Lesson 2.05 — for Loops

Overview

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

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

Assessments — *Students will...*

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

Homework — *Students will...*

- **Read** BJP 2.3 "Nested for Loops"
- **Complete** self-check questions 19-21 (4th edition: 22-24)

Materials & Prep

- **Projector and computer**
- **White paper and markers**
- **Classroom copies** of WS 2.5

Pacing Guide

Section	Total Time
Bell-work and attendance	5min
Introduction to for loops	15min

Procedure

For loops are very confusing in the beginning, so we've prepared a student handout that diagrams the parts of a loop. Encourage students to take supplemental notes in their notebook and on the handout.

Hook your students by betting them $100 that you can write a program that outputs the first 1,000 integers in fewer than 10 lines of code (adjust these numbers according to the gullibility/jadedness of your classroom).

Bell-work and Attendance [5 minutes]

Introduction to for Loops [15 minutes]

Emphasize with students...

Content - Management of complexity The chart in this activity is a flow chart that represents, organizes and communicates the control of a program. As you create programs with added complexity, it's important for you to have a way to manage this complexity. You can create flow charts to do this.

Flow charts can be used to show control in a loop, to show user input, to show processing and output, and to show conditional statements and decisions.

There are several flow chart creation programs available online, these can help you manage the complexity of your programs during the design stages.

---

1. If you haven't already, distribute the handout with loop diagrams to your students. Following along with the handout:

- Start building your for loop, narrating the parts as you go. Ask students for help with the different components of the program (starting with the public class):

```
public class Count1000 {

    public static void main (String[] args) {
        for …
```

(Break to work through the first page of the handout.)

2. Call on students to read the parts 1–4. Have students place their fingers on the part of the diagram that corresponds to the explanatory text.
3. Point out that even though Java is reading across the for loop at the top, Java jumps down to check out the body of the loop to check if the test is true. If it's true, it updates the loop and executes what was in the body.
   - Trace this flow of control on the board, and have students physically trace it on their sheets as you narrate the steps again.
   - This last step is very, very important: don't skip it! (humans have had communication by touch and sight long before we evolved language—the learning centers of our brain are better able to pick up new information if we involve movement, touch, or physical manipulation).
4. Ask students to do a quick Think-Pair-Share as to what the output will be for the sample code on the handout. Get them started by writing out the first and second line of output while you trace the loop as a whole class.

   If this goes well, move on the final example and the activity for the day. If it doesn't, re-teach the concepts using the flow chart on the back of the handout.

   Again, have students trace the diagram with their hands. Have them turn the sheet back over to the code diagram, and narrate the flow of control as they move their hands to where Java is reading the program.
5. Return to the program that will output the first 1,000 positive integers in fewer than 10 lines of code. Encourage students to guess how many lines it will take to write the whole program:

```
public class Count1000 {

    public static void main (String[] args) {
        for (int i = 1; i < 1000; i++) {
            System.out.println(i);
        }
    }
}
```

   Tweak different parts of the code and ask students to predict how it will change your output.

   - Change the continuation test to `i <= 1000`.
   - Change the variable `i` to `x`.
   - Change the update to `i += 2` (or `x += 2`, depending on which code you have up there).
   - If any of these examples stump your class (and they will), take them through the loop, one step at a time, writing down sample output to find the new pattern.

   Loops can be also analyzed to determine how many times they run. This is called run-time analysis or a statement execution count. For each of the above examples, ask how many times each loop executes.

Practice Activity [35 minutes]

1. Depending on the mood and frustration levels in the class, you may choose to have students work in pairs.
   a. If students are really having a rough time, work through the first practice question together as a whole group.
   b. Put soft, soothing (but upbeat) music on in the background to ease tension!

2. Have students complete the following self-check questions:

    a. Self-Check 2.21: Count2
    b. Self-Check 2.26: fingerTrap
    c. Self-Check 2.27: howManyLines

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.

Accommodation and Differentiation

If you have students who are speeding through this lesson, you should encourage them to complete Self-Check 2.25: numberTotal and Self-Check 2.28: blastOff.

Teacher Prior CS Knowledge

The textbook BJP introduces reducing redundancy and generalization early in the curriculum. Because of the early introduction, student have not been exposed to the logical operators and have only been introduced to the notion of boolean types. You can let students know that the class will go over the middle part of the for statement, the "continuation test", will be covered in detail in a future lesson.

Misconceptions

Common Syntax Errors

- Using commas as the delimiter in the for statement is a common error. This could stem from the difference in pattern between method calls that use comma and for statements that use the semi-colon.

  Specifying that each of the 3 parts of the for statement is a Java statement that ends in a semi-colon may help students remember the delimiter is a semi-colon.

- Unbalanced parentheses and/or curly brackets starts.

  Many classes use IDEs in the classroom. When parentheses or curly brackets are off, the editor starts indenting the code in non-standard ways. This is the first indication to the student that there is something wrong with the syntax. Letting students know that the IDE helps format, but if there is a syntax error in their code while typing, the editor will not format correctly. The student may want to take a closer look at their code, specifically matching parentheses and curly brackets.

- Incorrect ending for statement with semi-colon.

Conceptual Errors

- Variable scope for the counting variable is restricted to the for block.
- Order of execution of the initialization, continuation test, conditional statements, and update statement.

  When walking through examples of a for loop, it is tempting to use boxes to represent the loop variable. However, as the loop variable changes, the value is repeated erased to indicate the new value. If a table is used and the value of the loop variable crossed out, the students can see the progression of the variable through the iterations.

Video

- BJP 2–3, *for Loops* http://media.pearsoncmg.com/aw/aw_reges_bjp_2/videoPlayer.php?id=c2-3
- CSE 142, *The for Loop* (31:30–42:03) https://www.youtube.com/watch?v=0eUm1RFGkWw&start=1890

Forum discussion

Lesson 2.05 for Loops (TEALS Discourse account required)