COMSC-200 Lab 16
More STL

GOOD PROGRAMMING PRACTICES *show / hide*

ABOUT THIS ASSIGNMENT
In this lab session, you continue the ElevatorSimulation problem -- a problem that will be solved over labs 8-10 and 14-16.

As you complete this assignment, post the required files to the COMSC server. You may post them all at once, or one-at-a-time as you complete them, as you prefer. The individual labs are graded in order, starting with "a", which must be fully correct before "b" is graded, and so on. This assignment is worth 50-points, to be awarded after all labs are successfully completed. Use the "Submit your SP2015 work" link on the class website to post your file for this lab, using the **lab16** folder provided.

Lab 16 Completing The Elevator Simulation [`ElevatorSimulation6.cpp`]
Write `ElevatorSimulation6.cpp`, so that it creates a **Building** object and calls its `step` function in *only one* for-loop. Run the simulation for an indefinite period of simulated time -- no time limit in the for-loop, like there was in lab 15. Allow the user to quit by responding to the "Press ENTER to continue, X-ENTER to quit..." prompt, like in the 2nd for-loop in lab 15.

The purpose of the simulation is to test your elevator design (number, capacity, and speed), to see if it is adequate for your building with its **average rider arrival rate**. Your **design objective** is to include the "right" speed, size, and number of elevators for your building, for your chosen arrival rate of riders for this lab, and your number and location of floors from lab 15. Your simulation should appear to reach a "steady state" (without accumulating large crowds of riders on the floors) after 10 minutes or so. You may have to adjust your elevator configuration (number, size, speed) in your Building class constructor in order to accomplish this.

Here is what to expect:

1. Your program should place a randomly-selected number of riders into the building in every second of the simulation. That random number can be any non-negative whole number. It's expected to vary from second-to-second, but average to the **average rider arrival rate** over time.
2. During the simulation, elevators should be busy (that is, not idle) MOST of the time, but not ALL of the time.
3. After 10 minutes or so, you should *not* see riders accumulating on floors -- their numbers should reach an approximate "steady state".

Your `main` function should be very similar to the one you used for testing in lab 15. The main change is that you are to use a random number generator to decide whether to add any riders in any time step, and how many. You choose the **average rider arrival rate** for riders, for your building. You may also have to change the elevator configuration in your Building design from lab 15, to accommodate your chosen average rider arrival rate. If so, submit the new Building class file(s) to your lab16 folder.

Do *not* use friend relationships, except for overloaded stream insertion operators.

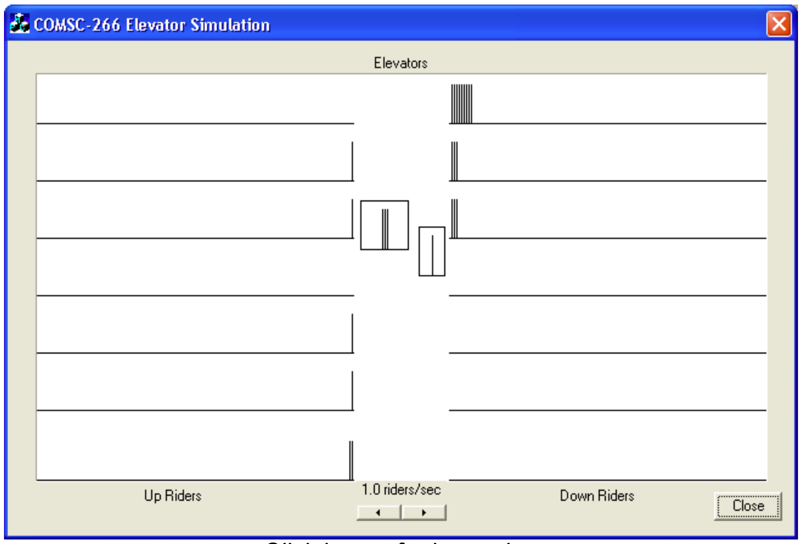
*Post **ElevatorSimulation6.cpp** (and optionally **ElevatorSimulation.exe** -- see below) to your lab16 folder on the COMSC server for credit.*

You might find this useful. The getArrivalsForThisSecond function's parameter is the average number of arrivals per second. It returns a randomly-selected number of arrivals for one second of simulation time, based on the Poisson distribution:

```
int getArrivalsForThisSecond(double averageRiderArrivalRate)
{
  int arrivals = 0;
  double probOfnArrivals = exp(-averageRiderArrivalRate); // for n=0 -- requires cmath
  for(double randomValue = (rand() % 1000) / 1000.0; // requires cstdlib AND srand in main
    (randomValue -= probOfnArrivals) > 0.0;
    probOfnArrivals *= averageRiderArrivalRate / ++arrivals);
  return arrivals;
}
```

Call this function *every second* of your simulation, to get a different number of riders every second, that averages the **averageRiderArrivalRate** over time.

You can create a GUI version of your elevator sumulation, using Visual C++. Right-click here to download a ZIP file with the project files. Expand it to your desktop, where it creates a folder named **ElevatorSimulation**. Copy the CPPs and Hs for your 4 classes (for a total of 8 files) to this new ElevatorSimulation folder. Open the folder and double-click the **ElevatorSimulation.dsw** file, which should start Visual C++. (If it does not, start it yourself and open the project in this folder.) Use Build->Rebuild Solution menu option to create the program.



Click image for large view

You do not have to submit this -- it's for your own enjoyment. But if you want to do so, submit the EXE only, as **ElevatorSimulation.exe**.

Elevator Simulation Index: where to find stuff...

class Rider, lab 8	class Elevator, lab 9	class Floor, lab 14
Rider::operator=, lecture 9	Elevator EXEs, lecture 9	Floor::addNewRider, lecture 14
Elevator::isNearDestination, lab 9	Elevator::moveToDestinationFloor, lab 9	Floor::isPreferredDirectionUp, lecture 14
class Building, lab 15	Elevator::addRiders, lecture 10	Floor::removeDownRiders, lecture 14
getDifferentInts(int, int&, int&), lab 14	Elevator::removeRidersForDestinationFloor, lecture 10	Floor::removeUpRiders, lecture 14
poissonHits(double), lab 15	Elevator::setDestinationBasedOnRiders, lecture 10	GUI simulation option, lab 15

How to pause a console program: *show / hide*

GRADING POLICY *show/hide*