**Lab-8**

**Page Replacement Algorithms Implementation**

**1. Objective**

* To implement FIFO (First-In-First-Out) & Optimal Page Replacement algorithms and To calculate and compare page faults for different algorithms

**2. Theory**

**Page Replacement Algorithms :**

When a page fault occurs and all frames are occupied, the operating system must select a page to remove from memory to make space for the requested page. The selection is done using a page replacement algorithm.

**FIFO Algorithm :**

The First-In-First-Out (FIFO) algorithm replaces the page that has been in memory the longest. It maintains a queue of pages in the order they were loaded into memory. When a page needs to be replaced, the page at the front of the queue (the oldest page) is selected for replacement.

**Characteristics:**

* Simple to implement
* Uses a queue data structure
* May suffer from Belady's Anomaly (increasing page faults with more frames)
* Not optimal in terms of performance

**Optimal Algorithm :**

The Optimal Page Replacement algorithm replaces the page that will not be used for the longest period in the future. This algorithm requires knowledge of the future page reference string.

**Characteristics:**

* Provides the lowest possible page fault rate for a fixed number of frames
* Used as a benchmark to compare other algorithms
* Not practical to implement in real systems as it requires future knowledge
* Also known as the Clairvoyant replacement algorithm

**Page Fault :**

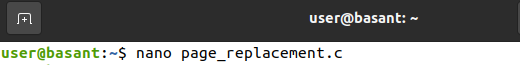
A page fault occurs when a program attempts to access a page that is not currently in physical memory. The operating system must then:

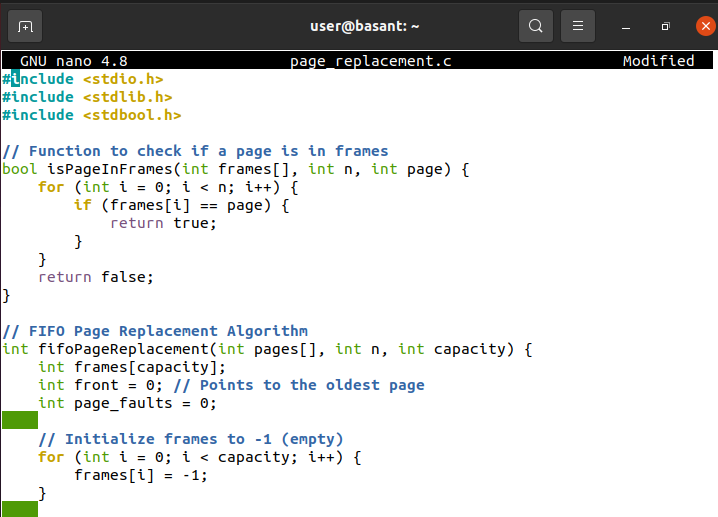
1. Find the location of the requested page on disk
2. Find a free frame or select a victim page to replace
3. Read the requested page into the frame
4. Update the page table
5. Restart the instruction that caused the page fault

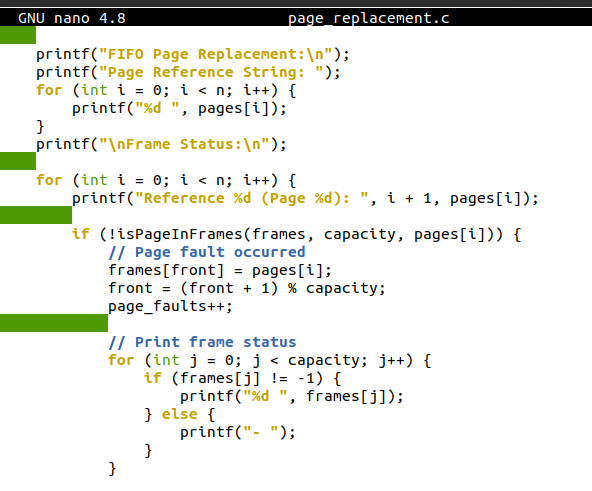
**3. Tools**

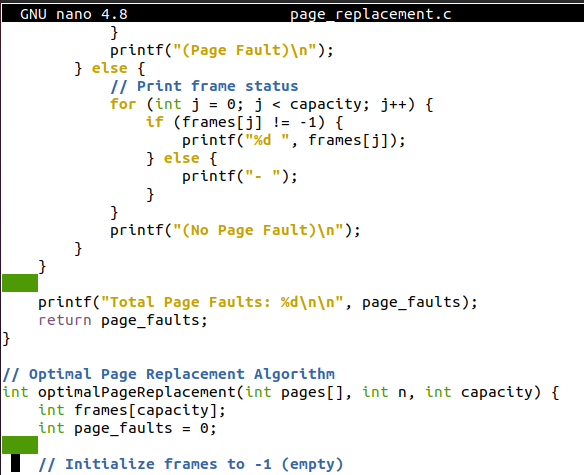
* gcc compiler
* Linux Terminal
* C Programming Language

