# CPSC 1100 – LAB 07

Implementing Methods with Calculations

This lab will deal with implementing mathematical concepts in java. The tasks will be assigned from the textbook. The entire lab will be due next week. **PLEASE COMMENT YOUR CODE.** You will have points taken off if you do not comment your code. Keep your code neat.

**Some useful links:**

BlueJ tutorial [www.bluej.org/tutorial/tutorial-201.pdf](http://www.bluej.org/tutorial/tutorial-201.pdf)

Java tutorial home page: <http://docs.oracle.com/javase/tutorial/>

Start here: <http://docs.oracle.com/javase/tutorial/java/index.html>

variables <http://docs.oracle.com/javase/tutorial/java/nutsandbolts/variables.html>

data types <http://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>

java math library <http://docs.oracle.com/javase/7/docs/api/java/lang/Math.html>

Simple Video on BlueJ Debugger <http://www.youtube.com/watch?v=LUUPTbWV0g8>

**Some helpful tips:**

1. Compile often – do it.
2. Perform the tasks by hand to verify your work.
3. You will create several classes in this lab, and create a single Lab07Tester class that will create objects from each of your classes, and test your classes.

## Tasks: Follow the directions below to complete your lab assignment

First you will need to create a new project, and then add a class named Lab07Tester.

You should then add a main method. Add a single line of code to your main method that prints a message to the screen stating that the tasks will now begin. As you complete each class in the assignment, you will add code to your main method to test your classes.

***Task01***: Modified problem E4.1 from Book. Create a class called Paper. Your class should have two instance variables to represent the height and width of the sheet of paper in inches. You should also have a public constant in your class that represents the number of millimeters per inch. (There are 25.4 millimeters per inch ). Create a constructor that accepts two values, the height and width of the paper. Create the following **accessor** methods: getWidthInMillimeters(), getHeightInMillimeters(), getWidthInInches(), getHeightInInches(). Recall that it is OK for an accessor method to calculate a value to return. However, accessor methods should NOT change the value of any instance variables.

Create a Paper object in your tester class. Use the dimensions 8.2 x 5.3. Test all four methods that you have created. Use printf to limit the number of decimal places in your printout.

***Task02***: Modified problems E4.4 and E4.5 from Book. Create a class called NumberCalculator. You should store 2 instance variables in this class that are integers. Create a constructor that accepts values for the two instance variables. Create the following accessor methods that will use the instance variables for calculations.

* getSum()
* getDifference()
* getProduct()
* getAverage() – note that this should return a double: average of 8 and 5 is 6.5.
* getDistance() (absolute value of the difference)
* getMaximum() (larger of the two)
* getMinimum() (the smaller of the two)

*Hint*: the max and min functions are declared in the Math class.

In your main method of your tester class prompt the user for two integers, and read the values in (use a Scanner object). Create a new NumberCalculator object in your tester class using the values entered by the user. Finally, call and print the results from all the methods that you created above. ***Properly format your output so that the numbers are properly aligned***. For your final output capture, use values 42 and 97 as input. ***E4.5 in the text book gives you an example of how your printout should look***. You will need to use printf. You do NOT need to print expected results for this problem, but you should know the expected results to verify your methods are correct.

***Task03***: Modified problem E4.10 from the Book. Create a class called CarSimulator. You will need three instance variables to store:

* The number of gallons of gas in the tank
* The fuel efficiency in miles per gallon
* The price of gas per gallon

Create the following methods:

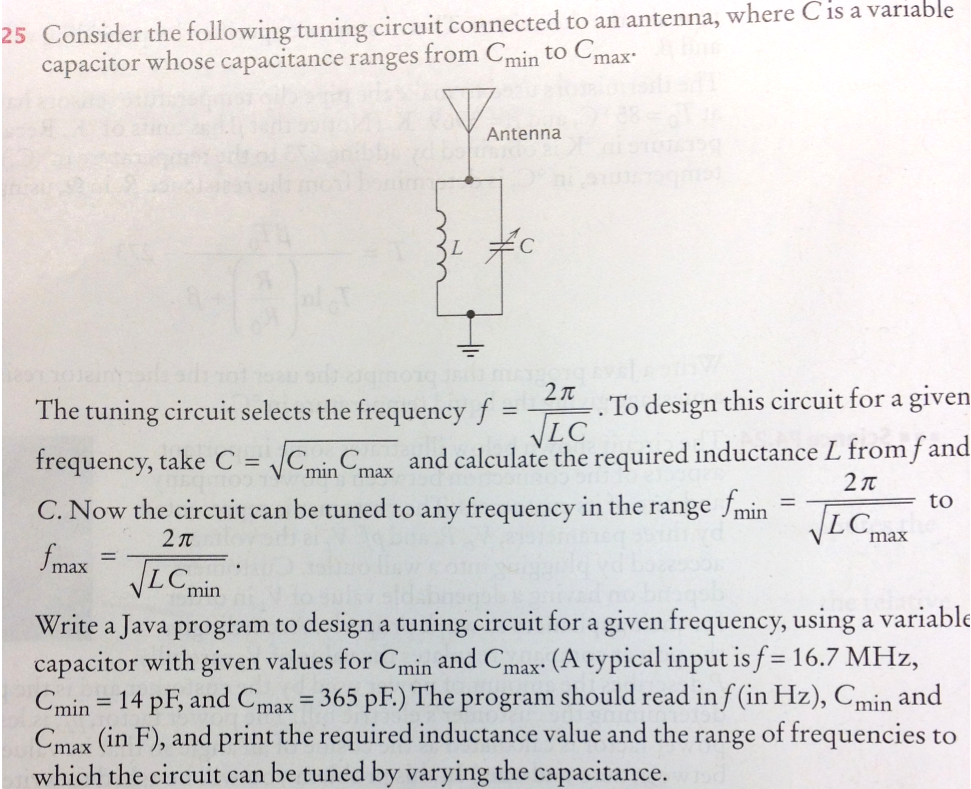
* getCostPer100Miles() – the cost to drive 100 miles
* getDistance() – how far the car can go with the gas in the tank

In your main method for your tester class insert code to prompt the user to enter values for the instance variables described above. Then use the Scanner object created in task02 to read in these values. You will then create a CarSimulator object using these values provided by the user. Finally, test the methods you created for this task. Use printf to limit the number of decimal places in your printouts. You do not need to print expected values. For your input values use 25 gallons in the tank, fuel efficiency of 26.35 MPG, and $2.13 cost per gallon.

***Task04***: Modified problem P4.25 from the Book. Create a class called TuningCircuit. You should have 3 instance variables that represent the **frequency**, **Cmin**, and **Cmax**. Create a constructor that accepts values for all 3 of the instance variables. Create two methods, getFMin() and getFMax()

Using the equations provided in the problem description from the book you can calculate these values. You do not need to understand the physics of this problem to solve it. Look at the problem closely, and see what values are known. One of the formulas will have a single variable that is unknown. You can solve for the unknown variable, and then use that value in the other equations as needed.

In your main method in your tester, prompt the user for values for the three instance variables, and create a new TuningCircuit object using the values entered by the user. The original problem from the book is posted below. You should again print your solution (the results from your two methods) neatly using printf. For your output capture to turn in use the typical values listed in the problem description below.



## To Turn In via Google Drive

You should turn in all of your .java files, as well as output from the BlueJ terminal window from your tester that should test all of your classes.