

Project 1 Report: Multithreaded Programming and Synchronization

CECS 326 - Operating Systems

Group members: Jorel Caoile
Len Quach

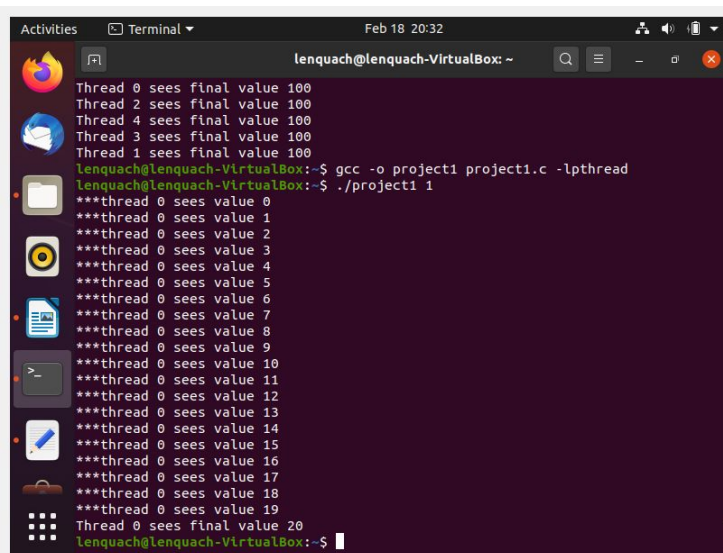
Abstract

The project focuses on process management while using the pthreads library in Linux. This part of Linux allows multiple concurrent activities within the kernel. This project has two parts: creating multiple threads without synchronization and creating multiple threads with synchronization. There is a problem when performing multithreaded programming without synchronization. The final result, once the program has finished executing, has it so that the threads will not have the same final values. Synchronization solves that problem.

1. Part 1: Simple Multi-Thread Programming without Synchronization

Create 1 process

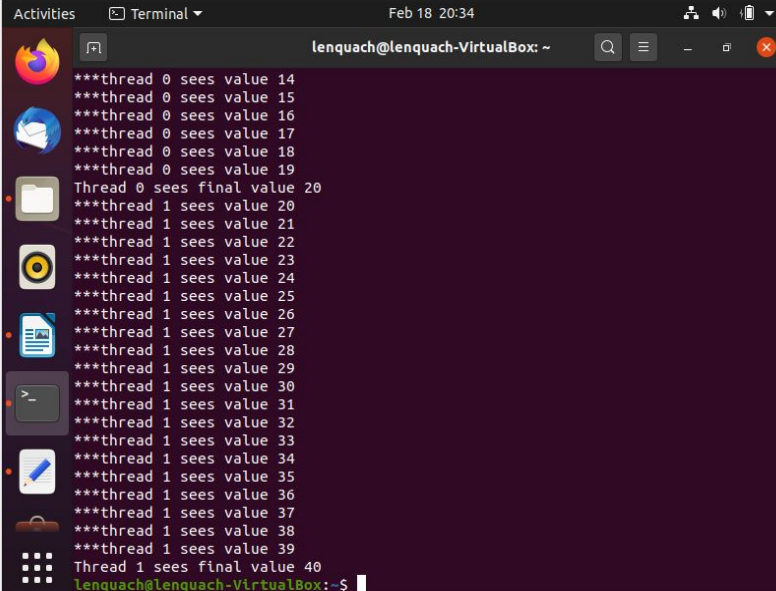
Fig.1



```
lenquach@lenquach-VirtualBox: ~  
Thread 0 sees final value 100  
Thread 2 sees final value 100  
Thread 4 sees final value 100  
Thread 3 sees final value 100  
Thread 1 sees final value 100  
lenquach@lenquach-VirtualBox:~$ gcc -o project1 project1.c -lpthread  
lenquach@lenquach-VirtualBox:~$ ./project1 1  
***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 0 sees final value 20  
lenquach@lenquach-VirtualBox:~$
```

