Cosc 4740
Program 3

Due Date: Oct 23

Goal: learn threading and semaphores.
Grade: 50 points.

There are 4 programming problems. You must use the threads as provided in the example code in class and lab. Use mutex for locks and semaphores for the rest. There cannot be any busy loops.

For 1, 2 and 4 you are writing them from scratch. Each process must sleep for random amount of time from 0 to 30 seconds (see semaphore_mutex_dock.cpp or unisex). **Each process MUST have an id number and it will be printed out with each event line. Prints are required before any sleep call, plus "events" like picking up a fork for example.**

1. Implement Dinning Philosopher problem using semaphores and avoid deadlock. Each philosopher will be their own thread.

2. *The Dining Savages Problem*. A tribe of savages eats communal dinners from a large pot that can hold M servings of stewed missionary. When a savage wants to eat, he helps himself from the pot, unless it is empty. If the pot is empty, the savage wakes up the cook and then waits until the cook has refilled the pot. Your solution should avoid deadlock and awaken the cook only when the pot is empty.

Implement the actions of the savages and cook using semaphores and avoid deadlock. There will be 1 cook thread and 10 savages threads.

3. Star port dock problem. Using the semaphore_mutex_dock.cpp in the example directory of the repo, correct the code, so that starvation can be avoided. You may NOT add any new variables or methods. Use what is already there.

4. The *Physical Plant Problem*
You will have three kinds of processes, client process, Help Desk process, Physical Plant Tech processes.

The client process will do something for a random amount of time. Then something brakes, so the client has to call physical plant to have them come out and fix it. In order for the client to get something fixed they must call the Help Desk. Then wait for the Physical Plant tech's to show up to fix the problem. Once the problem is fixed they go back to working for a random amount of time before the next problem.

The Help Desk has a simple job. Wait for calls from the clients and then tell the Physical Plant Tech to go fix it.

Physical Plant Tech process: They are in the break room having coffee for a random amount of time. When they are done with coffee, then wait for Help Desk to tell them to go out to clients. But they can only go to a client's when there are 3 tech's (union rules) done with coffee. Once there are 3 techs then the problem will get fixed, and they then tell the client the problem is fixed and go back to the break room for a random amount of time having coffee.

Implement the actions of the processes using semaphores. There will be 2 client threads, 1 help desk thread, and 5 Physical Plant Threads. Your code should avoid deadlock.

Turning in the Assignment:
Hard copy:

1. A cover page with your Name, cosc 4740 a repo name (see github and below for your repo name), Section #. Statement of help.
2. Only turn in a cover page, no code or output.
3. This is your marker that you are turning in the assignment.

Soft copy:

1. Use this link to create your repo https://classroom.github.com/a/ZZgTlwAZ
2. Upload the project to your repo.
3. Create/Edit the readme.md file, add the following:
    - Name
    - How to compile the code for each of the 4 problems, if there is no makefile.
    - List anything that doesn't work (that you know of)
4. Lastly ensure everything has been uploaded to the github website and not just the local repo.

Code will be graded on correctness, comments, and coding style.