# CPSC 1100 – LAB 12

Loops and Arrays

This lab will deal with using looping mechanisms. (Don't just do the problem in the book, read my description of the problem). 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>

relational operators <http://docs.oracle.com/javase/tutorial/java/nutsandbolts/op2.html>

if-then <http://docs.oracle.com/javase/tutorial/java/nutsandbolts/if.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 calculations by hand to verify your work.
3. Make sure to test several inputs to your program.

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

1. We will be doing problem **P6.3** from the book.

Create a new project, and then add a class named Fibonacci. Then create a new class called FibonacciTester. FibonacciTester will contain a main method.

The class Fibonacci should contain a single instance variable as an int. This variable represents which Fibonacci number of the sequence that your getFiboNumber() method will return. The Fibonacci sequence as follows. (Each value is the sum of the previous two values).

1, 1, 2, 3, 5, 8, 13, 21, …

So if your instance variable is 7, your getFiboNumber method should return 13 (which is the 7th number in the sequence as indicated by the instance variable). This method should accept no parameters, and return a long. (This value will directly tie into how many times your loop will run).

Complete the following task.

**P6.2** The *Fibonacci numbers* are defined by the sequence:

*f1 = 1*

*f2 = 1*

*…*

*fn = fn-1 + fn-2*

therefore

*f3 = f2 + f1 = 1 + 1 = 2 and*

*f4 = f3 + f2 = 1 + 2 = 3*

Each value is the sum of the two previous values.

Renaming the variables above in java terms we have the following (***use the* long *data type for all of these variables to prevent overflow***):

fold1 = 1; //initial value

fold2 = 1; //initial value

fnew = fold1 + fold2 //initially fnew is 3rd fibonacci number and //equals 2

***The next paragraph describes what you will need to do inside your loop. You will still need to figure out how many times to run your loop.***

After this *initial* set of values, to calculate the next *Fibonacci number*, we discard fold1, which is no longer needed, and set fold1 = fold2, and set fold2 = fnew. At this point, the next *Fibonacci number* can be calculated as above. (fnew = fold1 + fold2;). Hopefully you see a looping pattern at this point. Your code will repeat this process (loop) until you have reached the appropriate Fibonacci number.

If you consider things, if we want the 1st, 2nd, or 3rd Fibonacci number, our loop does not need to run at all. We already know these 3 values. If we want the 4th Fibonacci number, we need our loop to run once. If we want the 5th Fibonacci number, we need our loop to run twice. (etc …)

Implement a program that prompts the user for an integer ***n*** and prints the *n*th *Fibonacci number*, using the above algorithm. Your printout should be neat, i.e. something similar to the following:

Fibonacci number 7 is 13.

If at any point the user gives bad input (i.e. they enter a value other than an integer, or an integer less than 1) you should inform the user of the bad input, and then request they try again. (i.e. while the input is bad, continue to ask for input). Your program should not crash based on user input. (i.e. you should not get an exception). I will enter bad inputs to see if your program crashes.

Hints: You will need two while loops (you could also use do while, or for loops, any of the 3 looping mechanisms are acceptable). The first loop (in the main method) runs until the user gives a valid input, and the second loop (in the getFiboNumber method) will run until the nth Fibonacci number has been calculated.

In summary, your main method should input a value from a user (validate this input), then construct a Fibonacci object, call the getFiboNumber() method, and print the value returned by this method.

Run your program several times and capture the output (you should test bad input, as well as good input). You should have enough tests to verify that your code works properly.

1. In the same project, create a new class named MaxMinFromArray, declare two arrays as int type instance variables Array1D and Array2D, the first one is a one-dimensional array, and the second one is a two-dimensional array. In the constructor of this class, initialize the two arrays using operator new, you may use the number of dimension like Array1D[20], Array2D[10][10]. And fill the arrays with random integers between 1-100 generated by using Random class. Define four methods as following: getMaxFromArray1D, getMinFromArray1D, getMaxFromArray2D, getMinFromArray2D, separately they will return the maximum number from Array1D, the minimum number from Array1D, the maximum number from Array2D, the minimum number from Array2D. Use Advanced Loop to implement these get-\* functions and test them in FibonacciTester.

## To Turn In via Google Drive

You should turn in your java files and a document containing a capture of several runs of your program showing both invalid input and valid input.