ASSIGNMENT 1
First Java Assignment

COMP-202A, Fall 2011, All Sections

Due: Thursday, September 22nd, 2011 (23:30)

Please read the entire pdf before starting.

You must do this assignment individually and, unless otherwise specified, you must follow all the general instructions and regulations for assignments. Graders have the discretion to deduct up to 10% of the value of this assignment for deviations from the general instructions and regulations. These regulations are posted on the course website. Be sure to read them before starting.

Part 1:	0 points
Part 2, Question 1:	40 points
Part 2, Question 2:	20 points
Part 2, Question 3:	40 points
	100 points total

Before starting on the assignment, you should download from the course website the file `assignment1.zip`. Inside this file will be 3 directories:

- Question1: `Question1Testfile.txt`
- Question2 : This directory will contain the file `TaxCalulator.java`
- Question3: This directory will contain the files `TypeQuestions.java` and `typeexpressions.txt`

Each of these files will be used for different parts of the assignment.

It is very important that you follow the directions as closely as possible. The directions, while perhaps tedious, are designed to make it as easy as possible for the TAs to mark the assignments by letting them run your assignment through automated tests. While these tests will not determine your entire grade, it will speed up the process significantly, which will allow the TAs to provide better feedback and not waste time on administrative details. Plus, if the TA is in a good mood while he or she is grading, then that increases the chance of them giving out partial marks :)

Part 1 (0 points): Warm-up

Do **NOT** *submit this part, as it will not be graded. However, doing these exercises might help you to do the second part of the assignment, which will be graded. If you have difficulties with the questions of Part 1, then we suggest that you consult the TAs during their office hours; they can help you and work with you through the warm-up questions.*

Warm-up Question 1 (0 points)

Create a file called `HelloWorld.java`, and in this file, declare a class called `HelloWorld`. This class should define only one method called `main()`. In the body of this method, use `System.out.println()` to display "`Hello world!`". You can find such a class in the lecture slides; make sure you can compile and run it properly.

Warm-up Question 2 (0 points)

Create a file called `A.java`, and in this file, declare a class called `A`. This class should define only one method called `main()`. In the body of this method, use `System.out.println()` to display the following pattern:

```
   A
  A A
 AAAAA
A     A
A     A
```

Warm-up Question 3 (0 points)

Consider the following 2-d matrix:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Write a Java program that first reads 4 doubles, representing a,b,c, and d from the keyboard. It then outputs to the screen the determinant of the matrix.

For a 2x2 matrix, the determinant is always equal to $a * d - b * c$

Warm-up Question 4 (0 points)

Now consider the same question except on a 3x3 matrix:

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Write a Java program that first reads 9 doubles, representing the 9 letters above from the keyboard. It then outputs to the screen the determinant of the matrix.

For a 3x3 matrix, the determinant is always equal to

$$a * (i * e - f * h) - b * (d * i - f * g) + c * (h * d - e * g)$$

Part 2

The questions in this part of the assignment will be graded.

Question 1: Cheating on your calculus assignment (40 points)

The purpose of this question is to practice writing a full Java program from start to finish. You will practice reading and writing to and from the screen and keyboard, respectively.

The following code should be put into a class `Calculus` *and thus a file* `Calculus.java`

Your friend is taking calculus and is having a hard time taking derivatives and integrals. Unfortunately (s)he has an assignment due tomorrow that forces him to take hundreds of derivatives and integrals. You, as a new computer science student, realize that you can help by providing a computer program that will calculate derivatives and integrals.

Write a Java program that does the following:

1. Display a prompt and ask the user to enter his or her name.
2. Display a prompt and ask the user to enter three integers, called a,b,c respectively. These numbers will represent the coefficients of a quadratic equation:

$$ax^2 + bx + c$$

3. Print the formula they have entered on the screen.
4. Next print to the screen the derivative of the equation as a formula. For a quadratic equation above, the derivative will be

$$2ax + b$$

   Your output should fill in the actual values for a and b and perform any necessary multuplications.
5. Next ask the user to enter a value for x. x can be any number ,not just an integer, so you should choose an appropriate type for the variable.
6. Print to the screen the result of $2ax + b$
7. Next, print to the screen the formula for the *antiderivative* of the function. If the original function is $ax^2 + bx + c$ then the antiderivative will be:

$$\frac{a}{3.0}x^3 + \frac{b}{2.0}x^2 + cx$$

   Once again, your output should not include the letter a, b, or c. If you see what appears to be odd division results, try entering 3.0 instead of just 3 and 2.0 instead of just 2. This is a concept known as *integer division* in Java and will be covered later in the course.
8. Finally, print a message with your name followed by the name of your friend which was entered in the first step. This message should say that your friend permanently owes you for helping them out.

