Loops and Arrays

In this course, you will often work in teams of 3–4 students to learn new concepts. The "meta" activity will introduce you to the team framework. Then, we'll look at how to structure ***loops*** in Java. Finally, programs often need to store multiple values of the same type, such as a list of phone numbers or the names of students. Rather than create a separate variable for each one, we can store them together using an ***array***.

Content Learning Targets

After completing this activity, you should be able to say:

- I can describe the responsibility of each role and summarize the benefits of teamwork.
- I can identify the three main components of a while loop.
- I can rewrite a while loop as a for loop (and vice versa).
- I can declare and initialize array variables of primitive types.
- I can iterate (i.e., loop) over arrays of primitives.

Process Skill Goals

During the activity, you should make progress toward:

- Leveraging prior knowledge and experience of other students. (Teamwork)
- Justifying answers based on evidence provided in the model. (Problem Solving)
- Tracing execution of while/for loops and predicting their output. (Critical Thinking)

Facilitation Notes

The meta activity refers to Role Cards, which ideally should be printed out on a different color of card stock (one set per team). You might want to have each team complete a Team Report.

Model 1 addresses two misconceptions about assignment: reassigning and swapping.

When reporting out Model 2, have teams predict the answers to #15, #16, and #20, then run and discuss. Consider showing a debugger. Model 3 is a quick intro to for loops.

Model 4 introduces arrays for the first time, with a focus on array syntax. Explain how Java requires the `new` keyword when creating arrays, except for array initializers.

In Model 5, `new` zeros-out memory for the array. Default: 0 for `int`, 0.0 for `double`, etc. Model 6 shows how to loop over arrays.

Key questions: #11, #20, #25, #32, #33

Source files: *Assignment.java*, *WhileLoops.java*, *ForLoops.java*, *ArrayIteration.java*

Meta Activity: Team Roles

Decide who will be what role for today; we will rotate the roles each week. If you have only three people, one should have two roles. If you have five people, two may share the same role.

Manager:	Helen Hu	keeps track of time, all voices are heard
Presenter:	Clif Kussmaul	asks questions, gives the team's answers
Recorder:	Chris Mayfield	quality control and consensus building
Reflector:	Aman Yadav	team dynamics, suggest improvements

Questions (15 min) **Start time:**

1. What is the difference between **bold** and *italics* on the role cards?

The bold points describe what the responsibilities are. Examples of what that person would say are in italics.

2. Manager: invite each person to explain their role to the team. Recorder: take notes of the discussion by writing down key phrases next to the table above.

3. What responsibilities do two or more roles have in common?

Both the presenter and the recorder help the team reach consensus. The manager and reflector both monitor how the team is working.

4. For each role, give an example of how someone observing your team would know that a person is <u>not</u> doing their job well.

- Manager: The team is constantly getting behind.
- Presenter: The student doesn't know what to say.
- Recorder: Some team members aren't taking good notes.
- Reflector: The student never comments on team dynamics.

Model 1 Assignment

Consider the following Java statements. What is the resulting value of each variable?

A:

```
int x, y;
x = 1;
y = 2;
y = x;
x = y;
```

Value of x: 1

Value of y: 1

B:

```
int x, y, z;
x = 1;
y = 2;
z = y;
y = x;
x = z;
```

Value of x: 2

Value of y: 1

Value of z: 2

C:

```
int z, y;
z = 2;
z = z + 1;
z = z + 1;
y = y + 1;
```

Value of z: 4

Value of y: ?

Questions (15 min)

Start time:

5. In program A, why is the value of `x` not 2?

Each statement is executed one after the other, so the third assignment changes the value of `y` to 1. The last assignment then assigns 1 to the value `x`.

6. In program B, what happens to the values of `x` and `y`?

They get swapped; `x` was 1 and `y` was 2, but in the end `x` was 2 and `y` was 1.

7. In program B, what is the purpose of the variable `z`?

It is a temporary variable that makes it possible to swap the values of `x` and `y`.

8. If program C runs, what happens to the value of `z`?

It gets incremented twice; the value starts at 2, then it becomes 3, and then it becomes 4.

9. In program C, why is it possible to increment `z` but not `y`?

The variable `z` was initialized, but `y` was not. Java doesn't know what value to increment.

