

# CS 210, Fundamentals of Computer Science I

## Lab 5: while and for loops

### Introduction/Overview

In this lab, you'll write three Java programs, using the jGRASP Integrated Development Environment (IDE). In this lab you'll practice using while and for loops, which should also help prepare you for the midterm. The last part of the lab explains what needs to be submitted to Canvas for you to receive credit for completing this lab.

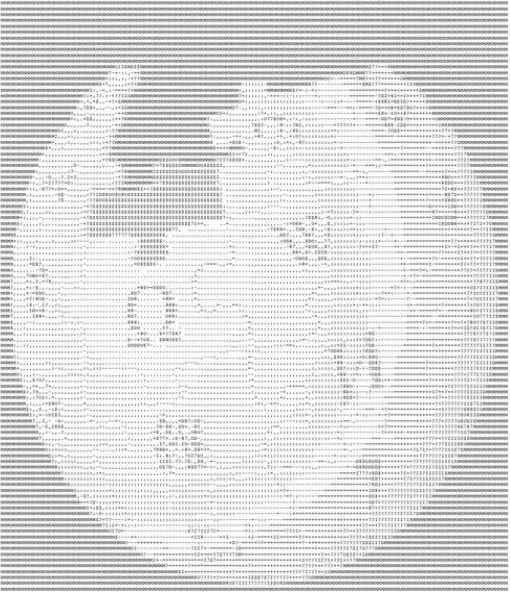
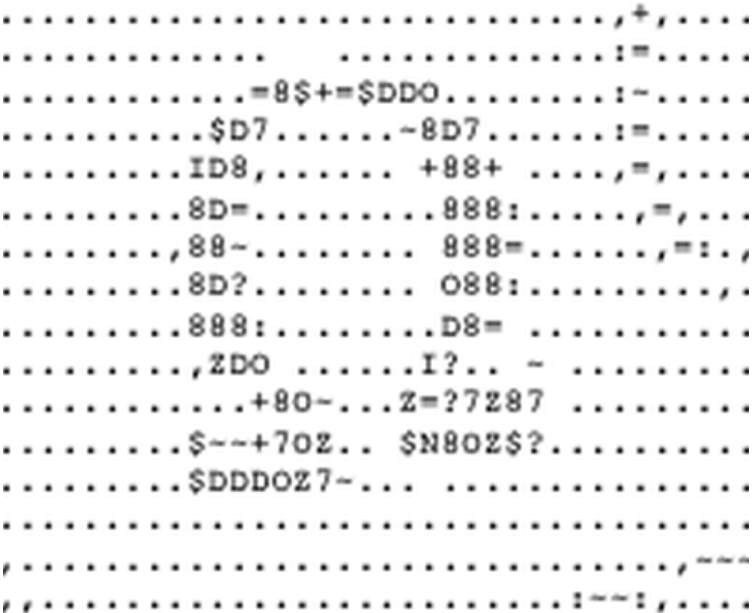
### Preliminaries

For this lab, create a folder on your computer, you can call it **lab5**. During lab you can save and work on your programs there, but only the .java files that you upload to Canvas will be graded.

### I. The while loop

You have learned in lecture about the while loop, that allows you to write code that repeats a set of statements.

For this part of the lab, you'll use a nested while loop (one inside of another), to generate a very simple ASCII art image. ASCII art is a graphic design technique for drawing pictures made up of the 95 printable characters that you can type on a standard keyboard. From far away, ASCII art looks like a normal (ish) art drawing (Figure 1a), but if you look closely, you'll notice that an ASCII art image is made up letters, numbers, characters, etc. (Figure 1b). (Images courtesy of [http://en.wikipedia.org/wiki/ASCII\\_art](http://en.wikipedia.org/wiki/ASCII_art))

	
<b>Figure 1a:</b> The Wikipedia Logo as ASCII art	<b>Figure 1b:</b> Close-up of the capital Omega in Figure aa

In this lab you won't be creating images that are as complicated as the one in Figure 1 (such images take a great deal of planning and practice). Instead, you'll begin by writing a program that generates a triangle, made up of

“at” characters (@). See Figure 3 for the output that your program will generate. To complete this part of the lab:

1. Create a new java file. Type the text that is shown in Figure 2 into the editor panel of the new java file that you just created. Add/modify the comments accordingly, so that your name is listed as an author, and so that you include a description of the program. As always, be generous with using comments.
2. Save the file as *AsciiTriangle.java* in your **lab5** folder. Compile the program and run it. Make sure that the program outputs what you see in Figure 3. If you have typed any part of the code incorrectly (and introduced a syntax error, you'll have to fix the error, until your code is error-free).

```
import java.util.Scanner; // import the Scanner class
public class AsciiTriangle { // Define a new class
    public static void main (String[] args) { // the main routine

        int triangleHeight = 0; // variable declaration and assignment
        String aSCIICharacter = "*"; // variable declaration and assignment
        Scanner keyboard = new Scanner(System.in); // create a scanner

        // print instructions, and ask user to input the height of the triangle
        System.out.println("This program prints a simple ASCII triangle. ");
        System.out.print("How tall should the triangle be? ");
        triangleHeight = keyboard.nextInt() + 1;

        // receive as input the ASCII character that should be used to print the ASCII art
        System.out.print("What ASCII character should be used to draw the triangle? ");
        aSCIICharacter = keyboard.next();

        int triangleRow = 1; // keep track of which row of the triangle you are printing
        String triangleRowChars = "";
        int rowCharacterPosition = 1;

        // at this point, triangleRow is 1
        while (triangleRow < triangleHeight ){
            triangleRowChars = "";
            rowCharacterPosition = 1;
            while (rowCharacterPosition <= triangleRow){
                triangleRowChars = triangleRowChars + aSCIICharacter;
                rowCharacterPosition++;
            }
            System.out.println(triangleRowChars);
            triangleRow++;
        }
    }
}
```

**Figure 2:** Java code to create triangle ASCII art image.

```

----jGRASP exec: java AsciiTriangle

This program prints a simple ASCII triangle.
How tall should the triangle be? 7
What ASCII character should be used to draw the triangle? +
+
++
+++
++++
+++++
++++++
+++++++
----jGRASP: operation complete.

