**EXPERIMENT 10**

**Aim:** Case study on reader writer problem Lab Objective: Students will be able to: Understand about reader writer problem

**Learning Objectives:**

Students will:

* Understand the Reader-Writer problem in concurrency.
* Learn to use semaphores for synchronization.

**Theory:**

**Introduction:** The Reader-Writer Problem is a classic synchronization problem in the field of concurrent programming. It highlights challenges related to share resource access by multiple threads or processes. In this case study, we will explore the Reader- Writer Problem, its real- world applications, and solutions to address these challenges.

Problem Statement: In the context of concurrent programming, the Reader-Writer Problem addresses the conflict between multiple readers and writers accessing a shared resource, such as a database or file. The problem can be summarized as follows:

1. Multiple readers can access the resource simultaneously without any issues.

2. Writers, however, need exclusive access to the resource. When a writer is writing, no other readers or writers should access the resource.

**Real-World Applications: The Reader-Writer Problem is not merely theoretical; it has practical implications in various domains:**

**1. Database Management Systems (DBMS):** In a database system, multiple users may need to read data simultaneously, while exclusive access is required during write operations. Efficient concurrency control mechanisms are necessary to ensure data integrity.

**2. File Systems:** When multiple processes access a shared file, concurrent read access is often allowed, but write access needs to be protected to prevent data corruption.

**3. Resource Management:** In operating systems and resource allocation systems, controlling access to shared resources is essential for maintaining system stability.

Solutions: Several solutions have been proposed to address the Reader-Writer Problem, balancing the need for concurrent access with the requirement for exclusive write access:

**1. First-Come-First-Serve (FCFS):** In this solution, writers are given exclusive access as soon as they request it, while readers are queued and granted access in the order they arrived. This method ensures that writers are not kept waiting indefinitely.Reader Priority: Readers are given preference unless a writer is waiting. This approach minimizes the chances of write starvation while still allowing concurrent read access.

**2. Writer Priority:** In this solution, writers are given preference, and once a writer arrives, no new readers are allowed to start reading. This approach ensures that readers do not lock writers out.

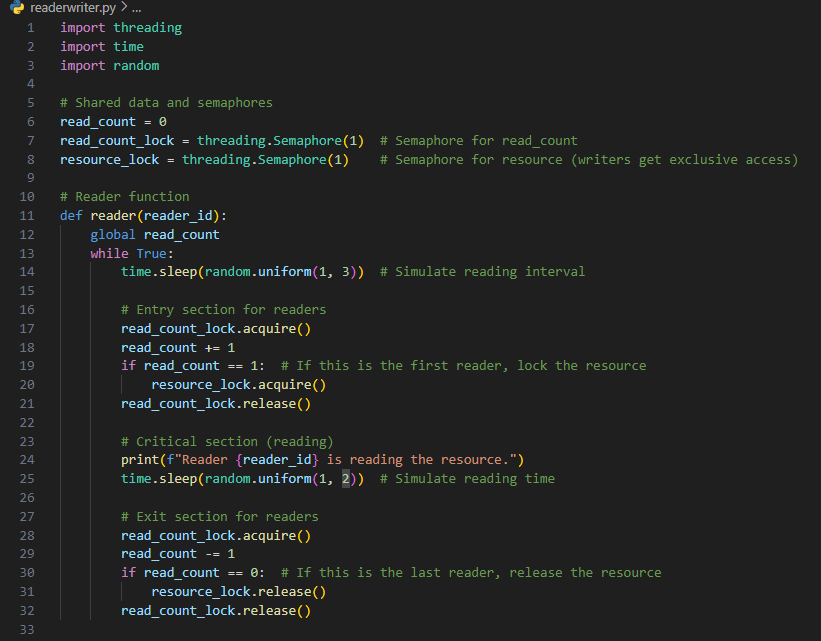
**3. Semaphore-Based Solution:** This method uses semaphores or other synchronization primitives to manage access to the shared resource, ensuring mutual exclusion between writers and allowing multiple readers.

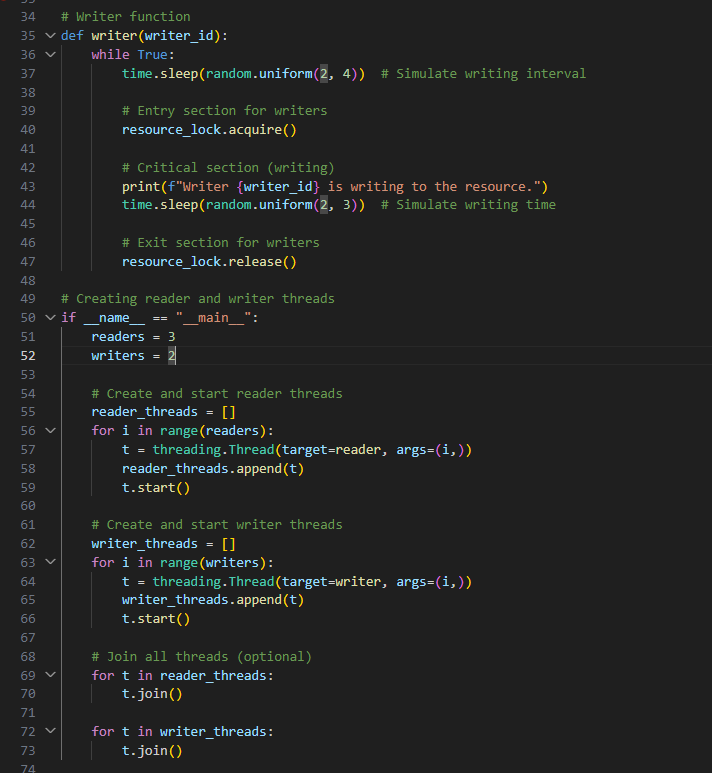
**4. Readers-Writers Locks:** Modern operating systems and programming libraries offer specialized readers-writers locks, which are designed for optimal performance and provide a balance between reader and writer access.

**Lab Outcome:** Students were able to:

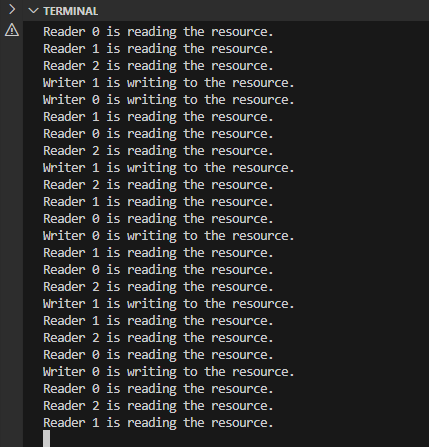
Understand the advantages and disadvantages of deadlock avoidance algorithm

**Code:**





**Output:**

****