1 Setup

Create a lab4 project in IntelliJ and create a new Java class called BrokenLoop (just like that, including the capital B and L). Open the BrokenLoop.java file on the course web page, and copy and paste the contents into your BrokenLoop in IntelliJ.

You should see the following code in the main method:

```java
int num = keyboard.nextInt();
int i = 0;
while(i < num) {
    System.out.print("*");
}
```

This piece of code is trying to print a line of num asterisks. Compile and run the program.

OH NO! Instead of printing num asterisks in a line, you’re probably seeing the window fill up with endless asterisks. You are stuck in an infinite loop: the condition i < num is never satisfied, so the computer will continue to print asterisks forever. Fortunately, there is a way to kill a program that is currently running. Click the red square to the left of the window where all the asterisks are printing. This will terminate the program (if you scroll back over to the left of the window, you should see the message “Process finished with exit code -1.” The -1 here indicates that something went wrong, in this case, that you manually terminated the program before it was done running.)

1. Fix the code so that it actually does print num asterisks in a line.

2 Printing with while loops

In this section you will use while loops to print some cool designs. Create a new Java class called While, then open the While.java file on the course web page and copy and paste the contents into your While. Currently all the program does is read an int from the keyboard and store it in a variable called size.

2. Use while loops to print a size by size square. Your square should look something like:

```
+++++
+++++
+++++
+++++
+++++
```
3. Use while loops to print a triangle with size rows in which the first row has one `^`, the second row has two `^`s, and so on. (You can use the value that’s already stored in size; no need to prompt the user for a new value.) Your triangle should look something like:

```
^ 
^^ 
^^^ 
^^^^ 
```

(Hint: For the square in task 2 above, you printed out the same thing on every line. Now, the number of `^`s you want to print depends on which line you are on. What variable in your code tells you what line you’re on? In terms of the current line, how many `^`s do you want to print?)

4. Use while loops to print another triangle with size rows, this time with the `^`s aligned to the right. Your triangle should look something like:

```
^ 
^^ 
^^^ 
^^^^ 
```

5. Currently, the code uses while loops to print the triangle. Modify this code so that it does the same thing, but uses for loops instead of while loops.

6. Use for loops to print an X. Begin by prompting the user to enter an odd number. Then print an X with the entered number of rows that looks something like this:
7. So far we are assuming that when told to enter an odd number, the user will be sensible and not enter 4 or 26 or 12. But at this point we have seen all the tools we need to check that the entered number is in fact odd. Before your code to print an X, write a loop to make sure that the user has entered an odd number. (You can assume that the user will be well behaved enough to enter an int as opposed to some other type.) Hint: is a for loop or a while loop a better choice for this task? Is there a fixed number of times that you’ll need to repeat your code?

4 A new design

Create a new Java class called MyPattern (just like that, including the capital M and P).

8. Come up with your own pattern to print, like those above. Your pattern should be scalable (i.e., the user should be able to enter a number that controls the size of the pattern) and it should involve multiple rows and columns. Be creative! Draw out your pattern on paper first, then write some for loops to print your pattern.

5 Submit your work

Submit your modified BrokenLoop.java, While.java, For.java, and MyPattern.java to the “Lab 4: Loops” assignment on Moodle.

This assignment is due on Thursday, February 27, 11:59 pm.