Recap of Java Programming Language

Variables, selection, iteration and more.

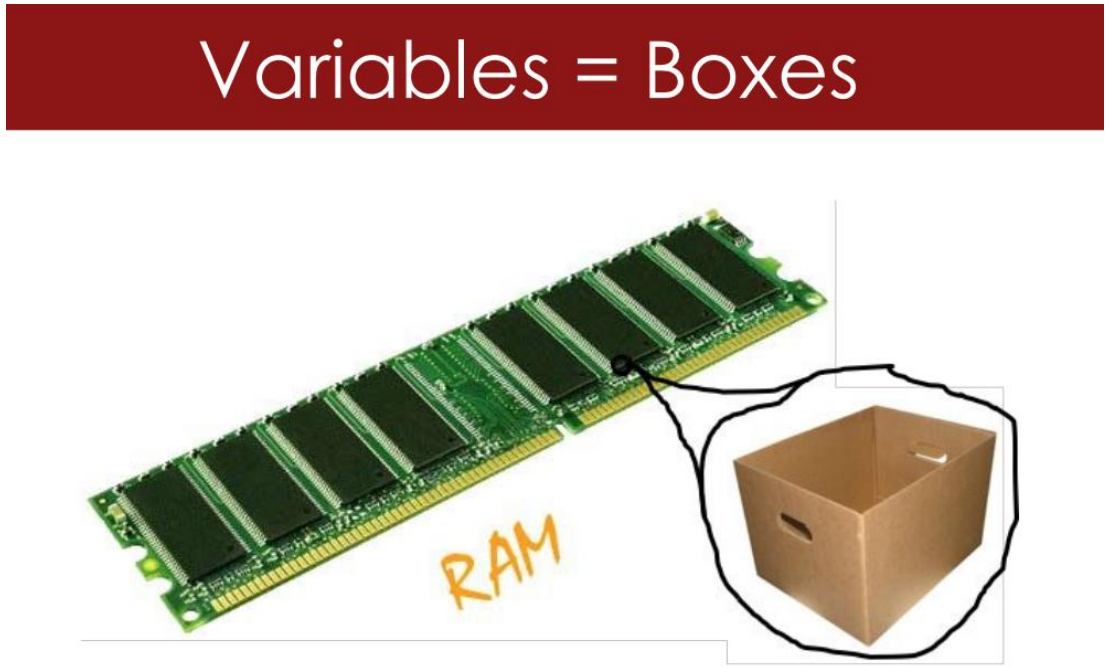
Produced by: Ms. Mairead Meagher
Ms Siobhan Roche

Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Department of Computing and Mathematics
http://www.setu.ie/

Variables

Variables are containers for storing data in your program. They have a type (e.g., int, String) and can be used to represent various types of information.



Primitive Data Types

1.byte:
- Size: 8 bits
- Range: -128 to 127

2.short:
- Size: 16 bits
- Range: -32,768 to 32,767

3.int:
- Size: 32 bits
- Range: -2^31 to 2^31 - 1

4.long:
- Size: 64 bits
- Range: -2^63 to 2^63 – 1

5.float:
- Size: 32 bits
- Range: Approximately ±3.4e^38 (7 significant digits)

6.double:
- Size: 64 bits
- Range: Approximately ±1.8e^308 (15 significant digits)

7.char:
- Size: 16 bits
- Represents a single Unicode character

8.boolean:
- Represents true or false

These primitive data types are the building blocks for storing numeric and boolean values in Java. It's important to choose the appropriate data type based on the range and precision needed for a particular variable.

Code Example

```java
// byte
byte myByte;  //declare variable
myByte= 42;  //initialise/assign it with a value

//declare and assign in one statement
short myShort = 1000; // short
int myInt = 123456; // int
long myLong = 123456789L; // long
float myFloat = 3.14f; // float
double myDouble = 3.141592653589793; // double
char myChar = 'A'; // char
boolean isJavaFun = true;  // boolean
// Print values
System.out.println("byte: " + myByte);
System.out.println("short: " + myShort);
System.out.println("int: " + myInt);
System.out.println("long: " + myLong);
System.out.println("float: " + myFloat);
System.out.println("double: " + myDouble);
System.out.println("char: " + myChar);
System.out.println("boolean: " + isJavaFun);
```

Variable Exercise

- Declare an integer variable named **myAge** and initialise it with your age.
- Declare a String variable named **myName** and initialise it with your name.
- Print both variables to the console.