A sample run of this program should look as follows:

```
richter:~$ java Calculus
Please enter your name
Jorg
Now enter 3 numbers
4
```

```
5
6
The formula you have entered is:
4x^2 + 5x + 6
The derivative of the equation is
8x+5
Enter a value for x
4
The derivative at the above point is:
37.0
The anti-derivative of the equation is
1.3333333333333333x^3 + 2.5x^2 + 6x
Jorg, you owe Dan Pomerantz forever!
```

Your program must adhere to this. You **must** verify this output by doing the following:

1. Find the file `Question1TestFile.txt` from the downloaded zip and move it to the same folder as the `Calculus.class` file. (This will normally be the same folder as the `Calculus.java` file.)
2. From the command prompt, navigate to the directory of your Java program and type

   ```
   java Calculus < Question1TestFile.txt
   ```

Except for your name, the output should be identical (with the exception of the number of decimal points or a rounding error) to the above code.

Correction: Depending on your operating system the test file may not "echo" the input. For example the output when running on the test file may not include the name "Jorg" or the numbers 4, 5, and 6. This is fine as well. Your program may look like the following instead:

```
richter:~$ java Calculus < Question1TestFile.txt
Please enter your name
Now enter 3 numbers
The formula you have entered is:
4x^2 + 5x + 6
The derivative of the equation is
8x+5
Enter a value for x
4
The derivative at the above point is:
37.0
The anti-derivative of the equation is
1.3333333333333333x^3 + 2.5x^2 + 6x
Jorg, you owe Dan Pomerantz forever!
```

Question 2: Tax Calculator (20 points)

The purpose of this question is to give you a quick sample of how Java could be used to create a real-world, if small application. It will also give you practice with looking at code that someone else has written and see how one piece can contribute to the entire puzzle.

In this question, you will modify an existing program by adding to it.

Inside the directory for assignment one, you will find a file `TaxCalculator.java` You can compile and run this program as any other program by typing `javac TaxCalculator.java` followed by `java TaxCalculator` . Note that since this program uses graphics it may be difficult although not impossible to run via telnet or ssh. When you do this, you will see a small window pop-up with several textfields. You can type numbers into the 3 white ones, but not into the 3 gray ones.

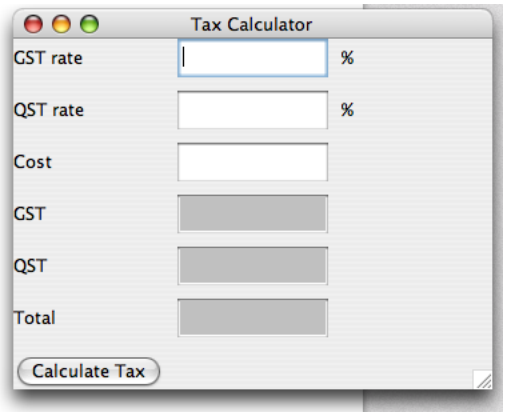

Figure 1: What the TaxCalculator program will look like when you first run it

First fill in values for the GST and QST (see important note below) rates as well as enter a price. Next, hit the button. You will see that it always shows zero. Your task on this question, will be to modify the two methods called `calculatePercentage` and `calculateTotal` so that the methods output the correct results.

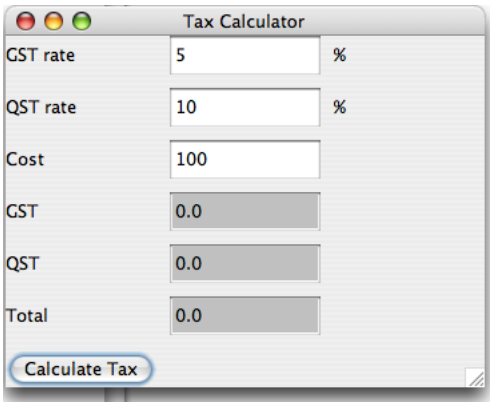

Figure 2: What the TaxCalculator program will look before you make any changes.

After changing these methods, a sample run of the program would look like the screen shot below.

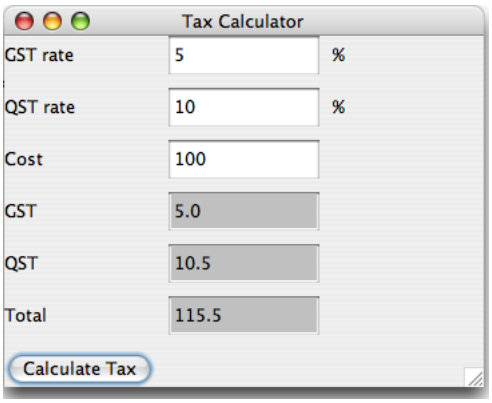

Figure 3: The correct results of the tax calculator after making the changes

You should change the methods `calculatePercentage` and `calculateTotal` as follow:

- `calculatePercentage` This method takes as input two doubles, `original` and `percent`. The method should return `percent` percent of `original`. For example, if `original` has the value 100 and `percent` is 5, the method should return 5
- `calculateTotal` This method takes as input three doubles `price`, `gstRate`, and `qstRate`. The method should return the total price after computing the GST and QST.

  Important note: The GST is calculated on the original cost, but the QST is calculated on the original cost *plus* the GST. So if, for example, the cost is $100, the GST is 10%, and the QST is 10%, then the cost after applying the GST is $110, and the cost after applying the QST is $121 . That is, we add 10 percent of 110.

