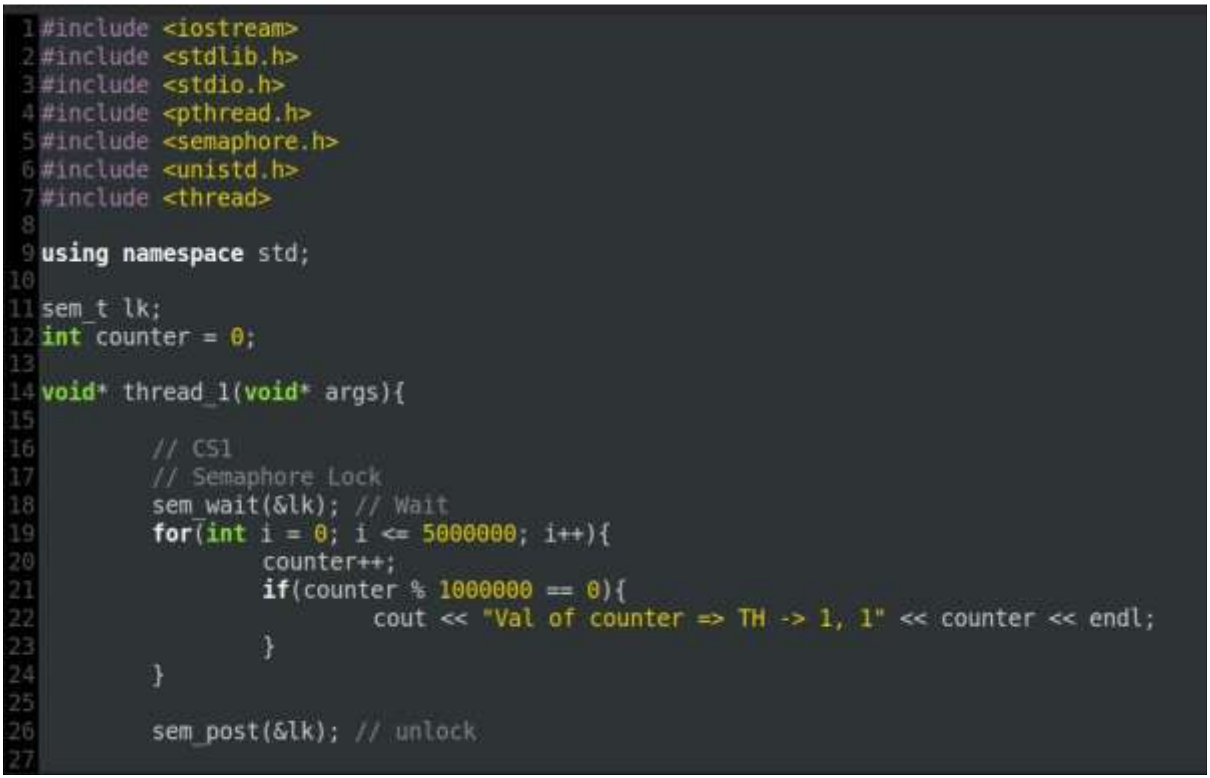
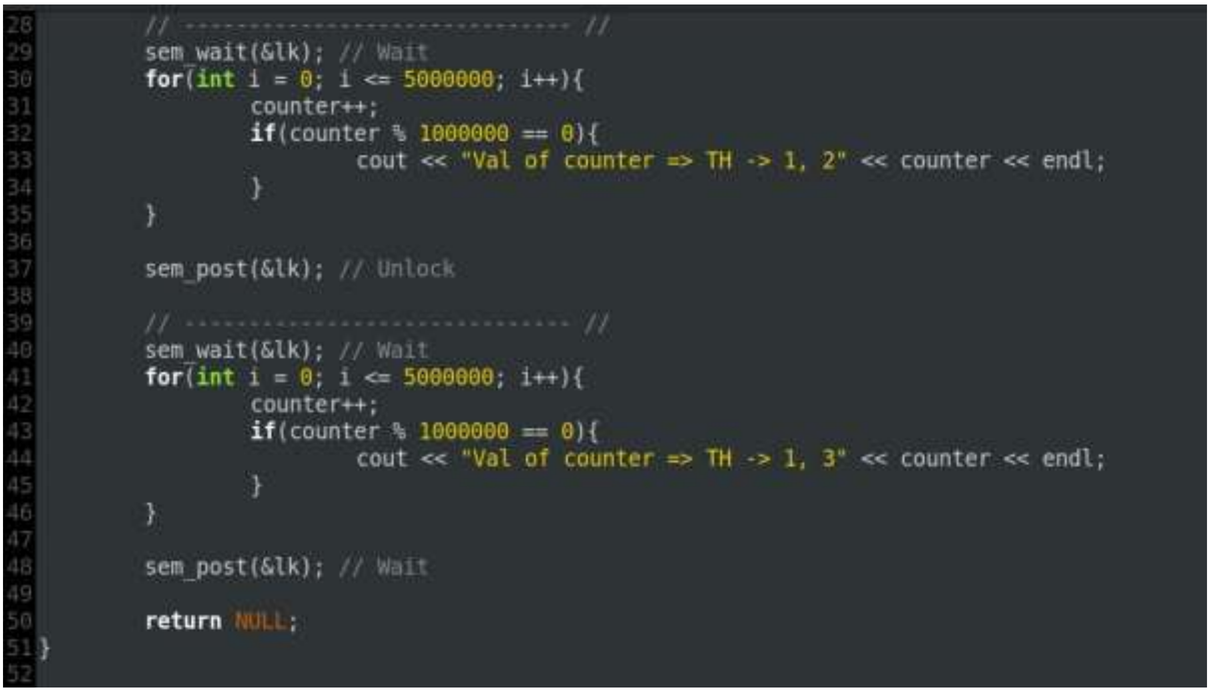
Synchronization: Two-Process