ASSIGNMENT 1

COMP-202, Winter 2018, All Sections

Due: Wednesday, January 31st, 11:59pm

Please read the entire PDF before starting. You must do this assignment individually.

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. :)

Up to 30% can be removed for bad indentation of your code as well as omitting comments, or poor coding structure.

To get full marks, you must:

- Follow all directions below
  - In particular, make sure that all classes and method names are **spelled and capitalized exactly** as described in this document. Otherwise, marks will be removed
- Make sure that your code compiles
  - Non-compiling code will receive a very low mark
- Write your name and student ID as a comment in all .java files you hand in
- Indent your code properly
- Name your variables appropriately
  - The purpose of each variable should be obvious from the name
- Comment your work
  - A comment every line is not needed, but there should be enough comments to fully understand your program

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

```
  22
2   2
   2
 2
22222
```

Use *Strings* composed out of the space character and the character '2'. For a bit of fun, try to draw the pattern '202'.

Warm-up Question 3 (0 points)
Practice with Binary:
We usually use base 10 in our daily lives, because we have ten fingers. When operating in base 10, numbers have a `one`s column, a `ten`s column, a `hundred`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 thirteen in base 5.
3. Represent the number thirteen in base 2.
4. What binary number is equal to the sum of these two binary numbers? 10101011 + 10010001
5. What is the number from part 4, in base 10?
6. What is the binary number for 11010010 + 11000101? Note: Don't worry about overflow here
7. And what is the number from part 6, in base 10?

Warm-up Question 4 (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*?

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

Part 2

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

Question 1: Calculator Program (50 points)

Attached to this assignment is a file called `Calculator.java`. Note that there is a marked section where your code must go. The code outside of this area must not be modified. Your name and student number must be written at the top of the file.

Let a, b, and c be the three numbers the program receives as *input arguments...*

Write Java code in the marked area to print the following calculations:

1. The sum of a and b.
2. The product of a and b.
3. The result of dividing a by b.
4. The result of dividing a by c.
5. A statement saying whether a is larger than b.
6. A statement saying whether a an odd number. Use the mod operator %.

For example, if the program is run with input arguments *5*, *5*, and *1*, the output should be:

```
The first argument a is:  5
The second argument b is:  5
The third argument c is:  1.0
Sum of a and b:  10
Product of a and b:  25
Dividing a by b:  1
Dividing a by c:  5.0
Is a larger than b:  false
Is a odd:  true
```

Be sure to include the specified text on each output line. That is, concatenate a *String* literal with the value of a variable.

For the division results in this question, don't worry about handling division by zero.

Input Arguments Note that this program is run by providing *input arguments*. That is, a, b, and c do not have a fixed value. Their values will depend on the input provided when running the program.

For example, once this program is compiled, it can be run by typing the text `run Calculator 5 5 1` in the Dr. Java *Interactions Pane* and pressing Enter. The output of the program must be different than if the text `run Calculator 6 4 2` was entered.

If you are using Eclipse, please read this for help with input arguments:
`http://www.cs.colostate.edu/helpdocs/eclipseCommLineArgs.html`.

Question 2: Creating a Lemonade Stand Sales Program (50 points)

The goal of this question is to write several *methods* to create a program for doing calculations for a lemonade stand. All the code for this question must be placed in a file named `LemonadeStand.java`. Note that this means the class must also be named `LemonadeStand`.

You have to write methods that perform the following tasks:

- Printing total sales
- Performing 'safe' division
- Determining the maximum of two numbers
- Calculating various statistics of the lemonade stand

Hint: To test these methods, create a main method. The main method will not be graded in LemonadeStand.java, but without it, you won't know whether or not your methods work! Your main method should call the below methods a few times using different input data so that you can verify the methods are working correctly. You should think of a few cases to test!

2a) Void Method for Printing Total Sales

Write a method `printTotalSales` that takes as input one `integer` parameter and one `double` parameter. The first parameter is an `integer` called `numSales`, while the second parameter is a `double` storing the `price` for each sale.

You must print these two numbers, as well as the result of multiplying them together. For example, if the method is called with the parameters `17` and `4.45`, then the message printed must be: `We sold 17 units at $4.45 each, which totals $75.65.`

Note that for full marks, this message **must** be written on one line. Research the `+` operator for `Strings`, or the `System.out.print()` and `System.out.println()` statements.

2b) Method for Division

Write a method `divide` inside of `LemonadeStand.java` that takes two `integer` parameters. This method should return the result of dividing the first parameter by the second[1]. The parameters names don't matter, but should be logical, such as `a` and `b` or `first` and `second`. The return value must be an `integer` value.

Note that division doesn't work properly if the second parameter is zero. Therefore, if the second parameter is zero, the method should print an appropriate error message before the division occurs, and then return zero. Note: this means that the program must not crash. Your method must **not** print anything if the second parameter is not zero.

To test your `divide` method, you will need to call it from your main method. Think about how you can call the method and then display the answer. Make sure to test the cases where the second parameter is 0, and where it is not 0.

2c) Maximum Method

Write another method `getMaximum` that takes two `integer` values as input, and returns an `integer` which is the larger value of the two parameters. In the case of a tie, return the value of either variable. Don't use the built-in *max* method.

[1]Using the slash division operator, as in the first assignment question.

2d) Statistics Method

Now use the `printTotalSales`, `divide`, and `getMaximum` methods to calculate some statistics for the lemonade stand.

Write a method `standStats` that takes as input two `integer` parameters and one `double` parameter. The return value must be `void`.

- The first parameter `numDays` corresponds to the number of days the stand was open.
- The second parameter `numSales` corresponds to the number of units sold.
- And the third parameter `price` corresponds to the price per unit.

First, call the `printTotalSales` method with `numSales` and `price` as parameters.

Second, create a variable `salesPerDay` to store the number of sales per day. This `integer` variable must store the result of calling the `division` method with the `numSales` and `numDays` parameters. Note that this will be integer division, without a decimal value.

Third, print the value of `salesPerDay` in a message. For example, the message could be `With 41 sales over 7 days, the sales per day were 5.`

Fourth, create an `integer` variable `targetSales`. Assign this variable the value obtained by calling the `getMaximum` method with the `salesPerDay` variable and the value `10`. This will set the target sales to be the larger of the sales per day or 10 sales.

Fifth, print a message detailing the target sales for the lemonade stand. For example, the message could be `The target sales are now: 10.`

Examples If the `standStats` method is called with the parameters 7, 17, and 4.45, the output should be:

```
We sold 17 units at $4.45 each, which totals $75.65.
With 17 sales over 7 days, the sales per day were 2.
The target sales are now:  10.
```

If the `standStats` method is called with the parameters 2, 37, and 2.25, the output should be:

```
We sold 37 units at $2.25 each, which totals $83.25.
With 37 sales over 2 days, the sales per day were 18.
The target sales are now:  18.
```

What To Submit

Please put all your files in a folder called Assignment1. Zip the folder (DO NOT RAR it) and submit it in MyCourses. If you do not know how to zip files, please ask any search engine or friends. Google will be your best friend with this, and a lot of different little problems as well.

Instead your zipped folder, there must be the following files. **Do not submit any other files, especially .class files.** Any deviation from these requirements may lead to lost marks.

`Calculator.java`
`LemonadeStand.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.