```c
#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>
#define randdelay 7

int data = 5;
int reader_count = 0;
sem_t count_mutex, writer_mutex;

void reader(int id)
{
    printf("READER %d : Started\n", id);
    while(1)
    {
        sem_wait(&count_mutex);
        reader_count++;
        if(reader_count == 1)
        {
            sem_wait(&writer_mutex);
        }
        sem_post(&count_mutex);

        printf("READER %d : Data is - %d\n", id, data);

        sem_wait(&count_mutex);
        reader_count--;
        if(reader_count == 0)
        {
            sem_post(&writer_mutex);
        }
        sem_post(&count_mutex);

        sleep(rand() % randdelay);
    }
}

void writer(int id)
{
    printf("WRITER %d : Started\n", id);
    while(1)
    {
        sem_wait(&writer_mutex);

        int x = (rand() % 10) +1;
        printf("WRITER %d : Writing data %d\n", id, x);
        data = x;

        sem_post(&writer_mutex);
        sleep(rand() % randdelay);
    }
}


int main()
{
    int numThreads = 5;
    pthread_t readerThread[numThreads], writerThread[numThreads];
    int res1 = sem_init(&count_mutex, 0, 1);
    int res2 = sem_init(&writer_mutex, 0, 1);

    if(res1 != 0 || res2 != 0)
    {
        printf("Semaphore Initialization Failed\n");
        return 1;
    }

    for(int i=0;i<numThreads;i++)
    {
        res1 = pthread_create(&readerThread[i], NULL, (void *) reader, i +1);
        res2 = pthread_create(&writerThread[i], NULL, (void *) writer, i +1);

        if(res1 != 0 || res2 != 0)
        {
            printf("Thread Creation Failed\n");
            return 1;
        }
    }

    for(int i=0;i<numThreads;i++)
    {
        res1 = pthread_join(writerThread[i], NULL);
```

```
        res2 = pthread_join(writerThread[i], NULL);

        if(res1 != 0 || res2 != 0)
        {
            printf("Thread Joining Failed\n");
            return 1;
        }
    }

    res1 = sem_destroy(&count_mutex);
    res2 = sem_destroy(&writer_mutex);

    if(res1 != 0 || res2 != 0)
    {
        printf("Destroying Semaphore Failed\n");
        return 1;
    }

    return 0;
}

/* OUTPUT -
READER 2 : Started
READER 2 : Data is - 5
READER 3 : Started
READER 3 : Data is - 5
WRITER 3 : Started
READER 1 : Started
READER 4 : Started
WRITER 4 : Started
WRITER 2 : Started
WRITER 5 : Started
WRITER 3 : Writing data 8
WRITER 1 : Started
WRITER 1 : Writing data 4
READER 1 : Data is - 4
READER 4 : Data is - 4
WRITER 2 : Writing data 10
READER 5 : Started
WRITER 4 : Writing data 3
READER 5 : Data is - 3
WRITER 5 : Writing data 10
READER 2 : Data is - 10
READER 2 : Data is - 10
READER 4 : Data is - 10
WRITER 1 : Writing data 3
READER 1 : Data is - 3
READER 4 : Data is - 3
WRITER 2 : Writing data 8
READER 4 : Data is - 8
READER 3 : Data is - 8
READER 2 : Data is - 8
WRITER 3 : Writing data 4
WRITER 4 : Writing data 6
WRITER 1 : Writing data 3
WRITER 2 : Writing data 9
READER 4 : Data is - 9
WRITER 3 : Writing data 4
READER 5 : Data is - 4
READER 1 : Data is - 4
WRITER 3 : Writing data 3
WRITER 5 : Writing data 2
WRITER 5 : Writing data 5
*/
```