Homework #3

Introduction to Java primitive data manipulation and `for` loops

1. Aim

The purpose of this lab is to:

a. Introduce you to primitive data manipulation.
b. Introduce you to the use of variables.
c. Introduce you to **`for`** loop structures.

It is recommended that you read and understand all the instructions below before starting this exercise.

2. Files Needed

You will be provided with one starter file for this assignment: **EinsteinTrick.java**. You will not be provided with starter files for the other programs. For the other programs, you will be creating your own Java programs from scratch. Create a project in Eclipse for this assignment. Add the **EinsteinTrick.java** file to the project (in the src directory) and refresh it. Then for each of the other programs below, you will create a new Java class in the project. When you create the classes in Eclipse, be sure to have Eclipse create the `public static void main` method for you. Make sure you name the class exactly as specified. Be sure to include the standard header comments at the top of each file. The standard header comments are:

```
// Name:
// Email:
// Date:
// Description: <insert short description of the program here>
```

3. To be Handed In

The files **EinsteinTrick.java**, **ForLoops.java**, and **TempConvert.java** should be sent to me when you have them completed. Be sure to name the files/classes exactly as specified.

4. Exercises

Part I: Declaring and using simple variables; performing integer arithmetic

Move the file **EinsteinTrick.java** to the “src” folder which was created for the project, and add the file to the project by refreshing the project (press F5). Be sure to fill in the comment header at the top of the file, and then complete the following exercise by writing appropriate Java code.

It is said that Albert Einstein used to take great delight in baffling friends with the puzzle below.

First, write the number 1089 on a piece of paper, fold it, and hand it to a friend for safekeeping. What you wrote down is not to be read until you have completed your amazing mental feat. Next, ask your friend to write down any three-digit number, emphasizing that the first and last digits must differ by at least two. Close your eyes or turn your back while this is being done. Better still, have someone blindfold you.

After your friend has written down the three-digit number, ask him to reverse it, then subtract the smaller from the larger.
Example: 654 - 456 = 198.

Once this is done, tell your friend to reverse the new number.
Example: 198 becomes 891.

Next ask your friend to add the new number and its reverse together.
Example: 198 + 891 = 1089.

If all goes as planned, your friend will be amazed. The number you wrote down at the start -- 1089 -- will always be the same as the end result of this mathematical trick.

Program Specifications

Your program will play the Einstein game as follows (the first two steps are done for you):

1. Print a message to the user about the game and explain the rules.
2. Prompt the user for a 3 digit number as described. Note: to enter a number when prompted, you will need to click your mouse in the "console" window where the prompt appears and then type your number and press enter.
3. Print out the user's number and the reverse of that number.
4. Print out the difference between the entered number and the reversed number. This should always be a positive number.
5. Print the reverse of the difference.
6. Print the sum of the difference and the reversed difference. It should be 1089 for any 3 digit number.

Assignment Notes:

One of the main issues here is that we are working with integers, and division with integer numbers is a little different. Integer division always returns an integer result. Thus 10/3 yields 3. The operation 4/5 yields 0.

There is another operator, called the modulus operator, which indicates the remainder after a division. The modulus operator is indicated by the % sign. Thus 10%3 is 1 (a remainder of 1). 4%5 is 4.

You can use these facts to gather the digits in the hundred's, ten's and one's place of a three digit integer, and then combine the digits mathematically to reverse the number. Hints: to get the one's digit of any integer you use modulus division by 10, and to remove the one's digit of any integer you use integer division by 10. For example: 1234%10 produces 4, while 1234/10 produces 123.

One other problem: The specification mentions that we should always subtract the smaller of the two numbers (the user number or its reverse) from the larger to produce a positive difference. Another way to do this is to take the absolute value of any subtraction. You can get the absolute value of any number by using the Math.abs() function. Thus Math.abs(-27) produces 27, and Math.abs(45) produces 45.

This program requires that you enter an integer value when the program is running. After you start the program, it will prompt you to enter data and then wait for you. You will need to click on the "Console" window at the bottom of Eclipse to make it the active window, and then you can type in your data and press enter. The code to read this number in has already been provided for you. You need to write the rest of the code to manipulate that number (steps 3-6 above).

