

Task 1

```
#include <iostream>
#include <thread>
#include <mutex>
using namespace std;
int globalVariable = 0;
int result = 0;
mutex mtx;
void threadFunction(int id)
{
    for (int i = 0; i < 100000; ++i) {
        mtx.lock();
        globalVariable++;
        result += globalVariable;
        mtx.unlock();
    }
}

int main() {
    thread t1(threadFunction, 1);
    thread t2(threadFunction, 2);
    thread t3(threadFunction, 3);

    t1.join();
    t2.join();
    t3.join();
    cout << "Final value of globalVariable: " << globalVariable << endl;
    cout << "Final value of result: " << result << endl;

    return 0;
}
```

```
vboxuser@Ubuntu: ~/Desktop
vboxuser@Ubuntu:~/Desktop$ g++ -o 1 1.cc
vboxuser@Ubuntu:~/Desktop$ ./1
Final value of globalVariable: 300000
Final value of result: 2050477040
vboxuser@Ubuntu:~/Desktop$
```

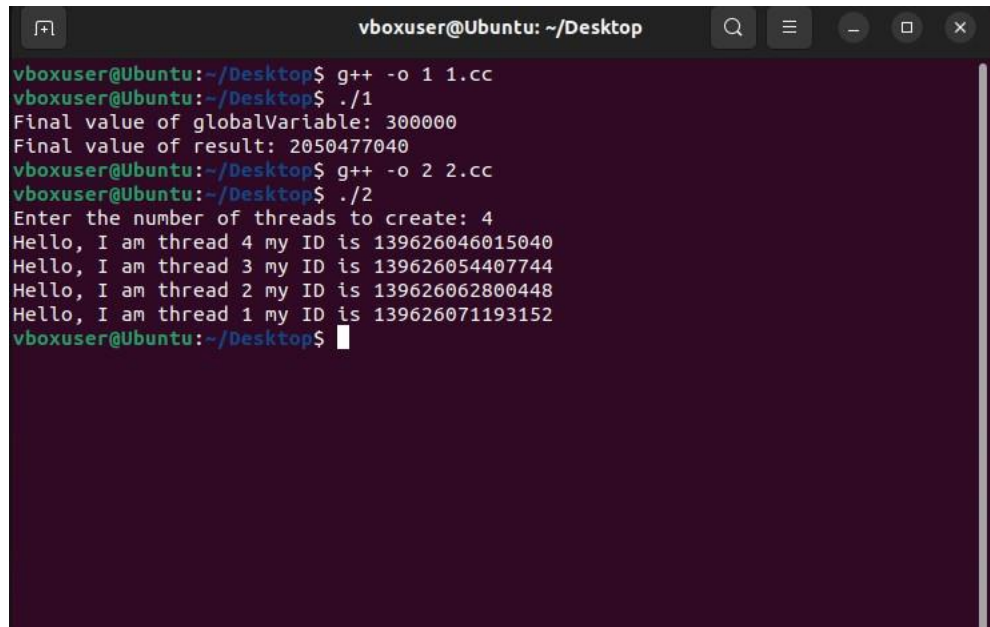
Task 2:

```
#include <iostream>
#include <pthread.h>
using namespace std;
void *print(void *arg)
{
    pthread_t id = pthread_self();
    cout << "Hello, I am thread " << *((int *)arg) << " my ID is " << id << endl;
    pthread_exit(NULL);
}

int main()
{
    int numThreads;
    cout << "Enter the number of threads to create: ";
    cin >> numThreads;

    pthread_t threads[numThreads];
    int threadArgs[numThreads];
    for (int i = 0; i < numThreads; ++i)
    {
        threadArgs[i] = i + 1;
        pthread_create(&threads[i], NULL, print, (void *)&threadArgs[i]);
    }
    for (int i = 0; i < numThreads; ++i)
```

```
{  
    pthread_join(threads[i], NULL);  
}  
  
exit(0);  
}
```



```
vboxuser@Ubuntu: ~/Desktop  
vboxuser@Ubuntu:~/Desktop$ g++ -o 1 1.cc  
vboxuser@Ubuntu:~/Desktop$ ./1  
Final value of globalVariable: 300000  
Final value of result: 2050477040  
vboxuser@Ubuntu:~/Desktop$ g++ -o 2 2.cc  
vboxuser@Ubuntu:~/Desktop$ ./2  
Enter the number of threads to create: 4  
Hello, I am thread 4 my ID is 139626046015040  
Hello, I am thread 3 my ID is 139626054407744  
Hello, I am thread 2 my ID is 139626062800448  
Hello, I am thread 1 my ID is 139626071193152  
vboxuser@Ubuntu:~/Desktop$
```

