Johnny LLerena
ID: 6279633
11/24/21

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.

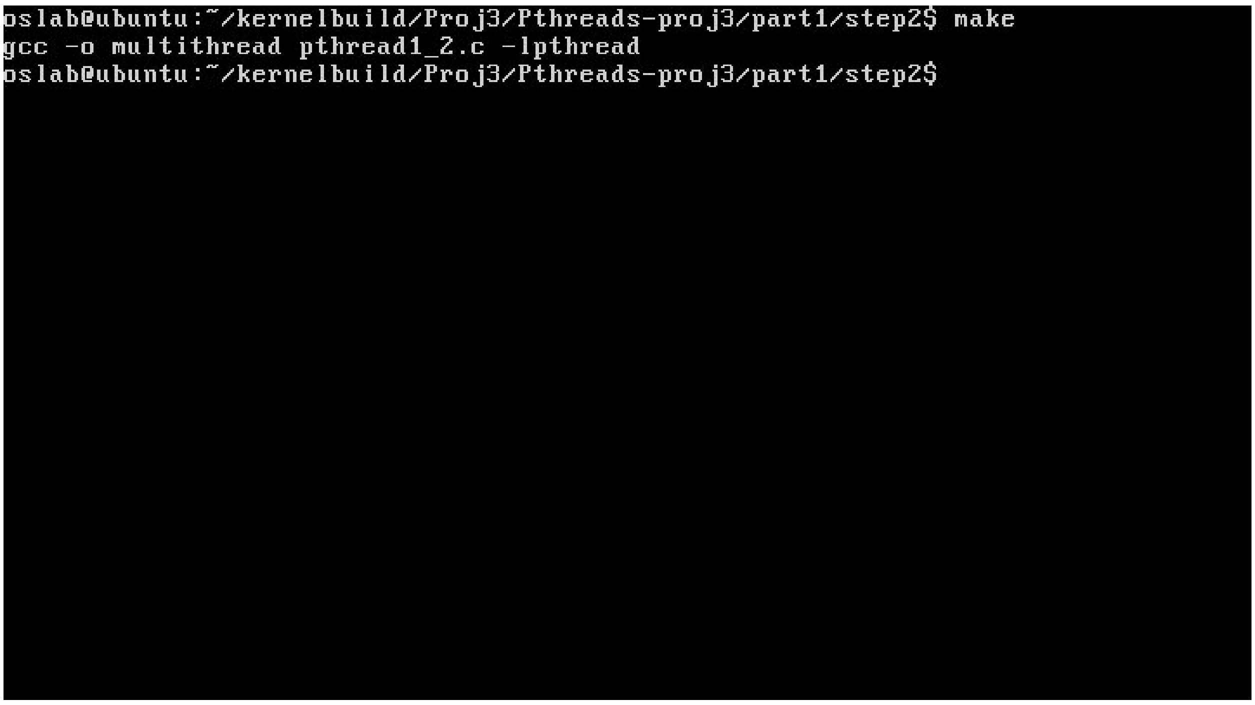
Barrier synchronization can be utilized in situations where a number of activities must be accomplished before an overall job can be finished. Barrier functions and a synchronization object called a barrier are defined in POSIX threads. The functions start up threads to do tasks and wait at the barrier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

