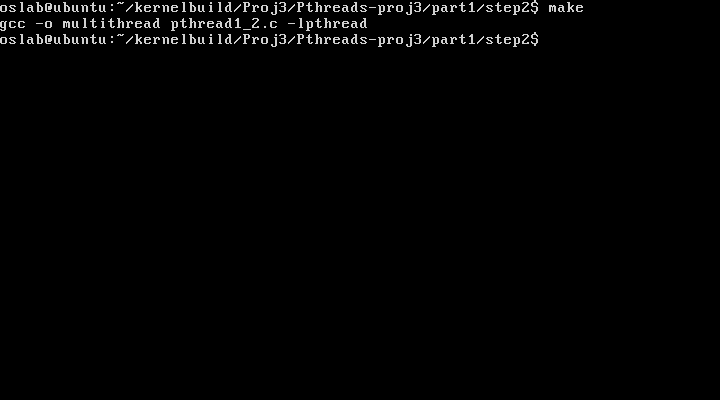
Just like step 1 of in this assignment we were assigned the task of creating a multi-threaded assignment that would use a shared variable amongst each thread. This shared variable stored a random integer. The final output of the program is a combination of the threads ID and the value that the current thread is seeing. This program does contain thread locking with barriers and as such the threads can be organized to report all assignments of one kind first before executing final actions such as printing the final values.

Barrier synchronization can be utilized in situations where a number of activities must be accomplished before an overall job can be finished. Barrier functions and a synchronization object called a barrier are defined in POSIX threads. The functions start up threads to do tasks and wait at the barrier until all threads reach the barrier, providing the number of threads that are synchronizing on the barrier.

All threads restart execution when the final thread reaches the barrier.

Thread synchronization is a method that assures that two or more concurrent processes or threads do not execute the same program segment, known as a crucial section, at the same time. Synchronization strategies are used to control the access of processes to crucial sections. When one thread begins executing the important section (a serialized part of the program), the second thread should wait until the first thread completes its execution.

**Make:**



**OUTPUT:**

