Loops and arrays tutorial

In this tutorial, you'll write for loops and work with an array.

For loops allow you to run the same lines of code a given number of times. Arrays hold a collection of related values of the same data type.

To get started, import this project into IntelliJ. You'll write your code in the `src/main/java/com/techelevator/Tutorial.java` file.

Step One: Write a for loop that increments a value

As stated earlier, a for loop runs the same code a given number of times. For loops typically use an `int` to keep track of the number of times the loop has run.

In the `main` method, create a new `int` and name it `i`:

```
int i;
```

You learned to use meaningful names and this may not seem very meaningful. However, for a loop counter, a single character is commonly used. Since you'll often type that variable name three times on one line at the beginning of the loop, brevity is convenient. You don't have to use `i`, but it's a convention that you'll see very often.

Next, add the following to create a for loop:

```
for (i = 0; i <= 5; i++) {

}
```

Inside the parentheses, there are three things going on:

- `i = 0`: this is the **initializer**. It sets the initial state before the loop runs. Typically, it sets the loop counter—`i` in this case—to the initial value you want to use.
- `i <= 5`: this is the **condition**. The loop runs as long as this expression evaluates to true.
- `i++`: this is the **iterator**. After the loop runs, this expression is evaluated. The `++` part means increment `i` by 1 each time.

Looking at the three parts of the loop statement, you can determine that the loop runs six times—for `i` values of `0` through `5`.

Next, add this line inside the loop to display the value `i`:

```
for (i = 0; i <= 5; i++) {
    System.out.println(i);
```

```
}
```

Run the application now. You'll see the values `0` through `5` printed on the console:

```
0
1
2
3
4
5
```

Step Two: Write a for loop that decrements a value

You don't always have to increment a value in a loop. You can also decrement a value.

Create another loop after the one you created in the previous step. This time, you'll start the loop counter at `10` and decrement the value to `0`:

```
for (i = 10; i >= 0; i--) {

}
```

Notice that you assign `i` the value of `10` at the start of the loop. As long as `i` is greater than or equal to `0`, the loop runs. After each loop, you decrement `i` by 1.

Inside this loop, add a line to display the value of `i` again:

```
for (i = 10; i >= 0; i--) {
    System.out.println(i);
}
```

Now when you run the application, you'll see the output of the first loop, and then the output of the second loop:

```
0
1
2
3
4
5
10
9
8
7
6
```

```
5
4
3
2
1
0
```

Step Three: Create an array and loop through its values

You can use loops to quickly iterate through values in an array. In this step, you'll create and populate an array that contains the forecast temperatures for the next five days, and then write a loop to display the values of the array.

First, create an array that holds five `int`s:

```
int[] forecastTemperatures = new int[5];
```

Next, you need to populate the values in the array.

You can access each element in an array using its index value, starting with `0`. This index value goes in square brackets—`[]`—after the array name.

You can assign a value to the first array element like this:

```
forecastTemperatures[0] = 72;
```

The first element in `forecastTemperatures` now has the value of `72`.

Populate the other four elements the same way:

```
forecastTemperatures[1] = 78;
forecastTemperatures[2] = 58;
forecastTemperatures[3] = 79;
forecastTemperatures[4] = 74;
```

You can always change values in an array after you've set them. For the third element, array index `2`, you'll add 10 to its value. Add this line next:

```
forecastTemperatures[2] = forecastTemperatures[2] + 10;
```

> You can also use the shorthand expression `forecastTemperatures[2] += 10;`.

Now, you can write a loop to iterate through its values. This loop looks similar the one you wrote in step one:

```
for (i = 0; i < forecastTemperatures.length; i++) {
    System.out.println(forecastTemperatures[i]);
}
```

In this code, you initialize `i` to `0` because that's the index of the first element in the array. `i` must be less than `forecastTemperatures.length` for the loop to run. Instead of printing `i`, you're using `i` to access elements in the array.

Notice that the condition is `<`, not `<=`. You don't want the loop to run all the way to `5` because the highest index in the array is `4` since the array counting starts at `0`.

When you run the application, it now displays this after the output from the previous steps:

```
72
78
68
79
74
```

Step Four: Write a loop to find the highest temperature

You can use arrays and loops for tasks that do more than print numbers to the console. In this step, you'll write a loop to find the highest temperature and the day that temperature occurs on.

Before writing the loop to find the highest value, you'll need two additional variables: one to hold the highest temperature, and one to hold the index of that value.

First, create an `int` to hold the highest temperature and assign it to the first element in the array—`forecastTemperatures[0]`:

```
int highestTemperatureValue = forecastTemperatures[0];
```

Next, create an `int` to hold the index of that value:

```
int highestTemperatureIndex = 0;
```

You're assigning these values to match the first element in the array for the moment and will change them later inside the loop if you find a higher value. In other words, you're assuming the first element is the highest one in the array unless you find a higher element.

Next, declare the loop:

```

```

```
for (int j = 1; j < forecastTemperatures.length; j++) {

}
```

Notice that this loop looks different from the previous ones you wrote. This loop uses a different loop counter variable—`j`—and sets the value to 1. It's not `0` because `highestTemperatureValue` already holds the first element, so you don't need to compare it again.

> This also shows you that you don't need to declare the loop counter variable first. You can declare it in the loop statement itself.

Inside the for loop, you need to compare each element to the highest temperature found so far, and save this value and index if this value is higher:

```
for (int j = 1; j < forecastTemperatures.length; j++) {
    if (forecastTemperatures[j] > highestTemperatureValue) {
        highestTemperatureValue = forecastTemperatures[j];
        highestTemperatureIndex = j;
    }
}
```

Each time the loop runs, `forecastTemperatures[j]` points to a different value in the array, based on the value of `j`. The `if` statement compares that to the value stored in `highestTemperatureValue`. If the value in the array is greater than the value stored in `highestTemperatureValue`, then it's updated with that value. The index of that value is stored in `highestTemperatureIndex` since you want to know on what day the highest temperature is going to occur.

Lastly, you need to display those values.

Add this line after the for loop to display the value of `highestTemperatureValue`:

```
System.out.println("The highest temperature is " + highestTemperatureValue);
```

This expression is called **string concatenation**. The Java compiler automatically converts the `int` value of `highestTemperatureValue` to a `String` so it can combine it with the text `The highest temperature is ` .

Then add this line to display how many days it takes to reach that temperature:

```
System.out.println("The highest temperature is in " + (highestTemperatureIndex +
1) + " days");
```

Notice in this line the expression `(highestTemperatureIndex + 1)`. Since array counting starts at `0`, and the forecast is for days *after* today, you don't want to display tomorrow's forecast and say it's today's. So, you need to increment the value by 1 when displaying it.

When you run the application, it now displays this after the output from the previous steps:

```
The highest temperature is 79
The highest temperature is in 4 days
```

Next steps

For further exploration, play around with the code you added today. What happens if you add or remove values from the `forecastTemperatures` array? What if the highest temperature is the first value, or the only value? If the highest value is the first value, can you get the output to be something like, "The highest temperature is tomorrow", and not display the number?