Bansilal Ramnath Agarwal Charitable Trust's
VISHWAKARMA INSTITUTE OF TECHNOLOGY, PUNE-37
(An Autonomous Institute of Savitribai Phule University)

Department of Artificial Intelligence & Data Science

Division	AI-A
Name	Mayank Dhananjay Kulkarni
Roll.no	78
PRN	12320056
Batch	B3

Title: Implement one Reader - One writer classical problem using Thread & Mutex

Code:-

```
#include <iostream>
#include <pthread.h>
#include <unistd.h>

pthread_mutex_t mutex;  // Mutex to control access to the shared resource
int sharedResource = 0; // Shared resource

// Writer function
void* writer(void* arg) {
    while (true) {
        // Lock the mutex to ensure exclusive access
        pthread_mutex_lock(&mutex);

        // Writing to the shared resource
        sharedResource++;
        std::cout << "Writer: wrote " << sharedResource << std::endl;

        // Unlock the mutex after writing
        pthread_mutex_unlock(&mutex);

        // Simulate time taken to perform other operations
        sleep(1);
    }
    pthread_exit(0);
}

// Reader function
void* reader(void* arg) {
    while (true) {
        // Lock the mutex to ensure exclusive access
        pthread_mutex_lock(&mutex);

        // Reading from the shared resource
        std::cout << "Reader: read " << sharedResource << std::endl;

        // Unlock the mutex after reading
        pthread_mutex_unlock(&mutex);
```

```
        // Simulate time taken to perform other operations
        sleep(1);
    }
    pthread_exit(0);
}

int main() {
    pthread_t writerThread, readerThread;

    // Initialize the mutex
    pthread_mutex_init(&mutex, NULL);

    // Create the writer and reader threads
    pthread_create(&writerThread, NULL, writer, NULL);
    pthread_create(&readerThread, NULL, reader, NULL);

    // Wait for the threads to finish (they actually run forever)
    pthread_join(writerThread, NULL);
    pthread_join(readerThread, NULL);

    // Destroy the mutex
    pthread_mutex_destroy(&mutex);

    return 0;
}
```

Output:

```
Writer: wrote 1
Reader: read 1
Reader: read 1
Writer: wrote 2
Reader: read 2
Writer: wrote 3
Reader: read 3
Writer: wrote 4
Writer: wrote 5
Reader: read 5
Writer: wrote 6
Reader: read 6
Reader: read 6
Writer: wrote 7
Reader: read 7
Writer: wrote 8
Writer: wrote 9
Reader: read 9
Reader: read 9
Writer: wrote 10
Reader: read 10
Writer: wrote 11
Reader: read 11
Writer: wrote 12
```