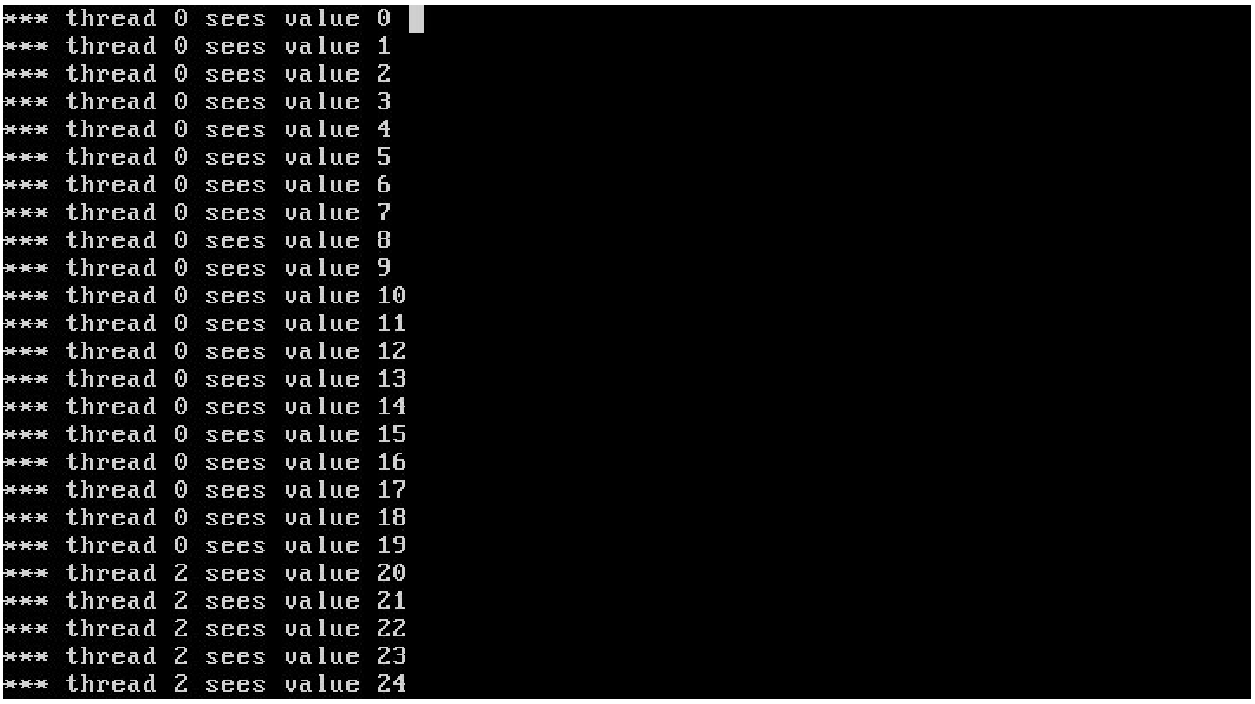
Johnny LLerena
ID: 6279633
11/24/21

Make:



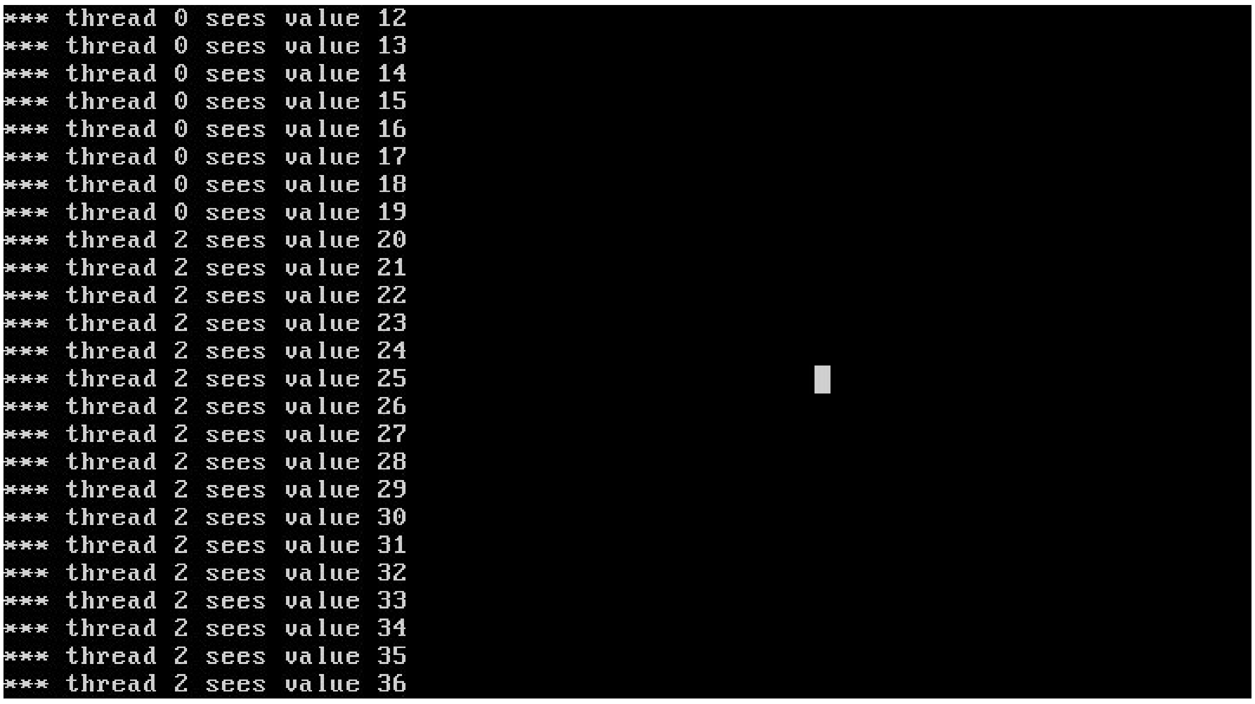
```
oslab@ubuntu:~/kernelbuild/Proj3/Pthreads-proj3/part1/step2$ make
gcc -o multithread pthread1_2.c -lpthread
oslab@ubuntu:~/kernelbuild/Proj3/Pthreads-proj3/part1/step2$
```

