**EXPERIMENT 20**

**Construct a C program to simulate Reader-Writer problem using semaphores**

AIM : To construct a C program to simulate Reader-Writer problem using semaphores

## ALGORITHM :

1. Include Libraries: Include necessary libraries for using semaphores, threads, and other required functionalities.
2. Initialize Semaphores: Create semaphores to control access to the shared resources:
   * Semaphore for Readers Count: Initialize a semaphore to 1 (binary semaphore).
   * Semaphore for Writers Count: Initialize a semaphore to 1 (binary semaphore).
   * Semaphore for Readers Waiting: Initialize a semaphore to 1 (binary semaphore).
   * Semaphore for Writers Waiting: Initialize a semaphore to 1 (binary semaphore).
   * Semaphore for Mutex: Initialize a semaphore to 1 (binary semaphore).
3. Reader Function: Create a function for readers to execute. This function should handle the logic for readers accessing the shared resource.
4. Writer Function: Create a function for writers to execute. This function should handle the logic for writers accessing the shared resource.
5. Implement Reader-Writer Logic: Inside the reader and writer functions, implement the logic that ensures proper synchronization using semaphores. Readers should check and update the readers count semaphore and writers should check and update the writers count semaphore.
6. Create Threads: In your main function, create multiple threads for readers and writers to simulate concurrent access.
7. Join Threads: Use thread joining functions to wait for all threads to complete their execution.

## Clean Up: Destroy the semaphores and perform any necessary clean-up operations before exiting the program

**OUTPUT :**

A screen shot of a computer

Description automatically generated