Session Goals

- User should be able to write conditional statements in Java, focusing on their correct/incorrect syntax.

- User should be able to use relational operators inside conditional statements easily.

- User should understand the meaning/importance of syntax (curly braces, semicolons etc).

- User should be able to debug syntax errors on their own.

- User should be able to define functions with correct parameter types and return types.

- User should be able to call functions and store values returned from it in variables.

- User should be able to trace the flow of a function call/invocation.

Session 2

Conditionals and Methods in Java

Session Agenda

- Relational Operators

- Conditional Statements

- Functions (Definition, Invocation)

- Passing parameters to a function

- Returning values from a function

- Introduction to Crio Java Platform

Comparison or Relational Operators

Operators	Meaning	Example	Result
<	Less than	5<2	False
>	Greater than	5>2	True
<=	Less than or equal to	5<=2	False
>=	Greater than or equal to	5>=2	True
==	Equal to	5==2	False
! =	Not equal to	5! =2	True

"==" not to be mistaken with **"="**

Equal to

Variable assignment

The If-else statement

The if statement is used to specify a block of code to be executed if a condition is true.

The else statement to specify a block of code to be executed if the condition is false.

Syntax:

```
if (condition) {
  //  block of code to be executed if the condition is true
} else {
  //  block of code to be executed if the condition is false
}
```

Example:
```
if (hour < 18) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

Dry Run of If-Else Statement

Dry Run the code snippet below and guess the output -

```java
int  n = 29;
if (n%2 == 0) {
   System.out. println(n + " is an even Number");
} else {
   System.out. println(n + " is an odd Number");
}
```

Recap - 6 Step Strategy

1. **Understand the problem** *(ask questions and get clarity)*

2. **Design test data/test cases** *(input and expected output)*

3. **Derive the solution - solve the problem** *(write pseudo code)*

4. **Test the solution** *(against the test data/case - dry run)*

5. **Write the program/code** *(using Java here)*

6. **Test the code** *(syntax errors, run time errors, logical errors)*

Activity: Find largest of two numbers

Write a java program to find the largest of takes two numbers a and b and parameters and return the largest one.

Link (Focus on writing the code and using "Run Code" to get output, there is no assessment yet!)

What will be your approach to the problem? (Step 3)

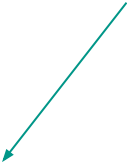*Quickly put your answers in the chat!*

Methods in Java

- In Java, a method is a block of code designed to perform a specific task. It's similar to a function in other programming languages.

main is a method here

```java
public class HelloWorld {
    public static void main ( String[] args ) {
        System.out.println ( "Hello World" );
    }
}
```

Methods Syntax in Java

Access Specifier (public): Determines who can access the method. public means any class can call the method.

Modifier (static): Specifies additional information about the method. static means it belongs to the class, not instances of the class.

Return-Type (void): Indicates what type of value the method will return. void means it returns no value.

Function Name (myFunc): The identifier that is used to call the method. Should be descriptive of what the method does.

Parameter List (int a, float b): The input the method accepts, with types and variable names specified. Parameters are optional; a method can have none.

Function Body: The block of code that defines what the method does. It contains all the logic and statements.
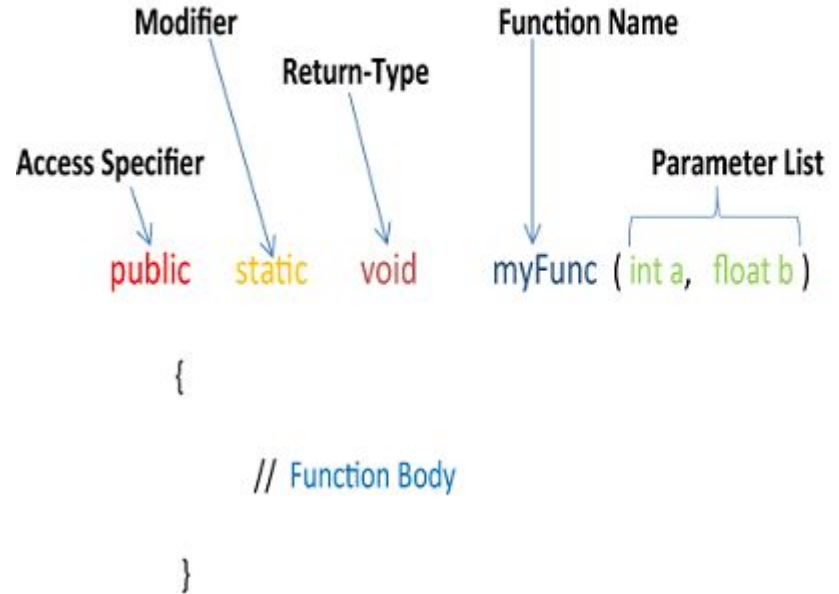
Note: The void keyword specifies that a method should not have a return value.

Function Definition and Invocation in Java

```java
public class Solution {
    // Method Definition
    public static void greet() {
        System.out.println("Hello There");
    }

    public static void main(String[] args) {
        // Calling a method/Invocation
        greet();
    }
}
```

- A method in Java is defined within a **class** and **begins with an access modifier**, followed by the **return type**, the **method name**, and a **pair of parentheses** which can contain **parameters**.

- The **body of function** is written within **{}**

Activity: Printing name on console

Write a function which takes firstName and lastName as parameters and returns fullName and print it on console.

Link (Clear your code from last exercise)

What will be your approach to the problem? (Step 3)

Quickly put your answers in the chat!

5 minute break

Java return keyword

```java
public class Main {
  static int myMethod(int x) {
    return 5 + x;
  }

  public static void main(String[] args) {
    System.out.println(myMethod(3));
  }
}
// Outputs 8 (5 + 3)
```

The `return` keyword finishes the execution of a method, and can be used to return a value from a method.

Activity: Square of a Number

Write a function to return the square of a number and print it on console.

Link (Clear your code from last exercise)

What will be your approach to the problem? (Step 3)

Quickly put your answers in the chat!

Introduction to Crio IDE to write Java code with Assessment

Write a java program to find sum of two numbers.

[Link](#)

What will be your approach to the problem? (Step 3)

Quickly put your answers in the chat!

Debrief: Methods/Functions

- A **method/function** is a **sub-program (block of code to perform a specific task)** that can be **executed as required** and **any number of times** within a larger program.

- The set of instructions that a function executes is called the **Function Definition.**

- Once you write a function, we can call that function into action by using its name - **Function Call** or **Function Invocation**.

- Inputs passed to the Function are called **Function Parameters**.

- Functions can be **parameterized** or **non-parameterized** functions.

One of the main purpose of function is **code Reusability.**

Summary

- A method in Java is defined within a class and begins with an access modifier, followed by the return type, the method name, and a pair of parentheses which can contain parameters.
- Methods can receive input data through parameters when called.
- Methods can provide output values using the return statement.
- The void keyword specifies that a method should not have a return value.

Quiz Time!

Question 1

Which block is executed if the condition provided is true?

- if block
- else block

Question 2

Parameters in a method are mandatory. True or False?

- True
- False

Question 3

The ___ keyword finishes the execution of a method, and can be used to yield a value from a method

- break;
- continue;
- return;
- yield;

Question 4

Which modifier is used to specify that the method will not yield a value at the end of execution?

- void
- static
- final
- abstract

Thank you!