OUTPUT:

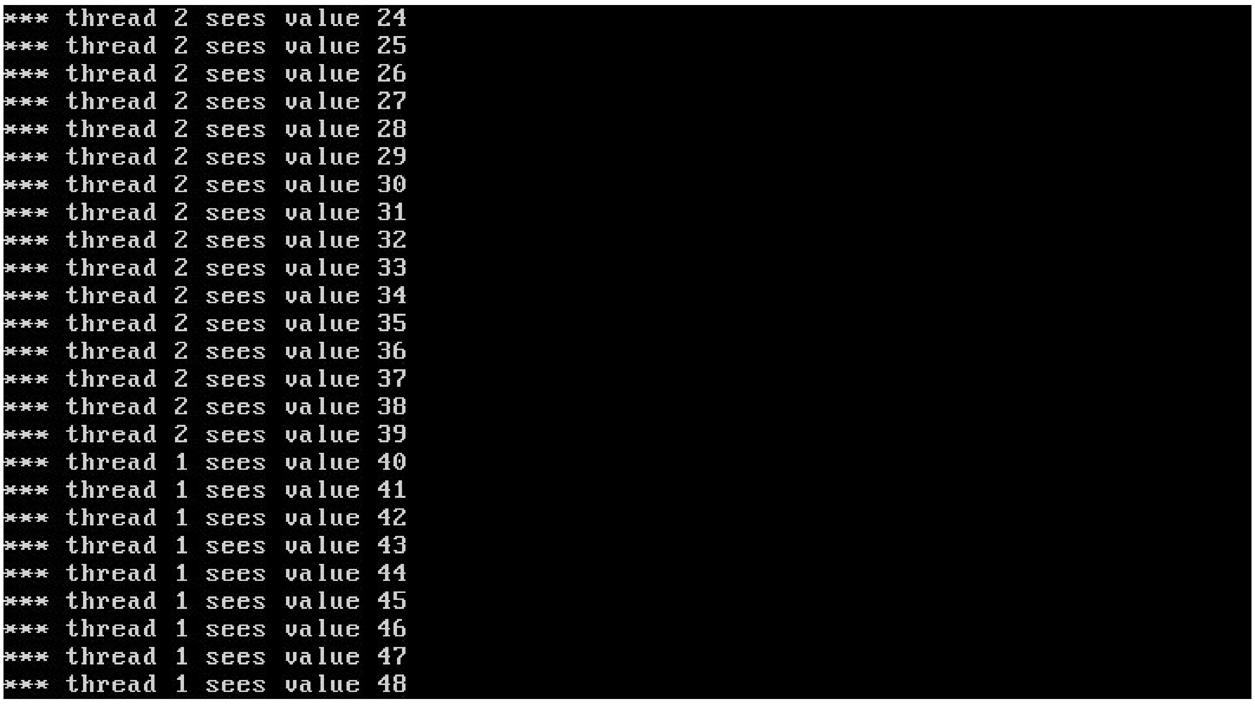


```
*** thread 0 sees value 0
*** thread 0 sees value 1
*** thread 0 sees value 2
*** thread 0 sees value 3
*** thread 0 sees value 4
*** thread 0 sees value 5
*** thread 0 sees value 6
*** thread 0 sees value 7
*** thread 0 sees value 8
*** thread 0 sees value 9
*** thread 0 sees value 10
*** thread 0 sees value 11
*** thread 0 sees value 12
*** thread 0 sees value 13
*** thread 0 sees value 14
*** thread 0 sees value 15
*** thread 0 sees value 16
*** thread 0 sees value 17
*** thread 0 sees value 18
*** thread 0 sees value 19
*** thread 2 sees value 20
*** thread 2 sees value 21
*** thread 2 sees value 22
*** thread 2 sees value 23
*** thread 2 sees value 24
```

