CSCI 3232 Systems Software Assignment 7

Upload all your files to the correct dropbox in Folio before the deadline --- **11:30PM, Jul 14, Wednesday, 2021.**

Note: You need to be able to figure out whether there are race conditions in a program and tell the possible outputs without actually running the code. You should also be able to apply mutex and semaphore techniques to synchronize your programs should any race conditions arise.

1. (30 pts) Read given code RaceOrNot1.c and write all possible outputs of the program. Assume there will be no thread creation or joining failures or mutex or semaphore failures. If you believe there is only one possible output, you just need to write that output. Pay attention to the note above.
2. (30 pts) Read given code RaceOrNot2.c and write all possible outputs of the program. Assume there will be no thread creation or joining failures or mutex or semaphore failures. If you believe there is only one possible output, you just need to write that output. Pay attention to the note above.
3. (40 pts) From previous homework you are already familiar with the math function *f* defined on positive integers as f(x)=(3x+1)/2 if x is odd and f(x)=x/2 if x is even. Given any integer *var*, iteratively applying this function *f* allows you to produce a list of integers starting from *var* and ending with 1. For example, when *var* is 6, this list of integers is 6,3,5,8,4,2,1, which has a length of 7 because this list contains 7 integers (call this list the Collatz list for 6). Write a C program **A7p3.c** that accepts three command line arguments *a*, *b* and *n* which are assumed to be positive integers with *a*<*b* and *n* between 2 and 6 inclusive. Use pthread to create *n* threads to count how many integers between *a* and *b* inclusive have their Collatz list length (A) less than the integer itself; (B) equal the integer itself; (C) greater than the integer itself. Notice that the integer 6 belongs to category (C). You should divide this list generation and length calculating task among the *n* threads as evenly as possible. For example, if *n* is 3 and there are 60 integers between *a* and *b* inclusive, then each thread is supposed to handle 60/3=20 Collatz lists. Print out the three numbers representing the counts for (A)(B)(C) mentioned above. Use mutex or semaphore to avoid race conditions if necessary. Note: if you do not use pthread to divide the task among the threads, you may get zero points. A sample run of the compiled program is shown below. You do NOT need to submit screen shots. Instead submit your source file.

[kwang@computer][~/temp]$./A7p3 11 70 3

using 3 threads.

The number of integers from 11~70 whose Collatz list length is < the integer is [xx]

The number of integers from 11~70 whose Collatz list length is = the integer is [yy]

The number of integers from 11~70 whose Collatz list length is > the integer is [zz]

Checklist of the two files to be submitted: A7p3.c, solution file for problems 1,2.