# CPSC 1110 – LAB 03

Inheritance (Chapter 9)

This lab will deal with implementing a simulation of a simple electronic circuit using the concepts of inheritance and subclassing. This will be a simple circuit comprised of only resistors (we will only be calculating resistance, not voltages or currents). The problem description is given in problem P9.7 in the Big Java 6th Edition book. You may use BlueJ or Eclipse to complete the lab. (If you want to use some other IDE please talk to the TA about it). **PLEASE COMMENT YOUR CODE.** You will have points taken off if you do not comment your code. You can see sample comments in my starter code for how you should comment your code. Keep your code neat.

Zip all of your .java files and a pdf containing your code’s output to submit to UTC Learn.

**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>

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

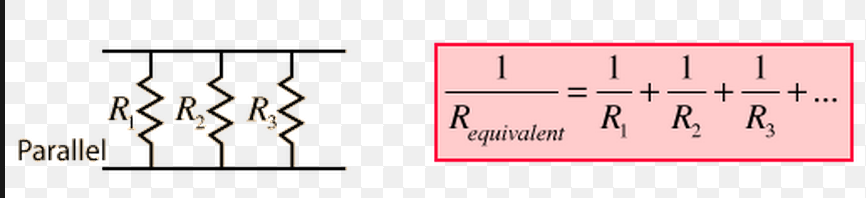
Array Lists <http://docs.oracle.com/javase/7/docs/api/java/util/ArrayList.html>

Inheritance <http://docs.oracle.com/javase/tutorial/java/IandI/subclasses.html>

**Some helpful tips:**

1. Compile often – do it.
2. You are responsible for a larger amount of the design process in this lab. You need to draw on paper how you want your subclasses to work and plan before you actually write code.
3. It may be helpful to use the Debugger or print statements to check your work. (always).
4. We will have a demo/tester class for this lab called CircuitDemo. You will need to fill in this class with enough tests to verify that your project works as expected. (Alternately, you may use the JUnit unit test framework from Section 8.7 in the book – more details below).
5. Resistor values should be stored as doubles.
6. There are two types of circuits we are creating, series and parallel (as well as a single resistor). The formula for computing the entire resistance of each type of circuit is given below. I googled these images from <http://hyperphysics.phy-astr.gsu.edu/hbase/electric/resis.html>

## 



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

For today's lab we will be completing Exercise P9.7 from the book.

***IMPORTANT!!*** All of your method/class names need to match what is shown in this document. This is the public interface of your class. The classes and methods you need to implement are as follows:

Circuit – single method getResistance. Note that this is simply an empty circuit with no resistors, and therefore the resistance is always 0 for objects of this class. This is the parent class.

Resistor – subclass of Circuit. Represents a single resistor. Needs to override the getResistance method from the superclass Circuit. To compute the value of the Resistor object you simply return the resistance value.

Serial – subclass of Circuit. Contains an ArrayList<Circuit> instance variable. Resistance is computed by the formula given above. Must override inherited getResistance method.

Parallel – subclass of Circuit. Contains an ArrayList<Circuit> instance variable. Resistance is computed by the formula given above. Must override inherited getResistance method.

**Read the problem description in the book for more details!** (additionally, you will want to reference the material from chapter 9).

I have included a file called CircuitDemo.java to the same folder. This demonstrates how your code should function. (Note that you will need to create appropriate constructors for each class shown above). This sample code builds the circuit shown in the book with the top resistor = 100 Ohms, the bottom left resistor = 100 Ohms, and the bottom right resistor = 200 Ohms. You should make sure you can calculate this circuit by hand. If you need help with this consult TA or me. (Often programmers have to write programs using formulas outside of their area of expertise. Make sure you can do calculations by hand or your program will never work!).

You can choose which way to test your circuit from the two choices below. You can either use standard test procedures as we have used previously (in the CircuitDemo.java file), or you can create a new class CircuitTester to run tests using the JUnit unit test framework. Using JUnit is bonus material. What I am saying is you must test your program in either case, but if you go to the extra work to use JUnit you will receive bonus points. I would suggest at the least you create 2-3 tests in the CircuitDemo.java file before attempting the JUnit tests.

**Choice 1**: You should add multiple circuits to the CircuitDemo.java file to verify that your program works as expected. Take screen-shots or capture the output text and include this in your project folder. Sample output from a working project using the given CircuitDemo.java file is shown below. (Look at the java file on UTC Learn to see the test code that produced this output).

Combined resistance: 75.0

Expected: 75.0

**Choice 2: BONUS**: Use the JUnit unit testing framework to run multiple tests on your circuit. **Use class name CircuitTester**. The completeness of your testing will determine the number of bonus points you receive. (Also, your tests should pass). Feel free to use google to figure out how to use junit and get it running. (You may also discuss this among yourselves – JUST HOW TO GET IT WORKING). Once junit is working, you must think of and implement ***YOUR OWN*** tests. Of course, I am available to help you with this as well. (The basic idea is that you will need to import two .jar files (org.junit.Test and org.junit.Assert) into your current project, and then write tests as demonstrated in section 8.7 of the book).

## To Turn In via UTC Learn

You should turn in 1 .ZIP file containing your java files and a PDF document with screen-shots (or text) of your output. 1 file should be uploaded to UTC Learn. ***IMPORTANT!!!*** You should name your file in the following manner. lastname-firstname-lab03.zip. So John Smith would submit smith-john-lab03.zip.