Create 2 processes

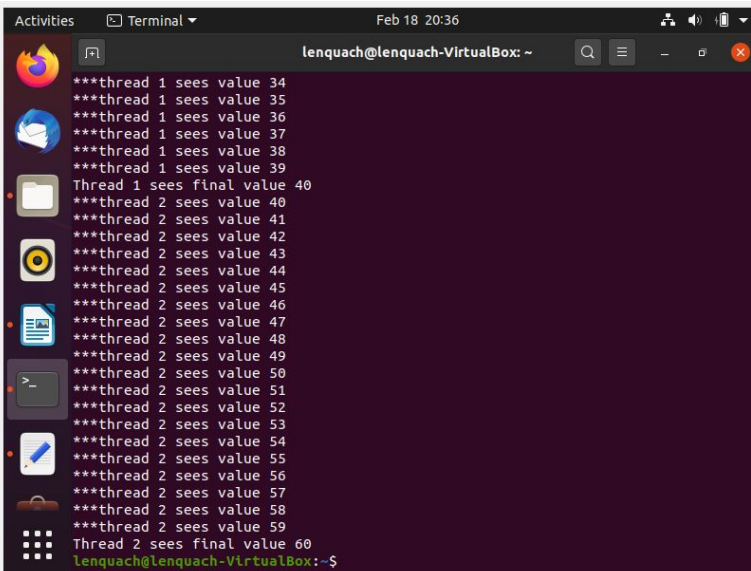
Fig. 2



A terminal window titled "lenquach@lenquach-VirtualBox: ~" showing the output of a program with two threads. The left sidebar contains icons for various applications. The terminal output is as follows:

```
***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 0 sees final value 20
***thread 1 sees value 20
***thread 1 sees value 21
***thread 1 sees value 22
***thread 1 sees value 23
***thread 1 sees value 24
***thread 1 sees value 25
***thread 1 sees value 26
***thread 1 sees value 27
***thread 1 sees value 28
***thread 1 sees value 29
***thread 1 sees value 30
***thread 1 sees value 31
***thread 1 sees value 32
***thread 1 sees value 33
***thread 1 sees value 34
***thread 1 sees value 35
***thread 1 sees value 36
***thread 1 sees value 37
***thread 1 sees value 38
***thread 1 sees value 39
Thread 1 sees final value 40
lenquach@lenquach-VirtualBox:~$
```

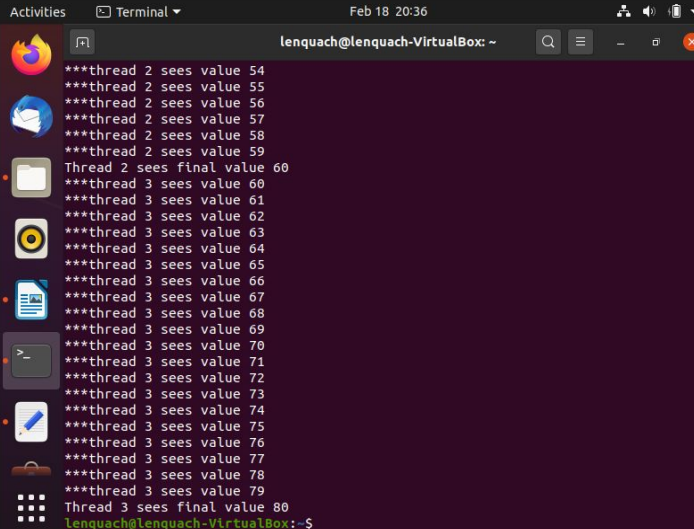
Create 3 processes
Fig.3



A terminal window titled "lenquach@lenquach-VirtualBox: ~" showing the output of a program with three threads. The left sidebar contains icons for various applications. The terminal output is as follows:

```
***thread 1 sees value 34
***thread 1 sees value 35
***thread 1 sees value 36
***thread 1 sees value 37
***thread 1 sees value 38
***thread 1 sees value 39
Thread 1 sees final value 40
***thread 2 sees value 40
***thread 2 sees value 41
***thread 2 sees value 42
***thread 2 sees value 43
***thread 2 sees value 44
***thread 2 sees value 45
***thread 2 sees value 46
***thread 2 sees value 47
***thread 2 sees value 48
***thread 2 sees value 49
***thread 2 sees value 50
***thread 2 sees value 51
***thread 2 sees value 52
***thread 2 sees value 53
***thread 2 sees value 54
***thread 2 sees value 55
***thread 2 sees value 56
***thread 2 sees value 57
***thread 2 sees value 58
***thread 2 sees value 59
Thread 2 sees final value 60
lenquach@lenquach-VirtualBox:~$
```

