

CS161: Introduction to Computer Science
Homework Assignment 4
due 2/14 by 11:59pm

Writing Java Classes

Up until now, we've used Java classes written by other people – e.g., the `String`, `Scanner`, and `Random` class. Although the Java library has many classes for you to use, about 99% of the programming you do out in the wild requires you to design and write your own classes. This is the skill we are currently practicing.

Create a new BlueJ project named `hw4`. All of your work should be contained inside of this project.

Written Exercises

Double click on the sheet of notebook paper icon in BlueJ. This will open a README file where you can type your answers to the following exercises:

- Complete exercises 4.4, 4.10, and 4.11
- Consider the following Java program that is missing certain keywords:

```
/**
 * This class stores information about a person
 */
public _____ Person {

    // Instance variables
    private String name;
    _____ int age;

    // The constructor
    public _____ (String theName, int theAge){
        name = theName;
        age = theAge;
    }

    public _____ introduceSelf(){
        System.out.println("Hi, my name is " name ".");
    }

    _____ int getAge(){
        _____ age;
    }

    public _____ getName(){
        return name;
    }
}
```

Fill in each blank with the missing Java keyword.

Tally Counter

Complete programming exercise 4.1 in the textbook. This asks you to write a class that represents a tally counter (shown in the picture). A tally counter is a simple device for counting objects – e.g., counting the number of people in an auditorium.



This question asks you to write a “driver class” called `CounterTest`. This means you should create a second Java class (called `CounterTest`) with a `main` method.

Light Bulb

Complete programming exercise 4.2 in the textbook. Notice this question does not give you as many details as the previous programming question. Your class should contain the following:

- appropriate instance variable(s)
- a constructor
- method(s) for turning the lightbulb on and off
- a `toString()` method



An ORCA Card

Write a Java class that represents an ORCA card. An ORCA card is used to pay for bus, train, and ferry trips in Pierce and King counties. When using the real card, you add funds to it and can then swipe the card to charge rides on the various services. In addition to keeping track of the balance and paying for rides, our ORCA card will also keep track of the number of trips that have been taken.



For full credit, your class should contain all of the methods described below. They should have exactly the same name as shown, take the correct arguments, and return the correct information. (I will run a program that creates instances of your class and tests them, and if your names or other details differ, my testing code won't compile.)

This question is less specific about the instance variables you'll need. Instead, you'll have to think about what variables you need in order to implement the methods below.

1. Create a Java class called `OrcaCard`.
2. Your class needs to be able to keep track of how much money is currently stored on the card. This balance should be set by the constructor, which should take a single argument (a double) specifying the initial balance.

3. Your class should have an `addMoney()` method that takes a single argument (the amount to add to the current balance) and adjusts the balance but doesn't return anything.
4. We'll simulate the process of "swiping" the card via the `buyTrip()` method. This method should take a double (representing the cost of the trip). Inside the method, you need to decrease the balance by the cost of the ticket *plus the cost of taxes*.
For example, in Washington state, taxes are 6.5%. Let's say I want to take a ferry from Tacoma to Vashon Island. The cost of the ticket itself is \$5.25. So, if I call your method and pass in 5.25, the balance on my card should decrease by \$5.25 plus an additional 0.34 cents for taxes.
Use a 6.5% tax rate in your code. A good idea is to store this value as a `final` variable.
5. We'll also add a `getAverageTripCost()` method. It doesn't need any arguments, but should return the average cost of the trips paid for by this card so far. This method will probably require the most thought from you – it may require you to add more instance variables to your class, or to change how other methods in the class work.
6. Finally, write a `toString()` method that returns a string representation of the ORCA card object. Feel free to personalize this as you see fit, but the string should contain at least the card's current balance and the number of trips taken.
7. For full credit, your code should contain Javadoc comments and inline (`//`) comments.