10. Because ***increment*** and ***decrement*** are so common in algorithms, Java provides the operators `++` and `--`. For example, `x++` is the same as `x = x + 1`, and `y--` is the same as `y = y - 1`. Write the value of `x` and `y` next to each statement below.

```
int x = 5;
x--;
x--;
```

```
x is 5
x is 4
x is 3
```

```
int y = -10;
y++;
y++;
```

```
y is -10
y is -9
y is -8
```

11. Like the assignment operator, the `++` and `--` operators replace the value of a variable. Java also has ***compound assignment*** operators for convenience. For example, the statement `x = x + 2` can be rewritten as `x += 2`. Simplify the following assignment statements.

```
step = step + 5;
size = size - 3;
total = total * 2;
change = change / 10;
hours = hours % 24;
```

```
step += 5;
size -= 3;
total *= 2;
change /= 10;
hours %= 24;
```

12. Which of the following assignment statements can also be rewritten like the ones in #11?

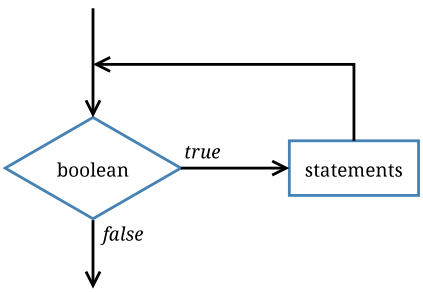
```
step = 5 + step;
size = 3 - size;
total = 2 * total;
change = 10 / change;
hours = 24 % hours;
```

```
step += 5;
NO
total *= 2;
NO
NO
```

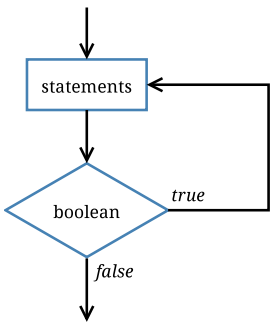
Model 2 While Loops

A loop is a set of instructions that are to be repeated. All loops have three main components: ***initialize***, ***test***, and ***update***. Identify these components in each of the examples below.

```
// pre-test loop
number = 1;
while (number <= 10) {
    System.out.println(number);
    number++;
}
```




```
// post-test loop
number = 1;
do {
    System.out.println(number);
    number++;
} while (number <= 10);
```



Questions (20 min) Start time:

13. Which loop component always happens first? Why?

The initialize step; you need to tell the loop where to begin. And variables cannot be updated until they have an initial value.

14. Explain why the `while` loop is called a ***pre-test*** and the `do while` loop is called a ***post-test***.

The `while` tests its condition before the loop body, whereas the `do while` tests its condition after the loop body.

15. What is output to the screen by each loop? Predict the output first, and then run the code to check your answer.

They both print the numbers 1 through 10, with each number on its own line.

16. What is the final value of `number` at the end of each loop? Make a prediction first, and then add print statements to the code to check your answer.

At the end of each loop, the value of `number` is 11.

17. How does the output change if you swap the `println` and `number++` statements?

Both loops print the values 2 through 11 instead.

18. What is the output to the screen if you remove the `number++` statement?

Both loops will print the value 1 forever, since `number` never reaches the stopping condition.

19. What is the difference between a `while` statement and an `if` statement?

They identical syntax and a similar meaning; the only difference is a `while` statement repeats the code between its braces as long as the condition is `true`.

20. What is output by the following loop? Explain how the code works.

```
number = 99;
do {
    System.out.println(number);
    number++;
} while (number <= 10);
System.out.println(number);
```

It will print the numbers 99 and 100; the `do while` loop does not repeat since 99 is greather than 10.

21. What is output by the following loop? And what mistake was made?

```
int i = 0;
while (i < 3)
    System.out.println("i = " + i);
    i = i + 1;
```

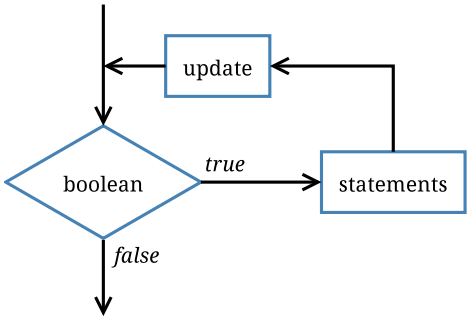
It will print `"i = 0"` forever. Without braces, the loop only executes the first statement, and `i = i + 1;` is never reached.

Model 3 For Loops

The `for` loop combines ***initialize***, ***test***, and ***update*** into one line of code.

```
// Loop A: count forwards
for (i = 1; i <= 10; i++) {
    System.out.println(i);
}

// Loop B: count backwards
for (i = 10; i >= 1; i--) {
    System.out.println(i);
}
```



Questions (10 min) Start time:

22. Identify the components of each `for` loop.

Loop A:

a) initialize `i = 1`

b) test `i <= 10`

c) update `i++`

Loop B:

a) initialize `i = 10`

b) test `i >= 1`

c) update `i--`

23. Rewrite each `for` loop as a `while` loop.

Loop A:

```
i = 1;
while (i <= 10) {
    System.out.println(i);
    i++;
}
```

Loop B:

```
i = 10;
while (i >= 1) {
    System.out.println(i);
    i--;
}
```

24. What do each of the `for` loops output to the screen? Be specific.

The first loop prints the numbers 1 to 10, and the second loop prints the numbers 10 to 1. Each number is on its own line.

25. Describe how to change the `for` loops to print even numbers only (i.e., the output should be 2 4 6 8 10 and 10 8 6 4 2).

