

return 0;

}

void printOuterStructure(int nf){

printf("Stream ");

for(int i = 0; i < nf; i++)

printf("Frame%d ", i+1);

}

void printCurrFrames(int item, int frame\_items[], int frame\_occupied, int nf){

printf("\n%d \t\t", item);

for(int i = 0; i < nf; i++){

if(i < frame\_occupied)

printf("%d \t\t", frame\_items[i]);

else

printf("- \t\t");

}

}

int predict(int string[], int frame\_items[], int refStrLen, int index, int frame\_occupied)

{

int result = -1, farthest = index;

for (int i = 0; i < frame\_occupied; i++) {

int j;

for (j = index; j < refStrLen; j++)

{

if (frame\_items[i] == string[j])

{

if (j > farthest) {

farthest = j;

result = i;

}

break;

}

}

if (j == refStrLen)

return i;

}

return (result == -1) ? 0 : result;

}

void optimalPage(int string[], int refStrLen, int frame\_items[], int nf)

{

int frame\_occupied = 0;

printOuterStructure(nf);

int hits = 0;

for (int i = 0; i < refStrLen; i++) {

if (search(string[i], frame\_items, frame\_occupied)) {

hits++;

printCurrFrames(string[i], frame\_items, frame\_occupied, nf);

continue;

}

if (frame\_occupied < nf){

frame\_items[frame\_occupied] = string[i];

frame\_occupied++;

printCurrFrames(string[i], frame\_items, frame\_occupied, nf);

}

else {

int pos = predict(string, frame\_items, refStrLen, i + 1, frame\_occupied);

frame\_items[pos] = string[i];

printCurrFrames(string[i], frame\_items, frame\_occupied, nf);

}

}

printf("\n\nHits: %d\n", hits);

printf("Misses: %d", refStrLen - hits);

}

int main()

{

int n;

printf("Enter the size of string\n");

scanf("%d",&n);

int string[n];

printf("Enter the string\n");

for(int i=0;i<n;i++){

scanf("%d",&string[i]);

}

int nf;

printf("Enter the number of frames\n");

scanf("%d",&nf);

int refStrLen = sizeof(string) / sizeof(string[0]);

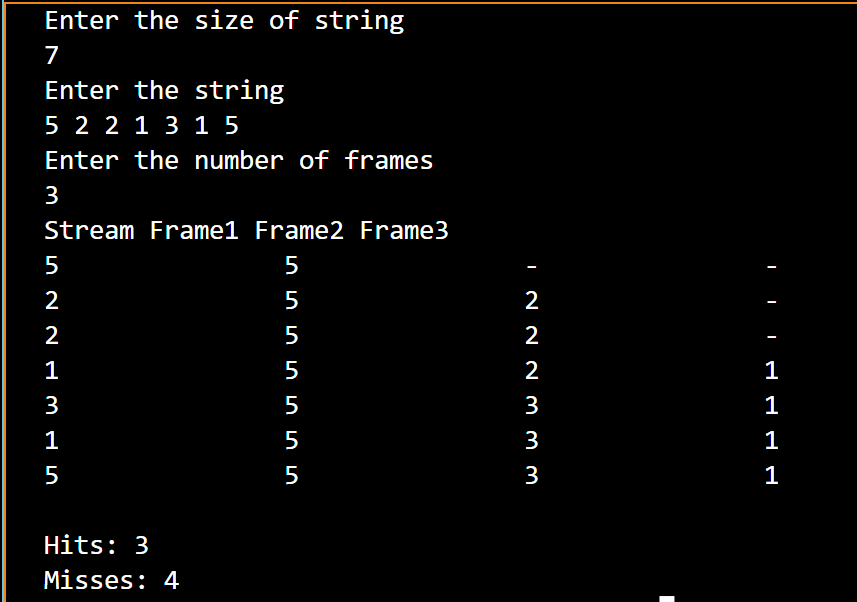
int frame\_items[nf];

optimalPage(string, refStrLen, frame\_items, nf);

return 0;

}

**Output:**

****

1. **Least recently Used**

**Code:**

import java.io.\*;

import java.util.\*;

public class Main {

public static void main(String[] args) throws IOException

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int frames,pointer = 0, hit = 0, fault = 0,ref\_len;

Boolean isFull = false;

int buffer[];

ArrayList<Integer> stack = new ArrayList<Integer>();

int reference[];

int mem\_layout[][];

System.out.println("Please enter the number of Frames: ");

frames = Integer.parseInt(br.readLine());

System.out.println("Please enter the length of the Reference string: ");

ref\_len = Integer.parseInt(br.readLine());

reference = new int[ref\_len];

mem\_layout = new int[ref\_len][frames];

buffer = new int[frames];

for(int j = 0; j < frames; j++)

buffer[j] = -1;

System.out.println("Please enter the reference string: ");

for(int i = 0; i < ref\_len; i++)

{

reference[i] = Integer.parseInt(br.readLine());

}

System.out.println();

for(int i = 0; i < ref\_len; i++)

{

if(stack.contains(reference[i]))

{

stack.remove(stack.indexOf(reference[i]));

}

stack.add(reference[i]);

int search = -1;

for(int j = 0; j < frames; j++)

{

if(buffer[j] == reference[i])

{

search = j;

hit++;

break;

}

}

if(search == -1)

{

if(isFull)

{

int min\_loc = ref\_len;

for(int j = 0; j < frames; j++)

{

if(stack.contains(buffer[j]))

{

int temp = stack.indexOf(buffer[j]);

if(temp < min\_loc)

{

min\_loc = temp;

pointer = j;

}

}

}

}

buffer[pointer] = reference[i];

fault++;

pointer++;

if(pointer == frames)

{

pointer = 0;

isFull = true;

}

}

for(int j = 0; j < frames; j++)

mem\_layout[i][j] = buffer[j];

}

for(int i = 0; i < frames; i++)

{

for(int j = 0; j < ref\_len; j++)

System.out.printf("%3d ",mem\_layout[j][i]);

System.out.println();

}

System.out.println("The number of Hits: " + hit);

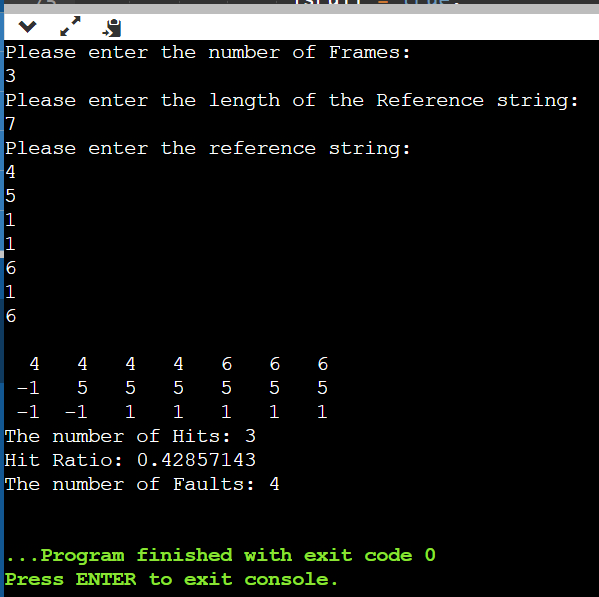
System.out.println("Hit Ratio: " + (float)((float)hit/ref\_len));

System.out.println("The number of Faults: " + fault);

}

}

**Output:**

****

**Conclusion:**

In the above experiment we learnt about the various page replacement algorithms and implemented the code in c and java.