ASSIGNMENT 1

COMP-202, Winter 2016, All Sections

Due: February1st, 2016 (23:59)

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 15% 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.

Question 1:	50 points
Question 2:	50 points
	100 points total

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, in some cases through automated tests. While these tests will never be used to determine your entire grade, they speed up the process significantly, which allows 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. You are responsible for knowing all of the material in these 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)

Practice with Binary:

Humans usually operate in base 10, probably because most of us have ten fingers. When operating in base 10, we think of numbers as having a `one`s column, a `ten`s column, a `100`s column, etc. These are all the powers of 10.

There is nothing special about 10 though. This can in fact be done with any number. In base 2, we have each column representing (from right to left) 1,2,4,8,16,etc. In base 3, it would be 1,3,9,27, etc.

Answer the following short questions about number representation and counting.

1. In base 10, what is the largest digit that you can put in each column? What about base 2? Base 3? Base n?
2. Represent the number seven in base 7.
3. Represent the number seven in base 2.
4. What binary number is equal to the sum of these two binary numbers? 11001101 + 100101010
5. What is the number from the previous part in base 10?
6. What is the binary number for 11010010 - 11000101?
7. And what is the number from the previous part in base 10?

Warm-up Question 4 (0 points)

This question is designed to help you practice with types.

Background: As discussed in the lectures, there are different types of values in programming languages such as Java. For instance, we can have integers like the number 5, real numbers like the number 5.1, boolean values like true and false, characters like 'm', or more complex types like Strings e.g. "we all live in a yellow submarine". We can compare, contrast, display and apply operators to values of these different types.

Recall also that we can assign values to variables that are declared with the appropriate type: char for storing characters, int for integers, double for real numbers, boolean for logical values, and several others.

When you declare a variable in Java, you need to specify the type. This involves thinking about a real world piece of data and figuring out what type you should use.

For each of the following, what type would best represent the data? Are there any that cannot be represented in Java?

1. The name of a day in the week.
2. Your letter grade in the course.
3. Your numeric grade in the course (as a percentage).
4. The name of the planet that your COMP 202 teacher is from.
5. The temperature outside.
6. The mathematical number π
7. The name of the author who wrote the Hitchhiker's Guide to the Galaxy?
8. The average number of brain cells in a human.
9. The average number of brain cells of a McGill student after three years of parties.
10. The (estimated) number of molecules in the universe.
11. The result of multiplying two integers together
12. The result of dividing an integer by another integer.
13. "Is the date today Sept 11?"
14. Does this string "i am happy" start with the character 'i'?

Warm-up Question 5 (0 points)
Logic

1. What does the following logical expression evaluate to?

```
(False or False) and (True and (not False))
```

2. Let a and b be boolean variables. Is it possible to set values for a and b to have the following expression evaluate as False?

```
a or (((not b) or (not b)) or (b or (not a)))
```

Warm-up Question 6 (0 points)
Write a method `swap` which takes as input two int values x and y. Your method should do 3 things:

1. Print the value of x and y
2. Swap the values of the variables x and y, so that whatever was in x is now in y and whatever was in y is now in x
3. Print the value of x and y again.

For example, if your method is called as follows: `swap(3,4)` the effect of calling your method should be the following printing

```
inside swap: x is:3 y is:4
inside swap: x is:4 y is:3
```

Now, create 2 int (integer) variables in the main method. Call them x and y. Assign values to them and call the swap method you wrote in the previous part.

After calling the `swap()` method—inside the main method— print the values of x and y. Are they different than before? Why or why not?

Part 2

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

Question 1: My first (graded) Java program (50 points)

The following should go inside a class called `ParseDigits` and thus a file called `ParseDigits.java`

Write a Java program that takes as input a 5 digit integer number and outputs three different values, one on each line:

1. The product of the digits
2. The sum of the digits
3. A statement saying which of those two numbers is larger.

The program should obtain its input from **String[] args** in the main method. That means that the input will initially be of type **String**. The input will be in the first position in a list of Strings. You can access this String-type list element and convert it to an integer using the following statement:

```
int number = Integer.parseInt(args[0]);
```

For example, if the number entered is 54321, the program should display:

```
The sum of the digits is 15
The product of the digits is 120
The product is larger than the sum
```

You can assume that the input is valid. That is, you can assume that the number entered has five digits, and that it is a valid non-negative integer number. Be sure to include descriptive text on each line, explaining what the following number corresponds to.

Question 2: Using Methods for Marking (50 points)

The goal of this question is to have you write several methods that are all very similar in nature in order to experiment with the different sorts of methods. You should put all of your code into a class `Marking` *and thus a file* `Marking.java`*.*

You cannot use any methods from the Math library in solving these problems.

2a)Void method that prints something

Write a method `printMaximum` that takes as input two `double` arguments and **prints** which is the larger number. You should include both numbers as part of the message. For example, your message could be "34.0 is larger than -12.2" or "5.0 and 5.0 are the same number". Note that for full marks, this message **must** be written on one line. (Remember that one way to do this is the `+` operator. Another way is to recall the difference between `System.out.print()` and `System.out.println()`).

Hint: To test your method, you can write a main method like you did in question one. The main method will not be graded, but without it, you won't know whether or not your method works! Your main method should call this method and verify the results. For example, your main method could be as follows. You should think of other cases to test!

```
public class Marking {
    public static void main(String[] args) {
        printMaximum(55,10);
        printMaximum(-44, -16);
    }
    public static void printMaximum(double num1, double num2) {
        // your method definition here
    }
}
```

2b)Method that returns something

The previous method that you wrote is not very general. Remember that one of the key ideas of methods is to write a general piece of code that can be re-used.

What stops it from being general? Well, your method is only useful if you want to **print** which number is larger. Suppose you wanted to use the larger of two numbers to perform some calculation. You don't have access to the result of this computation in any other method.

We will now make another method *inside the same Marking class* that is more general.

Write a method `maximum` inside of `Marking` that, like the previous method, takes as input two `double` values, but now returns the larger of the two numbers. Your method must **not** print anything.

To test your method, you will need to call it from your main method. Think about how you can call the method and get it to display the answer.

2c)A method calling another method

Now that you have a more general maximum method, you will use it as part of a more complicated operation.

Write a method `finalGrade` that takes as input four `int` values and returns the final grade of a comp-202 student. The first value corresponds to the total assignment grade. The second value is the quiz grade. The third is the midterm and the fourth is the final exam. Recall that if the student does better on the final than on the midterm, the mark for the final replaces the mark for the midterm. Be sure to use your `maximum` method when you do this. For the purposes of this question, all fractional marks must be rounded down.

For example, a student might have 28 out of 35 for assignments, 4 out of 5 for quizzes, 18 on 20 for the midterm, and 30 on 40 for the final. The `finalGrade` method takes these numbers `28, 4, 18, 30` as input, in order, and would output a final grade of 80. A student who instead had marks of `28, 4, 16, 38` would have a final grade of 89 (the midterm grade is dropped in this case, because the student's performance in the course is higher using the alternate grading scheme). Be careful with your types here! This method uses `int`s, whereas all the other methods used `double`s.

What To Submit

You have to submit one zip file with all your files in it to MyCourses under Assignment 1. If you do not know how to zip files, please ask any search engine or friends. Google might be your best friend with this, and a lot of different little problems as well.

These files should all be inside your zip.

`ParseDigits.java`
`Marking.java`
`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.