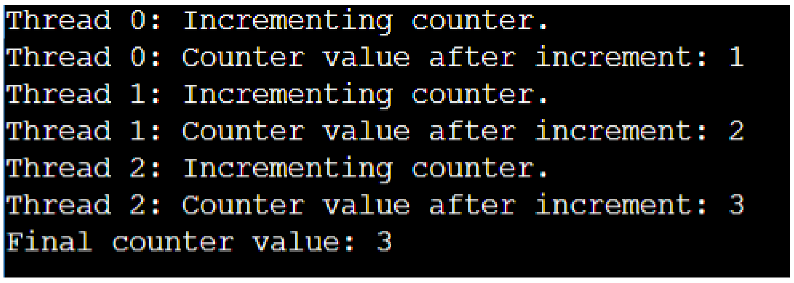
Task 7: Write a Program to Implementation of Classical problems using Threads and Semaphore (input Takes by user).

Similarly, you can replace the mutex with a semaphore. Here's a simple example:

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>

#define NUM_THREADS 3

int counter = 0;
sem_t semaphore;

void *incrementCounter(void *arg) {
   sem_wait(&semaphore);

   int thread_id = *((int *)arg);
   printf("Thread %d: Incrementing counter.\n", thread_id);

   counter++;

   printf("Thread %d: Counter value after increment: %d\n", thread_id, counter);

   sem_post(&semaphore);
   pthread_exit(NULL);
}

int main() {
   pthread_t threads[NUM_THREADS];
   int thread_ids[NUM_THREADS];
```

```
    sem_init(&semaphore, 0, 1); // Initialize semaphore with value 1

    for (int i = 0; i < NUM_THREADS; i++) {
        thread_ids[i] = i;
        pthread_create(&threads[i], NULL, incrementCounter, (void *)&thread_ids[i]);
    }

    for (int i = 0; i < NUM_THREADS; i++) {
        pthread_join(threads[i], NULL);
    }

    printf("Final counter value: %d\n", counter);

    sem_destroy(&semaphore);

    return 0;
}
```

Output-



```
Thread 0: Incrementing counter.
Thread 0: Counter value after increment: 1
Thread 1: Incrementing counter.
Thread 1: Counter value after increment: 2
Thread 2: Incrementing counter.
Thread 2: Counter value after increment: 3
Final counter value: 3
```