Johnny LLerena
ID: 6279633
11/24/21

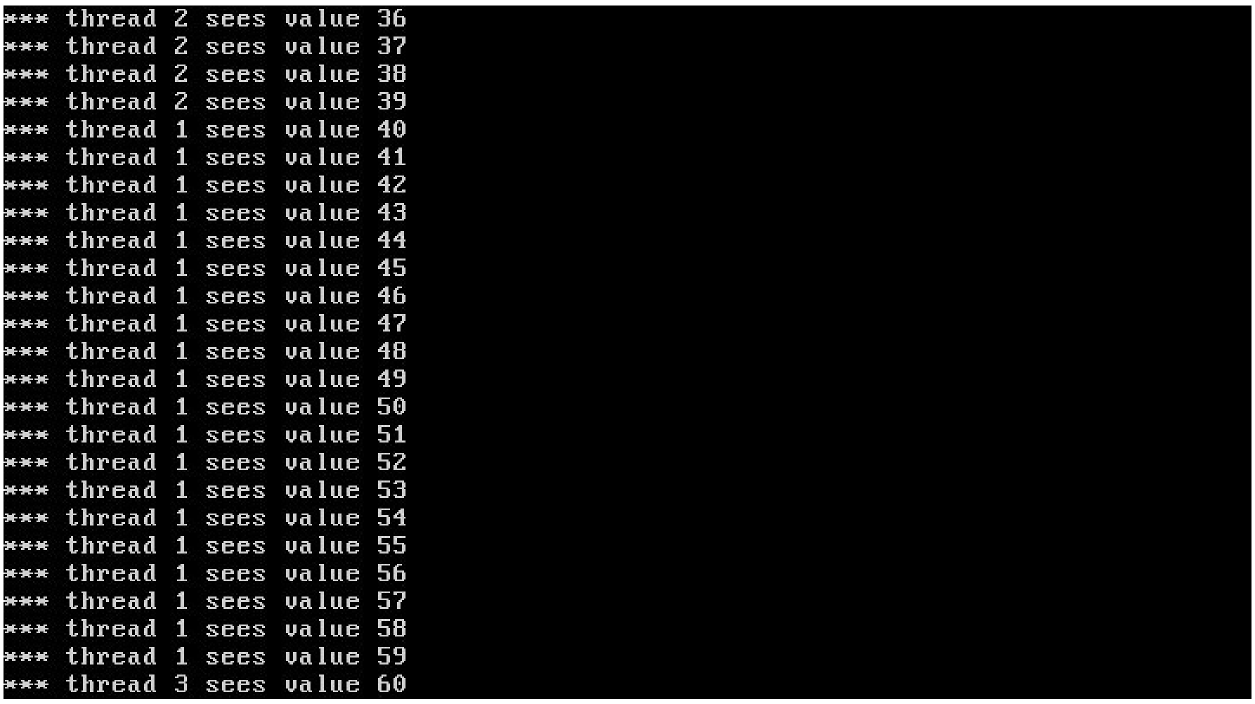


```
*** thread 0 sees value 12
*** thread 0 sees value 13
*** thread 0 sees value 14
*** thread 0 sees value 15
*** thread 0 sees value 16
*** thread 0 sees value 17
*** thread 0 sees value 18
*** thread 0 sees value 19
*** thread 2 sees value 20
*** thread 2 sees value 21
*** thread 2 sees value 22
*** thread 2 sees value 23
*** thread 2 sees value 24
*** thread 2 sees value 25
*** thread 2 sees value 26
*** thread 2 sees value 27
*** thread 2 sees value 28
*** thread 2 sees value 29
*** thread 2 sees value 30
*** thread 2 sees value 31
*** thread 2 sees value 32
*** thread 2 sees value 33
*** thread 2 sees value 34
*** thread 2 sees value 35
*** thread 2 sees value 36
```

