Lesson 2.06 — nested for Loops

Overview

Objectives — *Students will be able to...*

- **Trace** nested loops to predict program behavior.
- **Construct** loops to execute simple tasks.

Assessments — *Students will...*

- **Trace and construct** nested loops in practice problems.

Homework — *Students will...*

- **Read** BJP 2.4 "Scope" and "Pseudocode"
- **Complete** self-check questions 26, 27 (4th edition 29, 30) and exercise 4

Materials & Prep

- **Projector and computer**
- **Whiteboard and marker**

Pacing Guide

Section	Total Time
Bell-work and attendance	5min
Introduction to nested for loops	15min
Practice Activity	30min
Loop Challenge	5min

Procedure

Today your hook is another wager, only this time the students have to create the code to get the reward (TEALS swag, tickets to the raffle, bonus points, etc.). If you plan to use a badge system, this project is a good one to start with (a 'loop' badge, perhaps?).

Bell-work and Attendance [5 minutes]

Introduction to Nested for Loops [15 minutes]

1. Remind students of yesterday's bet, then tell your students that you'll award them [prize] if they can rewrite the program from yesterday in 11 lines. The catch is that student can't use a loop that executes more than 10 times! Explain that you're going to give them a new programming tool today, and then they'll have some time at the end of class to see if they can meet your challenge for the points.
2. Review the concept of "**control structure**," and have students explain to you that the loop controls the statements in the loop body. Ask students to walk you through the example on the board, narrating what Java is doing as you advance your hand along the loop:

   ```
   for (int i = 1; i <= 3; i++) {
       System.out.println("Calacas y calaveras!");
   }
   ```

   You might consider deliberately leaving in an error or two as you write the code on the board—students LOVE to catch you in an error! In general, leaving in errors is a good way to condition error-checking behavior in your students without them feeling forced. If students don't catch the error before you're ready to move on, point to it and ask students if the code is correct. Occasionally point to correct code and ask students to explain why it is a proper coding choice, or what other options would be.

If your class does not celebrate Día de los Muertos, change the string above to "Trick or Treat," "All Saints Day," "Kol Sana Wenta Tayeb" or whatever is most relevant.

If your class needs more physical engagement, have a volunteer come to the board to be console output, and point to them, directing them to write output on the board every time the loop is executed. You can also have another student walk through the flow of the loop in your place.

3. As you insert another loop to create a nested loop, explain that the great thing about control structures is that they can control other control structures!

   - (Engagement option: if your class is familiar with Xzibit/Pimp my Ride, this is a great opportunity for a yo-dawg meme, but at this point, this is probably only a reference that college-age and above will get.)

   ```
   for (int i = 0; i < 3; i++) {
       for (int j = 1; j <= 3; j++) {
           System.out.println("Calacas y calaveras!");
       }
   }
   ```

   - Point out that we always use a different control variable (j instead of i) so that Java knows we're writing a new loop.
   - As you write the inner loop, ask students how many times it executes, and briefly discuss the difference between initializing at 0 and 1, and how that relates to < or <= in the test.
   - Ask students how many lines Java will output (9), and walk through the loop showing flow of control and directing a student to produce the output.

Practice Activity [30 minutes]

Depending on the mood and frustration levels in the class, you may choose to have students work in pairs.

- If students are really having a rough time, work through the first practice question together as a whole group.
- Put soft, soothing (but upbeat) music on in the background to ease tension!

1. Have students complete the following self-check questions:
   a. Self-Check 2.31: starExclamation1
   b. Self-Check 2.32: starExclamation2
   c. Self-Check 2.33: starExclamation3
2. Have students complete the Exercise 2.5: starTriangle.
   a. You should emphasize the importance of constructing a structure diagram before they start coding.
   b. Although we haven't talked about pseudocode yet, suggest that students write out in English (or whatever their preferred language is) the steps that they will need to do to solve the problem. Encourage students to work on this in pairs if needed.
3. If more 25% or more of the class is struggling, return to whole group with the stipulation that students who get it may continue working independently.

Loop Challenge [30 minutes]

1. On the board or the projector, bring up the challenge you introduced in the beginning of class.
   - Students may begin on the challenge once they have finished their practice exercises.
   - In order to encourage all students to try the challenge, allow slower students to submit their challenge answer electronically by the end of the day. (This compromise gives them time to work on the code at lunch or after school, but dissuades students from directly copying others' answers.)

LOOP CHALLENGE Write a program that outputs the first 1,000 integers in 11 lines of code. You may not use a loop that executes more than 10 times.

```
public class Count1000 {
    public static void main (String[] args) {
        for (int i = 0; i < 10; i++) {
            for (int j = 0; j < 10; j++) {
                for (int q = 0; q < 10; q++) {
                    System.out.println((i*100)+(j*10)+q+1);
                }
            }
        }
    }
}
```

Accommodation and Differentiation

If you have students who are speeding through this lesson, you should encourage them to:

- Complete Exercise 2.3: fibonacci and Exercise 2.6: numberTriangle.
- Develop an algorithm for tackling complex coding problems. (What generalizable steps did they take to correctly build the nested loop programs?)

  Have the student turn these ideas into a mnemonic, poster, or checklist to share with the class.

Misconceptions

- Determining the purpose of the two variables counters in nested loops is confusing to students. Most introductions to nested loops use "i" and "j" as the loop variables. Starting with "i" is a carryover from the Fortran programming language where variables starting with the letters I to N were integers and loop variables are integers. However, Java does not have this restriction, integer variables can start with any letter.

  In order to help students grasp the two loop variables, use loop variables in context where students may be more familiar with: row/column, x/y. This affords using a graphical representation that students can plot while tracing through the nested loops.
- Confusion of the order of execution of the 3 parts of the for loop from a single for loop gets compounded with nested for loops. The order of execution for the nested loop is:
  - outer loop initialize variable
  - outer loop test condition
  - inner loop initialize variable
  - inner loop test condition
  - body of inner loop
  - inner loop update variable
  - repeat inner loop
  - outer loop update variable
  - repeat outer loop
- Attempting to use the inner loop variable in the outside loop block. It is not obvious to beginners that even though the inner block variable is declared inside the outer block, the inner block's scope is restricted to the inner block.

Video

- BJP 2–4, *Nested for Loops* http://media.pearsoncmg.com/aw/aw_reges_bjp_2/videoPlayer.php?id=c2-4
- CSE 142, *Nested for Loops* (16:18–37:50) https://www.youtube.com/watch?v=0eUm1RFGkWw&start=2524

Forum discussion

Lesson 2.06 Nested for Loops (TEALS Discourse account required)