CMPSC 472
Fall 2017
Homework #3

Due: Thursday, 10/12/2017
45 Points

Homework Information

Please note this list has changed for Homework #3.

- This homework is due at the beginning of the class period. Late papers receive no credit.
- This assignment is to be done on your own.
- Please be sure to email copies of your code AND hand in hard copies, one problem per file, printing on one side only. If a problem requires more than one page to print out, please staple the pages together.
- All code should have comments that include your name and the problem number.
- Put all solutions in a pocket folder, in order by problem number. (The folder should be a pocket folder, not an index folder or a folder with clamps or rings.)
- Your name and the course info (CMPSC 472) should be on the outside of the folder in the upper right hand corner.
- Files should be emailed using the `mail472` command. Place all files you want to email in a subdirectory and type `mail472`. You will be prompted on what to do. This command will email ALL files in the subdirectory, so be sure to delete any you do not wish to email (such as .lst and .pco files).
- Files must be named as per the instructions below, with no spaces using all lower case.
- **Failure to follow these rules may result in lost points.**

Important Things About BACI and Turning in Your Assignment

- Some versions of BACI don't like long (multiline) comments using the /* */ style. Therefore, you may wish to use the // comment style for long comments.
- In addition, some versions do not like semaphores to be initialized with the when they are declared, so you may wish to use INITIALSEM.
- However you set up comments and initialization, please note: ALL PROGRAMS MUST COMPILE ON THE LINUX MACHINES IN THE SUN LAB AND WILL BE GRADED ON THESE MACHINES. So please test your programs on these machines before submitting them.
- Your solutions should not unnecessarily restrict concurrency -- your solution will be graded not only on its correct output but also correct placement of the synchronization.
- Before you email your files, name the files using your first initial and last name followed by the question number. Place all files in a subdirectory and use the `mail472` command to email only the .cm (or .pm) files in the subdirectory (the command will email ALL files so be sure to delete ones that should not be mailed). (This command is located in /usr/bin.) For example, John Doe would email jdoe1a.cm, jdoe1b.cm, jdoe1c.cm, jdoe2.cm, jdoe3a.cm, and jdoe3b.cm by placing these 6 files in a subdirectory, changing to that directory, and typing `mail472`.
- Spaces in file names will cause problems with the `mail472` command, so do no use spaces in file names. **Spaces in the subject header may also cause problems.**
- NOTE: When you run mail472, you should get back an acknowledgement email almost immediately. It will be sent to your SUN lab account (unless you have your SUN lab email forwarded). If you do not get this email, it means the files were not sent.

The Problems

1. [15] Consider the cross.cm program.
   a. Modify the code to order the processes so they run in Stop → Look → Listen → Cross order. The output from the program should be:

      ```
      Stop Look Listen Cross
      ```

      Name the file using your first initial and last name, ALL LOWERCASE, followed by "1a" (i.e., jdoe1a.cm).

   b. Modify your answer to part (a) so each process will run 4 times, in sequence. The output from your program should be:

      ```
      Stop Stop Stop Stop
      Look Look Look Look
      Listen Listen Listen Listen
      Cross Cross Cross Cross
      ```

      Name the file using "1b" (i.e., jdoe1b.cm).

   c. Rewrite the code so that the processes print:

      ```
      Stop  Look  Listen Cross
      Stop  Look  Listen Cross
      Stop  Look  Listen Cross
      Stop  Look  Listen Cross
      ```

      Name the file using "1c" (i.e., jdoe1c.cm).

2. [15] The code onetwo.cm is a BACI program that simulates a mutual update on a shared variable. Sometimes, a race condition happens, causing the variable to be updated incorrectly (as you saw in the example from class).
   a. What are the possible values for *sum* when both procedures have completed without proper synchronization? Include your answer to this question as a comment labeled "Answer to 2a" in your program code for part c.
   b. The program onetwosem.cm is a BACI program that attempts to fix the race condition using semaphores. It doesn't work. Explain why this solution is incorrect. Include your answer to this question as a comment labeled "Answer to 2b" in your program code for part c.
   c. Modify the onetwo.cm program using semaphores so that it **never** updates the variable incorrectly. Do this is a way that maximizes concurrency for full credit. Name the file using "2" (i.e., jdoe2.cm).

3. [15] Look at the program ab.cm.
   a. By introducing one or more semaphores, modify this program so the processes print in the order ABAA. (Each A can be printed by any copy of process A.) Include comments near the top of your program to explain why the processes run in this order. Name the file using "3a" (i.e., jdoe3a.cm).
   b. Redo part (a), but have the processes print in the order AABA. Name the file using "3b" (i.e., jdoe3b.cm).