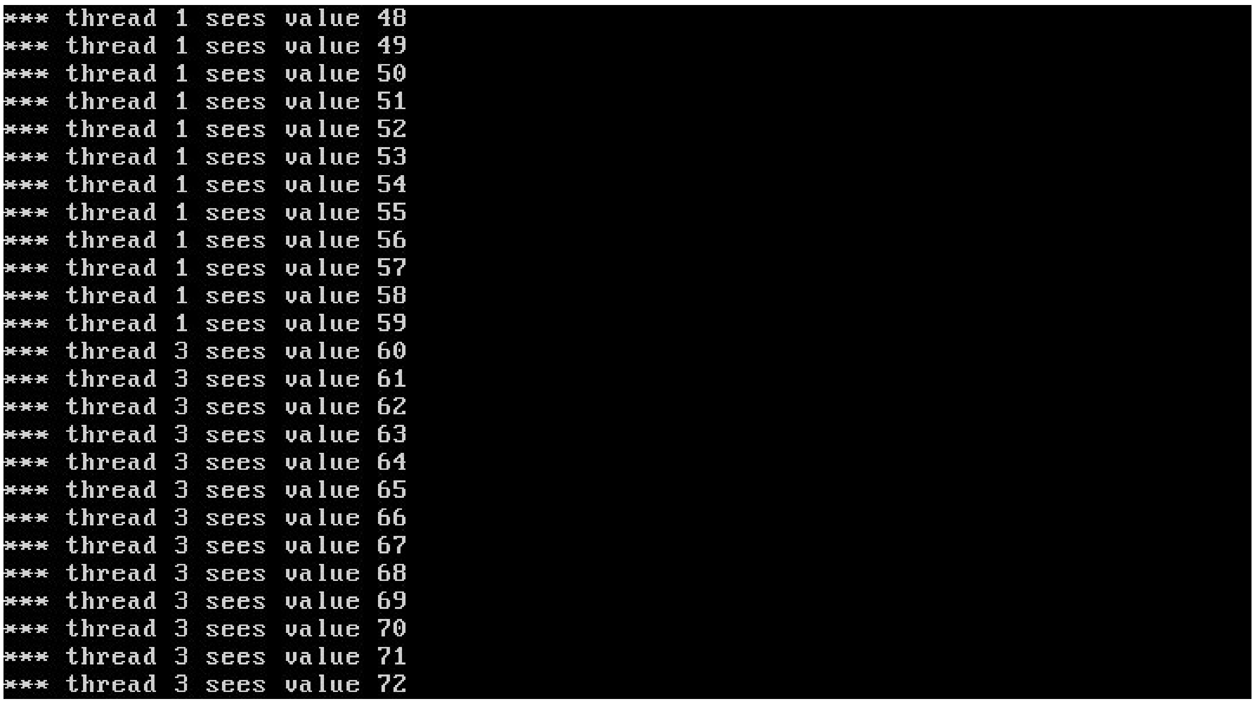


```
*** thread 2 sees value 24
*** thread 2 sees value 25
*** thread 2 sees value 26
*** thread 2 sees value 27
*** thread 2 sees value 28
*** thread 2 sees value 29
*** thread 2 sees value 30
*** thread 2 sees value 31
*** thread 2 sees value 32
*** thread 2 sees value 33
*** thread 2 sees value 34
*** thread 2 sees value 35
*** thread 2 sees value 36
*** thread 2 sees value 37
*** thread 2 sees value 38
*** thread 2 sees value 39
*** thread 1 sees value 40
*** thread 1 sees value 41
*** thread 1 sees value 42
*** thread 1 sees value 43
*** thread 1 sees value 44
*** thread 1 sees value 45
*** thread 1 sees value 46
*** thread 1 sees value 47
*** thread 1 sees value 48
```

