Lru

#include <stdio.h>

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

int min = time[0], pos = 0;

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

if (time[i] < min) {

min = time[i];

pos = i;

}

}

return pos;

}

int main() {

int frames, pages;

// Step 1: Input number of frames and pages

printf("Enter number of frames: ");

scanf("%d", &frames);

printf("Enter number of pages: ");

scanf("%d", &pages);

int pageSeq[100];

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

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

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

}

int memory[frames]; // Memory frames

int time[frames]; // To store the last used time

int counter = 0; // Global time counter

int pageFaults = 0;

// Step 2: Initialize memory with -1

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

memory[i] = -1;

time[i] = 0;

}

// Step 3: Traverse each page

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

int page = pageSeq[i];

int found = 0;

// Check if page is already in memory (Page Hit)

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

if (memory[j] == page) {

counter++;

time[j] = counter; // Update time of usage

found = 1;

break;

}

}

// If page not found (Page Fault)

if (!found) {

int pos = -1;

// Check for empty space first

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

if (memory[j] == -1) {

pos = j;

break;

}

}

// If no empty space, use LRU

if (pos == -1) {

pos = findLRU(time, frames); // Find LRU position

}

// Replace the page

memory[pos] = page;

counter++;

time[pos] = counter;

pageFaults++;

printf("Page %d → Page Fault\tMemory: ", page);

} else {

printf("Page %d → No Page Fault\tMemory: ", page);

}

// Print current memory state

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

if (memory[j] != -1)

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

else

printf("- ");

}

printf("\n");

}

printf("\nTotal Page Faults = %d\n", pageFaults);

return 0;

}

START

1. Input number of frames and pages

2. Input page reference string

3. Initialize memory with -1 and time array with 0

4. For each page in reference string:

a. If page is in memory:

→ It's a hit → update its time

b. Else:

→ It's a fault:

i. If memory has empty space:

- Add page to empty frame

ii. Else:

- Find the page with oldest time (Least Recently Used)

- Replace it with the new page

→ Update time and increment page fault count

c. Print current memory state

5. After all pages processed:

→ Print total number of page faults

END