

**Lab No: 1 Date: 2082/**

**Title: Write a program to calculate the number of page fault for user input reference string and frame size using FIFO page replacement algorithm.**

FIFO (First-In First-Out) is the simplest page replacement algorithm used in Operating Systems. It uses the principle of queue. When a page fault occurs and all frames are full, the oldest page (the one that entered first) is replaced with the new page.

Algorithm:

Step 1: Start with an empty queue (frames).

Step 2: Read the reference string (sequence of page requests).

Step 3: For each page request, check if the page is already present in frames.

* Step 3.1: If the page is found → Page Hit (do nothing).
* Step 3.2: If the page is not found → Page Fault occurs.
  + Step 3.2.1: If there is space in frames → insert the page into the frame.
  + Step 3.2.2: If there is no space → remove the oldest page (front of the queue) and insert the new page.

Step 4: Update the count of Page Faults and Page Hits accordingly.

Step 5: Repeat Step 3 for all pages in the reference string.

Step 6: After processing all pages, output the total Page Faults and total Page Hits.

Step 7: End.

**Language**: C++

**IDE**: VS Code

**Code:**

**#include <iostream>**

**#include <vector>**

**#include <queue>**

**#include <iomanip>**

**using namespace std;**

**void fifoPageReplacement(const string &referenceString, int frameSize) {**

**vector<char> frames;    // store characters in frames**

**queue<char> q;          // FIFO order**

**int pageFaults = 0, pageHits = 0;**

**cout << "\nStep-by-step Table (FIFO Page Replacement):\n";**

**cout << "---------------------------------------------------------------\n";**

**cout << setw(10) << "Page"**

**<< setw(20) << "Frames"**

**<< setw(15) << "Page Fault"**

**<< setw(15) << "Page Hit\n";**

**cout << "---------------------------------------------------------------\n";**

**for (char page : referenceString) {**

**bool found = false;**

**// Check if page is already in frames**

**for (char f : frames) {**

**if (f == page) {**

**found = true;**

**break;**

**}**

**}**

**if (!found) { // Page fault**

**if ((int)frames.size() < frameSize) {**

**frames.push\_back(page);**

**q.push(page);**

**} else {**

**// Replace oldest page (front of queue)**

**char oldPage = q.front();**

**q.pop();**

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

**if (frames[i] == oldPage) {**

**frames[i] = page;**

**break;**

**}**

**}**

**q.push(page);**

**}**

**pageFaults++;**

**} else {**

**pageHits++;**

**}**

**// Print current step**

**cout << setw(10) << page << setw(20);**

**for (char f : frames) cout << f << " ";**

**cout << setw(15) << (found ? "No" : "Yes")**

**<< setw(15) << (found ? "Yes" : "No") << "\n";**

**}**

**cout << "---------------------------------------------------------------\n";**

**cout << "Total Page Faults = " << pageFaults << endl;**

**cout << "Total Page Hits   = " << pageHits;**

**}**

**int main() {**

**int refSize;**

**string referenceString;**

**int frameSize;**

**cout << "Enter reference string size: ";**

**cin >> refSize;**

**cout << "Enter reference string: ";**

**for (int i = 0; i < refSize; ++i) {**

**char page;**

**cin >> page;**

**referenceString += page;**

**}**

**cout << "Enter frame size: ";**

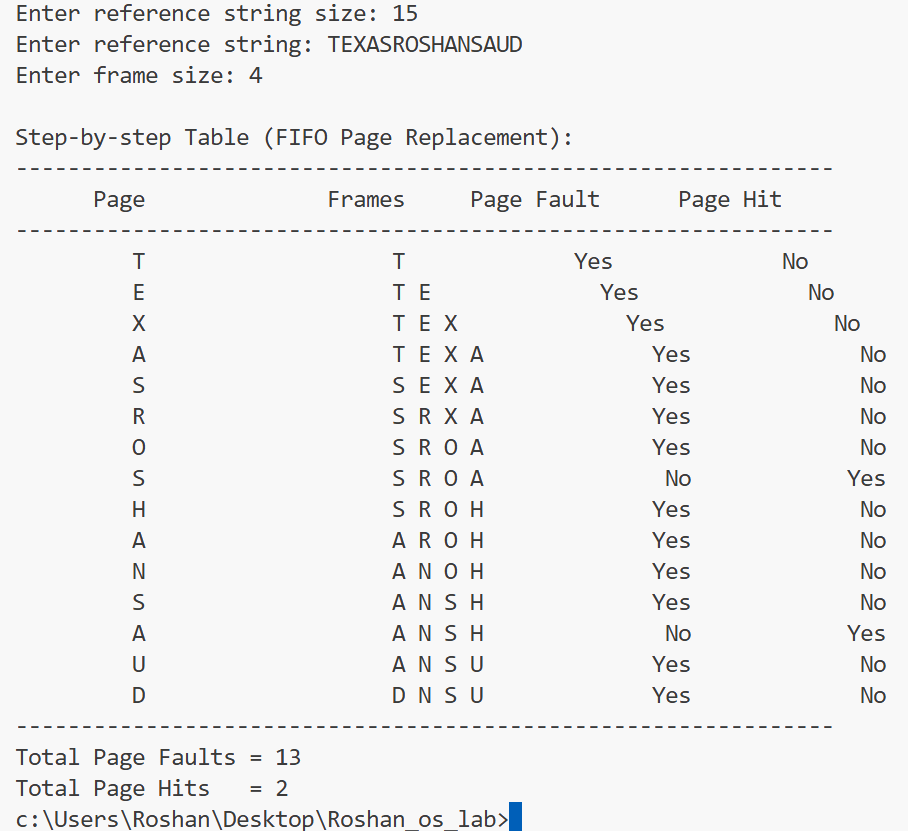
**cin >> frameSize;**

**fifoPageReplacement(referenceString, frameSize);**

**return 0;**

**}**

**Output:**

****

**Conclusion:**

The FIFO page replacement algorithm is simple to implement and replaces the oldest page when a new one arrives. While easy to use, it does not always give the best performance and may lead to more page faults compared to algorithms like LRU or Optimal.