



ASSIGNMENT # 5

NAME : SHAHERYAR ASHFAQ

ROLL NO : **20P-0128**

SECTION : BS-CS **4-B**

SUBJECT : OPERATING SYSTEM

```
1  #include<unistd.h>
2  #include<stdio.h>
3  #include<stdlib.h>
4  #include<string.h>
5  #include<pthread.h>
6  #include<semaphore.h>
7  #include<sys/types.h>
8  #include<errno.h>
9
10 #define NUM_RUNS 10000000
11
12 #define THREAD_NUMS 2
13
14 void* handler(void *ptr);
15
16 int counter;
17
18 sem_t mutex;      // Creating semaphore
19
20 int main() {
21
22     int i[THREAD_NUMS];
23     pthread_t th[THREAD_NUMS];
24
25     sem_init(&mutex, 0, 1); // Initializing semaphore with starting value of 1
26
27     i[0] = 0;
28     i[1] = 1;
29
30     for(int j = 0; j < THREAD_NUMS; j++) {
31         if (pthread_create(&th[j], NULL, &handler, &i[j]) != 0) {
32             perror("Failed to create thread");
33         }
34     }
```

```

    for (int j = 0; j < THREAD_NUMS; j++) {
        if (pthread_join(th[j], NULL) != 0) {
            perror("Failed to join thread");
        }
    }

    printf("-----\n");
    printf("Final counter value: %d\n", counter);
    printf("Error: %d\n", (NUM_RUNS * 2 - counter));

    sem_destroy(&mutex);

    return 0;
}

void* handler(void *ptr) {

    int iter = 0;
    int *thread_num = malloc( sizeof(int) );
    thread_num = (int *) ptr;

    printf("Starting thread: %d\n", *thread_num );

    sem_wait(&mutex);

    while(iter < NUM_RUNS) {
        counter++;
        iter++;
    }

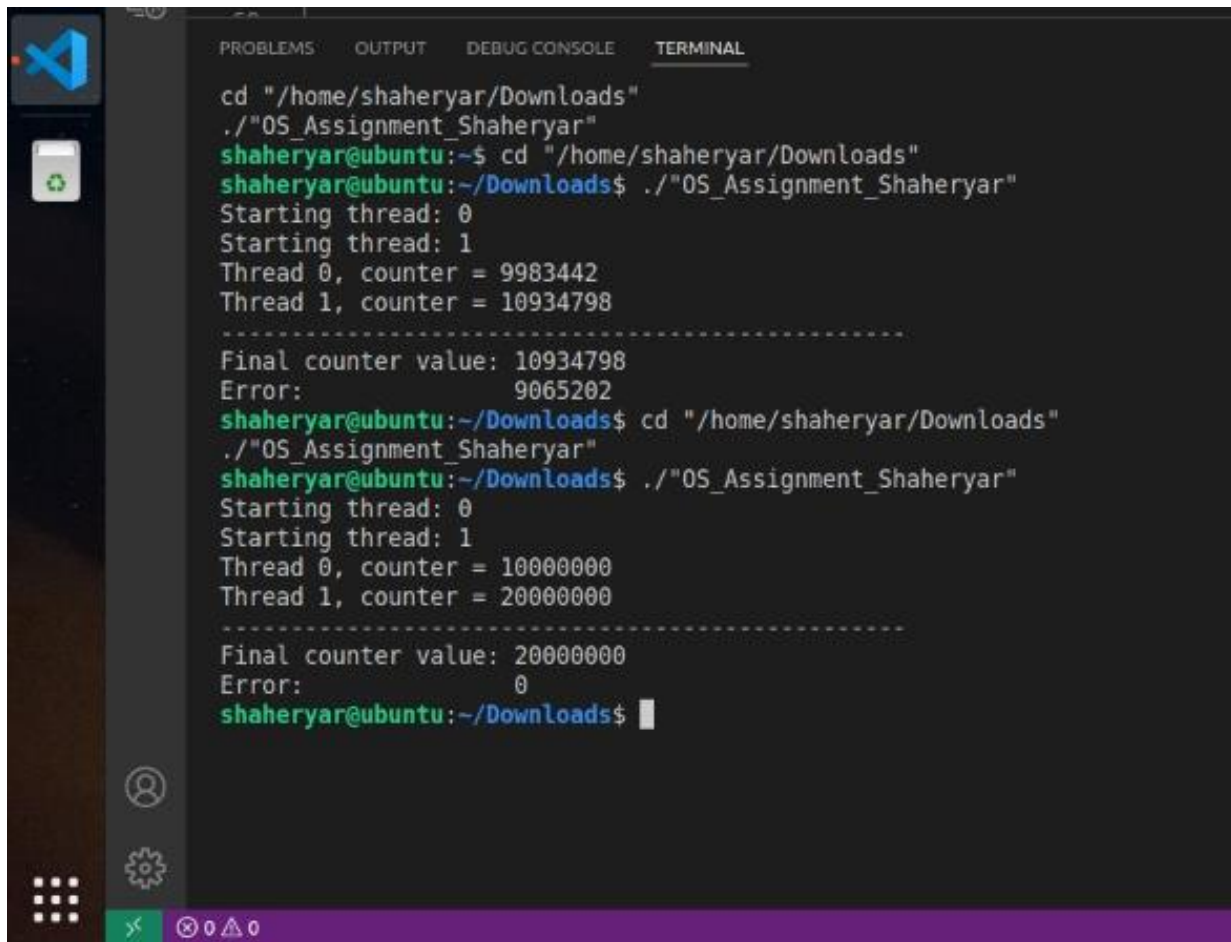
    sem_post(&mutex);

    printf("Thread %d, counter = %d\n", *thread_num, counter);

    pthread_exit(0);
}

```

OUTPUT



```
cd "/home/shaheryar/Downloads"
./"OS_Assignment_Shaheryar"
shaheryar@ubuntu:~$ cd "/home/shaheryar/Downloads"
shaheryar@ubuntu:~/Downloads$ ./"OS_Assignment_Shaheryar"
Starting thread: 0
Starting thread: 1
Thread 0, counter = 9983442
Thread 1, counter = 10934798
-----
Final counter value: 10934798
Error: 9065202
shaheryar@ubuntu:~/Downloads$ cd "/home/shaheryar/Downloads"
shaheryar@ubuntu:~/Downloads$ ./"OS_Assignment_Shaheryar"
Starting thread: 0
Starting thread: 1
Thread 0, counter = 10000000
Thread 1, counter = 20000000
-----
Final counter value: 20000000
Error: 0
shaheryar@ubuntu:~/Downloads$
```

Explanation

Locks are one synchronization technique. A lock is an abstraction that allows at most one thread to *own* it at a time. *Holding a lock* is how one thread tells other threads: "I'm changing this thing, don't touch it right now."