Homework 2 – Travel Buddy Service 1331

Problem Description

Hello! Please make sure to read all parts of this document carefully.

Welcome to CS 1331. In this assignment, you'll be using your knowledge of conditionals and Scanners to create a Travel Buddy Service. To do this, you will create and turn in a **TravelBuddy.java** file that runs code in a main method. You will be using both if-statements and switch statements for this code.

Solution Description

Create a TravelBuddy.java file that will simulate a Travel Buddy service. This service takes you to one of the wonderful Georgia Tech buildings for a fee. You should only create a single Java class called TravelBuddy.java with a main method. **You must have at least ONE switch statement and ONE if statement in your code.**

****Note that print statements indicated with quotes and outlined in the example output should match exactly with your code**.

`TravelBuddy.java`

Name your program TravelBuddy.java. It should work as follows:

1. The Travel Buddy starts out with the message ”Welcome to Travel Buddy Service! Please enter your name here: ”
    a. Notice the formatting of the name given in the example outputs.
    b. The name should have the first letter capitalized and the rest lowercased regardless of what the user inputs (i.e. goKu --> Goku, ash --> Ash)
2. Prompt the user "Where would you like to go, [name]?" and print the menu on individual lines as formatted in the output:

```
The Hive - $6.00
Invention Studio - $7.50
Klaus - $6.90
CULC - $3.20
```

3. Prompt the user to enter their destination of choice.
    a. If the user chooses CULC or Klaus, ask the user "Would you like to stop by Chick-fil-a on the way? (Yes/No)"
        - If they enter "Yes", then multiply their total cost by 1.5
        - If they enter "No", then go straight to the destination without doing anything additional to the cost.
        - If they enter an answer outside of "Yes" or "No" then terminate the service. (See Step 6)
    b. Add the corresponding price of that destination to the total cost and have the service print out "Your current total is $[totalCost]"
    c. If the user inputs an invalid destination, print out "That destination is invalid, goodbye!" and terminate the service. (See Step 6)
4. At the end of the ride prompt ask the user "Do you have a discount? (Yes/No)"
    a. If the user responds "Yes", ask them "What is the discount code?" (response is case sensitive)
        - Based on what they input give them their discount. Use an if statement.

- If the code is “CS1331”, give them a 25% discount on their entire ride. Print out “You have received 25% off your ride!”
- If the code is “IKNOWTHEOWNER”, subtract as much as the user wants.
    - Ask the user “How much would you like to take off?”
    - Subtract that number from the total
    - If the subtracted number is invalid (i.e. not a number), then take off 1.00
    - Print “Taking off $[subtractedNumber] from your total...”
- An invalid code will result in “That discount does not exist! Better luck next time!”

5. Once the service has ended, print the following message: “Thank you for riding with the Travel Buddy Service, [name]! You are now $[totalCost] poorer! We hope you ride with us again! :)"
    a. Round the total to **two decimal places.**
    b. If the total is negative, set the total to 0.
6. NOTE: If the service has been terminated, then the code itself should end.
    a. i.e. There are no further print statements, and the program exits.
    b. Do NOT use System.exit.

Example Output 1 – User Input is Bolded.

(Your program should look **exactly** like this)

```
Welcome to Travel Buddy Service! Please enter your name here: kelly
The Hive - $6.00
Invention Studio - $7.50
Klaus – $6.90
CULC - $3.20

Where would you like to go, Kelly?
The Hive

Your current total is $6.00
Do you have a discount? (Yes/No)
Yes

What is the discount code?
IKNOWTHEOWNER

How much would you like to take off?
2.00

Taking off $2.00 from your total...

Thank you for riding with the Travel Buddy Service, Kelly! You are now $4.00 poorer!
We hope you ride with us again! :)
```

Example Output 2 – User Input is Bolded.

```
Welcome to Travel Buddy Service! Please enter your name here: buZz
The Hive - $6.00
Invention Studio - $7.50
Klaus – $6.90
CULC - $3.20
```

```
Where would you like to go, Buzz?
Invention Studio

Your current total is $7.50
Do you have a discount? (Yes/No)
No

Thank you for riding with the Travel Buddy Service, Kelly! You are now $7.50 poorer!
We hope you ride with us again! :)
```

