CSC35500 Programming Project 2

Due: 11/8/2022, 11:59 PM

Early Submission Deadline: 11/6/2021, 11:59 PM

Problem Statement

You will be simulating an airport where 8 planes go out on tours repeatedly. Each tour will be for exactly 12 passengers and last approximately 30 seconds. In performing the simulation, you must make sure that no plane tries to leave a gate with less (or more) than 12 passengers and that no plane lands or takes off at the same time as another plane is landing or taking off.

The first thing your program should do is read the following from the command line (in the order specified):

- the total number of passengers that are in the airport.
- the sum total number of tours that all of the planes are to do. (NOT the number of tours that *each* plane is to do, the *total* number of tours that *all* planes will do when their number of completed tours is added together.)

You should then create 8 new threads, each one representing a (uniquely numbered) plane; the main function should wait for all of the threads to complete and then exit. Each thread represents an airplane, and should repeatedly follow these steps:

1. board. Waits for 12 passengers to get on the plane. Between each available passenger boarding a random delay of between 0 and 2 seconds should be enforced. This can be accomplished by calling sleep(rand()%3) after each passenger boards.
2. The associated plane should then be shown taxi-ing to the runway. *See the provided AirportAnimator class !*
3. The associate plane should then use the runway to take off, animating such (*see the provided AirportAnimator class!)* Of course, the plane should wait for exclusive access to the runway.
4. The plane should then go on tour. This basically just sleeps for between 15 and 45 seconds, and can be accomplished by calling sleep(15+rand()%31).
5. The plane should then request to land. Once exclusive access to the runway is granted land, the plane landing should be animated (*See the provided AirportAnimator class!)*
6. The plane should then taxi back to its gate. *See the provided AirportAnimator class!*
7. Each passenger should then deplane and thus Abe returned to the “pool” of available passengers. There should be a 1 second delay between each passenger deplaning.

After completing the above steps, you should update the number of completed tours (keeping in mind that two planes could conceivably attempt to do so at the same time) and go back to step 1 only if the number of tours required was not yet completed! Otherwise, the thread should return. *Don't forget to use the provided AirportAnimator class!*

You should appropriately update the status of the individual planes as each step is being processed.

Problem Details

- This problem once again requires that you use a Linux machine to complete
- You are being provided with code to display the requested animations. *Do not wast time trying to re-invent the wheel!*
- Should you have to stop your program (via either ctrl-C or ctrl-Z), you may find the resulting terminal in an unusable state. Typing **reset** (which may not be visible as you type it) should fix the underlying problem.

Submission

You should post to Canvas both your `C++` source code file and a plain text file (not an MS Word file) called `read.me in a zip or tgz file`. Make sure the "root" of your submission is a folder (not just a collection of files.)

The `read.me` file should contain:
• your name and any other identifying information.
• any special details on how to compile your project
• any special details on how to run your project - note that you cannot circumvent the project specifications here!
• any known bugs your project has, and possible fixes to those bugs (partial credit abounds here).
• an overview of how you solved the project.
• You may put any other information you like in this file, although you are not required to do so - one common additional set of entries is a "software engineering log" that indicates what you have done every time you sat down and worked on the project. Many programmers find that such actually helps you to finish projects faster!

The read.me file MUST also contain the answers to the following questions:

1. Try running your program with only 11 passengers. What happens? Why?
2. Try running your program with 12 passengers. What happens? Why?
3. Try running your program with 50 passengers and 100 tours. What happens? why?
4. Try running your program with 100 passengers and 20 tours. What happens? Why?
5. What is the minimum number of passengers that will guarantee avoidance of deadlock? Why?

Grading Breakdown

Correct Submission	10%
Successful Compilation	10%
Following Directions	20%
Correct Execution	40%
Comments/ read.me, including answers to required questions!	20%

Final Notes

- The provided code uses both the ncurses and the pthread libraries. So, you will need to add `-lncurses` and (possibly) `-lpthread` options to the end of your link command during compilation.
- Have you started this project yet? If not, *start now*!
- If you have any questions about this project, see me as soon as possible.
- Have you started this project yet? If not, ***start NOW*** !