```

**Figure 3:** Sample output of the program in Figure 2.

## II. The *for* loop

Another loop control structure, in addition to the *while* and *do-while* loops, is the **for** loop, which you've seen in lecture already. The **for** loop has a very similar functionality to the *while* loop, but the syntax is different. In this part of the lab, you'll recreate the triangle ASCII program that you wrote for the previous part of this lab, but you'll use a *for* loop instead of a *while* loop. To complete this part of the lab:

1. Create a new file, *ForLoopAsciiArt.java* inside of your **lab5** folder.
2. Import Java's Scanner functionality; at the top of your file, write: `import java.util.Scanner;`
3. Write a main method, and inside of it, set up a Scanner that is listening to the keyboard, and print to the screen the following questions: *How tall is the triangle?* Use the Scanner's `nextInt()` method, to retrieve from the keyboard, the number input by the user, which you should save into a variable `triangleHeight` of type *int*.
4. To output the “at” characters (@), you'll need to use two *for*-loops, an outer loop and an inner loop:
  - a. The outer loop should have an iterator variable, `row`, set to 0 in the initialization step. The update step should be `row++`, and the loop should iterate as long as `row <= triangleHeight`. Therefore, this loop will perform as many iterations as the number that was input by the user. The outer *for* loop should look like the following:

```

for (int row = 0; row <= triangleHeight; row++){
    // body of outer for loop
}

```

- b. Inside of the body of the outer *for* loop, write ANOTHER *for* loop. The idea is, that at row *x* of the triangle, you print *x* asterisks. The inner loop uses the value of the `row` variable, to determine how many columns of asterisks to print. The `println` is needed to generate a new line after each row of the triangle is printed. your inner *for* loop (the loop INSIDE the body of the outer loop), should be:

```

for (int column = 0; column < row; column++){
    System.out.print("@");
}

```

```
}  
System.out.println();
```

5. Complete the program by making sure that the main and class sections are correctly closed using the closing curly brackets. Compile and run your program. Make sure that your program reproduces the output that is shown in Figures 4.

```
----jGRASP exec: java ForLoopAsciiArt  
How tall is the triangle? 4  
  
@  
@@  
@@@  
@@@@  
  
----jGRASP: operation complete.
```

**Figure 4:** Sample output of the program *ForLoopAsciiArt.java*

### III. The `for` loop; decrementing

In this part of the lab, you'll again print a triangle, but this time, an upside-down triangle. See Figure 5, for example output. The code for this program is nearly identical to the program for printing a right-side-up triangle. The only difference is that the `for` loop decrements. To complete this part of the lab:

1. Using jGRASP, create a new file, *UpsideDownTriangle.java*, inside of your **lab5** folder.
2. Import the Scanner functionality, at the top of your java file. Write a main method, and inside of it, create a Scanner that is listening to the keyboard.
3. Print to the screen the following questions: *How tall is the upside-down triangle?*
4. Use the Scanner's `nextInt()` method, to retrieve from the keyboard, the number input by the user, which you should save into a variable *triangleHeight* of type `int`.
5. To output the asterisk characters (@), you'll again need two `for`-loops, an outer loop and an inner loop. The “first” row of the triangle, however, should have as many columns as the number that is placed into the variable `triangleHeight`. Then, each successive row should have one fewer asterisk.
  - a. The outer loop, should have an iterator variable, `row`, set to `triangleHeight` in the initialization step. The update step should be `row--`, and the loop should iterate as long as `row > 0`. Therefore, this loop will perform as many iterations as the number that was input by the user, but it is different than the `for` loop that you wrote for part I of this lab, because the “first” row has the most columns of asterisks, and each successive row has one fewer.

```
for (int row = triangleHeight; row > 0; row--){  
  
}
```

- b. Inside of the body of the outer `for` loop, write ANOTHER `for` loop, which is identical to the inner `for` loop that you wrote for part II of this lab.

6. Compile and run your program. Make sure that your program can reproduce the output in Figure 5.

```
----jGRASP exec: java UpsideDownTriangle
How tall is the upside-down triangle? 5
@@@@@
@@@@@
@@@
@@
@
----jGRASP: operation complete.
```

**Figure 5:** Sample output of the program *UpsideDownTriangle.java*

#### IV. What to hand in

The following files should be uploaded to Canvas (please upload the files individually):

*AsciiTriangle.java*  
*ForLoopAsciiArt.java*  
*UpsideDownTriangle.java*

Be sure that each .java file is commented and be sure to include your name at the top of each file. Code must be indented, so that it is easy to read. Finally, make sure that you have given your variables good descriptive names.

#### V. Rubric

File / task	Points
<i>AsciiTriangle.java</i> compiles and generates correct output when run, using a while loop	20
<i>ForLoopAsciiArt.java</i> compiles and generates correct output when run, using a for loop	35
<i>UpsideDownTriangle.java</i> compiles and generates correct output, using a for loop	35
All three .java files are commented, and contain your name and date	2
Variable names are adequate and descriptive in both java files, Code is indented properly in each .java file	8
Total	100