Fifo

#include <stdio.h>

int main() {

int frames, pages;

// Step 1: Input number of frames and number of pages

printf("Enter number of frames: ");

scanf("%d", &frames);

printf("Enter number of pages: ");

scanf("%d", &pages);

int pageSeq[100]; // To store page reference string

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

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

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

}

int memory[frames]; // Array to represent memory frames

int front = 0; // Queue front index for FIFO

int pageFaults = 0;

// Initialize memory frames with -1 (empty)

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

memory[i] = -1;

}

// Step 2: Go through each page in the reference string

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

int page = pageSeq[i];

int found = 0;

// Step 3: Check if the page is already in memory

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

if (memory[j] == page) {

found = 1; // Page hit

break;

}

}

// Step 4: If page not found → page fault occurs

if (!found) {

memory[front] = page; // Replace the oldest page (FIFO)

front = (front + 1) % frames; // Circular increment

pageFaults++;

// Print current memory state

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

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

if (memory[k] != -1)

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

else

printf("- ");

}

printf("\n");

} else {

// Page hit

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

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

if (memory[k] != -1)

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

else

printf("- ");

}

printf("\n");

}

}

// Final result

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

return 0;

}

START

1. Input number of frames (n)

2. Input number of pages (m) and page reference string

3. Initialize memory frames with -1

4. Set front = 0 and page\_faults = 0

5. For each page in the reference string:

a. If page is already in memory:

→ Print "No page fault"

b. Else:

i. Replace memory[front] with new page

ii. front = (front + 1) % n

iii. Increment page\_faults

iv. Print current memory state and "Page Fault"

6. After all pages, print total page\_faults

END