Change loop A to: `for (i = 2; i <= 10; i += 2)`

Change loop B to: `for (i = 10; i >= 2; i -= 2)`

26. In mathematics, the factorial of an integer n (denoted by $n!$) is the product of all positive integers less than or equal to n. For example, the factorial of 5 is:

$$5! = 5 * 4 * 3 * 2 * 1 = 120$$

The following code computes the factorial of 5:

```
fact = 1;
i = 5;
while (i > 1) {
    fact *= i;
    i--
}
```

a) Rewrite the code above using a `for` loop instead of a `while` loop.

```
fact = 1;
for (i = 5; i > 1; i--) {
    fact *= i;
}
```

b) How would you change the code to compute the factorial of 12?

Simply change `i = 5` to `i = 12`.

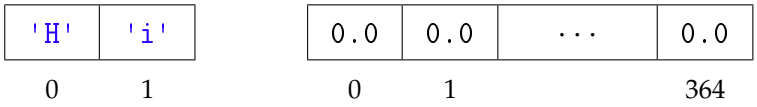
Model 4 Array Syntax

An ***array*** variable allows you to store multiple variables (of the same type). Each value in an array is known as an ***element***. The programmer must specify the ***length*** of the array (the number of array elements). Once the array is created, its length cannot be changed.

```
char[] letterArray = {'H', 'i'};
System.out.println(letterArray[0]);          // outputs H
System.out.println(letterArray.length);      // outputs 2

double[] numberArray = new double[365];
System.out.println(numberArray[0]);          // outputs 0.0
System.out.println(numberArray.length);      // outputs 365
```

Array elements are accessed by ***index*** number, starting at zero:



Questions (5 min) Start time:

27. Examine the results of the code.

a) What is the length of `letterArray`? 2

b) What is the length of `numberArray`? 365

c) What is the index of the element `'i'` in `letterArray`? 1

d) What is the index of the last element of `numberArray`? 364

28. Now examine the syntax of the code.

a) What are three ways that square brackets [] are used?

1) To declare the type: `double[]`
2) To specify the length: `double[365]`
3) To access an element: `numberArray[0]`

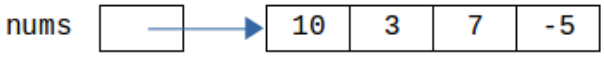
b) In contrast, how are curly braces {} used for an array?

To create an array with an initial set of values.

Model 5 Array Diagrams

Array elements are stored together in one contiguous block of memory. To show arrays in box-and-pointer memory diagrams, we simply draw adjacent boxes.

```
int[] nums = {10, 3, 7, -5};
```




```
double[] stats = new double[3];
```

Questions (15 min) Start time:

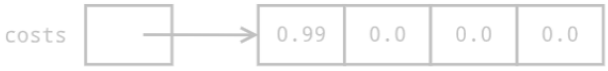
29. What is the default value for array elements?

Zero or equivalent value, depending on the data type. For numeric types like `int` and `double`, the default is 0; for `boolean`, it's `false`; for `char`, it's `'\u0000'`.

30. Draw a memory diagram for the following arrays. (*Hint:* You should have no empty boxes.)

a)
```
int[] sizes = new int[5];
sizes[2] = 7;
```

b)
```
double[] costs = new double[4];
costs[0] = 0.99;
```



31. Arrays are ***reference types***, not primitives. What is the ***value*** of an array variable?

An integer representing the memory location of the array.

32. Does the statement `int[] copy = nums;` create a new array? Justify your answer.

No. If you assign one array variable to another, you're only copying the reference, not the array itself.

33. Draw a memory diagram of the following array.

```
double[] prices = new double[4];
prices[1] = 10 / 4;
prices[3] = 3.49;
```

Model 6 Array Iteration

We can use loops with arrays to fill in initial values or access/modify current values. First, work these problems out on a whiteboard or paper. Once your team has reached a consensus, check your answers using `ArrayIteration.java`.

Questions (15 min) Start time:

34. Write a Java for loop to iterate through an array of integers named `arr` and add their values, storing the total in an integer called `sum`. (***Hint:*** How do you know when to stop?)

We use the length of `arr` to know when to stop:

```
int sum = 0;
for (int i = 0; i < arr.length; i++) {
    sum += arr[i];
}
```

35. Write a Java for loop to populate an empty (i.e., zeroed-out) array of integers named `triangles` with the triangular numbers up to the length of the array, starting with 0.

This is one of several reasonable solutions.

```
triangles[0] = 0;
for (int i = 1; i < triangles.length; i++) {
    triangles[i] = triangles[i-1] + i;
}
```

36. Write a java while loop to find the first negative value in an array of doubles called `earnings`. Print both the index and the value. if all values are nonnegative, print the message `No negative earnings.`

This is one of many ways to solve this problem.

```
int i = 0;
while (i < earnings.length && earnings[i] >= 0) {
    i++;
}
if (i >= earnings.length) {
    System.out.println("No negative earnings.");
} else {
    System.out.println(i + " " + earnings[i]);
}
```