PRACTICAL NO: 06

Name: Riyan Shaikh

Roll No: T512051 [B]

CODE:

#include <stdio.h>

#include <limits.h>

// Function to check if page is already present in frame

int isPageInFrame(int frames[], int frameSize, int page) {

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

if (frames[i] == page)

return 1;

}

return 0;

}

// Function to find index of the page to replace using LRU

int findLRU(int time[], int frameSize) {

int minTime = INT\_MAX, minIndex = 0;

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

if (time[i] < minTime) {

minTime = time[i];

minIndex = i;

}

}

return minIndex;

}

// Function to find index of the page to replace using Optimal

int findOptimal(int frames[], int frameSize, int pages[], int pageCount, int currentIndex) {

int farthest = currentIndex, replaceIndex = -1;

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

int j;

for (j = currentIndex; j < pageCount; j++) {

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

if (j > farthest) {

farthest = j;

replaceIndex = i;

}

break;

}

}

if (j == pageCount) // Page not found in future

return i;

}

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

}

// FCFS Page Replacement

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

int frames[frameSize], front = 0, pageFaults = 0;

for (int i = 0; i < frameSize; i++) frames[i] = -1;

printf("\nFCFS Page Replacement:\n");

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

if (!isPageInFrame(frames, frameSize, pages[i])) {

frames[front] = pages[i];

front = (front + 1) % frameSize;

pageFaults++;

printf("Page fault for page %d\n", pages[i]);

} else {

printf("No page fault for page %d\n", pages[i]);

}

}

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

}

// LRU Page Replacement

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

int frames[frameSize], time[frameSize], currentTime = 0, pageFaults = 0;

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

frames[i] = -1;

time[i] = 0;

}

printf("\nLRU Page Replacement:\n");

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

int replaceIndex;

if (!isPageInFrame(frames, frameSize, pages[i])) {

replaceIndex = findLRU(time, frameSize);

frames[replaceIndex] = pages[i];

pageFaults++;

printf("Page fault for page %d\n", pages[i]);

} else {

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

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

replaceIndex = j;

break;

}

}

printf("No page fault for page %d\n", pages[i]);

}

time[replaceIndex] = currentTime++; // Update time for the replaced page

}

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

}

// Optimal Page Replacement

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

int frames[frameSize], pageFaults = 0;

for (int i = 0; i < frameSize; i++) frames[i] = -1;

printf("\nOptimal Page Replacement:\n");

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

if (!isPageInFrame(frames, frameSize, pages[i])) {

int replaceIndex = findOptimal(frames, frameSize, pages, pageCount, i + 1);

frames[replaceIndex] = pages[i];

pageFaults++;

printf("Page fault for page %d\n", pages[i]);

} else {

printf("No page fault for page %d\n", pages[i]);

}

}

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

}

int main() {

int pageCount, frameSize;

// Minimum frame size = 3

frameSize = 3;

printf("Enter the number of pages: ");

scanf("%d", &pageCount);

int pages[pageCount];

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

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

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

}

FCFS(pages, pageCount, frameSize);

LRU(pages, pageCount, frameSize);

Optimal(pages, pageCount, frameSize);

return 0;

}

Output:

