Nested Loops

by Sophia

WHAT'S COVERED

In this lesson, you will learn about loops created within another loop, known as nested loops. Specifically, this lesson covers:

Table of Contents

1. Nested Loops

At a high level, a **nested loop** is just a loop within another loop. This is very similar to nested conditional statements, which was discussed in the previous Challenge 1.3. In that challenge, you used nested if statements nested in other if statements. The same nesting idea can be used on loops as well.

CONCEPT TO KNOW

A nested loop can be quite useful when there is a series of statements that includes a loop that you want to have repeated.

The functionality is no different than using a single loop except that the **"outer" loop** has one or more **"inner" loops** within it. There are many situations in which using nested loops are beneficial—for example, looping through two-dimensional arrays, which will be covered later in this tutorial. Another great example could be prompting a teacher for numeric grades to calculate the average grade for a student. However, instead of prompting a single student's grades, it could prompt the entire class. To handle this, you would add an outer loop that loops across all of the students, while the inner loop prompts the grades for each student.

The format of nested while loops would look something like this:

EXAMPLE

```
while(<expression>) {
  while(<expression>) {
    <statement(s)>
```

```
  }
  <statement(s)>
}
```

And the format of nested for loops would look something like this:

↗ EXAMPLE

```
for(<initialization>; <condition>; <increment/decrement>) {
  for(<initialization>; <condition>; <increment/decrement>) {
    <statement(s)>
  }
  <statement(s)>
}
```

In the examples above (as when these were first presented covering the different types of loops), the <expression>, <variable>, <iterable>, and <statement(s)> terms in angle brackets are just for information purposes; these are not keywords or actual code. These are just to explain what goes into each of the parts for the nested loop examples. If you notice in the nested examples, the inner loop would be executed one time for each iteration of the outer loop.

THINK ABOUT IT

What do you think would happen if the curly brackets were not paired correctly or not nested correctly?

If the curly brackets are not paired correctly, there could be syntax errors that keep the code from compiling. If the loops and their brackets are not nested correctly, it is possible that the code may compile but not run as expected. The placement of the curly brackets determines which loop a line of code is part of. Using correct indentation will help make sure the loops and brackets are organized correctly. This is especially true as the number of loops increases and selection statements with their blocks of code come into play.

There is no limitation on the use of nested loops or their iterations. In a nested loop, the number of total iterations of all loops included will be the total of the number of iterations in the outer loop multiplied by the iterations of the inner loop.

↗ EXAMPLE

100 x 100 = 10,000

In each iteration of the outer loop, the inner loop will execute all of its iterations. Then, for each iteration of an outer loop, the inner loop will restart and complete its execution before the outer loop can continue to its next iteration. These nested loops are especially useful when it comes to working with multidimensional arrays.

TERMS TO KNOW

Nested Loop
A nested loop is any loop that occurs inside the body of another loop.

Outer Loop
A loop that contains one or more other loops in its body.

Inner Loop
A loop that is contained in the body of another loop.

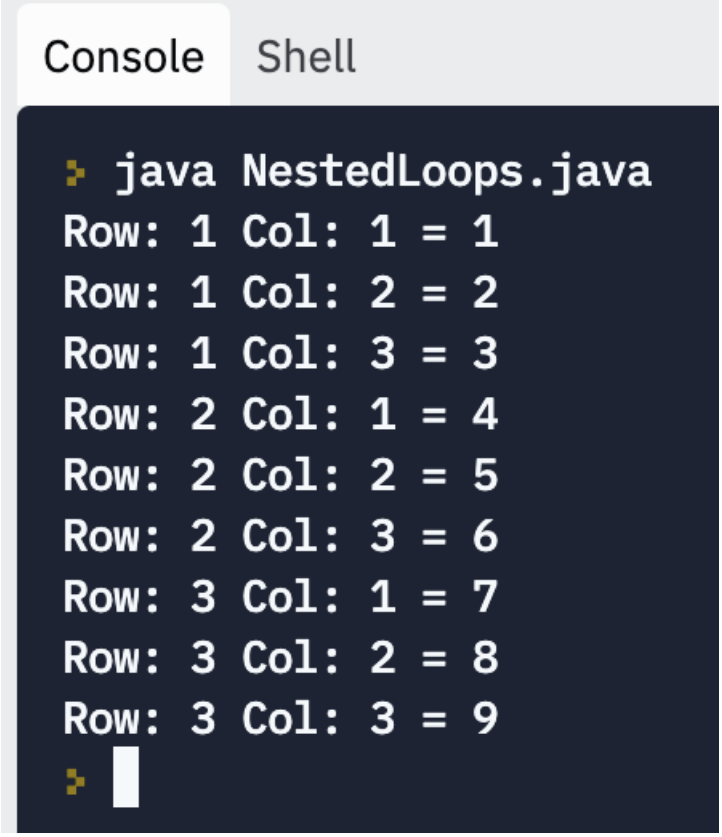
2. Nested for Loops

A two-dimensional array can be conceived of as an array of arrays, with each row in the array being a one-dimensional array. This is done by looping over the contents of an array, by first looping over the rows in the array. The array that makes up a given row then needs to be looped, or iterated over. This means that for each iteration of the loop that steps through the rows, there is a nested loop that goes through the elements in a given row. Since the number of rows and columns in a two-dimensional array is known, for loops are the appropriate choice for the outer and the inner loop.

Let's look at a simple example of a two-dimensional array and how to use nested for loops to print it out:

```
class NestedLoops {
  public static void main(String[] args) {
    int[][] numbers = {
      {1, 2, 3},
      {4, 5, 6},
      {7, 8, 9}
    };
    // Outer loop iterates over rows
    for(int row = 0; row < numbers.length; row++ ) {
      // Inner loop iterates over columns in each row
      for(int col = 0;  col < numbers[row].length; col++) {
        /* (row + 1) displays row # starting with 1 rather than 0.
           (col + 1) displays column # starting with 1 rather than 0.
           Parentheses needed so expression is evaluated before printing.
           These expressions do not change the values of row and col so
           array access works as expected. */
        System.out.println("Row: " + (row + 1) + " Col: " + (col + 1) +
          " = " + numbers[row][col]);
      }
    }
  }
}
```

The outer loop iterates over the three rows in the array. As each row is accessed, the inner loop iterates over the three columns like this:



Note how the outer loop gets the number of rows using this expression:

↱ EXAMPLE

```
numbers.length
```

Since the first dimension of a two-dimensional array is the number of rows, getting the length of the first dimension returns the number of rows.

The inner loop gets the number of columns in each row using this expression:

↱ EXAMPLE

```
numbers[row].length
```

Remember that the second dimension is the number of columns in each row. Using rows as the first dimension, we can then use length to get the number of columns in each row.

---

3. Nested Enhanced for Loops

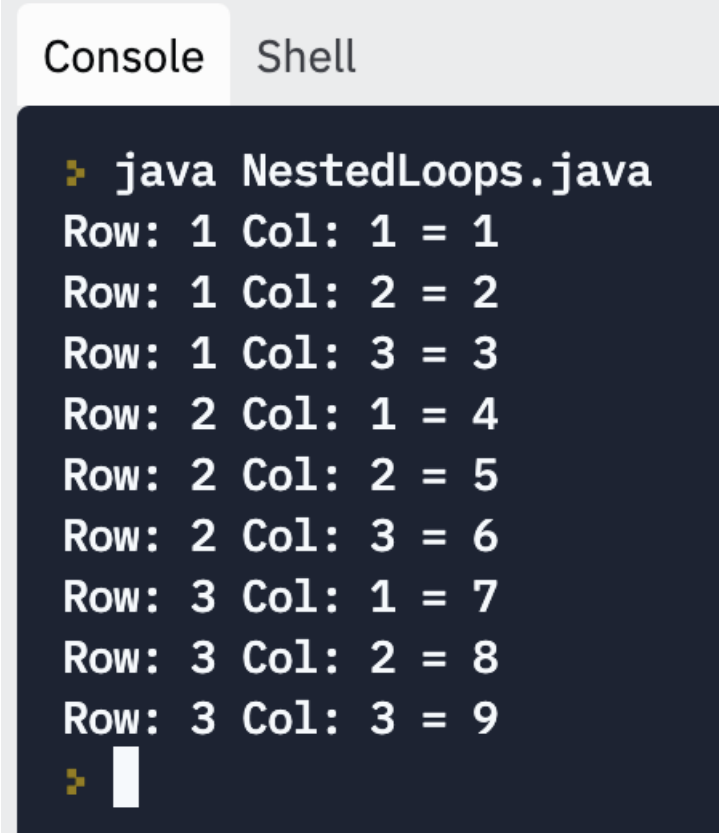
Compared to "plain" for loops, we have seen how enhanced for loops can simplify the code for iterating over an array or collection. Nested enhanced for loops can be used to iterate over a two-dimensional array, as this sample code shows.

TRY IT

Directions: Try typing in the following code in Replit in a file named NestedLoops.java:

```
class NestedLoops {
  public static void main(String[] args) {
    int[][] numbers = {
      {1, 2, 3},
      {4, 5, 6},
      {7, 8, 9}
    };
    // Without counters in loops, need rowNumber, colNumber
    int rowNumber = 1;
    int colNumber = 1;
    // Outer loop iterates over rows. Each row is a single-dimensional array
    for(int[] row : numbers) {
      // Inner loop iterates over columns in each row
      for(int value : row) {
        // Note space between colNumber++ and following +
        // Remember that ++ is increment operator, + is concatenation operator
        System.out.println("Row: " + rowNumber + " Col: " + colNumber++ +
          " = " + value);
      }
      colNumber = 1; // Reset colNumber after down with columns in row
      rowNumber++; // increment rowNumber after done with row
    }
  }
}
```

Running the code should produce the following output:



REFLECT

Nested loops may seem more complex at first, but how does using a pair of nested for loops simplify the processing of the Tic-Tac-Toe board in this code?

Since the enhanced for loops don't have counter variables, note the need to declare and initialize rowNumber and colNumber before the start of the loop. After the inner loop runs each time, the value of colNumber needs to be reset and the value of rowNumber needs to be incremented.

There is also an important detail to note in this statement:

↷ EXAMPLE

```
System.out.println("Row: " + rowNumber + " Col: " + colNumber++ +
  " = " + value);
```

The ++ is the increment operator that increases the value of colNumber by one after it has been accessed. The ++ is then followed by the concatenation operator (+).

BIG IDEA

Java doesn't recognize a triple plus sign as an operator, and the increment operator needs to be appended to the end of the variable name, since it is essential to include the space between the ++ and the + for the line to be syntactically valid and work as expected.

SUMMARY

In this lesson, you learned about using loops within other loops, also known as **nested loops**. You also

learned that using loops within other loops, or **nested for loops**, is similar to the nested conditional statements. You learned that **nested enhanced for loops** can be quite useful when you have a series of statements that includes a loop that you want to have repeated. You discovered that this can be useful when working with multidimensional arrays. Finally, you created and ran a few programs using nested loops.

Source: This content and supplemental material has been adapted from Java, Java, Java: Object-Oriented Problem Solving. Source **cs.trincoll.edu/~ram/jjj/jjj-os-20170625.pdf**

It has also been adapted from “Python for Everybody” By Dr. Charles R. Severance. Source **py4e.com/html3/**

TERMS TO KNOW

Inner Loop
A loop that is contained in the body of another loop.

Nested Loop
A nested loop is any loop that occurs inside the body of another loop.

Outer Loop
A loop that contains one or more other loops in its body.