Solutions, MUTEX, and Semaphore

Task#1

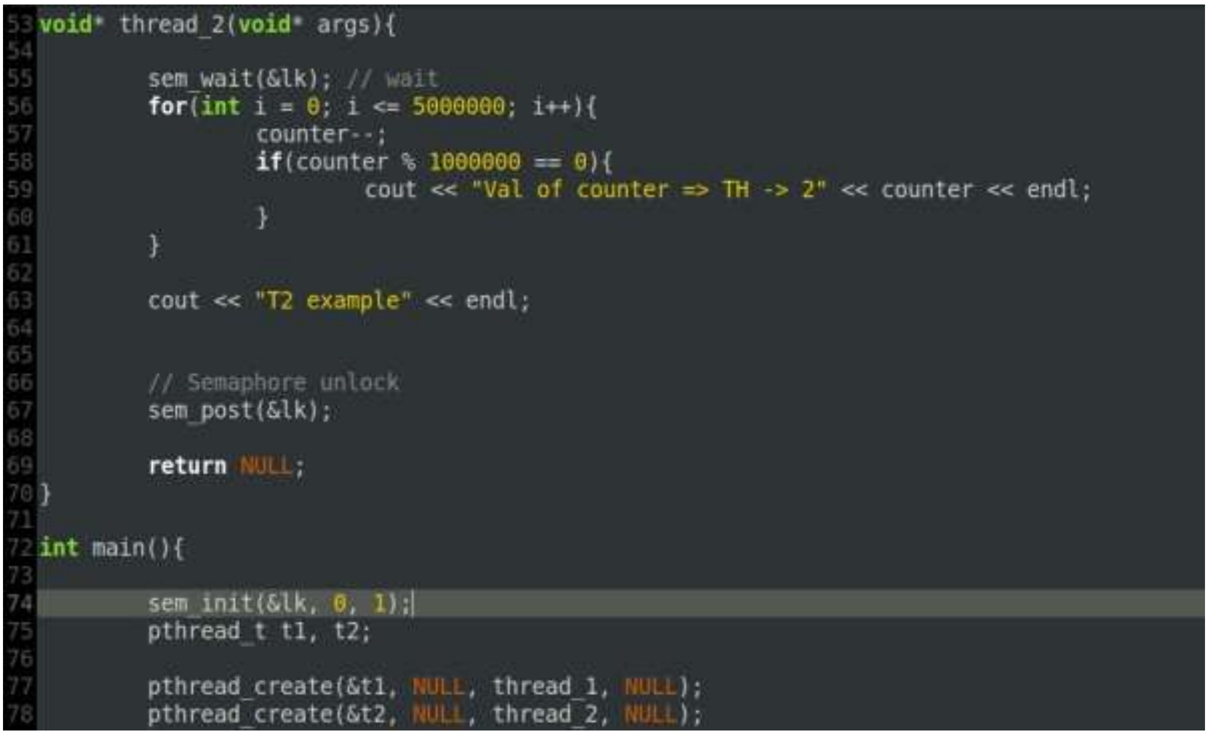
Solve CS problem using semaphore.



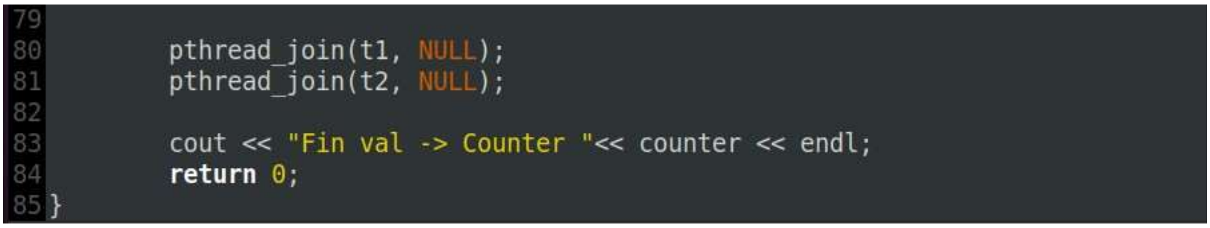
```
#include <iostream>
#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
#include <thread>

using namespace std;

sem_t lk;
int counter = 0;

void* thread_1(void* args){

        // CS1
        // Semaphore Lock
        sem_wait(&lk); // Wait
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 1, 1" << counter << endl;
                }
        }

        sem_post(&lk); // unlock

```



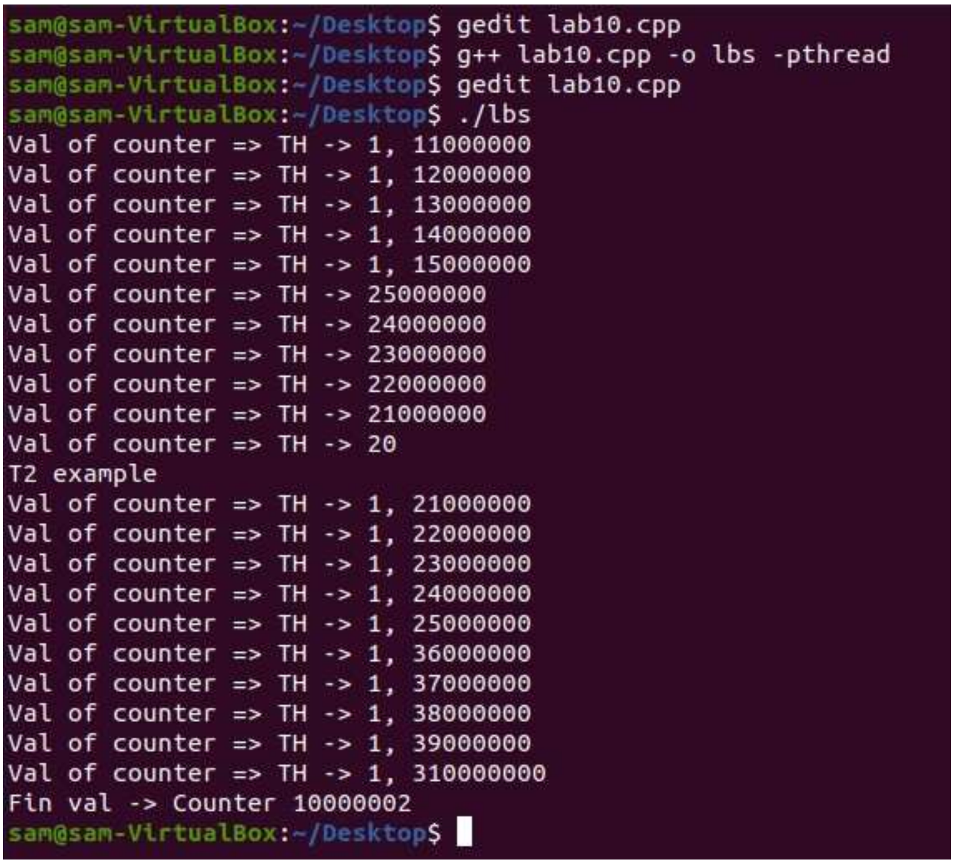
```
        // ------------------------------ //
        sem_wait(&lk); // Wait
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 1, 2" << counter << endl;
                }
        }

        sem_post(&lk); // Unlock

