**CODE >>**

#include<stdio.h>

#include<stdlib.h>

void printFrames(int frames[], int frameSize) {

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

if (frames[i] == -1) {

printf("X ");

} else {

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

}

}

printf("\n");

}

void FCFS(int pages[], int frameSize, int n) {

int frames[frameSize];

int counter = 0;

int pageFaults = 0;

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

frames[i] = -1;

}

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

int pageExists = 0;

for (int j = 0; j < frameSize; ++j) {

if (frames[j] == pages[i]) {

pageExists = 1;

break;

}

}

if (pageExists == 0) {

frames[counter] = pages[i];

++counter;

++pageFaults;

if (counter >= frameSize) {

counter = 0;

}

}

printf("Frames (FCFS): ");

printFrames(frames, frameSize);

}

printf("Number of page faults using FCFS: %d\n\n", pageFaults);

}

void LRU(int pages[], int frameSize, int n) {

int frames[frameSize];

int counter = 0;

int pageFaults = 0;

int recent[frameSize] = {-1, -1, -1};

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

frames[i] = -1;

}

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

int pageExists = 0;

for (int j = 0; j < frameSize; ++j) {

if (frames[j] == pages[i]) {

pageExists = 1;

recent[j] = i;

break;

}

}

if (pageExists == 0) {

int leastRecentIndex = 0;

for (int j = 1; j < frameSize; ++j) {

if (recent[j] < recent[leastRecentIndex]) {

leastRecentIndex = j;

}

}

frames[leastRecentIndex] = pages[i];

recent[leastRecentIndex] = i;

++pageFaults;

}

printf("Frames (LRU): ");

printFrames(frames, frameSize);

}

printf("Number of page faults using LRU: %d\n\n", pageFaults);

}

void Optimal(int pages[], int frameSize,int n) {

int frames[frameSize];

int pageFaults = 0;

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

frames[i] = -1;

}

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

int pageExists = 0;

for (int j = 0; j < frameSize; ++j) {

if (frames[j] == pages[i]) {

pageExists = 1;

break;

}

}

if (pageExists == 0) {

int maxDistance = -1;

int replaceIndex = -1;

for (int j = 0; j < frameSize; ++j) {

int futureIndex = 0;

for (int k = i + 1; k < 20; ++k) {

if (frames[j] == pages[k]) {

futureIndex = k;

break;

}

}

if (futureIndex == 0) {

replaceIndex = j;

break;

} else {

if (futureIndex > maxDistance) {

maxDistance = futureIndex;

replaceIndex = j;

}

}

}

frames[replaceIndex] = pages[i];

++pageFaults;

}

printf("Frames (Optimal): ");

printFrames(frames, frameSize);

}

printf("Number of page faults using Optimal: %d\n\n", pageFaults);

}

int main() {

int pages[20];

int n;

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

scanf("%d",&n);

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

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

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

}

int frameSize = 3;

FCFS(pages, frameSize,n);

LRU(pages, frameSize,n);

Optimal(pages, frameSize,n);

return 0;

}

**OUTPUT >>**

Enter the size of string

12

Enter the reference string:

5

6

7

8

5

6

9

5

6

7

8

9

Frames (FCFS): 5 X X

Frames (FCFS): 5 6 X

Frames (FCFS): 5 6 7

Frames (FCFS): 8 6 7

Frames (FCFS): 8 5 7

Frames (FCFS): 8 5 6

Frames (FCFS): 9 5 6

Frames (FCFS): 9 5 6

Frames (FCFS): 9 5 6

Frames (FCFS): 9 7 6

Frames (FCFS): 9 7 8

Frames (FCFS): 9 7 8

Number of page faults using FCFS: 9

Frames (LRU): 5 X X

Frames (LRU): 5 6 X

Frames (LRU): 5 6 7

Frames (LRU): 8 6 7

Frames (LRU): 8 5 7

Frames (LRU): 8 5 6

Frames (LRU): 9 5 6

Frames (LRU): 9 5 6

Frames (LRU): 9 5 6

Frames (LRU): 7 5 6

Frames (LRU): 7 8 6

Frames (LRU): 7 8 9

Number of page faults using LRU: 10

Frames (Optimal): 5 X X

Frames (Optimal): 5 6 X

Frames (Optimal): 5 6 7

Frames (Optimal): 5 6 8

Frames (Optimal): 5 6 8

Frames (Optimal): 5 6 8

Frames (Optimal): 5 6 9

Frames (Optimal): 5 6 9

Frames (Optimal): 5 6 9

Frames (Optimal): 5 7 9

Frames (Optimal): 5 8 9

Frames (Optimal): 5 8 9

Number of page faults using Optimal: 7