Variable Solution

```java
public class VariableExercise {
    public static void main(String[] args) {
        // Declare and initialise integer variable
        int myAge = 25;

        // Declare and initialise String variable
        String myName = "Alex";

        // Print variables
        System.out.println("Age: " + myAge);
        System.out.println("Name: " + myName);
    }
}
```

Selection (if statement)

- The 'if' statement allows us to make decisions in our code based on conditions. If a condition is true, one set of statements executes; otherwise, another set executes.

```
int x = 10;

// Simple if statement
if (x > 0) {
    System.out.println("x is positive");
} else {
    System.out.println("x is non-positive");
}
```

Exercise – if statement

Write a Java program that takes a user's age as input and prints "You are a minor" if the age is less than 18; otherwise, print "You are an adult."

```
Enter your age: 12
You are a minor.
```

```
Enter your age: 34
You are an adult.
```

```
Enter your age: 18
You are an adult.
```

Solution – if statement

```java
public class AgeCheck {
    public static void main(String[] args) {
        //use scanner to take user input from console
        Scanner scanner = new Scanner(System.in);
        // Ask user for age
        System.out.print("Enter your age: ");
        //Take user input for age
        int age = scanner.nextInt();
        // Check if the user is a minor or an adult
        if (age < 18) {
            System.out.println("You are a minor.");
        } else {
            System.out.println("You are an adult.");
        }
    }
}
```

Iteration in Java

- Loops are essential for repeating actions in your code.

- We learnt about :
  - for loop
  - while loop
  - do-while loop.

For loop

```java
// For Loop
System.out.println("For Loop:");
for (int i = 2; i <= 10; i += 2) {
    System.out.println(i);
}
```

While Loop

```java
// While Loop
System.out.println("\nWhile Loop:");
int j = 2;
while (j <= 10) {
    System.out.println(j);
    j += 2;
}
```

Do - While Loop

```java
// Do-While Loop
System.out.println("\nDo-While Loop:");
int k = 2;
do {
    System.out.println(k);
    k += 2;
} while (k <= 10);
```

Exercise - Loop

Use a loop to print the numbers from 1 to 5 (inclusive) to the console.

```
Numbers 1 to 5
1
2
3
4
5
```

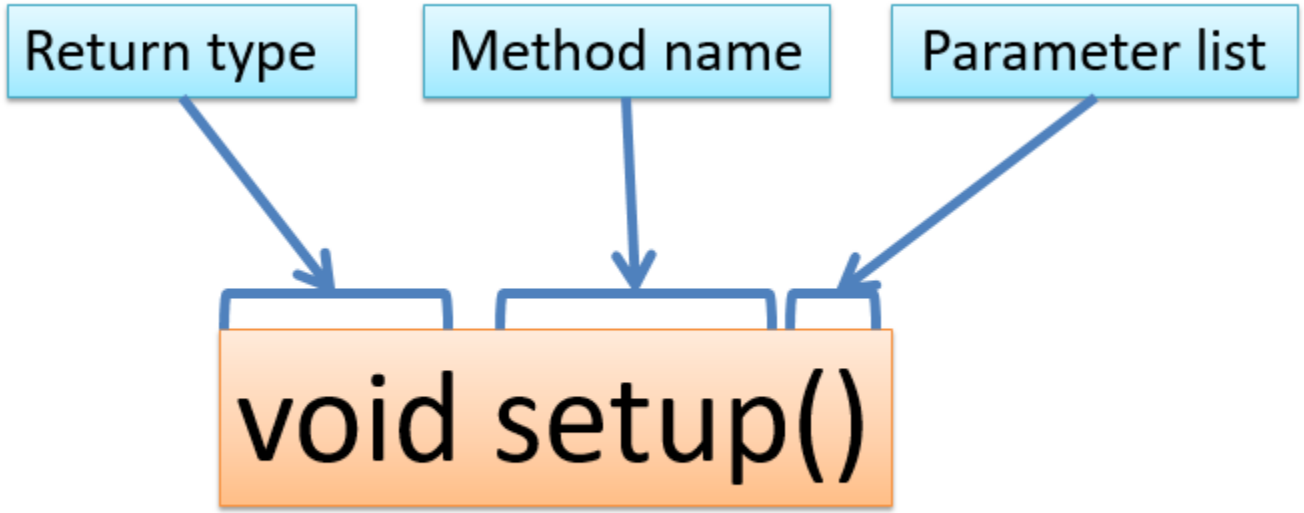
Solution – loop

Using **for** loop

```
System.out.println("Numbers 1 to 5");
for (int i=1; i<=5;i++){
    System.out.println(i);
}
```

Functions in Java (Methods)

- Functions, or methods in Java, are blocks of code that perform a specific task. They promote code modularity and reusability.



Method Definition and use

```
// Method definition
1 usage
int add(int a, int b) {
    return a + b;
}


// Method invocation
int result = add( a: 5,  b: 3);
```

Exercise - Methods

Create a method called **calculateSquare** that takes an integer parameter and returns its square. Use this method to calculate and print the squares of the numbers 1 to 5.

```
Square of 1 is: 1
Square of 2 is: 4
Square of 3 is: 9
Square of 4 is: 16
Square of 5 is: 25
```

Solution - Methods

```java
public class SquareCalculator {
    public static void main(String[] args) {
        // Calculate and print squares of numbers 1 to 5
        for (int i = 1; i <= 5; i++) {
            int square = calculateSquare(i);
            System.out.println("Square of " + i + " is: " + square);
        }
    }

    // Method to calculate square
    1 usage
    static int calculateSquare(int number) {
        return number * number;
    }
}
```

Understanding the 'static' Keyword

The 'static' keyword is used in Java for class-level elements. Static variables are shared among all instances of a class, and static methods belong to the class rather than an instance.

