<u>Assignments 5a and 5b:</u> Project number 5, page 233.

Write a simulation program for a small airport that has only one runway.

- There will be a queue of planes waiting to land and a queue of planes waiting to take off. However, only one plane can use the runway at a time, so there can be only one takeoff or one landing in progress at any one time.
- Assume that all takeoffs take the same amount of time.
- Assume that all landings take the same amount of time, but this need not be the same as the takeoff time.
- Assume that planes arrive for landing at random times but with a specified probability of a plane arriving during any given minute.
- Similarly, assume that planes arrive at the takeoff queue at random times but with a (possibly different) specified probability of a departure. (Despite the fact that takeoffs and landings are scheduled, delays make this a reasonable assumption.)
- Since it is more expensive and more dangerous to keep a plane waiting to land than it is to keep a plane waiting to take off, landings will have priority over takeoffs.
- Thus, as long as there is a plane waiting to land, no plane can take off.
- Use a clock that is an integer variable that counts the number of minutes simulated.
- Use a random number generator to simulate arrival and departure times of airplanes.

This simulation can be used, among other things, for deciding when the air traffic has become so heavy that a second runway must be built. Hence, the simulation will simulate disaster conditions in which planes crash because they run out of fuel while waiting too long in the landing queue. By examining the simulated situation, the airport authority hopes to avoid real tragedy.

- Assume all planes can remain in the landing queue for the same amount of time before they run out of fuel.
- If a plane runs out of fuel, your simulation will not discover this until the simulated plane is removed from the queue; at that point, the fact that the plane crashed is recorded, that plane is discarded, and the next plane is processed.
- A crashed plane is not considered in the calculation of waiting time.
- At the end of the simulated time, the landing queue is examined to see whether any of the planes in the simulated queue have crashed.
- You can disregard the planes left in the queue at the end of the simulation, other than those that crashed for lack of sufficient fuel.

Use the following input and output specifications:
Note: Ask the user to enter the input values in the same order as shown below. Your program will printout the output values.

Input: (1) The amount of time needed for one plane to land (in minutes);
(2) the amount of time needed for one plane to take off (in minutes);
(3) the probability of an arrival of a plane into the landing queue (this will be a decimal number between 0 and 1);
(4) the probability of an arrival of a plane into the takeoff queue (this will be a decimal number between 0 and 1);
(5) the maximum amount of time that a plane can stay in the landing queue without running out of fuel and crashing (in minutes); and
(6) the total length of time to be simulated (in minutes).
Note: you can assume that one minute is one iteration of your simulation loop.

Output: (1) The number of planes that took off in the simulated time;
(2) the number of planes that landed in the simulated time;
(3) the number of planes that crashed because they ran out of fuel before they could land;
(4) the average time that a plane spent in the takeoff queue; and
(5) the average time that a plane spent in the landing queue.

The output is to be printed to the screen.

Assignment 5 is to create a detailed design of this system. This is to be a text document that will contain the following:

1. A list of all of the classes needed for this simulation.
   - Each class will have a description of what the class is for and what it does.
   - The method headers for all of the methods in each class.
   - A description of what the method is for, and what it does: this will include the Preconditions and Postcondition. The method description should include a brief description of how the method is going to accomplish its functionality. This should not be code, but rather a textural description, although some pseudo code is acceptable. The method parameters and return values should be included, with a description of what they are.
   - A list of the class fields and what they are used for.
2. UML class diagrams for each class.
3. This detail design should be complete enough so that I can determine if your design is going to fulfill the requirements of the assignment.

Assignment 5a is due Feb 13.

Assignment 5b is to create the code from your detail design and to test it to verify that it works correctly. It is acceptable to deviate, as needed, from your detail design during the coding phase if need be, to make this work correctly. Assignment 5b is due Feb. 20.

The file with the main method is to be named Airport.java

Your design should have 3 files: Airplane.java, which will contain the Airplane class,
Runway.java, which will contain the Runway class,
Airport.java which will contain the Airport class.

I would envision that the Airport class will contain the main method, an inputData method, a runSimulation method that will contain the game loop, and an outputResults method that will printout the output.

Printing out the output will not tell you if your program is working correctly. You will need debug statements embedded in your runSimulation method to printout the times when planes entered and exited the queue as well as other information. You will need to send this data to a file to be analyzed after the simulation finishes because there will be more than one screen worth and it will scroll by too fast to analyze. The debugOutput.txt file is an example of the debug statements output you will need for determining if your program is working.
Leave the debug statements in the files you upload to D2L.