Task3:

```
#include <iostream>  
#include <pthread.h>  
using namespace std;  
void *task1(void *)  
{  
    cout << "Thread 1: Performing task 1" << endl;  
    pthread_exit(NULL);  
}  
  
void *task2(void *)  
{  
    cout << "Thread 2: Performing task 2" << endl;
```

```
    pthread_exit(NULL);
}

void *task3(void *)
{
    cout << "Thread 3: Performing task 3" << endl;
    pthread_exit(NULL);
}

void *task4(void *)
{
    cout << "Thread 4: Performing task 4" << endl;
    pthread_exit(NULL);
}

int main()
{
    pthread_t thread1, thread2, thread3, thread4;
    pthread_create(&thread1, NULL, task1, NULL);
    pthread_create(&thread2, NULL, task2, NULL);
    pthread_create(&thread3, NULL, task3, NULL);
    pthread_create(&thread4, NULL, task4, NULL);
    pthread_join(thread1, NULL);
    pthread_join(thread2, NULL);
    pthread_join(thread3, NULL);
    pthread_join(thread4, NULL);

    cout << "All threads have finished their tasks." << endl;

    return 0;
}
```

```
vboxuser@Ubuntu: ~/Desktop
vboxuser@Ubuntu:~/Desktop$ g++ -o 3 3.cc
vboxuser@Ubuntu:~/Desktop$ ./3
Thread 4: Performing task 4
Thread 3: Performing task 3
Thread 2: Performing task 2
Thread 1: Performing task 1
All threads have finished their tasks.
vboxuser@Ubuntu:~/Desktop$
```

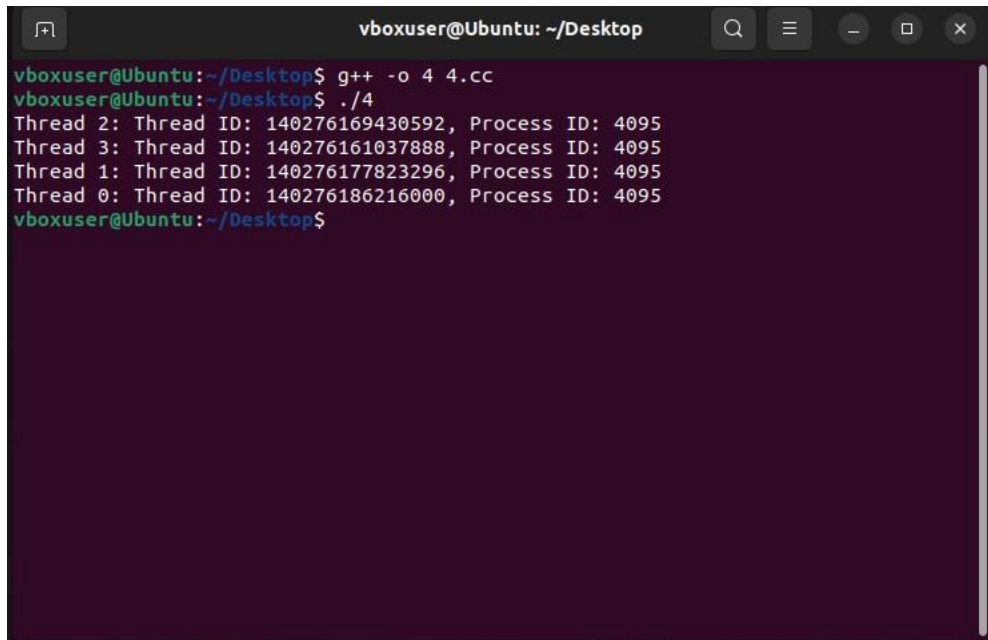
Task4:

```
#include <iostream>
#include <pthread.h>
#include <unistd.h>
using namespace std;
void *threadFunction(void *arg)
{
    long thread_id = (long)arg;
    pid_t process_id = getpid();
    cout << "Thread " << thread_id << ": Thread ID: " << pthread_self() << ",
Process ID: " << process_id << endl;
    pthread_exit(NULL);
}

int main()
{
    const int numThreads = 4;
    pthread_t threads[numThreads];
    for (long i = 0; i < numThreads; ++i)
    {
        pthread_create(&threads[i], NULL, threadFunction, (void *)i);
    }

    for (int i = 0; i < numThreads; ++i)
```

```
{  
    pthread_join(threads[i], NULL);  
}  
  
return 0;  
}
```



A terminal window titled "vboxuser@Ubuntu: ~/Desktop" with standard window controls. The terminal shows the following commands and output:

```
vboxuser@Ubuntu:~/Desktop$ g++ -o 4 4.cc  
vboxuser@Ubuntu:~/Desktop$ ./4  
Thread 2: Thread ID: 140276169430592, Process ID: 4095  
Thread 3: Thread ID: 140276161037888, Process ID: 4095  
Thread 1: Thread ID: 140276177823296, Process ID: 4095  
Thread 0: Thread ID: 140276186216000, Process ID: 4095  
vboxuser@Ubuntu:~/Desktop$
```