        // ------------------------------ //
        sem_wait(&lk); // Wait
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 1, 3" << counter << endl;
                }
        }

        sem_post(&lk); // Wait

        return NULL;
}

```



```cpp
void* thread_2(void* args){

        sem_wait(&lk); // wait
        for(int i = 0; i <= 5000000; i++){
                counter--;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 2" << counter << endl;
                }
        }

        cout << "T2 example" << endl;


        // Semaphore unlock
        sem_post(&lk);

        return NULL;
}

int main(){

        sem_init(&lk, 0, 1);
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread_1, NULL);
        pthread_create(&t2, NULL, thread_2, NULL);
```



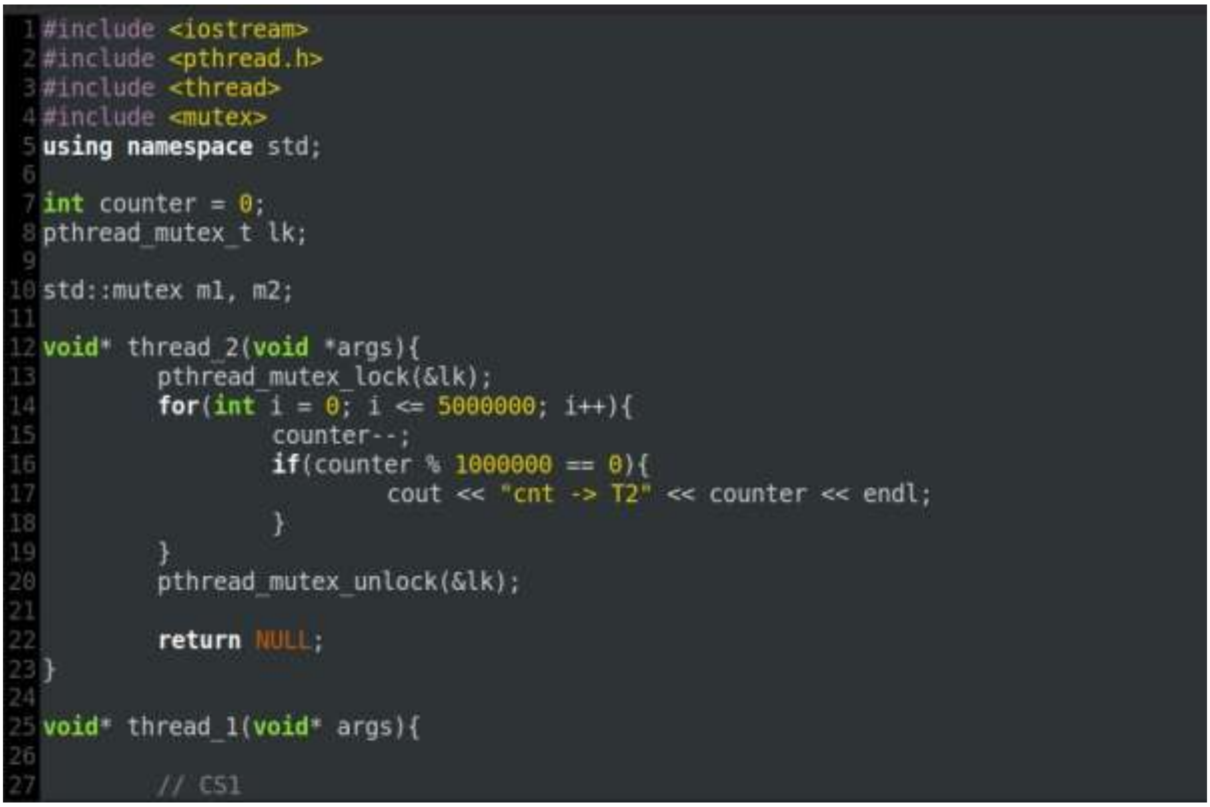
```cpp

        pthread_join(t1, NULL);
        pthread_join(t2, NULL);

        cout << "Fin val -> Counter "<< counter << endl;
        return 0;
}
```



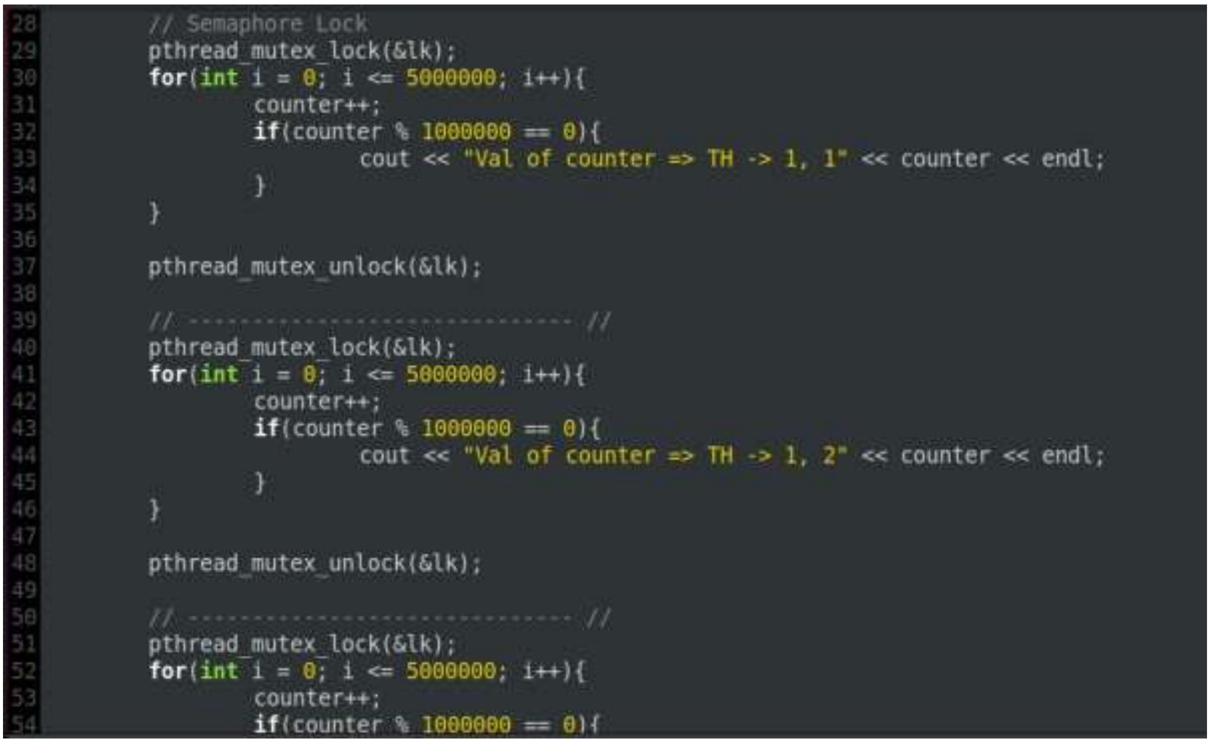
```
sam@sam-VirtualBox:~/Desktop$ gedit lab10.cpp
sam@sam-VirtualBox:~/Desktop$ g++ lab10.cpp -o lbs -pthread
sam@sam-VirtualBox:~/Desktop$ gedit lab10.cpp
sam@sam-VirtualBox:~/Desktop$ ./lbs
Val of counter => TH -> 1, 11000000
Val of counter => TH -> 1, 12000000
Val of counter => TH -> 1, 13000000
Val of counter => TH -> 1, 14000000
Val of counter => TH -> 1, 15000000
