**CSc 110 Lab 5**

**Review:**

From last lab, a solution to the TryAgain.java exercise is [here](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab5/code/TryAgainDone.java) for review. There are many ways to write programs, this is just one solution.

**Objectives:**

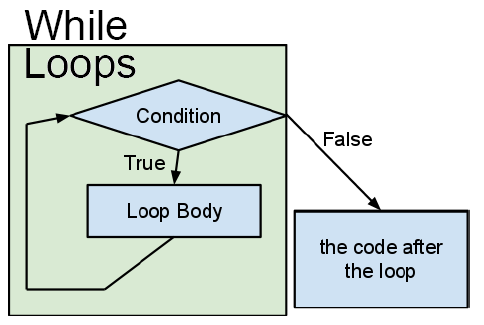
More emphasis is placed on actual coding practice this week.  Get out some blank paper since you will in all likelihood need to draw some pictures or make some notes to help you with your logic.

1. While loops
2. Method programming (more return statements and parameters)
3. Conditionals (if/else)

**Exercise 0) Quick coverage of while loops**

**While-Loops**

These are simple loops which keep on going *while* the condition is true:  
  
while(condition){ // It is important to have something in the loop body //that will make the condition false, so that it //is not an infinite loop }

You can write for-loops as while-loops and vice versa, but usually the situation will make one more appropriate than the other.  
  
  
  
initialization;  
while(condition){  
   body;  
   update;  
}  
  
for(initialization; condition; update){  
   body;  
}

|  |  |  |
| --- | --- | --- |
| Some examples: | | |
| String fromFile = "";{ while( in.hasNext() ){    fromFile += in.nextLine(); } | int i = 1; while(i <= 10){    System.out.println(i);    i++; } |  |

Try making the for-loop in TryAgain.java into a while-loop so that the user has as many chances as they need or so they still have only 3 chances. (Reminder: conditions can be !negated, and combined with && (and) and || (or).)

**EXERCISES**

**Exercise 1 Range checking**

a) Download [Ages.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab5/code/Ages.java) and predict what the three methods will do.

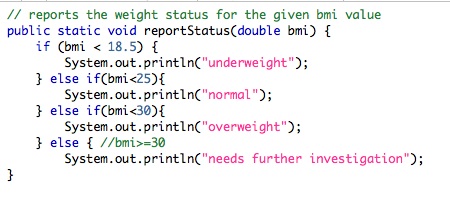
b) How can you test your predictions? Write a couple tests for each method and check the output.

c) Rewrite the third method, oldInCanada(), to only print out one message: the letters that you could have received from the highest authority. So, if your age is 100, you should only see that you could have one letter from the queen. If your age was 95 you would only see that you could have two letters from the Governor General.

After 5 minutes of individual thinking (not code)  the class should construct an approach to this.

**Exercise 2: If/Else -> If's with returns**

Type in this range checking method, compile and test it:



Change your method such that rather than using an else statement to control any further processing, you return immediately instead (just the statement:return;).  This is called a short circuit and can often be used to simplify conditional logic.  Note that a return statement is still legal even from a voided function. Run your tests again.

**Exercise 3 Methods and boolean logic**

Create, write and test the following methods.  First write and test a stub then apply the logic.  
  
a) Write a method that given a month name returns the number of days to the calling program. Assume that it is not a leap year.  
  
b) Write a method that given parameters start\_hour and start\_minutes and end\_hour and end\_minutes, returns the total number of minutes elapsed from start to end.  Assume that all times are within 24 hours AND given in 24hour format (eg 4:00pm is 16:00).

**Exercise 4  Parameters, Returns and Scanner: Many-sided Die**

1. Download this program, [Dice.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab5/code/Dice.java).
2. Compile it, run it.
3. Look at the code. More instructions in the comment **underneath the code**.
4. Consider adding a prompt for how many times to roll the die
5. Make sure that you understand what's going on in the method roll()

**Exercise 5 Building a Guesser program**

**First**

Write a new program Guesser. Save it as Guesser.java. It will let you try to guess the secret number 7 times, giving hints.

Guesser should declare a Scanner on the keyboard (System.in). Call the Scanner variable *input*.

Guesser should define an int variable and set it to a number between 1 and 100. You should be able to do this with Math.random()

The program should print "Guesser begins ...".

Compile and test the program so far.

**Second**

Create a method called getGuess that, given a Scanner variable, will prompt the user to guess a number between 1 and 100.

The method should return the guess to the main method.

Test the method.

**Third**

Create a method called winner that, given the guess and the secret number as parameters, prints "higher" if the guess is too low, "lower" if the guess is too high, and "WINNER" if the guess is correct.

The method should **return** false if the guess is high or low, and true if the guess is correct.

Test this method

**Finally**

Regulate the number of guesses that are allowed (set this to 7)

Print the number of guesses the user has left

**Possible Implementation:**

Set a variable called guesses to 7; // the number of guesses you get

Set a variable called guess to getGuess(input); // input is a Scanner variable

Set a boolean, youWon = false

while (!youWon && guesses > 0 )

print guesses

guess = getGuess(input)

guesses = guesses minus one

youWon = winner(guess, secretNumber);

if( youWon )

print "Nice job!!"

else

print "The number was..." + secretNumber

**Exercise 6 (time permitting) More Scanner and If/Else**

1. Download this program, [Quiz.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab5/code/Quiz.java).
2. Look at the code. Instructions are in the comment **above** the class.
3. There are some examples to look at, uncomment, compile and run (methods mammal and mathTest)
4. Write your own simple quiz, using a Scanner and if-statements

**Looking back...**

In this lab you'll have practiced or seen examples of:

* while loops
* if/else logic
* short circuits (returning out of an if-statement instead of using else)
* using Scanner and passing it as a parameter
* passing parameters to control loops and functionality inside methods
* Possible implementation of the guesser program here: [Guesser.java](http://webhome.csc.uvic.ca/~csc110l/2012_5/Lab5/code/Guesser.java)

**The End**