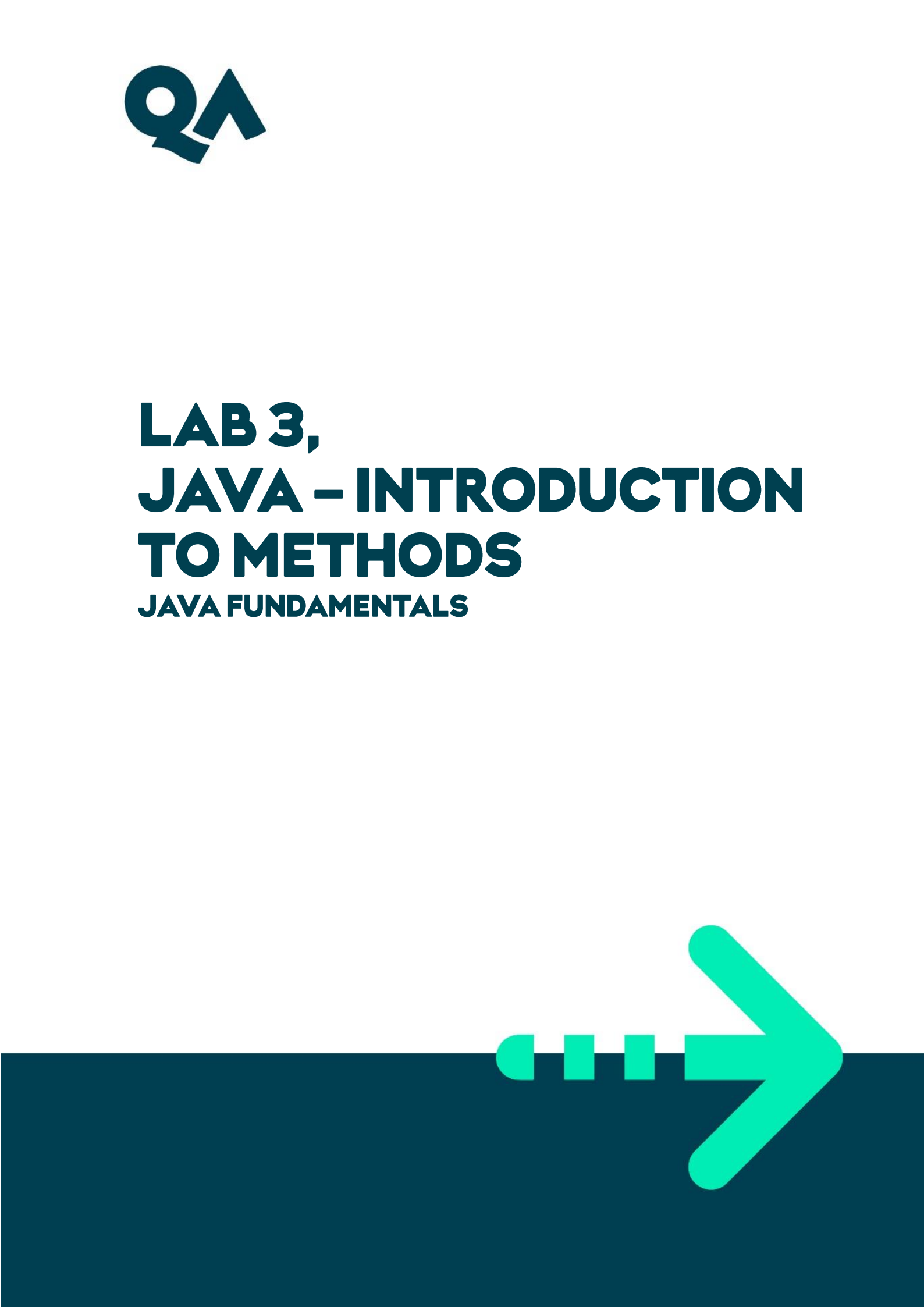

LAB 3, JAVA – INTRODUCTION TO METHODS

JAVA FUNDAMENTALS

Lab 3, Java – Introduction to methods

Objective

The objectives of this practical session are as follows.

- To be able to write and invoke methods that have a varying number of parameters, some of which return a value.
- To accept user input in response to a prompt and process that data further including converting it to a different type of data.
- You'll also create and use a new class

Part 1 – Authoring a helper method

Step by step.

1. Back in the labs project which you created in Lab1, add a new package called lab03.

   Please refer to lab1's instructions if you need help.

2. Add a new class called Program to the *lab03* package with a *main()* method.

3. Add a new method as public static int getInt(String prompt)

   This method has a String parameter called *prompt*, which it displays before getting an integer input from the user. It then returns an int.

   To get keyboard input (*System.in* stream), you'll use the Scanner object as:

   ```
   Scanner s = new Scanner(System.in);

   return s.nextInt();
   ```

   The Scanner class has to be resolved. Click on the word Scanner and press Ctrl-1 and choose *import Scanner*.

4. Create another method called String getString(String prompt)

   This method is similar to the getInt() method except you should change the s.nextInt to **`s.nextLine();`**

5. Call both methods in the main() method and then print the result to test your code.

Part 2 – Performing data conversions

The scenario is going to mimic a serving line at a lunch hall in that we are going to prompt the user to answer certain questions. What would you like as a main dish? Then how many Roast Potatoes? How many Brussel Sprouts? Then display what their lunch is.

Step by step.

1. Create a method called **`theLunchQueue`**. In the Program class.
2. Call the getString() method to display the following
   `What main dish would you like(Fish, Burgers or veg) ?`

   And get the answer into a variable called mainCourse.
3. Use the getInt() method to display the following prompts and capture the values in suitable variable names.

   ```
   How many roast potatos would you like?
   How many Brussel Sprouts would you like?
   ```

   Display the description for producing a bill. Something like:
   Hello, your lunch is xx with yy roast potatoes and zz Brussel sprouts.

   Replacing *xx*, *yy* and *zz* with your actual values of course!
4. Test your code by calling **`theLunchQueue() method`** from main().

Part 3 - Weight Conversions

1. Create a method as
   void **`convertInputToStonesPounds`**(int pounds).

   This method should

   a. Ask the user for a total weight in pounds in main() and pass the result to this method.

   b. Display the result (stones & pounds) in this method.

      Note: there are 14 pounds in a stone.
      Hint: Use division (/) and modulus (%)
2. Create another method as
   `void convertKgsToStonesPounds`(int kg).

   a. Ask the user for a weight in kilograms.

   b. Convert the weight and display it in stones and pounds

      Hint: 1 kilo = 2.20462 pounds
      Tip: convert the Kg to pounds and then call
      `convertInputToStonesPounds`(int kg)
3. Test your code at each stage

Part 4 – Move your code to a separate class

Does every method have to be in the Program class?
In this part you'll create a new class and move all the code to that class.

1. Create a new Class called Lab3Exercises without a main() method in the lab3 package.
2. Cut all the code outside of the main() method (Program class) and paste them inside the *Lab3Exercise* class.
3. Remove the static word from every method definition.
   We'll discuss static method at a later date. The only reason why every method was static was because main() is a static method() but we are now free of main()!
4. Back in the main() method, create an instance of *Lab3Exrcises* class and use it to call the methods.

   ```
   Lab3Exercise myLab3 = new Lab3Exercise();
   ```

5. At the start of each method call (in main) add "myLab3"
   For example:
   instead of **`theLunchQueue()`** `type` `myLab3.theLunchQueue()`
6. Run the application to make sure everything works.

** End **