Activity 6: Loops

Computers are often used to perform repetitive tasks. Running the same statements over and over again, without making any mistakes, is something that computers do very well.

Content Learning Objectives

After completing this activity, students should be able to:

- Explain what happens when re-assigning a variable.
- Identify the three main components of a while loop.
- Implement the factorial function using a for loop.

Process Skill Goals

During the activity, students should make progress toward:

- Tracing the execution of while/for loops and predict their final output. (Critical Thinking)

Facilitation Notes

Model 1 addresses two misconceptions about assignment: what happens when you reassign a variable, and how do you swap two variables. This understanding is essential for loop updates.

Consider bringing a pair of note cards labeled with a large 1 on the first and a large 2 on the second card. Have all team members put one hand behind their back, and then hand the cards to two students (student *X* gets the 1 and student *Y* gets the 2). Ask them to exchange cards, and if they struggle, say "What if you had a student *Z* who could help?" (Credit: Debra Duke)

When reporting out Model 2, have teams predict the answers to #11, #12, and #15. Then execute the code and have them discuss their results, if incorrect. Consider stepping through the code with a debugger (on the projector) during report-out.

Be sure to make connections between the flow charts in Model 2 and Model 3. You may need to guide students on #20 applying string methods in the context of a loop. On #23, students might not think to use a `long` variable to store the factorial. If time permits, show the output of the solution with `fact` declared as an `int`, and ask them to fix the code.

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: _____

1. 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.

2. 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.

3. 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.

4. 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.

5. 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.

6. 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 is 5          int y = -10;      y is -10
x--;              x is 4          y++;              y is -9
x--;              x is 3          y++;              y is -8
```

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

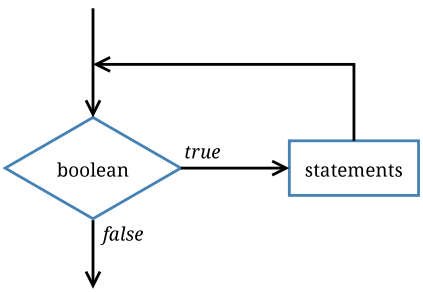
8. Which of the following assignment statements can also be rewritten like the ones in #7?

```
step = 5 + step;          step += 5;
size = 3 - size;          NO
total = 2 * total;        total *= 2;
change = 10 / change;     NO
hours = 24 % hours;       NO
```

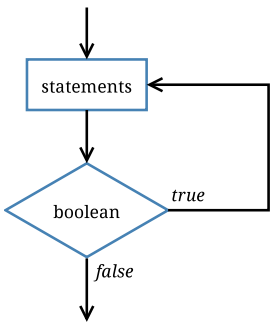
Model 2 While Loops

A loop is a set of instructions that are to be repeated. All loops have three main components: *initialize*, *test*, and *update*. Label each of these components in the two example loops 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 (15 min) Start time: _____

9. 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.

10. 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.

11. What is output (to the screen) by each loop?

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

12. What is the final value of `number` at the end of each loop?

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

13. What is output if you swap the `println` and `number++` statements?

Both loops print the values 2 through 11 instead.

14. What is the output if you remove the `number++` statement?

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

15. What is output by the loop below?

```
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.

16. What is the output of the following loop? (And what mistake was made?)

```
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.

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

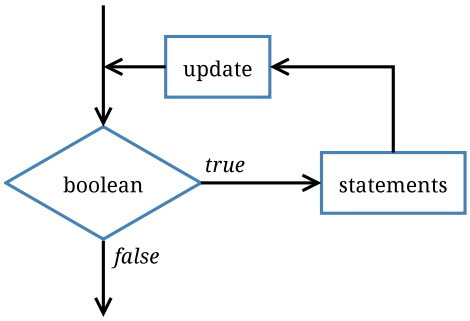
They identical syntax and a similar meaning; the only difference is a `while` statement repeats the code between its braces as long as the condition is `true`.

Model 3 For Loops

The for loop combines *initialize*, *test*, and *update* into one line of code. Label each of these components in the two example loops below. (Assume the variable `number` has already been declared.)

```
// count forwards
for (number = 1; number <= 10; number++) {
    System.out.println(number);
}

// count backwards
for (number = 10; number >= 1; number--) {
    System.out.println(number);
}
```



Questions (15 min) Start time: _____

18. 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.

19. Describe how to make these loops display even numbers only (2 4 6 8 10 and 10 8 6 4 2).

Change the first loop: `for (number = 2; number <= 10; number += 2)`.
Change the second loop: `for (number = 10; number >= 2; number -= 2)`.

20. Write a `for` loop that prints each character of a string on a separate line. You will need to invoke the `length()` and `charAt()` methods. Assume the string variable is named `word`.

```
for (int i = 0; i < word.length(); i++) {
    System.out.println(word.charAt(i));
}
```

21. Rewrite your `for` loop in #20 as a `while` loop.

```
int i = 0;
while (i < word.length()) {
    System.out.println(word.charAt(i));
    i++;
}
```

22. Write a loop that computes the factorial of a given integer `n`. Recall that $n! = n * (n-1) * (n-2) * \ldots * 1$. Store your result in a variable named `fact`.

```
long fact = 1;  // not int
for (int i = n; i > 1; i--) {
    fact *= i;
}
```

23. A *nested loop* is one that exists within the scope of another loop. This construct is often used when there are two variables for which all combinations must be examined.

```
for (int i = 0; i < 10; i++) {
    for (int j = 0; j < 10; j++) {
        System.out.printf("The product of %d and %d is %d\n", i, j, i * j);
    }
    System.out.println();
}
```

Write nested loops that compute and display the factorial of each integer from 1 to 20. (Reuse your code from the previous question.) Your output should be in this format:

```
The factorial of 1 is 1
The factorial of 2 is 2
The factorial of 3 is 6
The factorial of 4 is 24
```

```
for (int n = 1; n <= 20; n++) {
    long fact = 1;  // not int
    for (int i = n; i > 1; i--) {
        fact *= i;
    }
    System.out.printf("The factorial of %d is %d\n", n, fact);
}
```