Here's an example of how I might use your `OrcaCard` class:

```
public static void main(String[] args){

    // Create a new card with a balance of $20.50
    System.out.println("Creating a new ORCA Card:");
    OrcaCard oc = new OrcaCard(20.50);
    System.out.println(oc.toString());

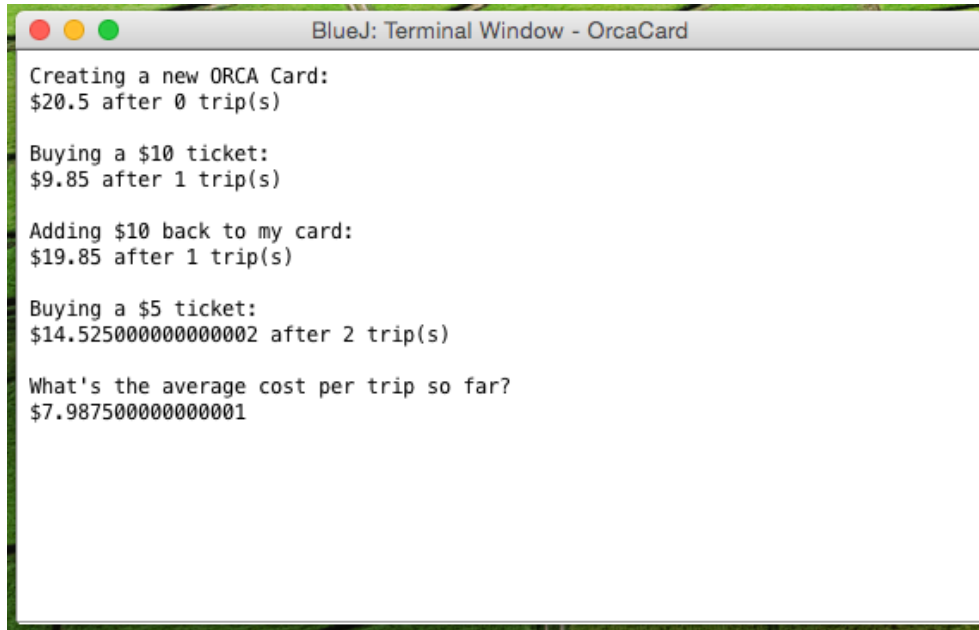
    System.out.println("\nBuying a $10 ticket: ");
    oc.buyTrip(10.00);
    System.out.println(oc.toString());

    System.out.println("\nAdding $10 back to my card: ");
    oc.addMoney(10.00);
    System.out.println(oc.toString());

    System.out.println("\nBuying a $5 ticket: ");
    oc.buyTrip(5.00);
    System.out.println(oc.toString());

    System.out.println("\nWhat's the average cost per trip so far?");
    double avg = oc.getAverageTripCost();
    System.out.println("$" + avg);
}
```

Here is what would be printed to the screen:



```
BlueJ: Terminal Window - OrcaCard
Creating a new ORCA Card:
$20.5 after 0 trip(s)

Buying a $10 ticket:
$9.85 after 1 trip(s)

Adding $10 back to my card:
$19.85 after 1 trip(s)

Buying a $5 ticket:
$14.525000000000002 after 2 trip(s)

What's the average cost per trip so far?
$7.987500000000001
```

Note that we purchased two tickets for \$10 and \$5 dollars respectively. If we don't take taxes into consideration, the average trip cost is \$7.50 dollars per trip. However, if we take taxes into consideration, we're averaging about \$7.99 dollars per trip.

Also, you don't need to worry about the long decimal point numbers.

Common Mistakes

Here are some common mistakes you might want to check before you submit your assignment:

- Having instance variables that aren't actually being used! Once you're done, go through and make sure you're actually using each instance variable. Delete an unused variables.
- Not initializing all of your instance variables in the constructor. Every variable that is not `final` should be given an initial value in the constructor.
- Making a variable an instance variable when really it should be a local variable. Is an instance variable only ever used in one method? This means it should be moved into that method – i.e., it should be a local variable.

Extras

Looking for something more to do? How about adding code to your `buyTrip()` method so that it also prints out a simulated ticket that shows (for example): the cost of the ticket, the amount paid in taxes, the remaining balance on the card, and a random time of day. This would be very useful so that we don't have to call the `toString` method after buying a ticket to see our remaining balance.

Submitting your assignment

Your `hw4` folder should contain your written exercises and the classes you wrote. Please remember to rename your folder

`hw4_firstName_lastName`

before you zip it.