Val of counter => TH -> 25000000
Val of counter => TH -> 24000000
Val of counter => TH -> 23000000
Val of counter => TH -> 22000000
Val of counter => TH -> 21000000
Val of counter => TH -> 20
T2 example
Val of counter => TH -> 1, 21000000
Val of counter => TH -> 1, 22000000
Val of counter => TH -> 1, 23000000
Val of counter => TH -> 1, 24000000
Val of counter => TH -> 1, 25000000
Val of counter => TH -> 1, 36000000
Val of counter => TH -> 1, 37000000
Val of counter => TH -> 1, 38000000
Val of counter => TH -> 1, 39000000
Val of counter => TH -> 1, 310000000
Fin val -> Counter 10000002
sam@sam-VirtualBox:~/Desktop$
```

Task#2

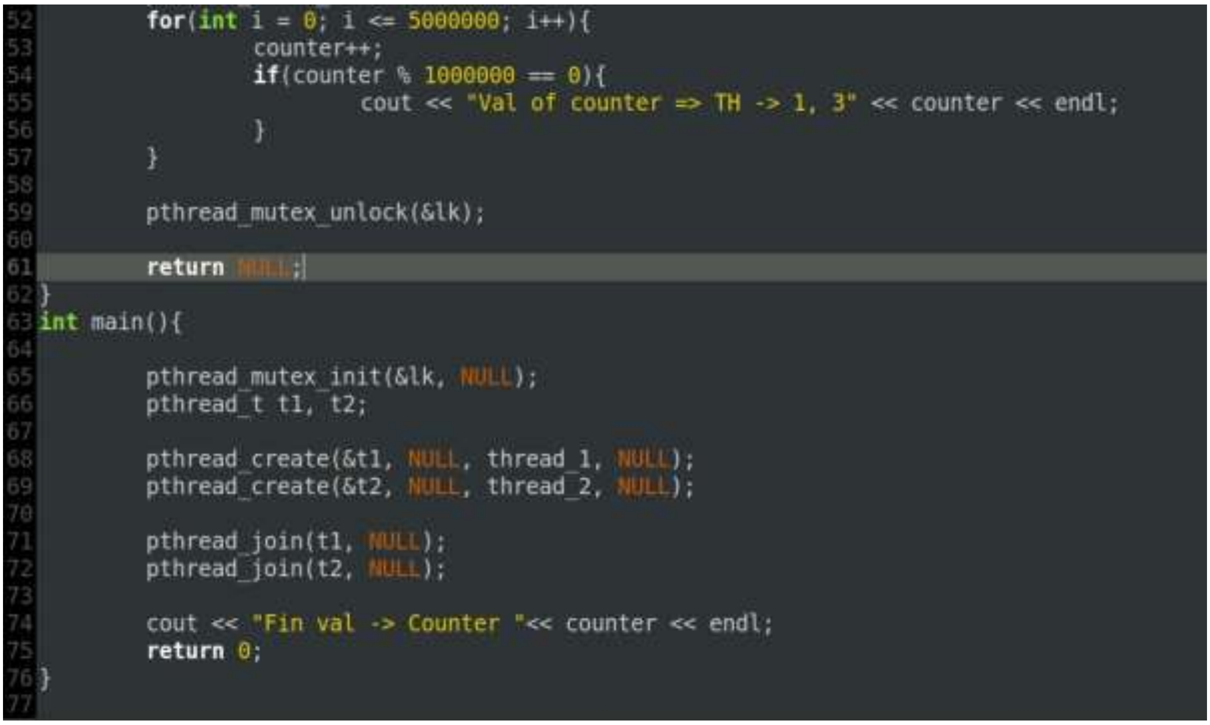
Solve CS problem using MUTEX.



