Week 3: Computer Science 1

Control Structures

Program Flow

Even the simplest programs need to make decisions. **Control structures** allow us to control the flow of a program and determine which statements are executed and when.

Java uses **if statements** to control the flow of a program. Their are three versions, the simple **if statement**, the **if-else statement**, and the **else-if** statment.

Let's review the simple **if statement** first.

```
if (condition)
  // statement
```

We can read this as "if the condition is true, then execute the statement".

Recap of **Relational & Boolean Operators**

Operator	Description
==	equal to
!=	not equal to
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
!	not
&&	and
||	or

```
if(count <= 10)
  System.out.println("Count is less than or equal to 10");
```

When the statement is executed, the condition is evaluated. If the condition is true, the statements are executed. If the condition is false, the statements are skipped. So as long as count is less than or equal to 10, the statement will be executed.

```
if(count <= 10)
  System.out.println("Count is less than or equal to 10");
System.out.println("End");
```

In this example, the first statement is executed if count is less than or equal to 10. The second statement is executed regardless of the value of count. This is because the second statement is not part of the if statement. It is not indented and is not enclosed in curly braces.

If count is less than or equal to 10 the condition is true, the output will be:

```
Count is less than or equal to 10
End
```

If count is greater than 10, the condition is false, the output will be:

```
End
```

An **if-else statement** allows us to execute one set of statements if the condition is true and another set of statements if the condition is false.

```
if (condition)
  // statements
else
  // statements
```

```
if(count <= 10)
  System.out.println("Count is less than or equal to 10");
else
  System.out.println("Count is greater than 10");
System.out.println("End");
```

If count is less than or equal to 10 the condition is true, the output will be:

```
Count is less than or equal to 10
End
```

If count is greater than 10, the condition is false, the output will be:

```
Count is greater than 10
End
```

The **if** and **if-else** statements do not have to be single statements. They can be **compound statements** represented by a block of statements enclosed in curly braces.

```
if (condition) {
  // statements
  // ...
  // ...
}
```

```
if (count <= 10)
{
    System.out.println("incrementing count");
    count++;
}
else
{
    System.out.println("decrementing count");
    count--;
}
System.out.println("End");
```

If count is less than or equal to 10 the condition is true, the output will be:

```
incrementing count
End
```

If count is greater than 10, the condition is false, the output will be:

```
decrementing count
End
```

Notice the format of the **if-else** statement. The **if** and **else** are aligned. The **if** and **else** are indented. The **if** and **else** are enclosed in curly braces. The statements in the **if** and **else** are indented.

You can use braces to enclose a single statement, but it is not required. It is a good idea to use braces to enclose a single statement. It makes the code easier to read and less prone to errors.

The statements within **if** and **if-else** statements can be other **if** and **if-else** statements.

This is called **nesting**.

```
if (condition1)
  if (condition2)
    // statements
  else
    // statements
else
  // statements
```

```
if (count <= 10)
  if (count == 10)
    System.out.println("Count is equal to 10");
  else
    System.out.println("Count is less than 10");
else
  System.out.println("Count is greater than 10");
```

If count is less than or equal to 10 the first condition is true, the output will be:

```
Count is less than 10
```

If count is greater than 10, the first condition is false, the output will be:

```
Count is greater than 10
```

If count is equal to 10, the both conditions are true, the output will be:

```
Count is equal to 10
```

Can we write the previous example without nesting the **if** and **if-else** statements?

Along with if and if-else statements, Java has an **else-if** statement. It is used when you have more than two possible values for a variable.

```
if (condition1)
  // statements
else if (condition2)
  // statements
else
  // statements
```

```
if (condition1)
  // statements
else if (condition2)
  // statements
else
  // statements
```

With an else-if statement, the conditions are evaluated in order. If the first condition is true, the statements are executed and the rest of the else-if statement is skipped. If the first condition is false, the second condition is evaluated. If the second condition is true, the statements are executed and the rest of the else-if statement is skipped. If the second condition is false, the else statements are executed.

```
if (count == 10)
  System.out.println("Count is equal to 10");
else if (count < 10)
  System.out.println("Count is less than 10");
else
  System.out.println("Count is greater than 10");
```

We also can combine multiple conditions together with the **logical operators**.

