# Loops and Iteration

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.

| | | | |
|---|---|---|---|
| Manager: | | Recorder: | |
| Presenter: | | Reflector: | |

## 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.
- Rewrite a while loop as a for loop (and vice versa).

## 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 to swap two variables. This understanding is essential for loop updates.

Optional warm-up activity: Bring a pair of note cards labeled with a large "1" on the first card 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 when they struggle, ask them "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 #16. 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.

Model 3 is a quick introduction to for loops. Be sure to point out that the update step happens after the loop body. If time permits, have students run their code on a computer to check their work. You could also ask teams to discuss which loop syntax they like better, and why.

Key questions: #7, #16, #21

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

# 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--;
x--;
```

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

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

```
y is -10
y is -9
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;
size = size - 3;
total = total * 2;
change = change / 10;
hours = hours % 24;
```

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

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

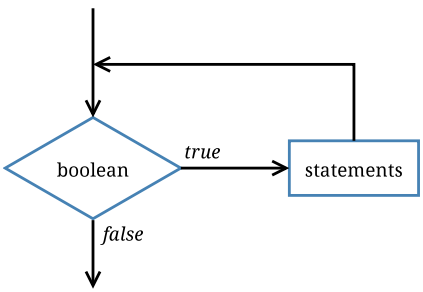
```
step = 5 + step;
size = 3 - size;
total = 2 * total;
change = 10 / change;
hours = 24 % hours;
```

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

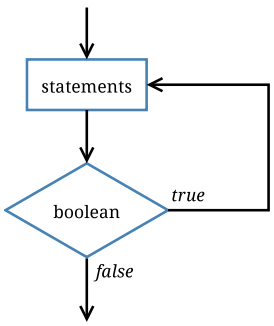
## Model 2 While Loops

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:**

**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? 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.

**12**. 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.

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

Both loops print the values 2 through 11 instead.

**14**. 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.

**15**. 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`.

**16.** 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.

**17.** 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;
```

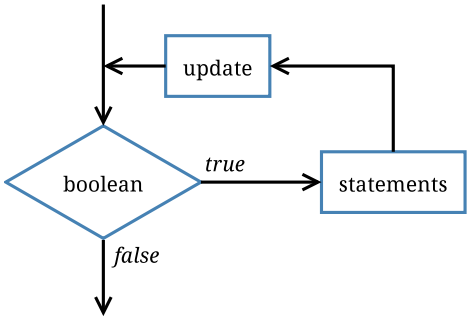
It will print `"i = 0"` forever. Without braces, the loop only executes the first statement, and `i = i + 1;` is never reached.

# 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:

**18.** 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--`

**19**. 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--;
}
```

**20**. 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.

**21**. 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)`

**22**. 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`.