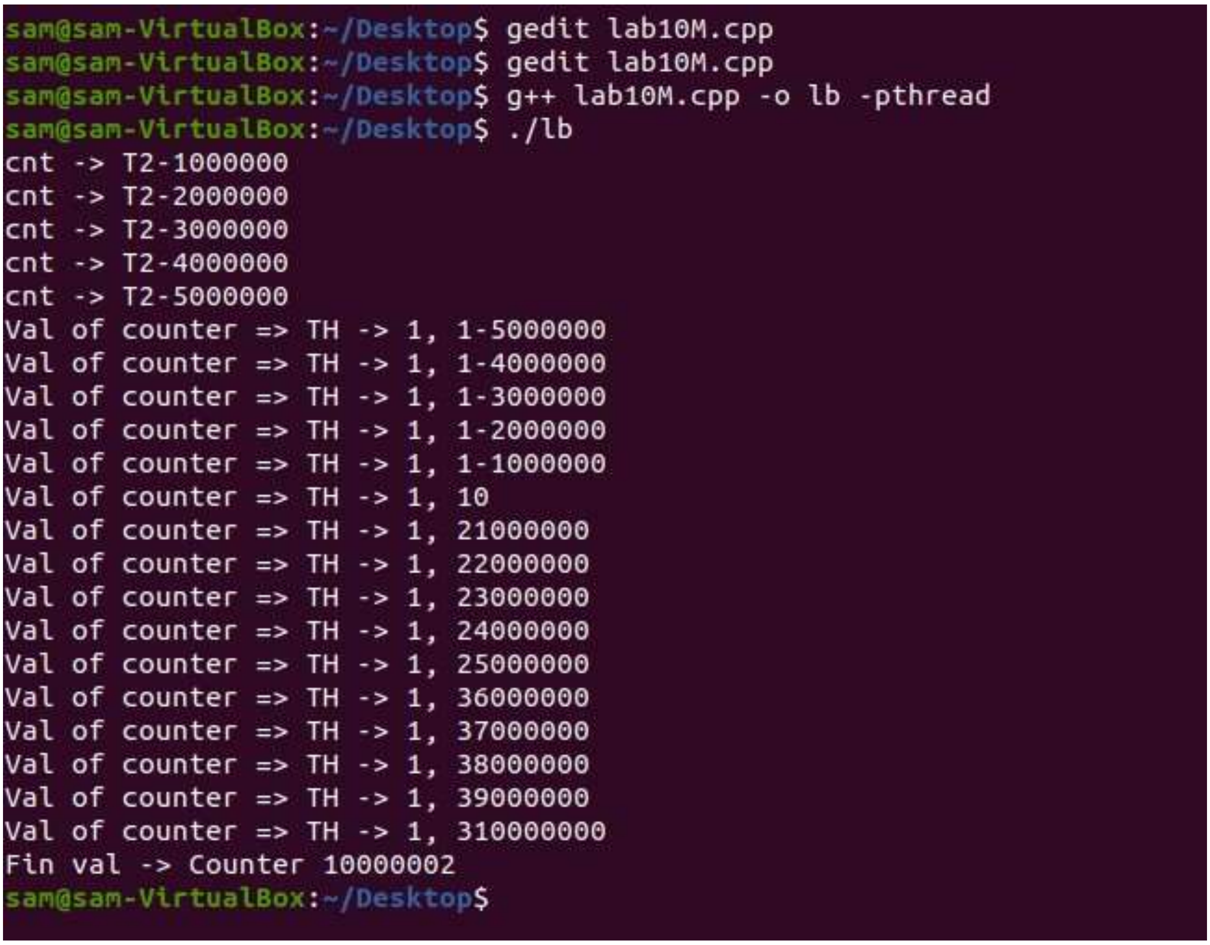
```
#include <iostream>
#include <pthread.h>
#include <thread>
#include <mutex>
using namespace std;

int counter = 0;
pthread_mutex_t lk;

std::mutex m1, m2;

void* thread_2(void *args){
        pthread_mutex_lock(&lk);
        for(int i = 0; i <= 5000000; i++){
                counter--;
                if(counter % 1000000 == 0){
                        cout << "cnt -> T2" << counter << endl;
                }
        }
        pthread_mutex_unlock(&lk);

        return NULL;
}

void* thread_1(void* args){

        // CS1
```



```
        // Semaphore Lock
        pthread_mutex_lock(&lk);
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 1, 1" << counter << endl;
                }
        }

        pthread_mutex_unlock(&lk);

        // ----------------------------- //
        pthread_mutex_lock(&lk);
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 1, 2" << counter << endl;
                }
        }

        pthread_mutex_unlock(&lk);

        // ----------------------------- //
        pthread_mutex_lock(&lk);
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
```



```
        for(int i = 0; i <= 5000000; i++){
                counter++;
                if(counter % 1000000 == 0){
                        cout << "Val of counter => TH -> 1, 3" << counter << endl;