Here is a sample execution to show you how your program should behave (the user input is in **green**):

```
This program demonstrates a favorite mathematical trick of Albert Einstein.
When prompted, enter a 3 digit number such that the first and last digits
of the number differ by at least 2. The final result will always be 1089.

Please enter a 3-digit integer: 396
The reverse of 396 is: 693
The difference between 396 and 693 is: 297
The reverse of the 297 is: 792
The sum of 297 and 792 is: 1089
```

Here is another sample execution:

```
This program demonstrates a favorite mathematical trick of Albert Einstein.
When prompted, enter a 3 digit number such that the first and last digits
of the number differ by at least 2. The final result will always be 1089.

Please enter a 3-digit integer: 872
The reverse of 872 is: 278
The difference between 872 and 278 is: 594
The reverse of the 594 is: 495
The sum of 594 and 495 is: 1089
```

Part II: Writing for-loops

Create a Java class called **ForLoops.java.** In that file, write `for` loops to produce the following output, with each line 60 characters wide:

```
------------------------------------------------------------
_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-_-^-
001122334455667788990011223344556677889900112233445566778899
------------------------------------------------------------
```

Even though you could produce the above output with four simple println statements, you must use for loops to capture the repeated patterns (where each repeated pattern is as small as possible; e.g., the first line is 60 repeated dashes and not 30 repeated double dashes). You should also use methods as appropriate to eliminate redundancy (i.e., the first and last lines are identical). Additionally, you are required to use nested for-loops to print the individual digits that appear on the 3rd line of the output; you are not allowed to simply print the pattern "`00112233445566778899`" three times, rather you will need an additional loop to generate the digits 0 through 9, and then another loop to repeat that digit twice.

Part III: Performing calculations in a for-loop

Create a Java class called **TempConvert.java.** In that file, complete the following exercise by writing appropriate Java code.

With the economy still struggling in a recession, people are having to sell things at a deep discount to make ends meet. This is bad for you if you are the seller, but this is great for you if you happen to be in the market for something and you have some cash to spend. In fact, you just happened to find the steal of a deal while at home over the summer break. You got a great deal on a two-year-old Honda Civic. It turns out there was only one problem: the car was a Canadian model which was originally purchased in Toronto (you bought it while at home in Buffalo, NY). This is not a big problem, just a nuisance – since the dash is all in metric. The speedometer is no big deal, as it lists MPH right next to the KM/H. But what bothers you is the climate control system. The digital cabin thermostat displays temperatures in Celsius rather than Fahrenheit, and there is no way to change it.

Your task is to write a program, **TempConvert.java**, which will produce a conversion table that you can tape to your dash. The table should list Celsius temperatures from 10 to 40 (inclusive) going in steps of 2 degrees, and for each temperature print the equivalent Fahrenheit temperature. Both the Celsius and Fahrenheit temperatures are floating point values, so be sure to use variables of type **double**. The program should be written in such a way that it would be easy for someone to change it to print a different range of temperatures or to change the step value (i.e., the program should use variables for those three values).

Mathematically, the formula for the temperature conversion is:

```
tempF = (9/5)*tempC+32;
```

Where `tempC` is the temperature in degrees Celsius, and `tempF` is the temperature in degrees Fahrenheit.

Here is a sample of the output your program should produce. The values are separated by a single tab character.

```
Temperature conversion table
============================

Celsius	Fahrenheit
10.0	50.0
12.0	53.6
14.0	57.2
16.0	60.8
18.0	64.4
20.0	68.0
22.0	71.6
24.0	75.2
26.0	78.80000000000001
28.0	82.4
```

```
30.0     86.0
32.0     89.6
34.0     93.2
36.0     96.8
38.0     100.4
40.0     104.0
```

Many students wonder why the one number is printed as 78.80000000000001. As was mentioned in lecture when discussing data types, the computer cannot represent all floating point values exactly. This is an example where a calculation produced a result that was close to the correct floating point value – but off just a bit.

5. Additional requirements

A. You must start your program with header comments that provide your name, VUnetID, email address, the date the program was last modified, an honor statement ("I have neither given nor received unauthorized help on this assignment"), and a short description of the program (see the examples distributed with homework #2).
B. You should use a consistent programming style. This should include the following.
   a. Meaningful variable & method names
   b. Consistent indenting
   c. Use of "white-space" and blank lines to make the code more readable
   d. Use of comments to explain pieces of tricky code
   e. A descriptive comment before each method (other than main, which already has a short description of the program)

See the code examples in the class text for a good formatting style.

6. Submission for grading

Once you have completed the exercise, submit the files **EinsteinTrick.java**, **ForLoops.java**, and **TempConvert.java** for grading by emailing all three files at once to me at roth.jerry@gmail.com. Be sure to submit the **source** files (the **.java** files in the **src** directories) and not the compiled byte-codes (the **.class** files in the **bin** directories).

7. Grading

This lab is worth 30 points, each part being worth 10 points. Your grade will be based on whether your solution is correct or not, and on how closely you followed the directions above. Remember that programming style will now be a part of your grade (on this and all subsequent assignments).