**Institute of Technology Tralee**

**Computing Department**

**Object Oriented Programming 1**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Practical 7 – User-defined Methods**

We will continue with user-defined methods in this lab session. Hopefully at this stage you have got to grips with their syntax. The exercises in this lab sheet will combine user-defined methods with many of the control structures you have covered. Some of them are tricky so take your time working them out and ask for help if you get bogged down. Hopefully it will serve as good revision for the upcoming assessment.

**Organising your Work**

Within OOP1Stuff create a folder called **Lab7** to save your work from this lab session.

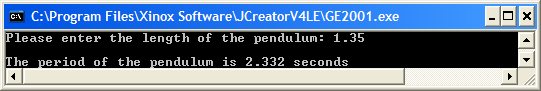
**Exercise 1**

Write a program that first of all reads in the length,, of a pendulum and then calls a user-defined method **periodCalculator**() that calculates the period of a pendulum, T. This is determined using the formula:

where g is the acceleration due to gravity which, on Earth, is 9.8 m/s/s and π is the number PI which you should try to access from the **Math** class. You can define as a **constant** in your program, **within the periodCalculator**() method.

You will also need to use the **sqrt**() method from the Math class for this calculation. Note that, like PI, it is defined to be static.

The **periodCalculator**() method takes a single double argument representing the length of the pendulum and returns the period calculated to main(). The main() then displays the period to **3 decimal places**. Your program would run as follows:

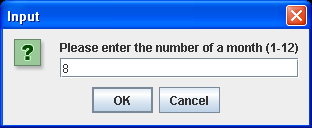
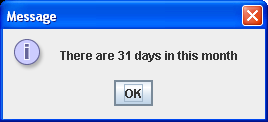


**Exercise 2**

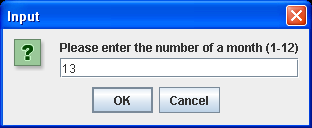
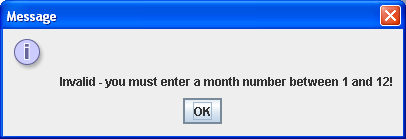
Write a program called **Exercise2.java** that takes in the month number (a whole number in the range 1-12) and then calls a user-defined method **numberDaysIn**() which takes as an argument the month number and returns the number of days in that month to main(). The main() then displays this value.

The method should use a **switch** statement for its decision making. It should have a **default** case that returns the value 0 – this will be used within main() to indicate that an **invalid month** has been entered, when the method is called. The program should operate as indicated in the following sample runs and note that **no looping process** is required in your program.

Run 1:

Run 2:

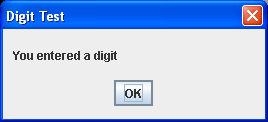
 ****

**Exercise 3**

You have seen many pre-defined methods at this stage e.g. **isDigit**() from the Character class, which deals with testing to see if the character it receives as an argument is a digit and returning true if it is, false otherwise. Write the code for this pre-defined method yourself from first principles.

You should call the method **myIsDigit**() and, like isDigit() it should return a **boolean** value of true or false depending on the outcome of the test. Obviously, you must also write a main() in order to test out the method. The prompt for the keyboard character will take place within main(), as will the displaying of the result on the message dialog, through the call to myIsDigit(). You can call the program **Exercise3.java** and it should run as indicated in the following sample runs – note that your program requires **no looping process**.

Run 1:

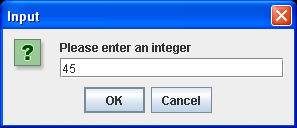
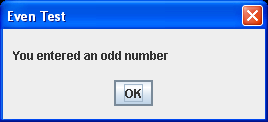
 

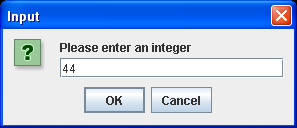
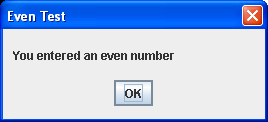
Run 2:

** **

**Exercise 4**

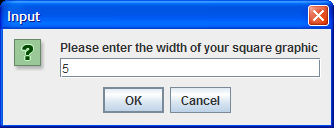
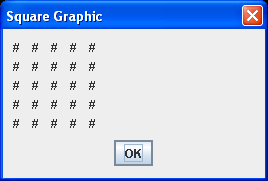
Write a program called **Exercise4.java** that prompts for and reads in a series of integers of unknown size and passes them one at a time to a user-defined method called **isEven**() which determines if the integer is even. The method should take an integer argument and return true if the integer is even, false otherwise. The outcome should then be displayed within main(). The loop in this program should stop when the user hits return when asked for a number. It should run as indicated in the following sample screenshots:

** **

** **

**Exercise 5**

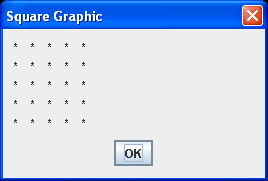
Write a program called **Exercise5.java** containing a method **squareGraphic()** that will display a solid square of hashes whose side is supplied by the user in main(). The method will take a single integer argument and return nothing to main(). Try to use **for loops** when coding your method here. It should run as indicated below:

**Note**: put **4 blank spaces** between each # character when displaying to ensure your output resembles a square rather than a rectangle.

**Exercise 6**

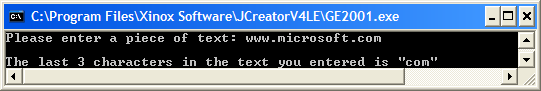
Save the previous program as **Exercise6.java** and modify it so that the program now gives the user the chance to decide what the graphics fill character will be. Use a String variable for reading in this extra information in main(). Your method will also require modification so that it now takes in this String information as a second argument when called. It will also need some internal changes. Thus, if the side value entered is 5 and the fill characteris "\*" then the method should display

****

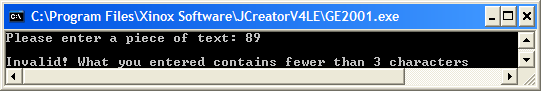
**Exercise 7**

Write a program called **Exercise7.java** that first of all asks the user to enter a piece of text in main(). Then a method **last3Characters()** should be called which takes the user-supplied text as an argument and returns the last 3 characters it contains. The result will then be displayed in main(). If it turns out that the user entered text that contains fewer than 3 characters, then the method should return the empty string “” to indicate this abnormality and main() should then display an appropriate message in this case. Your program should run as indicated below:

Run 1:



Run 2:



**Exercise 8**

Write a program called **Exercise8.java** that first of all asks the user how many times they wish to toss a coin. The program should then use a **for loop** to iterate this number of times. Each time the loop iterates, a user-defined method called **coinToss**() should be called which simulates the tossing of a coin. In this case the method will work by simply generating a **random number** of either 0 or 1 and returning this to main() when the method is called. We will take it here that 0 represents “heads” and 1 represents “tails”. Then the for loop will keep track of the number of heads or tails and display the outcome for each toss. Finally, once the loop stops, the main() will display the percentage of heads and the percentage of tails to the **nearest whole number**. Your program will run as indicated below:

