

NAME _____

DATE _____

PERIOD _____

UNIT 2 · TOPIC 2.7 · WORKSHEET

Loop the Loop

Trace the checks, rebuild a countdown, fill a savings loop, and catch the loop that's off by one.

About 35 minutes

10 questions

Cumulative through 2.7

TOPIC

2.7

**DIRECTIONS**

Work each puzzle in pencil. For code, write exactly what prints. Some puzzles include pieces you won't need.

PART A · 2 QUESTIONS

A

Warm up your brain

1

YOU'RE THE JVM

You are the Java Virtual Machine. Run this code in your head. What prints?

```
int n = 50;
int steps = 0;
while (n > 5) {
    n = n / 2;
    steps++;
}
System.out.println(steps + " " + n);
```

A 4 3**B** 3 6**C** 5 1**D** 4 6**THINK FIRST**

Does a while loop's condition get checked the same number of times as the body runs, or one more time? Guess in the margin; question 2 settles it.

2 PENCIL CHECK

Trace this loop. Each row is one check of the condition, in order. Fill in the values at the moment of the check and whether the condition is true or false.

```
int x = 1;
int total = 0;
while (total < 20) {
    total += x;
    x *= 2;
}
```

Check	x	total	total < 20
1	1	0	true
2			
3			
4			
5			
6			

How many times did the body run? _____

PART B · 4 PUZZLES

B Puzzles

3 CODE TILES

Mission control's countdown fell off the fridge. The first line stayed put. Put the tiles back in order so the program prints the output below. One tile is a decoy and isn't used.

```
int count = 3;
```

count--;	}
while (count >= 0) {	System.out.println("Liftoff!");
System.out.println(count + "...");	while (count > 0) {

Target output

3...

2...

1...

Liftoff!

YOUR PROGRAM

4 SNIPPET POOL

Hopper has \$10 and saves \$15 more each week. The program should count the weeks until Hopper has **at least** \$100 and print 6 weeks, \$100. Fill each hole from the pool.

Savings.java

```
int savings = 10;
int weeks = 0;
while ( (1) ) {
    savings (2);
    (3);
}
System.out.println(weeks + " weeks, $" + savings);
```

THE POOL

Each snippet can be used only once, and not every snippet is needed.

savings <= 100	+= 15	weeks--	savings > 100
= 15	weeks++	savings < 100	



FAIR QUESTION

Couldn't I just write the adding step six times?

Only if you already knew the answer was six. Change the goal to \$250, or the weekly amount to \$12, and the count changes. A while loop's condition says *when to keep going*, so the loop works out how many passes it needs on its own.

5 MATCH-UP

Four loops look almost the same. Each changes one thing. Match each loop to what it prints. Each output is used once, and one output in the list isn't used.

Loop	Code	Prints
1	<pre>int n = 1; while (n <= 3) { System.out.print(n + " "); n++; }</pre>	
2	<pre>int n = 1; while (n < 3) { System.out.print(n + " "); n++; }</pre>	
3	<pre>int n = 1; while (n <= 3) { n++; System.out.print(n + " "); }</pre>	
4	<pre>int n = 4; while (n <= 3) { System.out.print(n + " "); n++; }</pre>	

- **P** 2 3 4
- **Q** 1 2 3
- **R** nothing at all
- **S** 1 2
- **T** 1 2 3 4

6 YOU'RE THE COMPILER

You are the compiler. For each snippet, decide whether it compiles. If it does, write what it prints, or write *runs forever*. If it doesn't compile, say why in a few words.

Snippet	Compiles?	Output, or why not
<pre>1 int lap = 1; while (lap = 3) { lap++; }</pre>		
<pre>2 int n = 10; while (n > 20) { n++; } System.out.println(n);</pre>		
<pre>3 int count; while (count < 5) { count++; }</pre>		
<pre>4 String word = "hop"; while (word.length() < 6) { word += "!"; } System.out.println(word);</pre>		
<pre>5 int x = 7; while (x != 0) { x -= 2; } System.out.println(x);</pre>		

PART C · 4 QUESTIONS

C

Catch the mistake

7

ONE-TOKEN FIX

Coach wants Lap 1 through Lap 5, but this program stops at Lap 4. Change exactly one token so it prints all five laps.

```
1  int lap = 1;
2  while (lap < 5) {
3      System.out.println("Lap " + lap);
4      lap++;
5  }
```

LINE, OLD TOKEN, NEW TOKEN



WATCH OUT

if and while look alike, but if checks its condition **once** and never comes back. if (lap <= 5) prints one lap and moves on. When a step must happen again and again, you need while.

8 TWO TRUTHS AND A LIE

Hopper hikes until his energy runs out. Two of these claims about the code are true and one is a lie. Circle the lie, then rewrite it so it's true.

```
int energy = 10;
int steps = 0;
while (energy > 0) {
    energy -= 4;
    steps++;
}
```

- **1** The body runs 3 times.
- **2** After the loop, energy holds -2.
- **3** On the last pass, the loop stops as soon as energy drops below 1, so that pass never reaches steps++.

THE LIE, REWRITTEN

9 WHO AM I?

Some loop ideas are playing a party game. Each gives a clue about itself. Name the idea.

Clue	I am...
I check my condition before every pass, even the very first one.	
I'm one trip through a loop's body.	
My condition is true at every check, so I never end.	
I make a loop run one time too many or one time too few.	
I'm the boolean expression in the loop header that decides whether the body runs again.	
I'm the values a loop's variables need before it starts, so it works as intended.	

10 WRAP-UP**REFLECTION**

In one sentence: what has to happen inside a while loop's body for the loop to ever stop?

YOUR ANSWER