Loops and Arrays

Programs often need to store multiple values of the same type, such as a list of phone numbers, or the names of your top 20 favorite songs.

Manager: Recorder:

Presenter: Reflector:

Content Learning Objectives

After completing this activity, students should be able to:

- Compare the components of a while loop and a for loop.
- Declare and initialize array variables of primitive types.
- Predict the output of methods called on a string object.

Process Skill Goals

During the activity, students should make progress toward:

- Tracing the execution of loops over arrays and strings. (Critical Thinking)

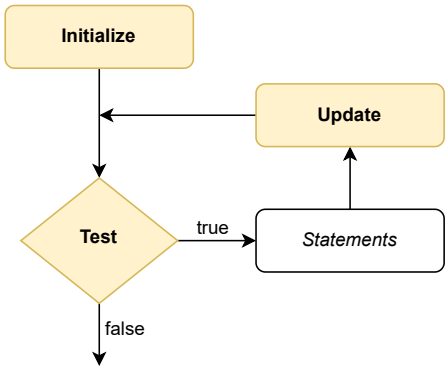
Model 1 Loop Components

A loop is a block of statements that might run multiple times. A loop has three components:

a) The loop variable is initialized.

b) The loop variable is tested:
   - If true, the loop continues.
   - If false, the loop ends.

c) The loop variable is updated.



Note: In Java, `i++` is equivalent to `i = i + 1`, and `i--` is equivalent to `i = i - 1`.

Questions (15 min) Start time:

1. Identify the components of each `while` loop:

Loop A:

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

a) initialize:

b) test:

c) update:

Loop B:

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

a) initialize:

b) test:

c) update:

2. What does each `while` loop output on the screen?

3. What is the final value of `i` at the end of each loop?

4. Identify the components of each `for` loop:

Loop A:

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

a) initialize:

b) test:

c) update:

Loop B:

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

a) initialize:

b) test:

c) update:

5. Explain how the loop components are arranged differently in a for loop, in comparison to a while loop.

6. Describe how to change the `for` loops to print even numbers only. The output for Loop A should be 2 4 6 8 10, and the output for Loop B should be 10 8 6 4 2.

7. In math, the factorial of an integer n is the product of all positive integers less than or equal to n. For example, the factorial of 5 is:

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

The following while loop computes the factorial of 5:

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

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

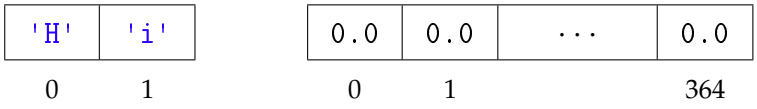
Model 2 Array Syntax

An ***array*** variable allows you to store multiple values 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, the 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 (15 min) Start time:

8. Examine the results of the code.

a) What is the length of `letterArray`?

b) What is the length of `numberArray`?

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

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

9. Now examine the syntax of the code.

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

b) What is the only way that curly braces {} can be used?

10. What are the resulting type and value of the following expressions? Show your work by writing the value of each array element in the space provided.

```
int[] a = {3, 6, 15, 22, 100, 0};
double[] b = {3.5, 4.5, 2.0, 2.0, 2.0};
String[] c = {"alpha", "beta", "gamma"};
```

a) `a[3] + a[2]` Type: Value:

b) `b[2] - b[0] + a[4]` Type: Value:

c) `c[1].charAt(a[0])` Type: Value:

d) `a[4] * b[1] <= a[5] * a[0]` Type: Value:

As shown in #10, an array variable can be declared and initialized without using the `new` keyword. However, to assign an array variable that was previously declared, the `new` keyword is required:

```
a = new int[] {3, 6, 15, 22, 100, 0};
c = new String[] {"alpha", "beta", "gamma"};
```

11. Write statements that declare and initialize variables with the following arrays (without using the `new` keyword).

a)

0	14	1024	127	3	5521

b)

3.23	1.52	4.23	32.5	2.45	5.23	3.33

12. Write statements that assign the following `new` arrays to the variables you declared in #11.

a)

0	14	1024	127	3	5521

b)

3.23	1.52	4.23	32.5	2.45	5.23	3.33

Model 3 String Methods

Method	Returns	Description
`charAt(int)`	`char`	Returns the char value at the specified index of `this` string.
`indexOf(String)`	`int`	Returns the index within `this` string of the first occurrence of the specified substring.
`length()`	`int`	Returns the length of `this` string.
`substring(int, int)`	`String`	Returns a new string that is a substring of `this` string (from `beginIndex` to `endIndex - 1`).
`toUpperCase()`	`String`	Returns a copy of `this` string with all the characters converted to upper case.

Each method listed above is non-`static`. That is, they have an ***implicit parameter*** named `this` that is passed automatically. (Note: There are many other `String` methods not listed above.)

Questions (15 min) Start time:

13. If the variable `str` references the string `"hello world"`, then what is the return value of the following method calls?

a) `str.charAt(8)`

b) `str.indexOf("wo")`

c) `str.length()`

d) `str.substring(4, 7)`

e) `str.toUpperCase()`

14. Explain what precedes the dot operator (`.`) in the expressions above. What does it have to do with the keyword `this` in Model 3?

15. How many arguments does each method call in #13 have? (Hint: None of the answers are zero; don't forget to count the implicit argument.)

a)

b)

c)

d)

e)

To call a `static` method, you write *ClassName.methodName()*. Example: `Math.abs(-5)`

To call a non-`static` method, you write *objectName.methodName()*. Example: `str.charAt(8)`

Methods can be designed either way. Most `String` methods are non-`static`, because that makes the code easier to read.

Static (`str` passed explicitly)	**Non-Static (`str` passed implicitly)**
`String.charAt(str, 8) // wrong`	`str.charAt(8)`

16. Label each method call below as `static` or non-`static`.

a) `color.indexOf("RED")`

b) `String.format("%3d", x)`

c) `title.substring(0, 10)`

d) `String.valueOf(3.14)`

e) `name.charAt(0)`

17. Consider the following statement and compiler error. Why is it impossible to call `charAt()` this way? How would you correct the error?

```
char c = String.charAt(0);
```

Error: non-static method charAt(int) cannot be referenced from a static context

18. For each method in #16, what object is referenced by `this`? Write N/A if `this` is not passed to the method.

a)

b)

c)

d)

e)