

2.11 Nested Iteration

Lesson Plan

Pacing: 3 class periods of 45 minutes

Learning objectives: 2.11.A

Essential knowledge: 2.11.A.1

Materials

- Projector and the Loop-de-Loop slide deck
- Whiteboard and markers
- Mini whiteboards or scrap paper
- IDE on every computer (Days 1 to 3)
- Printed Loop-de-Loop puzzle worksheet (answer key for you) and exit ticket

Agenda

Minutes	Activity
5	Day 1 warm-up — write out the whole plan (slide 2)
15	Day 1 direct instruction — the inner loop finishes first; don't reuse the name (slides 3–7)
5	Day 1 pause and try — count down, then count up (slide 8)
15	Day 1 guided practice — predict, then run OrderTrace and a nested loop of your own
5	Day 1 closure — who moves first?
5	Day 2 warm-up — how many lines print?
17	Day 2 direct instruction — rows and columns, which variable to print, where println goes (slides 9–12)
5	Day 2 pause and try — where do the X's land? (slide 13)
13	Day 2 guided practice — draw a rectangle and a numbered grid in the IDE
5	Day 2 closure — println inside or after?
5	Day 3 warm-up — how many times does the body run?
15	Day 3 direct instruction — an inner loop that depends on the outer variable (slides 14–17)
5	Day 3 pause and try — where does the inner loop start? (slide 18)
10	Day 3 putting it together — every letter against every later letter (slide 19)
5	Day 3 closure — slides 20 and 21
5	Day 3 exit ticket

Overview

A nested loop is a loop in the body of another loop (2.11.A.1). Day 1: the inner loop runs completely, from its start to its end, each time the outer body runs, so the outer variable holds still while the inner variable moves. Day 2: rows and columns, where the inner loop prints one row and the outer

loop repeats it for every row. Day 3: an inner loop whose bounds use the outer variable, so the number of inner passes changes on each outer pass, and counting the total passes.

Students know `for` and `while` loops (2.7, 2.8), `print` versus `println`, and counters. Stay inside the boundary: no arrays, no `ArrayList`, no `char`, and no run-time analysis vocabulary beyond counting passes (formal comparison of run times is Topic 2.12).

TEACHER NOTE

The slides are the Loop-de-Loop deck (21 slides); slide numbers refer to it. Day 1 uses slides 2–8, Day 2 slides 9–13, Day 3 slides 14–21. The worksheet (10 questions, about 35 minutes) is homework: questions 1, 2, and 6 after Day 1; questions 3, 4, and 5 after Day 2; questions 7, 8, 9, and 10 after Day 3. Question 9 settles the Think First guess about an inner `while` loop's counter.

Demonstrations are in the topic's `examples/` folder (`.snippet.java` files hold only the statements for `main`): `OrderTrace`, `SameNameFixed`, `CountDown` (Day 1); `Rectangle`, `RowNumbers`, `PrintlnInside`, `PrintlnFixed` (Day 2); `Triangle`, `ShrinkingNumbers`, `TriangleOffByOne`, `CountNoReset`, `CountReset`, `TriangleWhileBug`, `TriangleWhileFixed` (Day 3).

Day 1 warm-up: write out the whole plan

Show slide 2. Students write, in their own words, the full plan the slide describes, and notice that one repeated task is itself made of a repeated smaller task. Take two answers and keep them for the closure.

Day 1 direct instruction: the inner loop finishes first

Show slides 3 and 5, then run `OrderTrace`: outer 1 pairs with inner 1 and 2, then outer 2, then outer 3, six lines in all. Slide 6 shows the check-and-print sequence; have students call out each line before it appears. Slide 7: the inner loop cannot reuse the outer loop's variable name.

MISCONCEPTION Reusing the outer loop's variable name

Students paste a loop header with `int i` inside another loop that already declares `i`. The outer variable still exists inside the outer body, so Java reports that `i` is already defined; the code does not compile. Compare `SameName` and `SameNameFixed`: rename the inner variable.

Day 1 pause and try

Slide 8: count down, then count up. Students write the exact output first, then say how many times the inner body ran per outer pass.

Day 1 guided practice

Students predict the output of `OrderTrace` on mini whiteboards, then run it in the IDE. Then each student writes a nested loop of their own (for example, three outer values and four inner values, printing both numbers) and trades with a partner, who predicts the output before running it.

Day 1 independent practice

Puzzle worksheet questions 1, 2, and 6 are homework: the JVM loop with two prints per pass, the pencil check table for `c <= r`, and the outer-versus-inner face-off.

Day 1 closure

Revisit the warm-up answers. Cold-call: "The outer loop runs 5 times and the inner runs 2 times each time. How many times does the inner body run in all?" (10.)

Day 2 warm-up

Show

```
for (int r = 1; r <= 3; r++) { for (int c = 1; c <= 4; c++) { System.out.print("*"); } }  
with no println. Students say how many stars print (12) and on how many lines (one).
```

Day 2 direct instruction: rows and columns

Slide 10: the inner loop prints one row; the outer loop repeats it for every row;

`System.out.println()` after the inner loop ends each row. Slide 11 (worked example): print the inner variable for column numbers, the outer variable for row numbers; run `RowNumbers`. Slide 12: `println` inside the inner braces ends the line after every star.

MISCONCEPTION `println` in the wrong place

A student wants 2 rows of 4 stars but puts `println()` inside the inner braces, so every star gets its own line. Run `PrintLnInside` (8 lines of `*`), then `PrintLnFixed`. Teach: ending a row belongs after the inner loop, in the outer body.

Day 2 pause and try

Slide 13: students mark where the X's land in the grid and write the output. Check by tracing the first two rows aloud.

Day 2 guided practice

In the IDE, students draw a rectangle (`Rectangle`) with their own row and column counts, then a grid of numbers (`RowNumbers`), then change which variable is printed. Move around the room:

braces decide which loop a line belongs to, not indentation (worksheet WATCH OUT after question 6).

Day 2 independent practice

Puzzle worksheet questions 3, 4, and 5 (code tiles for the star rows, the snippet pool for the shrinking number rows, and the match-up that counts passes) are homework.

Day 2 closure

Show `PrintLnInside`'s code. Ask: "Where should the `println()` go to print two rows of four stars?" (Right after the inner closing brace.)

Day 3 warm-up

Show the `count++` match-up idea from worksheet question 5, version 2, on the board: `for i` from 1 to 4, `for j` from 1 to `i`. Students predict the final count (10) and keep their guess for slide 15.

Day 3 direct instruction: inner depends on outer

Slide 15: when the inner condition uses the outer variable, the inner loop runs a different number of times on every pass (1, 2, 3, 4, 5 for the triangle). Run `Triangle`. Slide 16: two inner loops in one row. Slide 17: a mistake in the inner condition repeats on every row; run `TriangleOffByOne`.

MISCONCEPTION An inner counter that never starts over

With `while` loops, you write the inner starting value yourself. Put `int game = 1;` before the outer loop and it runs once, so after the first outer pass the inner condition is false forever. Run `CountNoReset` (prints Games played: 3, not 12), then `CountReset`. The same slip is `TriangleWhileBug`, which prints only one star per row. The inner counter must be set inside the outer body. Worksheet question 9 settles it.

MISCONCEPTION An off-by-one inner condition

`<` instead of `<=` in the inner loop drops one item from every row, so the output is wrong in every row, not once. Have students count the missing stars in `TriangleOffByOne`.

Day 3 pause and try

Slide 18: students find where the inner loop starts and write the output. Ask what changes if it starts at `row` instead of 1.

Day 3 putting it together

Slide 19: every letter paired with every later letter, so the inner loop starts after the outer index. Students trace the first two outer passes before the output is revealed.

Day 3 independent practice

Puzzle worksheet questions 7, 8, 9, and 10 are homework (you're the compiler, two truths and a lie, the one-line move, the wrap-up).

Day 3 closure and exit ticket

Show slide 20 (vocabulary), then slide 21: partners explain the order in which a nested loop runs. Give the exit ticket (about five minutes). Question 1 checks tracing a nested loop with a dependent inner bound, question 2 the inner counter that never restarts, and question 3 the same bug in a `while` nest (2.11.A.1). Re-teach the weakest one at the start of Topic 2.12.