

**Program:**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include <limits.h>

// Function to check if page is in frame

bool isPagePresent(int frames[], int n, int page) {

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

if (frames[i] == page) return true;

}

return false;

}

// Function to print the frames

void printFrames(int frames[], int n, bool isPageFault) {

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

if (frames[i] == -1) printf("- ");

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

}

printf("%s\n", isPageFault ? "(Page Fault)" : "(Page Hit)");

}

void printSummary(int pageFaults, int pageHits, int total) {

float faultRatio = (float)pageFaults / total \* 100;

float hitRatio = (float)pageHits / total \* 100;

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

printf("Total Page Hits: %d\n", pageHits);

printf("Page Fault Ratio: %.2f%%\n", faultRatio);

printf("Page Hit Ratio: %.2f%%\n", hitRatio);

}

// FIFO

void fifo(int pages[], int n, int frameCount) {

int frames[frameCount], pageFaults = 0, pageHits = 0, pointer = 0;

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

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

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

printf("Reference %d (Page %d): ", i + 1, pages[i]);

if (!isPagePresent(frames, frameCount, pages[i])) {

frames[pointer] = pages[i];

pointer = (pointer + 1) % frameCount;

pageFaults++;

printFrames(frames, frameCount, true);

} else {

pageHits++;

printFrames(frames, frameCount, false);

}

}

printSummary(pageFaults, pageHits, n);

}

// LRU

void lru(int pages[], int n, int frameCount) {

int frames[frameCount], counter[frameCount], time = 0, pageFaults = 0, pageHits = 0;

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

frames[i] = -1;

counter[i] = 0;

}

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

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

time++;

printf("Reference %d (Page %d): ", i + 1, pages[i]);

if (!isPagePresent(frames, frameCount, pages[i])) {

int lruIndex = 0;

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

if (counter[j] < counter[lruIndex]) lruIndex = j;

}

frames[lruIndex] = pages[i];

counter[lruIndex] = time;

pageFaults++;

printFrames(frames, frameCount, true);

} else {

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

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

counter[j] = time;

break;

}

}

pageHits++;

printFrames(frames, frameCount, false);

}

}

printSummary(pageFaults, pageHits, n);

}

// Optimal

void optimal(int pages[], int n, int frameCount) {

int frames[frameCount], pageFaults = 0, pageHits = 0;

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

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

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

printf("Reference %d (Page %d): ", i + 1, pages[i]);

if (!isPagePresent(frames, frameCount, pages[i])) {

int farthest = i, replaceIndex = 0;

bool found;

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

found = false;

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

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

if (k > farthest) {

farthest = k;

replaceIndex = j;

}

found = true;

break;

}

}

if (!found) {

replaceIndex = j;

break;

}

}

frames[replaceIndex] = pages[i];

pageFaults++;

printFrames(frames, frameCount, true);

} else {

pageHits++;

printFrames(frames, frameCount, false);

}

}

printSummary(pageFaults, pageHits, n);

}

// LFU

void lfu(int pages[], int n, int frameCount) {

int frames[frameCount], frequency[frameCount], pageFaults = 0, pageHits = 0;

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

frames[i] = -1;

frequency[i] = 0;

}

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

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

printf("Reference %d (Page %d): ", i + 1, pages[i]);

if (!isPagePresent(frames, frameCount, pages[i])) {

int lfuIndex = 0;

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

if (frequency[j] < frequency[lfuIndex]) {

lfuIndex = j;

}

}

frames[lfuIndex] = pages[i];

frequency[lfuIndex] = 1;

pageFaults++;

printFrames(frames, frameCount, true);

} else {

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

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

frequency[j]++;

break;

}

}

pageHits++;

printFrames(frames, frameCount, false);

}

}

printSummary(pageFaults, pageHits, n);

}

// Main Menu

int main() {

int n, frameCount, choice;

char cont;

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

scanf("%d", &n);

int pages[n];

printf("Enter the page reference sequence: ");

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

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

}

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

scanf("%d", &frameCount);

do {

printf("\n--- Page Replacement Menu ---\n");

printf("1. FIFO\n");

printf("2. LRU\n");

printf("3. Optimal\n");

printf("4. LFU\n");

printf("5. Run All\n");

printf("6. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1: fifo(pages, n, frameCount); break;

case 2: lru(pages, n, frameCount); break;

case 3: optimal(pages, n, frameCount); break;

case 4: lfu(pages, n, frameCount); break;

case 5:

fifo(pages, n, frameCount);

lru(pages, n, frameCount);

optimal(pages, n, frameCount);

lfu(pages, n, frameCount);

break;

case 6: exit(0);

default: printf("Invalid choice!\n");

}

printf("\nDo you want to continue? (y/n): ");

scanf(" %c", &cont);

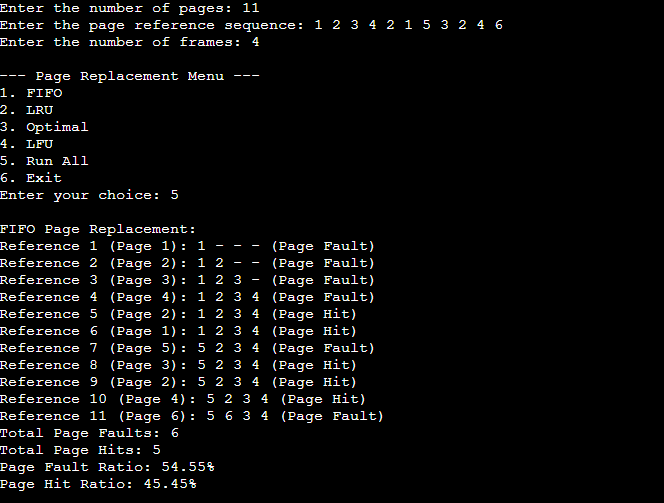
} while (cont == 'y' || cont == 'Y');

return 0;

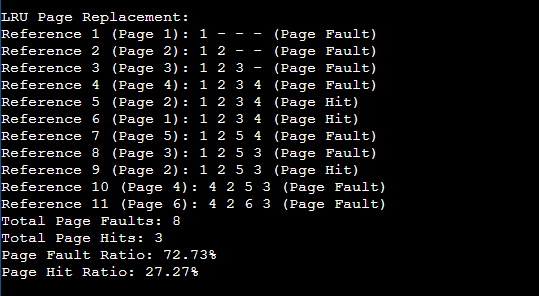
}

**Output :**

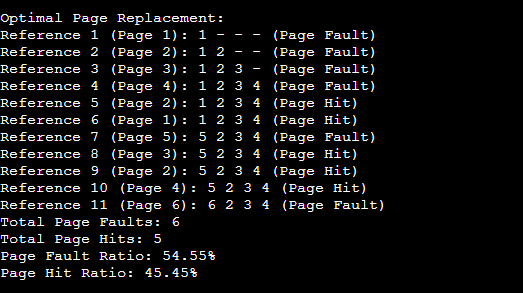
**FIFO:**



**LRU:**



**OPTIMAL:**



**LFU:**