To find the part of the file to edit, you should look inside the file `TaxCalculator.java` for the words "YOUR CODE GOES HERE"

Question 3: Types and Expressions (40 points)

In the following, you will run a Java program to answer several questions about types. Some of these may require you to do some investigation on your own. Of course, you are welcome to run the code in a Java program that you write to confirm the answers.

Make sure that the file `typeexpressions.txt` and `TypeQuestions.java` are in the same folder. Compile the program `TypeQuestions.java` as you normally would. It should produce a file `TypeQuestions.class`. Once again, verify that `typeexpressions.txt` is in the same folder as this.

Next, run the class `TypeQuestions` by typing `java TypeQuestions`. This will open up a form once again (see figure). After the header where you can fill in your name and student ID, there are 4 columns in the window:

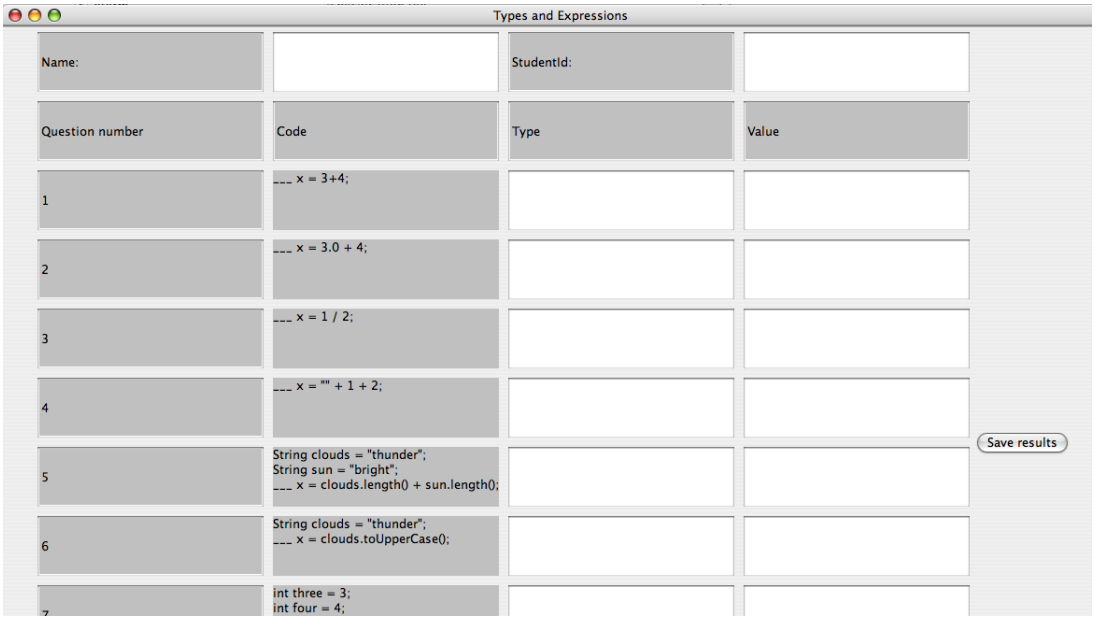

Figure 4: The program where you can enter your answers for question 3.

1. The question number
2. A piece of Java code or an expressions
3. A blank value where you will write the type of the expression
4. A blank value where you will write the value of the expression

In each case, the final line of the Java code will "declare" a variable `x`, but it's type will be "invisible". Your task is to fill in both the variable's type and value in the two columns.

For example, if the code shows as:

```
___ x = 3;
```

you should write under "Type" *int* and under "Value" *3*. For full marks on the question, you should write the closest type to the expression. For example, although the literal `3` can be converted to a `double` implicitly, you should write `int` since the literal 3 is an int. On the other hand the literal `3.0` should be considered a `double` since although it could be explicitly cast to an `int` by writing `(int) 3.0` , the literal `3.0` is itself a double.

After you finish, hit the submit button. The program will verify that you have entered valid types in all of the type fields (`int`, `double`, `short`, `long`, `boolean`, `char`, `byte`, `float`, or `String`) and that you have not left any values blank. *It will not check for correctness, only that your answers are all filled in.*

If there is an error, it will be displayed with a short description of the problem. If you want to fix it, you may. If not, you may proceed.

At this point, a file will be created in your folder called `TypeExpressionResults.txt`. It will contain your name, student id, as well as your answers. It is your responsibility to verify that this file looks like it contains all your answers. **Before exiting the program, you should verify this file exists and that you can find it in order to avoid potentially losing your work.** Then submit this file along with the rest of the assignment.

If, after trying to enter this on the computer, you have not been able to generate a file using the computer program, you may, **as a last resort**, create the file manually. If you do this, you should put one question per line and the format of your line should be the type followed by the value. You also must include your Name and StudentId at the top of this file and should mention the problems you encountered in the `Confessions.txt` file (see description below).

What To Submit

`Calculus.java`
`TaxCalculator.java`
`TypeExpressionResults.txt`
`Confession.txt` (optional) In this file, you can tell the TA about any issues you ran into doing this assignment. If you point out an error that you know occurs in your problem, it may lead the TA to give you more partial credit. On the other hand, it also may lead the TA to notice something that otherwise he or she would not.