Example Output 3 – User Input is Bolded.

```
Welcome to Travel Buddy Service!  Please enter your name here: gAnDaLf
The Hive - $6.00
Invention Studio - $7.50
Klaus – $6.90
CULC - $3.20

Where would you like to go, Gandalf?
North Ave

That destination is invalid, goodbye!
```

Rubric

[100] `TravelBuddy.java`

- [10] Program compiles and is able to run (main method)
- [10] Correctly does the following:
  - [3] Stores the user's name in the correct format
  - [3] Options are printed in the correct format
  - [4] Tracks the running total of the fee
- [10] Correctly uses a switch statement
- [10] Correctly uses an if statement
- [10] Asks the questions in the correct order
  - [2] points per question (including prompt for name)
- [20] Exact formatting of output following each question
  - [5] points per question (excluding prompt for name)
- [20] Correct program behavior
  - [10] Correctly adds or subtract points based on destination
  - [10] Asks additional question only when the 2 specified locations are mentioned
- [10] Correct formatting of output at the end of the program

We reserve the right to adjust the rubric, but this is typically only done for correcting mistakes.

Allowed Imports

To prevent trivialization of the assignment, you are only allowed to import **java.util.Scanner**, **java.text.NumberFormat**, **java.util.Locale**, and j**ava.text.DecimalFormat**. You are ***not*** allowed to import any other classes or packages.

Feature Restriction

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or break our auto grader. For that reason, do not use any of the following in your final submission:

- `var` (the reserved keyword)
- `System.exit`

Checkstyle

For this assignment, we will be enforcing style guidelines with Checkstyle, a program we use to check Java style. Checkstyle can be downloaded here.

To run Checkstyle, put the jar file in the same folder as your homework files and run

```
java -jar checkstyle-6.2.2.jar -a *.java
```

The Checkstyle cap for this assignment is **0 points**. This means that up to 0 points can be lost from Checkstyle errors. Any deductions for Javadoc comments will not be counted under Checkstyle until later homework assignments.

Collaboration

Collaboration Statement

To ensure that you acknowledge a collaboration and give credit where credit is due, **we require that you place a collaboration statement as a comment at the top of at least one .java file that you submit**. That collaboration statement should say either:

I worked on the homework assignment alone, using only course materials.

or

In order to help learn course concepts, I worked on the homework with [give the names of the people you worked with], discussed homework topics and issues with [provide names of people], and/or consulted related material that can be found at [cite any other materials not provided as course materials for CS 1331 that assisted your learning].

Allowed Collaboration

When completing homeworks for CS1331 you may talk with other students about:

- What general strategies or algorithms you used to solve problems in the homeworks
- Parts of the homework you are unsure of and need more explanation
- Online resources that helped you find a solution
- Key course concepts and Java language features used in your solution

You may **not** discuss, show, or share by other means the specifics of your code, including screenshots, file sharing, or showing someone else the code on your computer, or use code shared by others.

Examples of approved/disapproved collaboration:

- **approved**: "Hey, I'm really confused on how we are supposed to implement this part of the homework. What strategies/resources did you use to solve it?"
- **disapproved**: "Hey, it's 10:40 on Thursday... Can I see your code? I won't copy it directly I promise"

In addition to the above rules, note that it is not allowed to upload your code to any sort of public repository. This could be considered an Honor Code violation, even if it is after the homework is due.

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradecope:

- TravelBuddy.java

Make sure you see the message stating "HW2 submitted successfully". From this point, Gradescope will run a basic autograder on your submission as discussed in the next section.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the homework. We will only grade your last submission: be sure to **submit every file each time you resubmit**.

Gradescope Autograder

For each submission, you will be able to see the results of a few basic test cases on your code. Each test typically corresponds to a rubric item, and the score returned represents the performance of your code on those rubric items only. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

- Prevent upload mistakes (e.g. non-compiling code)
- Provide basic formatting and usage validation

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or checkstyle your code; you can do that locally on your machine.

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Piazza for a note containing all official clarifications