Topic 1.2: Methods

Learning Goals (Week 1):

- Identify data types based on value
- Map variables to the current values
- Perform basic operations on variables
- **Create and use Java and user-defined methods**
- Format Printed Output
- Obtain and process user input from the console
- Use booleans, conditionals, and compound conditionals correctly
- Select and implement different types of loops depending on scenario
- Use special String and Math operations
- Successfully implement and manipulate java arrays

Method Parts

- Methods are comprised of a signature and a body
- Signature
  - method header: includes the modifier, return type, name of method and parameters
- Body
  - collection of statements/commands called when method is performed

```
public class MethodTest {
     public static void main(String[] args) {
          System.out.println(“Hello, World!”);
    }
}
```

Methods are always placed within a class.

Methods we’ve already seen

- main(String[] args)
  - main method with args array as the parameter
  - implicitly called when we run our program (java MainClass)
- System.out.println(String)
  - String parameter which prints out the value to console
  - Ignore the System.out. part for now, the println() is the method call

- If you see round brackets you are either defining a method or calling it

Writing & Calling our own methods

E.g 1: No return value

```
public class MethodTest {
     public static void main(String[] args) {
          System.out.println(“In Main”);
          String message = “Hello, World!”;
          printMessage(message);
    }
    public static void printMessage(String msg) {
         System.out.println(“In printMessage”);
         System.out.println(msg);
    }
}
```

Output:

```
In Main
In printMessage
Hello, World!
```

E.g 2: With return value

```
public class MethodTest {
     public static void main(String[] args) {
          String message = “Hello, World!”;
          System.out.println(“Main: ” + message);
          message = changeMessage(message);
          System.out.println(“After method call: ” + message);
    }
    public static String changeMessage(String msg) {
         msg = msg + “ changed in method”;
         return msg;
    }
}
```

Output:

```
Main: Hello, World!
After method call: Hello, World! changed in method
```

Why not put all my statements/commands in main()?

- You ***COULD,*** but ***SHOULD*** you?
  - NO!
- main() is like a CEO of a company; it shouldn’t do low-level work
  - main() delegates work to lower-level workers who may delegate work further
- Ask yourself what your code is doing.
  - Each behaviour should have a function, with managers and middle managers calling the shots.
- One more indicator: duplication
  - If you find yourself copying/pasting code, you probably need a method.

Why not put all my statements/commands in main()?

Make a program that takes in two positive temperature values in Celsius. It should convert the temperatures to Fahrenheit figure out the average temperature and then print that value.

NOT GREAT

```
public class TemperatureChecker {
     public static void main(String[] args) {
          double celsius1 = ?;
          double celsius2 = ?;
          double farenheit1 = (celsius1 * (9.0/5.0)) + 32;
          double farenheit2 = (celsius2 * (9.0/5.0)) + 32;
          double average = (farentheit1 + farenheit2) / 2;
          System.out.println(celsius1 + “C converts to ” + farenheit1 + “F”);
          System.out.println(celsius2 + “C converts to ” + farenheit2 + “F”);
          System.out.println(“The average of the two temperatures is ” + average);
    }

}
```

Why not put all my statements/commands in main()?

Make a program that takes in two positive temperature values in Celsius. It should convert the temperatures to Fahrenheit figure out the average temperature and then print that value.

BETTER

```
public class TemperatureChecker {
     public static void main(String[] args) {
          double celsius1 = ?;
          double celsius2 = ?;
          double farenheit1 = convertToF(celsius1);
          double farenheit2 = convertToF(celsius2);
          double average = getAverage(farenheit1, farenheit2);
          printConvert(celsius1, farentheit1);
          printConvert(celsius2, farentheit2);
          System.out.println(“The average of the two ” +
               “temperatures is ” + average);
    }
```

```
    public static double convertToF(double c) {
       return (c * (9.0/5.0)) + 32;
    }

    public static double getAverage(double f1, double f2) {
        return (f1 + f2) / 2;
    }

    public static void printConvert(double c, double f) {
       System.out.println(c + “C converts to ” + f + “F”);
    }
}
```

Why not put all my statements/commands in main()?

BEST

```
public class TemperatureChecker {
    public static void main(String[] args) {
        double celsius1 = ?;
        double celsius2 = ?;
        double farenheit1 = convertToF(celsius1);
        double farenheit2 = convertToF(celsius2);
        printFarenheits(farneheit1, farenheit2);
        printAverage(farenheit1, farenheit2);
    }
    public static void printFarenheits(double f1, double f1) {
        printConvert(c1, f1);
        printConvert(c2, f2);
    }
```

```
    public void print printAverage(double f1, double f2) {
        double avg = getAverage(f1, f2);
        System.out.println(“The average of the two ” +
            “temperatures is ” + avg);
    }

    public static double convertToF(double c) {
      return (c * (9.0/5.0)) + 32;
    }

    public static double getAverage(double f1, double f2) {
       return (f1 + f2) / 2;
    }

    public static void printConvert(double c, double f) {
      System.out.println(c + “C converts to ” + f + “F”);
    }
}
```

Pause and Practice (see BasicCalculator.java for solution)

Create a basic calculator that can perform the four fundamental arithmetic operations: addition, subtraction, multiplication, and division. Focus solely on writing methods with appropriate return types.

You can have two int variables in main with the values 8 and 17 (which you could change and re-run your program as you see fit).

The output should clearly display the operation being performed and the result. Here's how you can structure the output for each arithmetic operation:
(Example: 8 + 17 = 25)

- Operation: Addition (**+**)
- Output Format: "**firstNumber + secondNumber = result**"
- Example: If **firstNumber** is 8 and **secondNumber** is 17, the output should be "8 + 17 = 25"