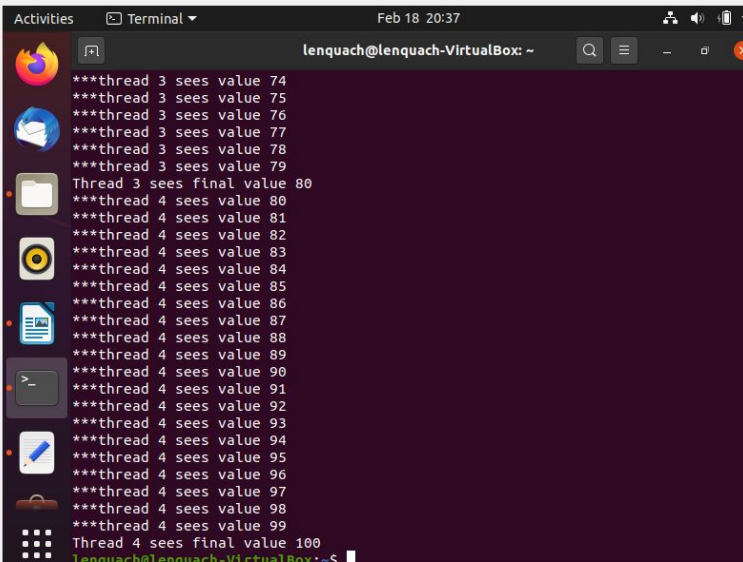
Create 4 processes
Fig. 4



```
lenquach@lenquach-VirtualBox: ~  
***thread 2 sees value 54  
***thread 2 sees value 55  
***thread 2 sees value 56  
***thread 2 sees value 57  
***thread 2 sees value 58  
***thread 2 sees value 59  
Thread 2 sees final value 60  
***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 sees final value 80  
lenquach@lenquach-VirtualBox:~$
```

Create 5 processes

Fig.5



```
lenquach@lenquach-VirtualBox: ~  
***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 sees final value 80  
***thread 4 sees value 80  
***thread 4 sees value 81  
***thread 4 sees value 82  
***thread 4 sees value 83  
***thread 4 sees value 84  
***thread 4 sees value 85  
***thread 4 sees value 86  
***thread 4 sees value 87  
***thread 4 sees value 88  
***thread 4 sees value 89  
***thread 4 sees value 90  
***thread 4 sees value 91  
***thread 4 sees value 92  
***thread 4 sees value 93  
***thread 4 sees value 94  
***thread 4 sees value 95  
***thread 4 sees value 96  
***thread 4 sees value 97  
***thread 4 sees value 98  
***thread 4 sees value 99  
Thread 4 sees final value 100  
lenquach@lenquach-VirtualBox:~$
```

The program in part 1 runs without Synchronization, and therefore, the number of threads does not depend on the other one or the threads do not see the same final value. The last thread will see a final value 20 times the number of threads.

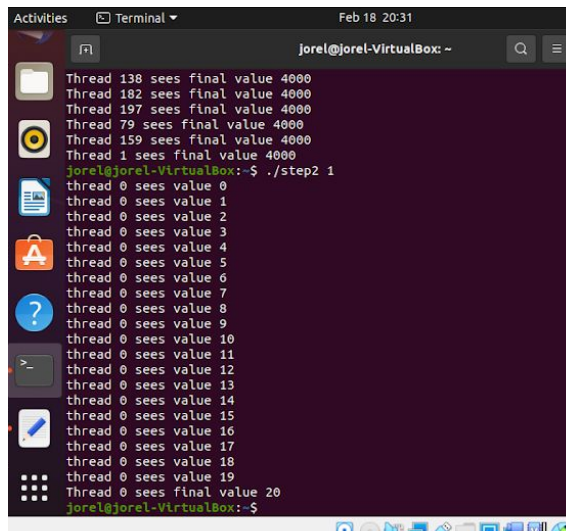
2. Part 2: Simple Multi-Thread Programming with Proper Synchronization

The mutex lock prevents multiple threads from executing simultaneously. This ensures synchronized access of shared resources in the code. The mutex barrier causes a certain number of threads to be at the barrier before starting the next part of the program. Using both,

the program runs in a more organized way and helps prevent errors in the shared variable. Without synchronization, there can be a loss of data and race conditions may occur. However, with the synchronization, the threads will all see a value of 20 times the number of threads.

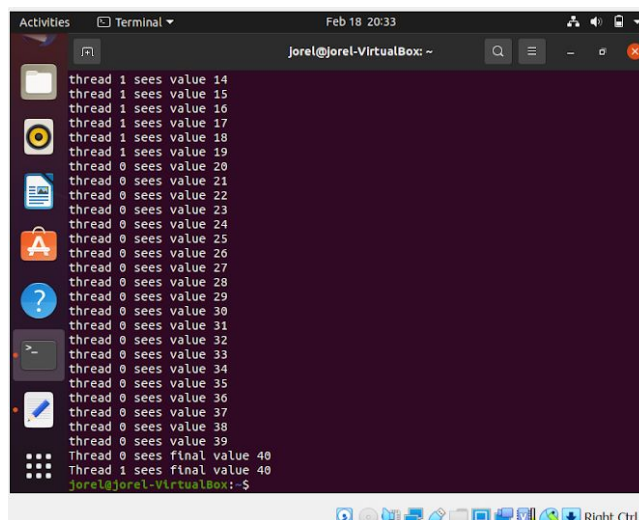
1 thread

Fig.1

A terminal window titled 'jorel@jorel-VirtualBox: ~' showing the output of a program with 1 thread. The output lists several threads (138, 182, 197, 79, 159, 1) all seeing a final value of 4000. Then, thread 0 is shown seeing values from 0 to 19, and finally reaching a final value of 20. The prompt is 'jorel@jorel-VirtualBox:~\$'.

