MIST 4600 – Spring 2014 Assignment B 10 Points. Due: Wednesday by 11:00 p.m. (See eLC.)

Scenario:

You are to modify the the BetterTicketMachine BlueJ project of Chapter 2 to sell tickets to either of **two different destinations**. You will find the start project on eLC both in the Assignments and in Course Content. The start BlueJ folder is named LastnameFirstname-AssignmentB. Before you begin, download the start folder, and rename it by substituting your own Lastname and Firstname. The new machine that you create will

The new name of the class (you change it) will be TwoDestinationMachine (as opposed to "TicketMachine", and it will require four parameters to create a new TwoDestinationMachine object (as opposed to one parameter for the NaiveTicketMachine.) Therefore the constructor signature will now be

```
public TwoDestinationMachine (String pDest1Name, int pDest1Price,
        String pDest2Name, int pDest2Price)
```

Make sure you name this absolutely correctly (spelling not only counts, but it required), and the parameters must be exactly like you see it above or the code will not run when tested. I will not debug an incorrect class signature. I simply mark it completely wrong.

In more colloquial English, the four parameters are: the name of the first destination, price of the first destination in cents, the name of the second destination, and the price of the second destination in cents. For example, the names and prices might be the following:

Destination#	Name	Price
1	Atlanta	500
2	Stone Mountain	400

You will need to make the following changes to what was the BetterTicketMachine class (Every time you make a couple of changes, recompile. Always add {}. (), [] in pairs.):

1. In the field definition part of the code, (the wrapper statement above the constructor) you will need to add several new fields: two String attributes/fields: *dest1Name* and *dest2Name*; two int attributes/fields" *dest1Price* and *dest2Price*. Keep the existing attributes/fields for balance and total.

2. In the constructor, change the constructor to add the new parameters. Use the following parameter names: pDest1Name, pDest1Price, etc. (each of which will start with a small case "p") to initialize the four of the six fields described above. The p indicates 'parameter.'

3. Change the printTicket method.
a. Add a *parameter* to the method (an integer for the pDesiredDestination, either a one or a two). So the method signature will look like:

```
printTicket(int pDesiredDestination)
{
```

b. Use an if statement to ask if the pDesiredDestination is == 1. If the answer to this question is yes, then (all within the curly brackets for this if statement) copy the code from the old BetterTicketMachine printTicket method and modify it to make sure it does the following: Check to make sure the balance is greater to or equal to dest1Price, and if so prints a ticket (to destination 1, with destination 1's price). For example

```
##################
# The BlueJ Line
# Ticket To Museum
# 400 cents.
##################
```

If the balance is not high enough, print out a message indicating how much more must be entered to go destination 1.
c. Add another if statement in the printTicket method to ask if the pDesiredDestination is == 2. If the answer to this question is yes, then (all within the curly brackets for this if statement) copy the code from the old BetterTicketMachine printTicket method and modify it to make sure it does the following: Checks to make sure the balance is greater to or equal to dest2Price, and if so prints a ticket (to destination 2, with destination 2's price). If the balance is not high enough, print out a message indicating how much more must be entered to go destination 2.

d. Add another if statement to ask if the desiredDestination is > 2. If so, have the method print out a message indicating that you must specify the destination by entering either a 1 or a 2.

e. Add another if statement to ask if the desiredDestination is < 1. If so, have the method print out a message indicating that you must specify the destination by entering either a 1 or a 2.

4. For now, delete the getPrice() method.

5. I have put an AssignmentBTester class in the same project. Do not change this tester. When you have finished your program and believe it is correct, you must compile and run the tester to see if it works. Make sure the results are what they should be. Look at the last page of this assignment sheet. Using this tester is similar to what I will do when I grade your program. In addition, I have added a second tester class called AssignmentBTester2. You may use this; I will not give you the expected output. As a challenge, you may wish to create a third tester. Go for it.

(You can also run the tester if you want to check how your program works at any time, though note that it will not compile correctly until your program has all the needed methods, even if the methods are empty – just an open bracket followed immediately by a closed bracket {}, and the constructor's parameters are correct.)

6. Add your name and a comment at the top of your program telling me whether you got the correct output from the tester or not. Indicate what does or does not work. (I already added a place in the top comment (/**   */) so you can do this.)

I suggest that you do your work as follows:

1. Download the zipped BlueJ project and this Word file to your hard drive. Unzip the BlueJ project folder. Rename your copy of this original BlueJ project, saving it as LastnameFirstname-AssignmentB . Keep the zip file as a backup of the start folder.
2. Enter your name as an additional author under @author
3. Enter the date and Assignment B under @version
4. Type your name in the required place in the comment.
5. Modify the code: fix the wrapper and constructor, and add the Java statements. needed to complete the exercise.
6. Compile the TwoDestinationMachine class correcting any errors.
7. Create an object of the TwoDestMachine class and test the new method you wrote.
8. Continue testing and correcting until it works correctly.
9. Test your project using the tester class.
10. **Make sure you have renamed your LastnameFirstname-AssignmentB**
11. Make sure you have made a copy of the project that you are going to submit. You should save this copy on your flash drive or email it to yourself.
12. Correctly zip your BlueJ project folder.
13. Submit your project by uploading and submitting it to the eLC Assignment.

Janine's Programming Tips:

- Compile your code after every couple of changes. That way you know where the error is.
- Use Edit > Auto layout to adjust the indentation. This will help identify missing or extra {}.
- Put the curser to the right of a { or }. The BlueJ editor will highlight the match.
- Always enter {}. (), and [] in pairs to avoid compiler errors.

Correct output from "AssignmentBTester":

```
##################
# The BlueJ Line
# Ticket to Stone Mountain
# 400 cents.
##################

##################
```

```
# The BlueJ Line
# Ticket to Atlanta
# 500 cents.
##################

You must insert at least: 300 more cents.
You must insert at least: 400 more cents.
```