

Project2: Going up?

Due Date: Wednesday April 4, 2007.

Using Greenfoot, you are to create an elevator simulation.

Consider a building n stories tall; to make sure things get interesting: $n \geq 5$. On each floor there are two areas: a waiting room and the elevator lobby. In the building there are P persons randomly distributed among the n floors. Each person $p \in P$ follows the same, though randomized behavior, over and over:

1. Enter the waiting room and wait there for some random amount of time. i.e. A random number of calls to `run`.
2. Exit the waiting room for the elevator lobby and decide, randomly, on another floor to travel to. No person should randomly decide to stay on the floor they are on. After deciding on a floor, the user “presses” the appropriate up or down elevator call button.
3. Eventually, an elevator shows up and takes the person to the desired floor, at which point the person exits the elevator and begins the cycle over again.

This project is open-ended in that one should feel free to be creative. While the description is in terms of a building and a bank of elevators, it could also be cast as taxis (elevators) and passengers. The floors are just a finite set of taxi destinations/known buildings. The people could be ants, the elevators could be bumble bees and the floors could be flowers. Hopefully, you get the idea.

As you did in Project 1, you are to create and eventually hand in a “programming journal.” A Programming Journal is a journal that you will use to record your work (and reflections) on this project. Every time you sit down to work on this project you will generate an entry in your journal. An entry should indicate the date and time you began the work session, the duration of the session, what you worked on, and finally, what you intend to work on during your next work session.

Remember that not all the time spent working on the project is “computer time.” It is recommended that you spend time thinking about and designing your solution prior to actually sitting down at a computer. Your journal should reflect this time as well.

In order to earn a:

- C: You must have a correctly working and reasonably well documented program with a single elevator. You do not need to implement waiting rooms, nor have persistent people. One can create a person on a random floor, have that person take one elevator trip and then be removed from the world. At which point you would create a new person on a random floor (or better yet, on that floor) who will take one elevator trip, etc. You should implement a reasonably efficient elevator control algorithm.
- B: The requirements for a C in addition to excellent documentation. Furthermore, your set of P people should be persistent; don't remove and repeatedly recreate them. You should also have the above described waiting rooms, and at least two reasonably efficiently scheduled elevators.
- A: The requirements for a B in addition to some other open-ended improvements. Here is your opportunity to think outside the box. Here are some examples. These are meant as guidelines/suggestions and not as a set of necessary nor sufficient conditions for an A.
 - Implement a multi-building scenario where each building has a different number of floors. Persons p on the ground floor can randomly elect to go to a different building and starting riding the new building's elevators. This is a real challenge in class design: to take your solution and have it work for multiple building instances with little or no changes!
 - The elevator call panel on each floor's elevator lobby has $n - 1$ buttons instead of just two: up/down. The person hits the correct button of the desired floor. Implement an "optimal" elevator scheduling algorithm.
 - Make one elevator a "single gender" elevator. This world is populated by two different genders; M and F . An M person can only enter the single-gender elevator if and only if it is empty, or there are no F persons in it. The same goes for the F designated people.

1 Extra Credit Opportunity:

For a 10% bump, implement the Blackjack program discussed in class.

Good Luck