The Main Method in Java

```java
public class MainMethodExample {
    public static void main(String[] args) {
        // Code inside the main method
        System.out.println("Hello, Java!");
    }
}
```

public: The main method must be declared as public so that it can be accessed by the Java runtime environment when starting the program.

static: The main method is declared as static to allow it to be called without creating an instance of the class. This is necessary because the Java runtime environment invokes the main method directly.

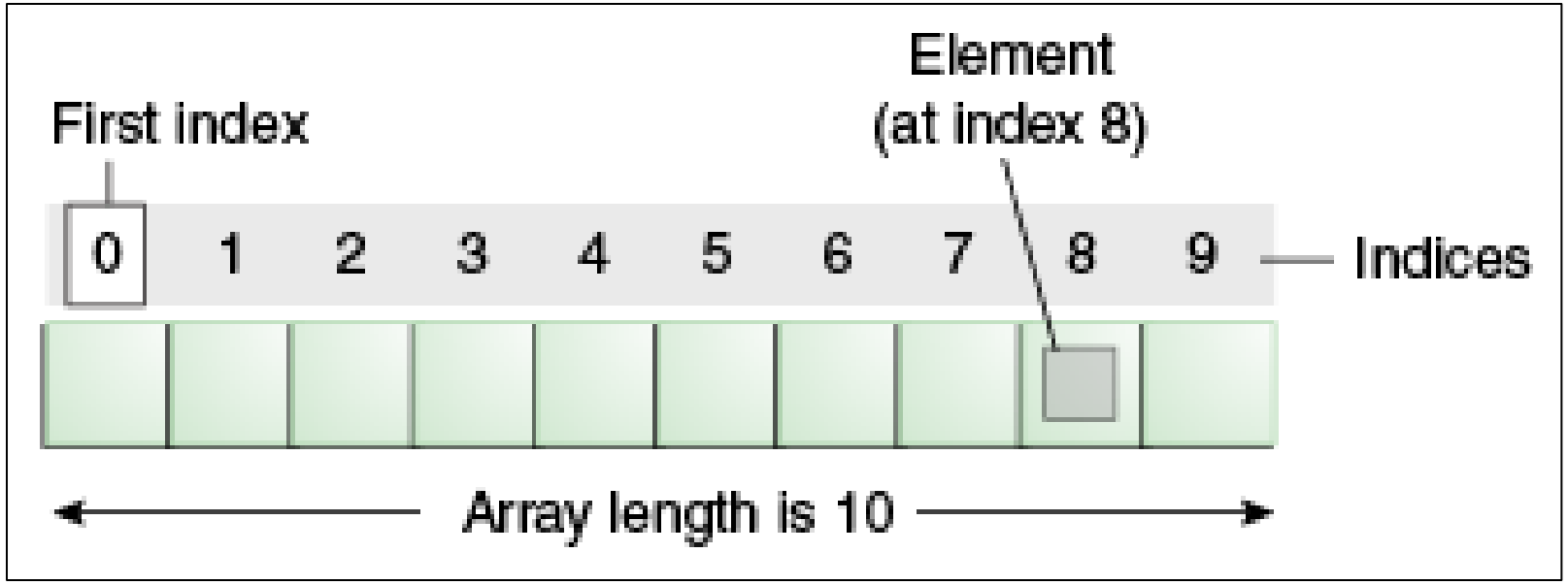
void: The main method does not return any value (returns void). It is meant to execute a series of statements and does not produce a result.

main: The name of the method. It serves as the entry point for the Java program.

(String[] args): The method parameter. args is an array of strings that can be used to pass command-line arguments when running the Java program.

Arrays

- Arrays allow us to store multiple values of the same type under a single name.



Arrays Example

```
//int[] numbers = {3, 7, 2, 8, 5};
int[] numbers = new int[5];
numbers[0]= 3;
numbers[1]= 7;
numbers[2]= 2;
numbers[3]= 8;
numbers[4]= 5;
int sum = 0;
for (int i = 0; i < 5; i++)  {
    sum += numbers[i];
}
System.out.println("Sum of numbers: " + sum);
```

Exercise - Array

- Declare an array of integers named numbers with values 10, 5, 12, 18, and 15. Write a loop to print each of these numbers

```
Number 1 : 10
Number 2 : 5
Number 3 : 12
Number 4 : 18
Number 5 : 15
```

Solution - Array

```
int[] numbers = {10, 5, 12, 18, 15};

int sum = 0;
for (int i = 0; i < 5; i++)  {
    System.out.println("Number "+ (i+1) + " : " + numbers[i]);
}
```

Summary

```java
public class SummaryExercise {
    public static void main(String[] args) {
        // Call the average method with an array of integers
        int[] values = {10, 20, 30, 40, 50};
        double averageValue = calculateAverage(values);

        // Print the result
        System.out.println("Average value: " + averageValue);
    }

    // Method to calculate the average of an array of integers
    1 usage
    static double calculateAverage(int[] theArray) {
        int sum = 0;
        for (int value : theArray) {
            sum += value;
        }
        return (double) sum / theArray.length;
    }
}
```

Questions?