Johnny LLerena
ID: 6279633
11/24/21

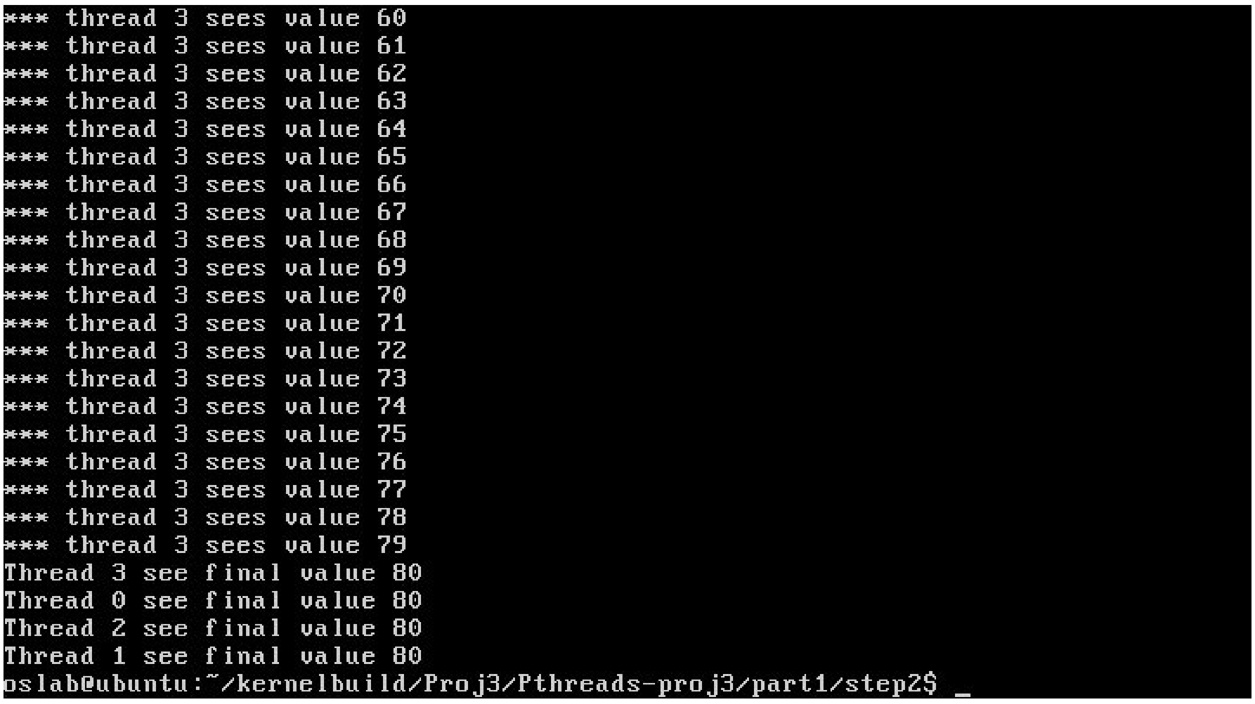


```
*** thread 2 sees value 36
*** thread 2 sees value 37
*** thread 2 sees value 38
*** thread 2 sees value 39
*** thread 1 sees value 40
*** thread 1 sees value 41
*** thread 1 sees value 42
*** thread 1 sees value 43
*** thread 1 sees value 44
*** thread 1 sees value 45
*** thread 1 sees value 46
*** thread 1 sees value 47
*** thread 1 sees value 48
*** thread 1 sees value 49
*** thread 1 sees value 50
*** thread 1 sees value 51
*** thread 1 sees value 52
*** thread 1 sees value 53
*** thread 1 sees value 54
*** thread 1 sees value 55
*** thread 1 sees value 56
*** thread 1 sees value 57
*** thread 1 sees value 58
*** thread 1 sees value 59
*** thread 3 sees value 60
```



```
*** thread 1 sees value 48
*** thread 1 sees value 49
*** thread 1 sees value 50
*** thread 1 sees value 51
*** thread 1 sees value 52
*** thread 1 sees value 53
*** thread 1 sees value 54
*** thread 1 sees value 55
*** thread 1 sees value 56
*** thread 1 sees value 57
*** thread 1 sees value 58
*** thread 1 sees value 59
*** thread 3 sees value 60
*** thread 3 sees value 61
*** thread 3 sees value 62
*** thread 3 sees value 63
*** thread 3 sees value 64
*** thread 3 sees value 65
*** thread 3 sees value 66
*** thread 3 sees value 67
*** thread 3 sees value 68
*** thread 3 sees value 69
*** thread 3 sees value 70
*** thread 3 sees value 71
*** thread 3 sees value 72
```

Johnny LLerena
ID: 6279633
11/24/21



```
*** thread 3 sees value 60
*** thread 3 sees value 61
*** thread 3 sees value 62
*** thread 3 sees value 63
*** thread 3 sees value 64
*** thread 3 sees value 65
*** thread 3 sees value 66
*** thread 3 sees value 67
*** thread 3 sees value 68
*** thread 3 sees value 69
*** thread 3 sees value 70
*** thread 3 sees value 71
*** thread 3 sees value 72
*** thread 3 sees value 73
*** thread 3 sees value 74
*** thread 3 sees value 75
*** thread 3 sees value 76
*** thread 3 sees value 77
*** thread 3 sees value 78
*** thread 3 sees value 79
Thread 3 see final value 80
Thread 0 see final value 80
Thread 2 see final value 80
Thread 1 see final value 80
oslab@ubuntu:~/kernelbuild/Proj3/Pthreads-proj3/part1/step2$ _
```