Operator	Description
!	not
&&	and
||	or

```
if (condition1 && condition2)
  // statements
else if (condition1 || condition2)
  // statements
else if (!condition1)
  // statements
else
  // statements
```

```
if (count == 10 && count < 20)
  System.out.println("Count is equal to 10 and less than 20");
else if (count == 10 || count < 20)
  System.out.println("Count is equal to 10 or less than 20");
else if (!(count == 10))
  System.out.println("Count is not equal to 10");
else
  System.out.println("Count is greater than 20");
```

A switch statement is another way to control the flow of a program. It is used when you have a limited number of possible values for a variable. The switch statement is similar to a series of if-else statements. I'm going to include it here, but concentrate on the if-else statements for now.

```
switch (variable) {
  case value1:
    // statements
    break;
  case value2:
    // statements
    break;
  case value3:
    // statements
    break;
  default:
    // statements
}
```

If we wanted to use a switch statement to convert a character to a day of the week, we could use the following code.

```
char date = 'M';
switch (date) {
  case 'M':
    System.out.println("Monday");
    break;
  case 'T':
    System.out.println("Tuesday");
    break;
  case 'W':
    System.out.println("Wednesday");
    break;
  case 'R':
    System.out.println("Thursday");
    break;
  case 'F':
    System.out.println("Friday");
    break;
  default:
    System.out.println("Invalid day");
}
```

Exercises

Create a program that checks to see if a number is positive or negative. It should print either "true" or "false" depending on the number. Format your print statement like this:

```
10 is positive: true
```

Think about what you need to do to determine if a number is positive or negative. What math operation do you need to use? What is the condition? What is the statement? What math operation do you need to use?

```java
int num = 10;
boolean isPositive;

if (num >= 0)
  isPositive = true;
else
  isPositive = false;

System.out.println(num + " is positive: " + isPositive);
```

Create a program that checks to see if a number is even or odd. It should print either "true" or "false" depending on the number. Format your print statement like this:

```
10 is even: true
```

Think about what you need to do to determine if a number is even or odd. What math operation do you need to use? What is the condition? What is the statement? What math operation do you need to use?

```java
int num = 10;
boolean isEven;

if (num % 2 == 0)
  isEven = true;
else
  isEven = false;

System.out.println(num + " is even: " + isEven);
```

Create a program that checks if a number is between -1 and 5 inclusive.

Mathematically, we can write this as:

$-1 \leq x \leq 5$

There are multiple ways to write this program. Explore more than one way to write it.

```
int num = 10;
boolean isBetween;

if (num >= -1 && num <= 5)
  isBetween = true;
else
  isBetween = false;

System.out.println(num + " is between -1 and 5 inclusive: " + isBetween);
```

```
int num = 10;
boolean isBetween;

if (num >= -1)
  if (num <= 5)
    isBetween = true;
  else
    isBetween = false;
else
  isBetween = false;

System.out.println(num + " is between -1 and 5 inclusive: " + isBetween);
```

The following table shows grade letters and their corresponding grade points. Write a program that converts grade points to a grade letter.

Grade Letter	Grade Points
A	100-90
B	89-80
C	79-70
D	69-60
F	59-0

```
int grade = 85;
char gradeLetter;

if (grade >= 90)
  gradeLetter = 'A';
else if (grade >= 80)
  gradeLetter = 'B';
else if (grade >= 70)
  gradeLetter = 'C';
else if (grade >= 60)
  gradeLetter = 'D';
else
  gradeLetter = 'F';
```

Create a program that computes the BMI of a person from who is 70 inches tall and 150 pounds. The formula for BMI is:

$$BMI = \frac{weight \times 703}{height^2}$$

The program should print the BMI and the category the person falls into. The categories are:

Category	BMI Range
Low	0-18.5
Normal	18.5-24.9
High	25-29.9
Obese	30-39.9
Morbid	40+

```
int height = 70;
int weight = 150;
double bmi;
String category;

bmi = weight * 703 / (height * height);

if (bmi < 18.5)
  category = "Low";
else if (bmi < 25)
  category = "Normal";
else if (bmi < 30)
  category = "High";
else if (bmi < 40)
  category = "Obese";
else
  category = "Morbid";
```