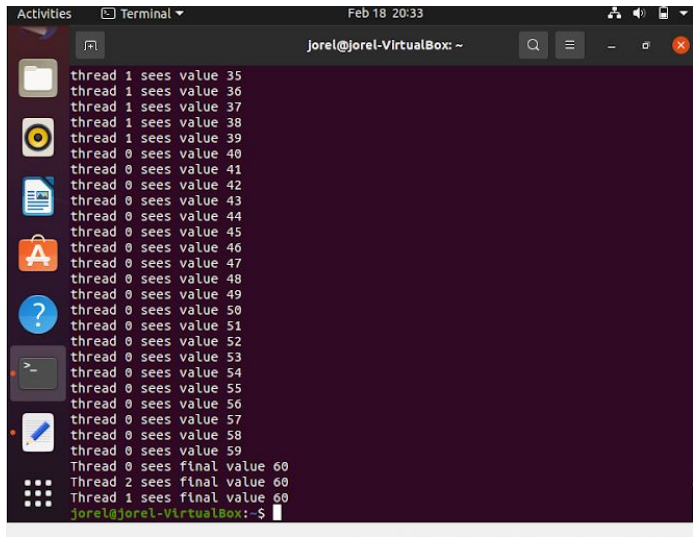
2 threads

Fig. 2

A terminal window titled 'jorel@jorel-VirtualBox: ~' showing the output of a program with 2 threads. Thread 1 sees values 14 through 19, and thread 0 sees values 20 through 39. Both threads reach a final value of 40. The prompt is 'jorel@jorel-VirtualBox:~\$'.

3 threads

Fig.3

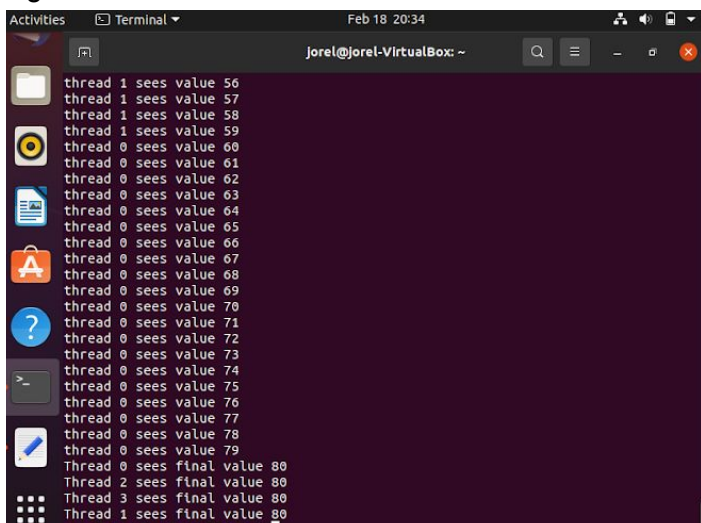


A terminal window titled "jorel@jorel-VirtualBox: ~" showing the output of a program with 4 threads. The output shows thread 1 seeing values 35 through 59, and thread 0 seeing values 40 through 59. All threads eventually see a final value of 60. The prompt is "jorel@jorel-VirtualBox:~\$".

```
thread 1 sees value 35
thread 1 sees value 36
thread 1 sees value 37
thread 1 sees value 38
thread 1 sees value 39
thread 0 sees value 40
thread 0 sees value 41
thread 0 sees value 42
thread 0 sees value 43
thread 0 sees value 44
thread 0 sees value 45
thread 0 sees value 46
thread 0 sees value 47
thread 0 sees value 48
thread 0 sees value 49
thread 0 sees value 50
thread 0 sees value 51
thread 0 sees value 52
thread 0 sees value 53
thread 0 sees value 54
thread 0 sees value 55
thread 0 sees value 56
thread 0 sees value 57
thread 0 sees value 58
thread 0 sees value 59
Thread 0 sees final value 60
Thread 2 sees final value 60
Thread 1 sees final value 60
jorel@jorel-VirtualBox:~$
```

4 threads

Fig. 4

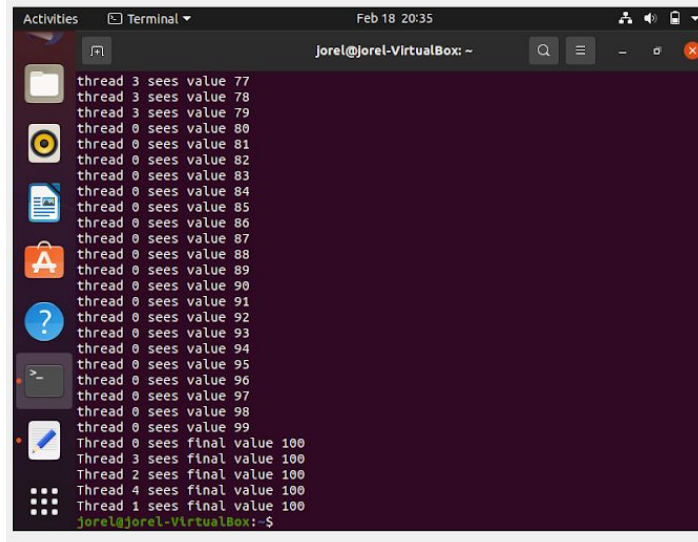


A terminal window titled "jorel@jorel-VirtualBox: ~" showing the output of a program with 5 threads. The output shows thread 1 seeing values 56 through 79, and thread 0 seeing values 60 through 79. All threads eventually see a final value of 80. The prompt is "jorel@jorel-VirtualBox:~\$".

```
thread 1 sees value 56
thread 1 sees value 57
thread 1 sees value 58
thread 1 sees value 59
thread 0 sees value 60
thread 0 sees value 61
thread 0 sees value 62
thread 0 sees value 63
thread 0 sees value 64
thread 0 sees value 65
thread 0 sees value 66
thread 0 sees value 67
thread 0 sees value 68
thread 0 sees value 69
thread 0 sees value 70
thread 0 sees value 71
thread 0 sees value 72
thread 0 sees value 73
thread 0 sees value 74
thread 0 sees value 75
thread 0 sees value 76
thread 0 sees value 77
thread 0 sees value 78
thread 0 sees value 79
Thread 0 sees final value 80
Thread 2 sees final value 80
Thread 3 sees final value 80
Thread 1 sees final value 80
jorel@jorel-VirtualBox:~$
```

5 threads

Fig.5

A screenshot of a terminal window titled "Terminal" with a date and time of "Feb 18 20:35". The terminal shows a series of log messages from a program. The messages are: "thread 3 sees value 77", "thread 3 sees value 78", "thread 3 sees value 79", "thread 0 sees value 80", "thread 0 sees value 81", "thread 0 sees value 82", "thread 0 sees value 83", "thread 0 sees value 84", "thread 0 sees value 85", "thread 0 sees value 86", "thread 0 sees value 87", "thread 0 sees value 88", "thread 0 sees value 89", "thread 0 sees value 90", "thread 0 sees value 91", "thread 0 sees value 92", "thread 0 sees value 93", "thread 0 sees value 94", "thread 0 sees value 95", "thread 0 sees value 96", "thread 0 sees value 97", "thread 0 sees value 98", "thread 0 sees value 99", "Thread 0 sees final value 100", "Thread 3 sees final value 100", "Thread 2 sees final value 100", "Thread 4 sees final value 100", and "Thread 1 sees final value 100". The terminal prompt is "jorel@jorel-VirtualBox: ~\$".

```
thread 3 sees value 77
thread 3 sees value 78
thread 3 sees value 79
thread 0 sees value 80
thread 0 sees value 81
thread 0 sees value 82
thread 0 sees value 83
thread 0 sees value 84
thread 0 sees value 85
thread 0 sees value 86
thread 0 sees value 87
thread 0 sees value 88
thread 0 sees value 89
thread 0 sees value 90
thread 0 sees value 91
thread 0 sees value 92
thread 0 sees value 93
thread 0 sees value 94
thread 0 sees value 95
thread 0 sees value 96
thread 0 sees value 97
thread 0 sees value 98
thread 0 sees value 99
Thread 0 sees final value 100
Thread 3 sees final value 100
Thread 2 sees final value 100
Thread 4 sees final value 100
Thread 1 sees final value 100
jorel@jorel-VirtualBox: ~$
```

3. Individual Contributions

Jorel Caoile:

- Write code for part 2 and write the report
- Do ReadMe file

Len Quach:

- Write code for part 1 and write the report
- Record video

Video Link: <https://youtu.be/kppKK4N6QHk>