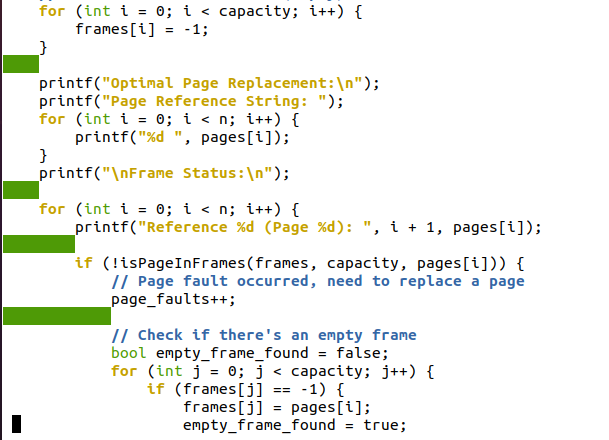
**4. Program  
a. Create the C file:**

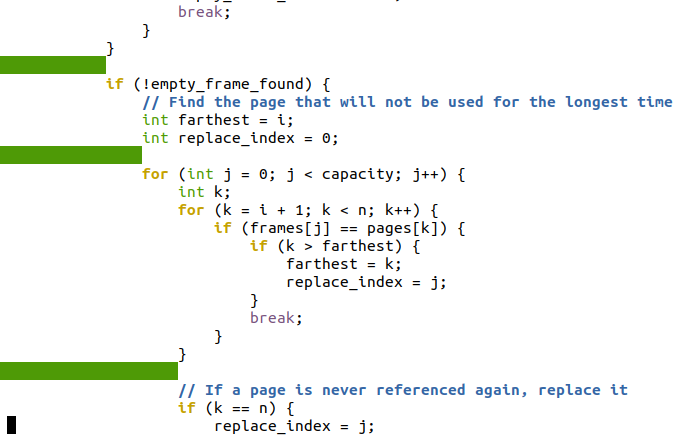
**  
b. Compile the program:**

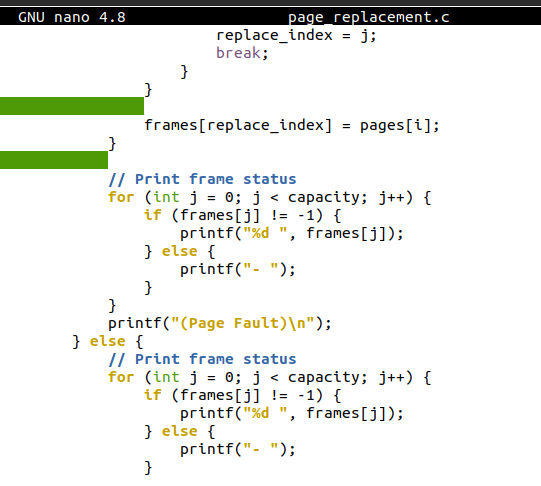
****

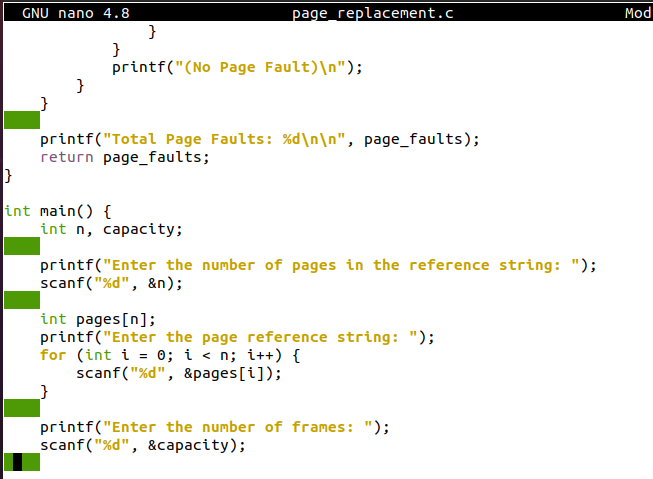
****

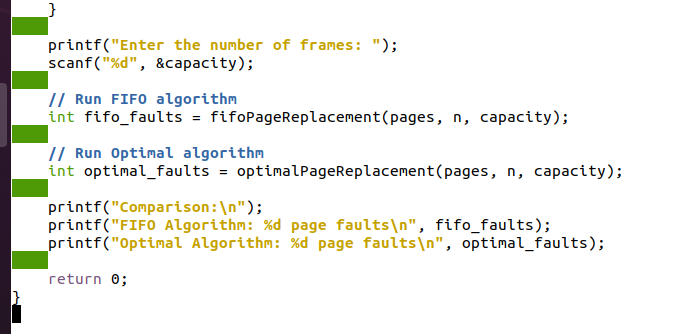
****

****

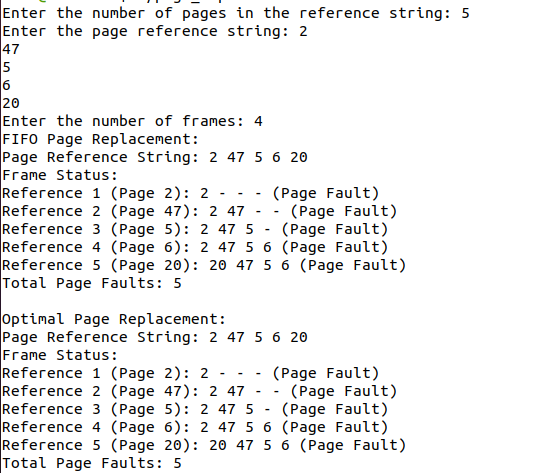
****

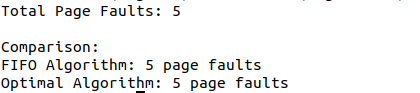
****

****

**  
c. Run the program:**

**  
d. Example input and output:**

****

****

1. **Conclusion**

The FIFO and Optimal page replacement algorithms were successfully implemented. The program correctly calculates and compares page faults for both algorithms, demonstrating the Optimal algorithm's superior efficiency due to its future-knowledge approach. This implementation effectively illustrates different page replacement strategies used in operating systems.