

# 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)



# Model 1 Assignment

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

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

Value of x: \_\_\_\_\_

Value of y: \_\_\_\_\_

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

Value of x: \_\_\_\_\_

Value of y: \_\_\_\_\_

Value of z: \_\_\_\_\_

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

Value of z: \_\_\_\_\_

Value of a: \_\_\_\_\_

## Questions (15 min)

Start time: \_\_\_\_\_

1. Why is the value of x not 2 in the first section of code?
2. What happens to the values of x and y in the second section of code?
3. What is the purpose of the variable z in the second section of code?
4. What happens to the value of z in the third section of code?
5. Why is it possible to increment z but not a in the third section of code?

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

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

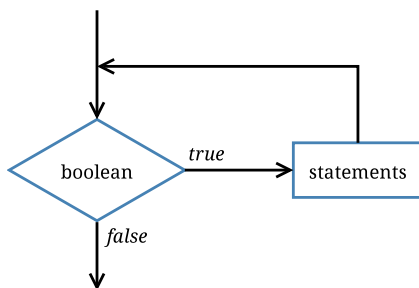
8. Which of the following assignment statements can also be rewritten like the ones in #7?

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

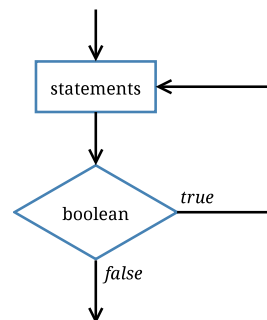
## Model 2 While Loops

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);
```





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

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

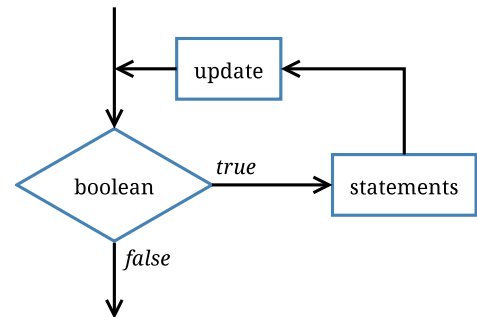
17. What is the difference between a `while` statement and an `if` statement?

### 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.

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

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

22. 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);  
    }  
}
```

Write loops that compute and display the factorial of each integer from 1 to 20. Recall that  $n! = n * (n - 1) * (n - 2) * \dots * 1$ . 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  
The factorial of 5 is 120
```