# CPSC 1110 – LAB 3

Class Design (Chapter 8)

This lab will deal with implementing a combination lock class. The initial problem description is given in the book as P8.1. I have added a little clarification, as well as a getCurrentNumber() accessor method. The trick to this lab is that you must learn how to store information about the state of your combination lock. (A state diagram may be useful to help solve this problem – we will cover the state diagram in class). You may use BlueJ or Eclipse to complete the lab. (If you want to use some other IDE please talk to me 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.

You should zip all your .java files along with a PDF containing your codes output into a single file for submission.

**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>

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>

**Some helpful tips:**

1. Compile often – do it.
2. Perform the tasks by hand to verify your work. Run the algorithms on paper with small input sizes to make sure your algorithm works. (i.e. You may want to draw a combo lock with only 5 possible values rather than 40 to see how to deal with spinning the lock left and right).
3. It may be helpful to use the Debugger or print statements to check your work.
4. We will have a tester class for this lab as well, called ComboLockTest. When you run the “main()” method it should allow you to interact with the ComboLock class and verify your work. You can manually type in menu commands to interact, or you can pipe an input text file into the program for “batch” testing.

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

For today's lab we will be completing Exercise P8.1 from the book with some slight modifications. Starter code is posted in Labs/Lab3 folder in two files ComboLock.java and ComboLockTest.java.

Please study the following image to have better understanding of the lock.

A picture containing metalware, lock

Description automatically generated

Follow the directions given for P8.1, and fill in the gaps in the starter code on UTC Learn. More details are given inside the comments for the ComboLock class.

Here is sample output of an interaction with the program.

Please enter 3 values for the new combo lock: 12 12 12

Combo is: 12 12 12

Current Number: 0

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

28

Current Number: 12

Enter number of ticks to turn to the left 0 - 40. Enter an invalid number to quit (negative, or >40).

40

Current Number: 12

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

40

You opened the lock!

Would you like to run simulation again? (Yes or No)

y

Please enter 3 values for the new combo lock: 12 13 14

Combo is: 12 13 14

Current Number: 0

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

28

Current Number: 12

Enter number of ticks to turn to the left 0 - 40. Enter an invalid number to quit (negative, or >40).

1

Current Number: 13

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

39

You opened the lock!

Would you like to run simulation again? (Yes or No)

y

Please enter 3 values for the new combo lock: 1 2 3

Combo is: 1 2 3

Current Number: 0

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

39

Current Number: 1

Enter number of ticks to turn to the left 0 - 40. Enter an invalid number to quit (negative, or >40).

3

Current Number: 4

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

3

Current Number: 1

Enter number of ticks to turn to the left 0 - 40. Enter an invalid number to quit (negative, or >40).

3

Current Number: 4

Enter number of ticks to turn to the right 0 - 40. Enter an invalid number to quit (negative, or >40).

333

Invalid entry. The program will now exit.

Some things to note are that an entry of “40” will spin the lock all the way back around to the same number you are on. Our ComboLockTest class will verify that values sent to the turnLeft() and turnRight() methods receive a value between 0 – 40 (inclusive).

ComboLockTest is not written with a lot of error checking. This is ok because we are simply using it as a testing class. So if you feed bad input to ComboLockTest this may cause an exception. (i.e. entering a double value instead of an int while scanning for input). You just need to be careful that you use valid inputs and if you cause an exception be aware of what caused it.

Notice what happens when a user spins the lock right, then right, then right. Also assume our combination is 1,3,5.

myLock.turnRight(39); //spins once to the right, OK! First Value locked in!

myLock.turnRight(3); //spins the dial and updates value, but now we have //wiped out our initial good entry,

//looking for a right spin to 1 again now.

myLock.turnRight(37); //spins the dial and updates value, but the middle //spin “reset” the tumblers.

This implies that you will need to use an instance variable to keep track of what state your lock is in.

If you have other questions about how your lock should operate, please ask me.

## 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-lab02.zip. So John Smith would submit smith-john-lab02.zip.