                }
        }

        pthread_mutex_unlock(&lk);

        return NULL;
}
int main(){

        pthread_mutex_init(&lk, NULL);
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread_1, NULL);
        pthread_create(&t2, NULL, thread_2, NULL);

        pthread_join(t1, NULL);
        pthread_join(t2, NULL);

        cout << "Fin val -> Counter "<< counter << endl;
        return 0;
}
```



```
sam@sam-VirtualBox:~/Desktop$ gedit lab10M.cpp
sam@sam-VirtualBox:~/Desktop$ gedit lab10M.cpp
sam@sam-VirtualBox:~/Desktop$ g++ lab10M.cpp -o lb -pthread
sam@sam-VirtualBox:~/Desktop$ ./lb
cnt -> T2-1000000
cnt -> T2-2000000
cnt -> T2-3000000
cnt -> T2-4000000
cnt -> T2-5000000
Val of counter => TH -> 1, 1-5000000
Val of counter => TH -> 1, 1-4000000
Val of counter => TH -> 1, 1-3000000
Val of counter => TH -> 1, 1-2000000
Val of counter => TH -> 1, 1-1000000
Val of counter => TH -> 1, 10
Val of counter => TH -> 1, 21000000
Val of counter => TH -> 1, 22000000
Val of counter => TH -> 1, 23000000
Val of counter => TH -> 1, 24000000
Val of counter => TH -> 1, 25000000
Val of counter => TH -> 1, 36000000
Val of counter => TH -> 1, 37000000
Val of counter => TH -> 1, 38000000
Val of counter => TH -> 1, 39000000
Val of counter => TH -> 1, 310000000
Fin val -> Counter 10000002
sam@sam-VirtualBox:~/Desktop$
```