Emilio Quiambao

Homework 5

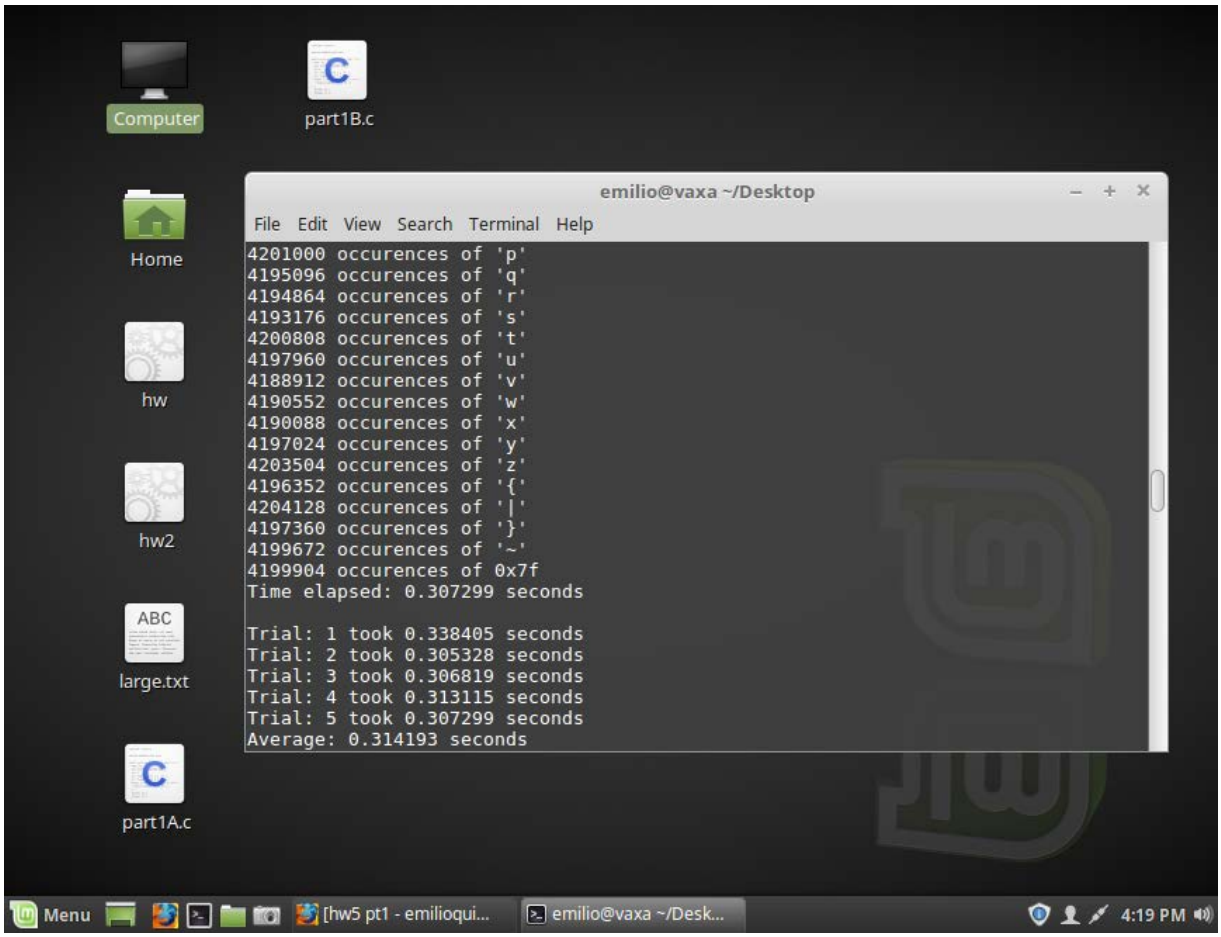
Part 1

This first part of the assignment modifies the previous assignments's character count program into two programs. Instead of a two-dimensional array, the program instead uses a one-dimensional array to hold all of the character counts with no synchronization. The second program is the same, but this time the increments are protected by a mutex. I ran tests from both programs side by side using the same 'large.txt' file from the previous assignment.

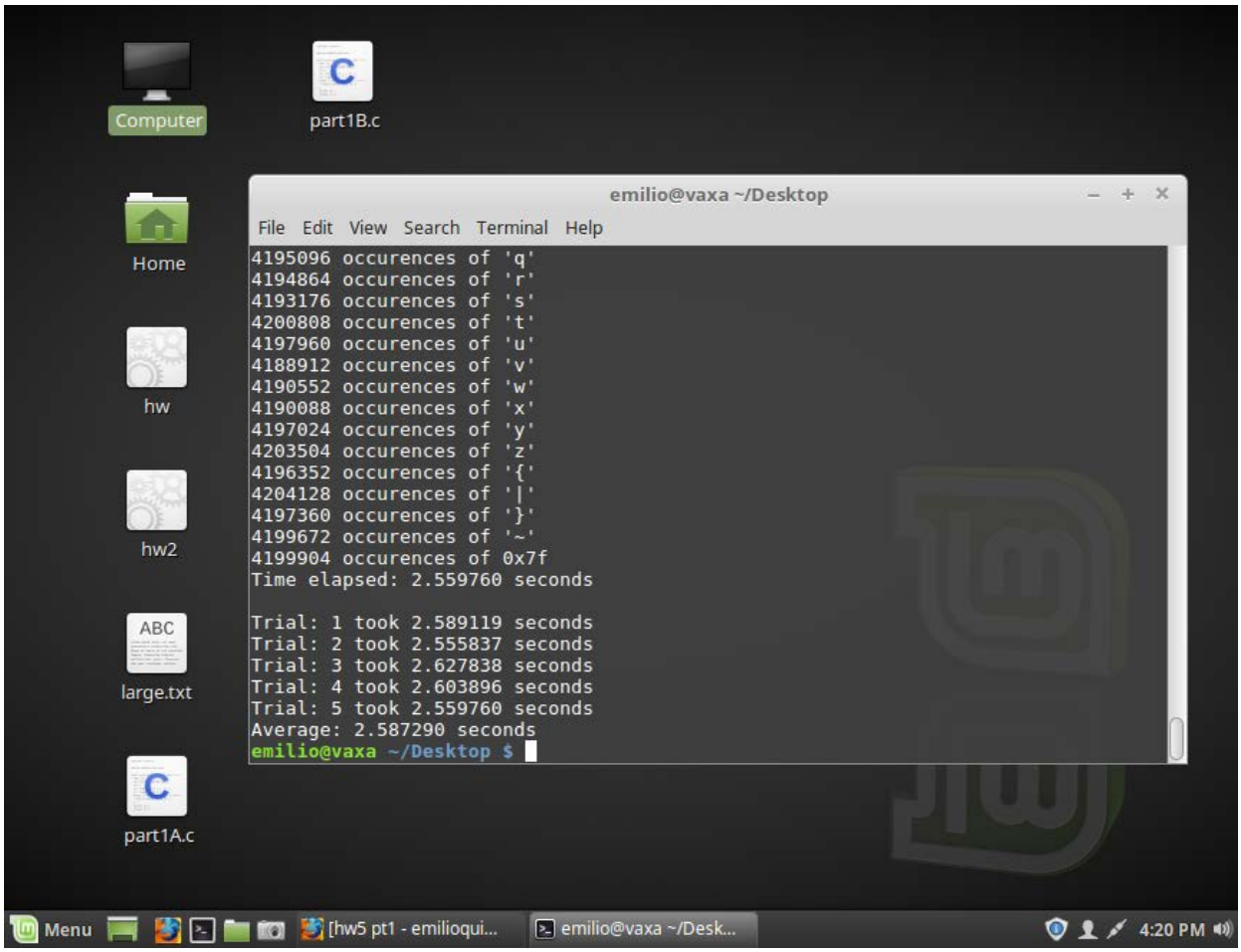
My prediction was that the second program with protection and synchronization will have larger occurrences because the mutex will prevent race conditions from occurring. The program with no protection risks unexpected outputs (or smaller numbers) because its output is dependent on sequence of events or timing, and with no synchronization, the program can count incorrect number of occurrences in the file. In both the programs however, I have scanned through their outputs and have found that they have the same count. However, interestingly enough, I have found that the first program is much more faster than the second program by an average of 2 seconds! My theory of why the times are so different is because the mutex takes up so much time ensuring that no threads meet a critical condition, while having no protection sacrifices the risk with faster times.

The following pages contain screenshots of the output for both programs, as well as part 2 of the assignment.

First program output (no mutex)

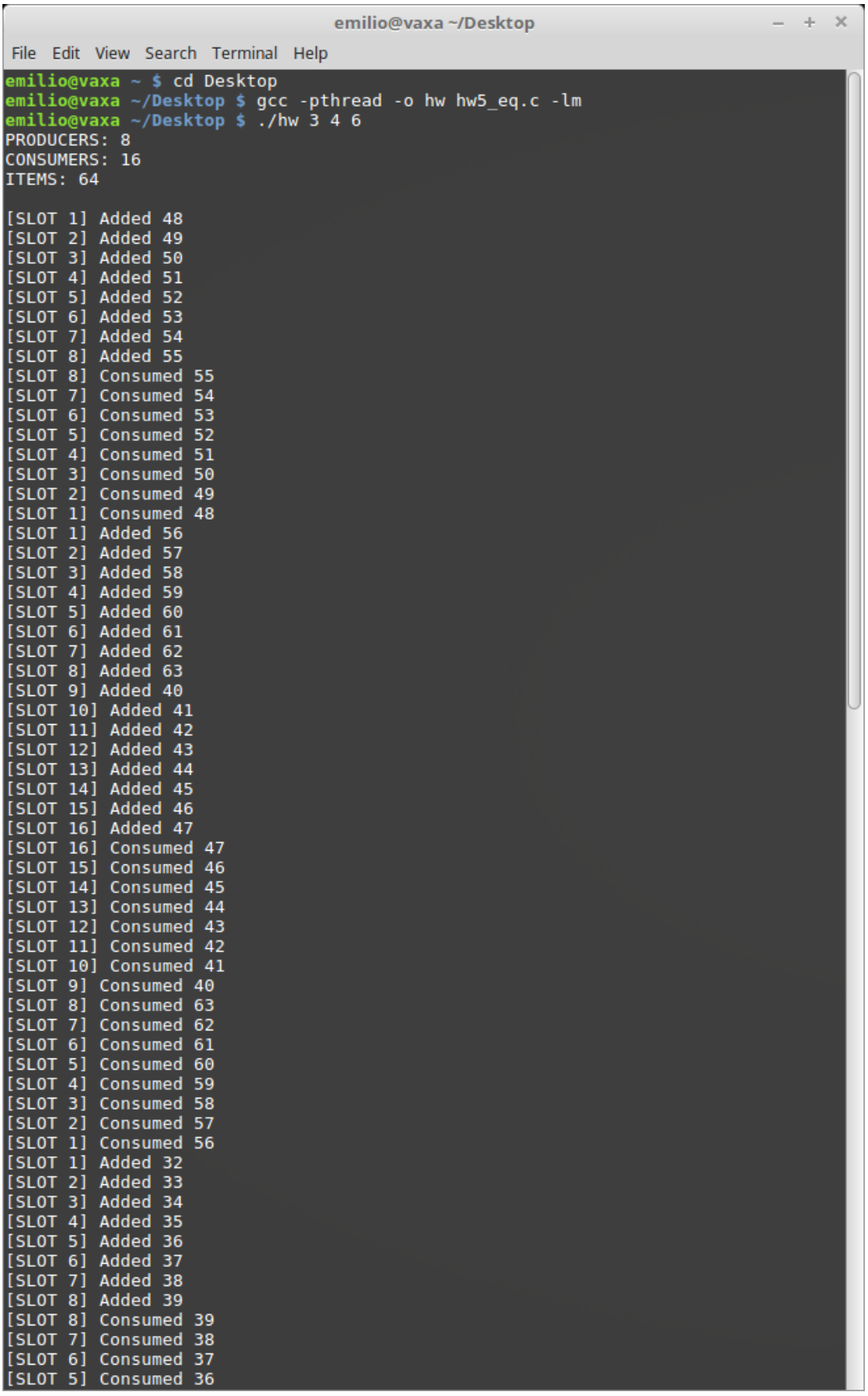


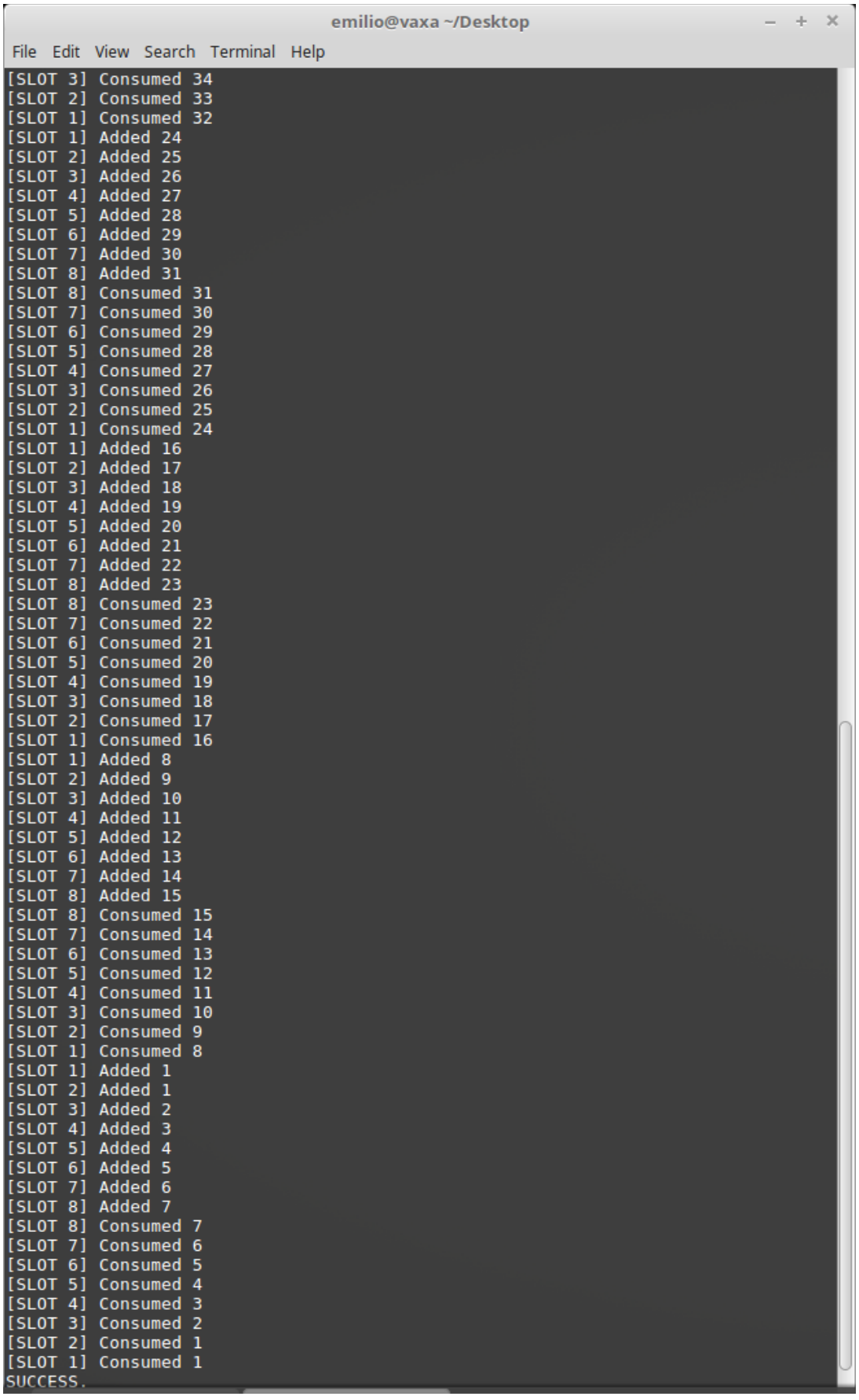
Second program output (with mutex)



Part 2

The second part of the assignment implements a multi-threaded producer-consumer program using a bounded buffer. It uses two condition variables (‘ready_to_read’ and ‘ready_to_write’) supported by a mutex in order to control access to the buffer. Two functions were made to check if the buffer is full or empty. The idea is that consumers wait for a ‘ready_to_read’ signal if the buffer is empty, and producers wait for a ‘ready_to_write’ signal if the buffer is full. Both wait conditions and their push or pull actions are surrounded by a mutex. The user enters the number of producers, consumers, and items which is specified by their binary log. If the user fails to input valid arguments, if the mutex fails to initialize, or if any threads fail to create, an error will be displayed. Below is an example of entering parameters 3, 4, 6, which equates to 8 producers, 16 consumers, and 64 items.


emilio@vaxa ~/Desktop
File Edit View Search Terminal Help
emilio@vaxa ~ $ cd Desktop
emilio@vaxa ~/Desktop $ gcc -pthread -o hw hw5_eq.c -lm
emilio@vaxa ~/Desktop $ ./hw 3 4 6
PRODUCERS: 8
CONSUMERS: 16
ITEMS: 64
[SLOT 1] Added 48
[SLOT 2] Added 49
[SLOT 3] Added 50
[SLOT 4] Added 51
[SLOT 5] Added 52
[SLOT 6] Added 53
[SLOT 7] Added 54
[SLOT 8] Added 55
[SLOT 8] Consumed 55
[SLOT 7] Consumed 54
[SLOT 6] Consumed 53
[SLOT 5] Consumed 52
[SLOT 4] Consumed 51
[SLOT 3] Consumed 50
[SLOT 2] Consumed 49
[SLOT 1] Consumed 48
[SLOT 1] Added 56
[SLOT 2] Added 57
[SLOT 3] Added 58
[SLOT 4] Added 59
[SLOT 5] Added 60
[SLOT 6] Added 61
[SLOT 7] Added 62
[SLOT 8] Added 63
[SLOT 9] Added 40
[SLOT 10] Added 41
[SLOT 11] Added 42
[SLOT 12] Added 43
[SLOT 13] Added 44
[SLOT 14] Added 45
[SLOT 15] Added 46
[SLOT 16] Added 47
[SLOT 16] Consumed 47
[SLOT 15] Consumed 46
[SLOT 14] Consumed 45
[SLOT 13] Consumed 44
[SLOT 12] Consumed 43
[SLOT 11] Consumed 42
[SLOT 10] Consumed 41
[SLOT 9] Consumed 40
[SLOT 8] Consumed 63
[SLOT 7] Consumed 62
[SLOT 6] Consumed 61
[SLOT 5] Consumed 60
[SLOT 4] Consumed 59
[SLOT 3] Consumed 58
[SLOT 2] Consumed 57
[SLOT 1] Consumed 56
[SLOT 1] Added 32
[SLOT 2] Added 33
[SLOT 3] Added 34
[SLOT 4] Added 35
[SLOT 5] Added 36
[SLOT 6] Added 37
[SLOT 7] Added 38
[SLOT 8] Added 39
[SLOT 8] Consumed 39
[SLOT 7] Consumed 38
[SLOT 6] Consumed 37
[SLOT 5] Consumed 36


emilio@vaxa ~/Desktop
File Edit View Search Terminal Help
[SLOT 3] Consumed 34
[SLOT 2] Consumed 33
[SLOT 1] Consumed 32
[SLOT 1] Added 24
[SLOT 2] Added 25
[SLOT 3] Added 26
[SLOT 4] Added 27
[SLOT 5] Added 28
[SLOT 6] Added 29
[SLOT 7] Added 30
[SLOT 8] Added 31
[SLOT 8] Consumed 31
[SLOT 7] Consumed 30
[SLOT 6] Consumed 29
[SLOT 5] Consumed 28
[SLOT 4] Consumed 27
[SLOT 3] Consumed 26
[SLOT 2] Consumed 25
[SLOT 1] Consumed 24
[SLOT 1] Added 16
[SLOT 2] Added 17
[SLOT 3] Added 18
[SLOT 4] Added 19
[SLOT 5] Added 20
[SLOT 6] Added 21
[SLOT 7] Added 22
[SLOT 8] Added 23
[SLOT 8] Consumed 23
[SLOT 7] Consumed 22
[SLOT 6] Consumed 21
[SLOT 5] Consumed 20
[SLOT 4] Consumed 19
[SLOT 3] Consumed 18
[SLOT 2] Consumed 17
[SLOT 1] Consumed 16
[SLOT 1] Added 8
[SLOT 2] Added 9
[SLOT 3] Added 10
[SLOT 4] Added 11
[SLOT 5] Added 12
[SLOT 6] Added 13
[SLOT 7] Added 14
[SLOT 8] Added 15
[SLOT 8] Consumed 15
[SLOT 7] Consumed 14
[SLOT 6] Consumed 13
[SLOT 5] Consumed 12
[SLOT 4] Consumed 11
[SLOT 3] Consumed 10
[SLOT 2] Consumed 9
[SLOT 1] Consumed 8
[SLOT 1] Added 1
[SLOT 2] Added 1
[SLOT 3] Added 2
[SLOT 4] Added 3
[SLOT 5] Added 4
[SLOT 6] Added 5
[SLOT 7] Added 6
[SLOT 8] Added 7
[SLOT 8] Consumed 7
[SLOT 7] Consumed 6
[SLOT 6] Consumed 5
[SLOT 5] Consumed 4
[SLOT 4] Consumed 3
[SLOT 3] Consumed 2
[SLOT 2] Consumed 1
[SLOT 1] Consumed 1
SUCCESS.