ELIJAH KAYONGO

20/BCC/BU/R/0017

**Conditional statements** begin with the keyword if followed by parentheses. An expression is placed inside the parentheses, which is evaluated when the conditional statement is reached. The result of the evaluation is a boolean value. No evaluation occurred above. Instead, a boolean value was explicitly used in the conditional statement.

The parentheses are followed by a block, which is defined inside opening- { and closing } curly brackets. The source code inside the block is executed if the expression inside the parentheses evaluates to true.

An example to compare numbers in the conditional statement.

int number = 11;

if (number > 10) {

System.out.println("The number was greater than 10");

}

If the expression in the conditional statement evaluates to true, the execution of the program progresses to the block defined by the conditional statement.

NB! An if -statement is not followed by a semicolon since the statement doesn't end after the conditional.

A conditional statement also marks the start of a new code block.

An example of a correctly indented block.

if (number > 10) {

number = 9;

}

Program control structures **are programming blocks that can change the path we take through those instructions.**

**Examples of program controls**

1. Switch

If we have multiple cases to choose from, we can use a *switch*statement.

Let's again see a simple example:

**int** count = 3;

**switch** (count) {

**case** 0:

System.out.println("Count is equal to 0");

**break**;

**case** 1:

System.out.println("Count is equal to 1");

**break**;

**default**:

System.out.println("Count is either negative, or higher than 1");

**break**;

}

Three or more *if/else*statements can be hard to read. As one of the possible workarounds, we can use *switch,*as seen above.

## Loops

**We use**[loops](https://www.baeldung.com/java-loops)**when we need to repeat the same code multiple times in succession.**

Let's see a quick example of comparable for and while type of loops:

**for** (**int** i = 1; i <= 50; i++) {

methodToRepeat();

}

**int** whileCounter = 1;

**while** (whileCounter <= 50) {

methodToRepeat();

whileCounter++;

}

Both code blocks above will call methodToRepeat 50 times

## Break

**We need to use**[break](https://www.baeldung.com/java-continue-and-break)**to exit early from a loop.**

Let's see a quick example:

List<String> names = getNameList();

String name = "John Doe";

**int** index = 0;

**for** ( ; index < names.length; index++) {

**if** (names[index].equals(name)) {

**break**;

}

}

Here, we are looking for a name in a list of names, and we want to stop looking once we've found it.

A loop would normally go to completion, but we've used break here to short-circuit that and exit early.