Computational Problem Solving CSCI-603
AiRIT Lab 6

6/16/2021

1 Introduction

The powers that be at RIT have decided to start a new airline dedicated to getting students to their Spring Break destination free of charge. The airline will of course be named "AiRIT."

From AiRIT's perspective, each student passenger comprises three important pieces of information:

1. **Full Name** - A first name and last name.
2. **Ticket Number** - A potentially variable length string comprising digits and possibly letters. The first character is always a digit and indicates the passenger's *boarding zone*, which will be a number between 1 and 4.
3. **Carry On** - Indicates whether or not the passenger has a carry-on bag that will need to be stowed in the overhead compartment on the plane.

AiRIT operates a single gate at the Greater Rochester International Airport. There is a separate line at the gate for each boarding zone. As passengers arrive at the gate, they take the next position at the back of the line that corresponds with their boarding zone. Local fire codes insist that the gate only allow a set maximum number of passengers to line up at any one time. Once the gate is full, any remaining passengers must wait outside of the gate until *all* of the passengers already in line have boarded a plane.

AiRIT's boarding policy is similar to that of other airlines. When the aircraft is ready for boarding, passengers will board based on their *boarding zone*, with the highest numbered boarding zone boarding *first*. That is to say that passengers assigned to boarding zone 4 will be the first to board the plane, followed by boarding zone 3, and so on, until all passengers have boarded or the plane is full. Passengers in each zone will board the plane in the order that they arrived at the gate and will load the plane from *back to front*, meaning that, as passengers board the plane, they will move to the unoccupied seat that is closest to the back of the plane. It is assumed that the passengers with carry-on luggage will stow their bags in an available space in the overhead compartments as they make their way to their seats. If the gate empties before the aircraft is full, the aircraft will take off with whatever passengers are on board *before* the next set of passengers lines up.

AiRIT's policy for deplaning is somewhat unique. Once the plane arrives at its destination, passengers *without* carry-on luggage deplane from *front to back*. This means that any passengers in boarding zone 1 should depart first, followed by passengers in boarding

zone 2, and so on. Once all of the passengers without carry-on luggage have disembarked, the remaining passengers will begin deplaning again from *front to back*.

The AiRIT aircraft includes a maximum passenger capacity, but AiRIT runs the plane continuously, 24 hours a day, 7 days a week. As the plane fills up with passengers, it departs, drops its passengers off at their destination, and immediately returns to the gate ready for the next set of passengers. This continues until all student passengers have been safely sent along to their destinations.

1.1 Problem Solving Session

You will work in a group of three or four students as determined by the instructor. Each team will work together to complete the following activities.

1. Write code for a class `Passenger` to represent a student passenger.
   (a) Each passenger has a name (a string), a ticket number (a string), and whether or not the passenger has a carry-on bag. Make sure you use `__slots__` to limit the valid data member names.
   (b) Write a constructor that takes these arguments and initializes the data members.
   (c) Write a method that is suitable when printing out a `Passenger` object. It should return a string that contains the passenger's full name followed by its ticket number and whether or not has a carry-on.
2. Design a class `Gate` to represent the airline gate.
   (a) What data structure(s) will the gate need to maintain in order to allow the passengers to line up for their respective boarding zones?
   (b) There is a fire-code mandated maximum number of passengers allowed in the gate at a time. How will you enforce this?
3. Write the **pseudocode** for the function that, given a passenger and a gate, places the passenger into the appropriate line at the gate. This function should return `True` if there is still room at the gate, or `False` if the gate is full after the passenger gets into line.
4. What data structure(s) will the plane need to keep track of passengers as they load and unload the plane?
5. **(Optional)** Write the **pseudocode** for the function that, given a gate and an AiRIT aircraft, loads passengers onto the plane. For the purposes of this exercise, you can ignore whether or not the passenger has a carry-on.

When your team has completed the required tasks, validate your solutions with your instructor or an SLI.