**EXPERIMENT 9**

**Aim:** Implementation of FIFO and LRU page replacement algorithms

**Lab Objective:** Students will be able to: Implement page replacement algorithms

**Theory:**

FIFO (First-In-First-Out) Page Replacement Algorithm: FIFO is one of the simplest page replacement algorithms used in computer operating systems. It works on the principle of first-come, first-served. When the operating system needs to replace a page in memory, it selects the page that has been in memory the longest, i.e., the page that was brought into memory first. Here's some theory about FIFO:

**Operation:** FIFO maintains a queue of pages in memory, and when a page replacement is required, it selects the page at the front of the queue (the oldest page) for removal.

**Advantages:**

* Simplicity: FIFO is easy to implement and understand.
* Low overhead: It doesn't require additional data structures beyond a queue.

**Disadvantages:**

* Belady's Anomaly: FIFO can suffer from Belady's Anomaly, where increasing the number of page frames may lead to an increase in page faults.
* Poor Performance: FIFO may not always make optimal choices regarding page replacement, as it doesn't consider the actual usage patterns of pages.
* LRU (Least Recently Used) Page Replacement Algorithm: LRU is a more sophisticated page replacement algorithm that selects the page for replacement based on its usage history. It assumes that the page that has been least recently used is the best candidate for replacement. Here's some theory about LRU:
* Operation: LRU keeps track of the order in which pages are accessed. When a page replacement is needed, LRU selects the page that has not been accessed for the longest time.

**Advantages:**

* Improved Performance: LRU is designed to minimize page faults by considering the actual usage patterns of pages.
* Fairness: LRU gives priority to pages that are actively used, promoting fair allocation of memory resources.

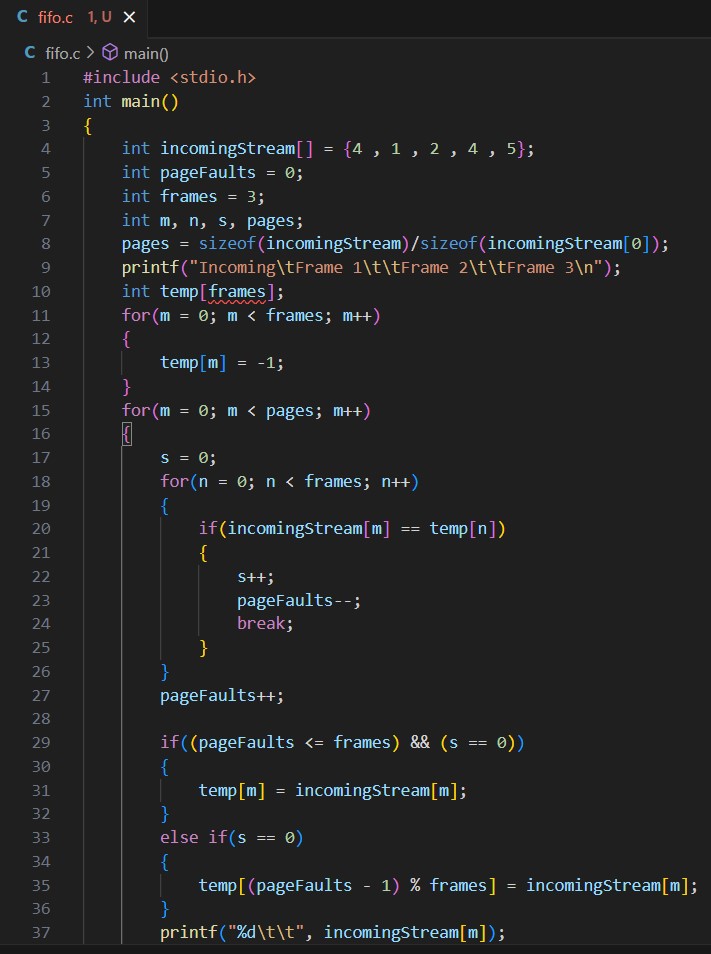
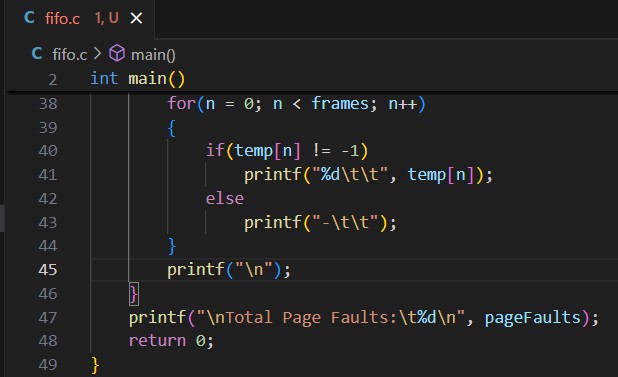
**Disadvantages:**

* Implementation Complexity: Implementing an efficient LRU algorithm can be more complex than FIFO. Maintaining the usage history of pages can be resource-intensive.
* High Overhead: LRU requires additional data structures, such as a queue or a list, to keep track of page access order.

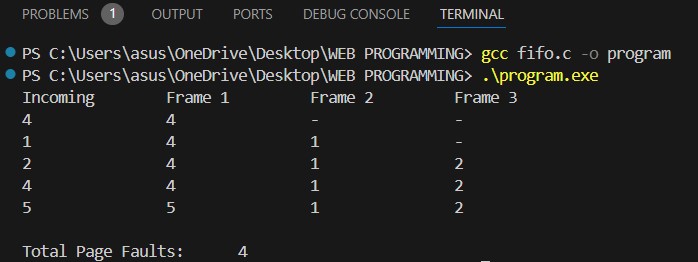
**Variants:** Various LRU variants, such as the clock algorithm or the second chance algorithm, aim to strike a balance between performance and implementation complexity.

In practice, the choice between FIFO and LRU depends on the specific requirements of the system and the available hardware resources. LRU is preferred when optimizing for performance, while FIFO may be chosen when simplicity is more important, or when hardware limitations restrict the use of more advanced algorithms.

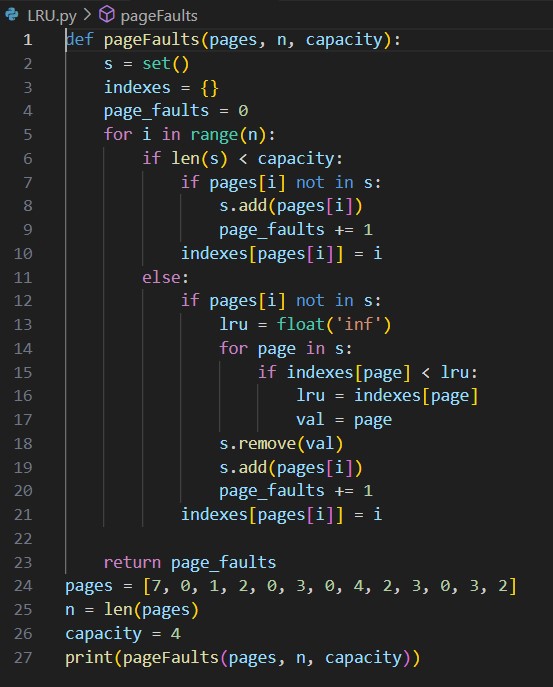
FIFO CODE :



OUTPUT:



LRU CODE :



Output:

