**Processes/Threads Shared Memory Homework problem (assignment #7) for Computer Operating Systems**

This assignment is worth 5 points.

Make a C / C++, Java, or Python program with two processes, a producer and a consumer. If you want to use another language, clear it with me first. This is similar to assignment 2, except you are to use locks to protect the shared variable from being clobbered. This means that the output should always be the same.

The producer process/thread consists of a loop that writes the loop count plus 1 (a value from 1 to 5) into a variable that it shares with the consumer process/thread (this variable is to be initialized to 0). On each pass through the loop, before the producer writes into the shared variable, it does a random wait of from one to three seconds (compute a new random wait value on each pass through the loop). The loop is to be executed five times. But before you place a value into the shared variable, you must check a shared lock. If the lock is set, wait. If the lock is not set, set it, check the value of the shared variable, if it is nonzero, release the lock and wait 1 second and check the lock and the variable again (repeat this until the variable is 0). If it is zero, place the value (the number from 5 to 1) into the shared variable and clear the lock. Note that if you a spin lock, you will not have to do the 1 second waits when the lock is set.

The consumer process consists of a loop that reads the variable it shares with the producer five times and computes the sum of the values it reads. On each pass through the loop before it reads the shared variable, it does a random wait of from one to three seconds (compute a new random value on each pass through the loop). But before you read a value from the shared variable, you must check a shared lock. If the lock is set, wait 1 second and check it again (repeat until it is cleared). If the lock is not set, lock it, and read the value from the shared variable. If the value is 0, clear the lock and do a wait of 1 second and then do the lock and check it again (repeat until it is nonzero). If the value is not zero, add it to the sum being accumulated and set the shared variable to 0 and then clear the shared lock. Note that if you a spin lock, you will not have to do the 1 second waits when the lock is set.

When the loop finishes, the program is to write the sum into a file.

You must run the program twice and submit the output from both runs. Note that the output from the runs will be the same (15 the sum of 5, 4, 3, 2, and 1) if you have written your program correctly. Place a zip file with the code, a copy of the data file output from each of your runs for grading in the D2L Dropbox for the 7th assignment. Also include a word or text file with your observations on the assignment: What did you notice about the runs? What if anything did you learn doing this assignment? Roughly how long (in hours) did the assignment take?

Please put comments at the top of your program with your name, the date, the assignment number, and a brief description of the program. I also want to see comments within the program. The first items in your data file and observations file must be your name and the assignment number (have your program output this information into your output file). You are to zip your files together and place the zip file in the drop box for assignment 7 on D2L - do not send me a tar file or another type of compression file or place the files out on the Internet or the Cloud for me to pick up.

If you do not know how to do locking in Java, C/C++, or Python, it is your responsibility to learn how. This information is available on the Internet.

The contents of the output file should look something like this.  If you have the output from both runs in the same file, you do not need to repeat your name and the assignment number.

John Doe
Computer Operating Systems Assignment #7

The sum is 15