

Garfield AP CS

for loops

Shorthand operators

- There's a lot of arithmetic in programming, so we need shortcuts

increment

variable++;

decrement

variable--;

variable = variable + 1;

variable = variable - 1;

variable += value;

variable -= value;

variable *= value;

variable /= value;

variable %= value;

variable = variable + value;

variable = variable - value;

variable = variable * value;

variable = variable / value;

variable = variable % value;

Repetition over a range

```
System.out.println("1 squared = " + 1 * 1);  
System.out.println("2 squared = " + 2 * 2);  
System.out.println("3 squared = " + 3 * 3);  
System.out.println("4 squared = " + 4 * 4);  
System.out.println("5 squared = " + 5 * 5);  
System.out.println("6 squared = " + 6 * 6);
```

- Intuition: "I want to print a line for each number from 1 to 6"
- There's a statement, the `for` loop, that does just that!

```
for (int i = 1; i <= 6; i++) {  
    System.out.println(i + " squared = " + (i * i));  
}
```

- "For each integer `i` from 1 through 6, print ..."

Initialization

```
for (int i = 1; i <= 6; i++) {  
    System.out.println(i + " squared = " + (i * i));  
}
```

- Tells Java what variable to use in the loop
 - Called a *loop counter*
 - Can use any variable name, not just *i*
 - Can start at any value, not just 1

Test

```
for (int i = 1; i <= 6; i++) {  
    System.out.println(i + " squared = " + (i * i));  
}
```

- Tests the loop counter variable against a bound
 - Uses comparison operators:
 - < less than
 - <= less than or equal to
 - > greater than
 - >= greater than or equal to

Update

```
for (int i = 1; i <= 6; i++) {  
    System.out.println(i + " squared = " + (i * i));  
}
```

- Changes loop counter's value after each repetition
 - Without an update, you would have an *infinite loop*
 - Can be any expression:

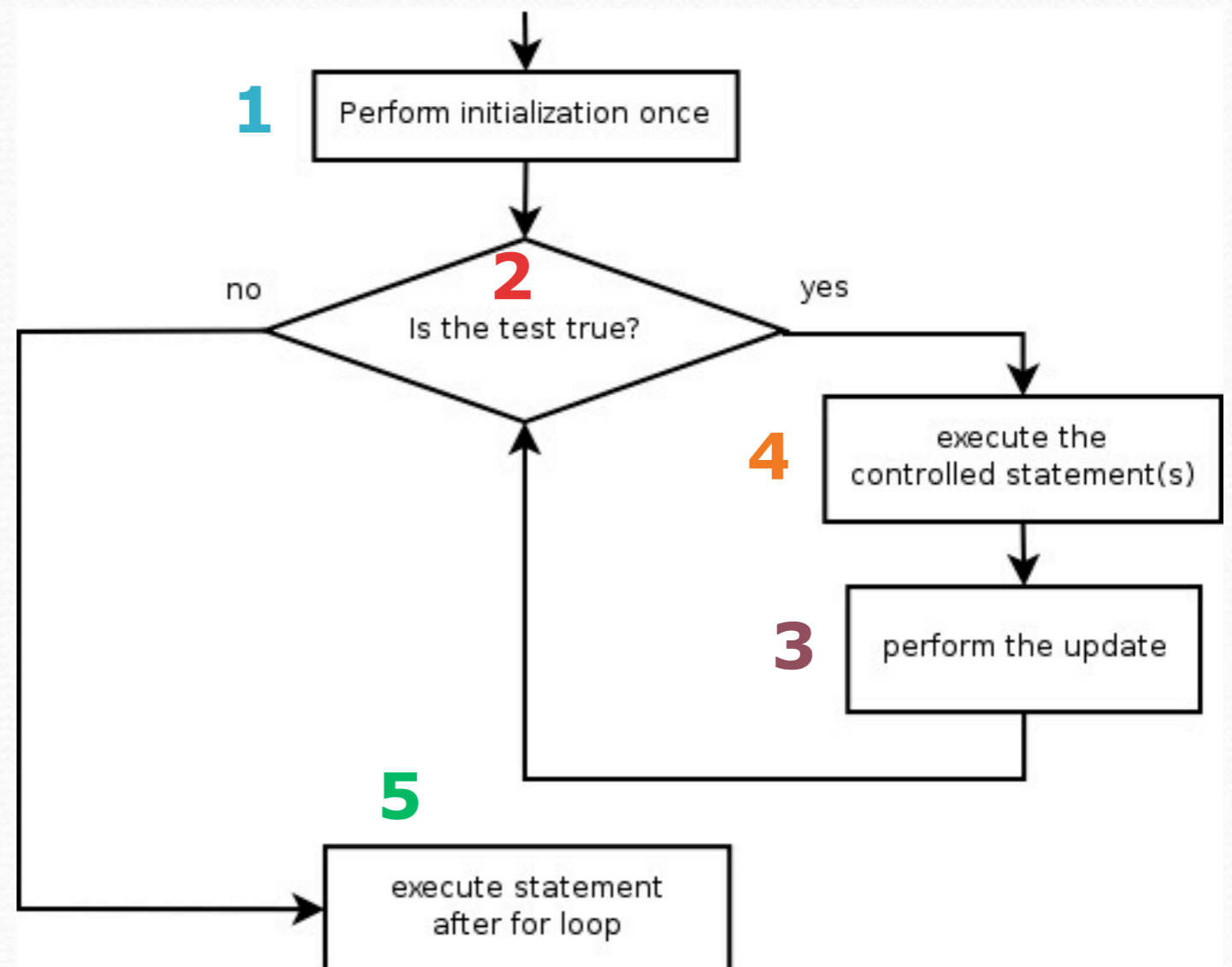
```
for (int i = 1; i <= 9; i += 2) {  
    System.out.println(i);  
}
```

Loop walkthrough

```
1 for (int i = 1; i <= 4; i++) {  
  4 System.out.println(i + " squared = " + (i * i));  
}  
5 System.out.println("Whoo!");
```

Output:

```
1 squared = 1  
2 squared = 4  
3 squared = 9  
4 squared = 16  
Whoo!
```



Loop body

- Can have multiple statements
- Doesn't have to use counter variable

Loop counter

- Can have any name
- Can start at any value
- Can increment, decrement
- Can test against an expression

System.out.print

- Prints without moving to a new line
 - allows you to print partial messages on the same line

```
int highestTemp = 5;
for (int i = -3; i <= highestTemp / 2; i++) {
    System.out.print((i * 1.8 + 32) + " ");
}
```

- Output:

26.6 28.4 30.2 32.0 33.8 35.6

Mapping loops to numbers

- What for loop would output
4 7 10 13 16
- What for loop would output
2 7 12 17 22

Loop tables

2 7 12 17 22

- To see patterns, make a table of `count` and the numbers.
 - Each time `count` goes up by 1, the number should go up by 5.
 - But `count * 5` is too great by 3, so we subtract 3.

<code>count</code>	number to print	<code>5 * count</code>	<code>5 * count - 3</code>
1	2	5	2
2	7	10	7
3	12	15	12
4	17	20	17
5	22	25	22

You try 17 13 9 5 1

Loop errors

- Forgetting curly braces
- Infinite loops
- Loops that never run