# ASSESSED Lab 6

# User-defined methods

## Assessment information

**Congratulations! There is only one assessed lab sheet left!**

**This worksheet *is* one of the seven assessed lab sheets.**

**It can be assessed within the next *5* weeks. Let me know *in advance* when you’d like to be assessed.**

**Do not forget to have it ‘signed off’ after you have been assessed.**

## 1 introduction

This laboratory worksheet covers the use of user-defined methods and classes within the Java programming environment. This laboratory involves the creation of a number of Java programs. Make sure that you save any code you write. Also make sure you save any results or notes that you observe about your work.

## 2 Preliminaries

For all of the following exercises you **must** **implement** each program and **test** each program. You will need more time than just one laboratory session to complete this worksheet.

This worksheet is marked slightly differently to the other worksheets. To **pass** the worksheet you must complete all of the non-recursive methods (section 4) and the Finch Robot method (section 5).

Unless stated in the text of the exercises, your solutions should have **no side effects** (that is, they should have **return** statements).

## 3 Writing User Defined methods

There are a number of programming exercises in this worksheet. Some of the methods will need some previous exercises to have been completed. Full marks will be given if you implement tests for all your programs. An example is provided for Exercise 1 below. This code tests your solution by generating 25 numbers and checking if they are odd or even.

## 4 methods (without recursion)

Design, implement and test the following seven problems/exercises using non-recursive methods.

### Exercise 1: Even/Odd

Write a method that **returns** true if an integer is even and false if the number is odd.

Use the code below to test that your method works as desired.

**public** **class** Lab6\_Ex1

{

**public static** **void** main(String args[])

{

**int** i;

**boolean** result;

**String** s = "";

**for**(i=1;i<=25;i++)

{

result = *EvenOdd*(i);

s = (result)?"Even":"Odd";

System.*out*.println(i + " is " + s);

}

}

**public** **static** **boolean** EvenOdd(**int** x)

{

//add your code here to check if x is odd or even. Note that the return type is Boolean. So it should return true if it is even and false if it false.

}

}

### Exercise 2: Reverse String

Write a method that when passed a string, returns the string in reverse order. Call your method from the main method:

**public static** **void** main(String args[])

{

System.out.println(reverse("desserts"));

}

Your method could be declared as follows:

**public** **static** String reverse(String ipx)

{

//add your code here to reverse the string

}

Can you think of possible erroneous input? For example, what if the input is an empty string? How would your solution be able to handle it?

Having a look at String methods may be useful:

<http://docs.oracle.com/javase/6/docs/api/java/lang/String.html>

### Exercise 3: Leap Year

Write a method that returns true if a year is a leap year and false otherwise. Call and test your method as above. Again, full marks will be given if your code can handle erroneous input.

Look at the following web page for the definition (and algorithm) of a leap year:

<http://en.wikipedia.org/wiki/Leap_year>

Check that the code works using this online calculator:

<http://www.dataip.co.uk/Reference/LeapYear.php>

### Exercise 4: Days in a Month

For a given a *month* and *year* write a program that returns the number of days in that month (so your method should takes two parameters).

Hint: use the method you wrote for calculating whether a year is a leap year or not. It may be a good idea to have the leap year method inside the same class.

Just to remind you:

1 - January - 31 days

2 - February - 28 days; 29 days in leap years

3 - March - 31 days

4 - April - 30 days

5 - May - 31 days

6 - June - 30 days

7 - July - 31 days

8 - August - 31 days

9 - September - 30 days

10 - October - 31 days

11 - November - 30 days

12 - December - 31 days

### Exercise 5: Factorial

Write a method that calculates the factorial of a long number *n*. It is very similar to the sum(n) method that you implemented in lab 6.

**public static** **void** main(String args[])

{

System.out.println(factorial(5));

}

*The factorial of a number n (n!) is defined as the* ***product*** *of all the positive integers less than or equal to n. For example, the factorial of 5 is:*

*1 x 2 x 3 x 4 x 5 =* ***120***

You should use a loop to complete this exercise. You will be asked to do the recursive version in the next section.

### Exercise 6: Character count in a string

Write a method that returns the number of times that a character occurs in a string. For example, if the string is “abracadabra” and the character is ‘a’ then the method returns 5. Call it from the main method as follows.

**public static** **void** main(String args[])

{

System.*out*.println(*charcount(*"abracadabra", 'a'));

}

Your method could be declared as follows:

**public** **static** int charcount(String ipx, char c)

{

//add your code here to count the frequency of c in ipx

}

## 5 Recursive methods

The purpose of this section of the laboratory worksheet is to develop a number of recursive methods.

### Exercise 7: Recursive Factorial

Now develop another method that returns the factorial for a given number *x,* but this time using recursion.

Note that this requires a very small change to be made to the recursive method we discussed in the lecture. Can you find the factorial of 40? Why?

### Exercise 8: Palindrome

A palindrome is a word or phrase that reads the same backwards or forwards. Write a method called *palindrome* that takes as input a character string and returns true or false depending on whether the string is a palindrome or not.

**You can use recursion or a loop – it is up to you!**

**public static** **void** main(String args[])

{

System.*out*.println(*palindrome(*"anna"));

System.*out*.println(*palindrome(*"a toyota"));

}

### Exercise 9: Simple Calculator

Write a java program for a GUI-based calculator that performs the basic arithmetic operations (+, -, \*, /). The calculator will have 10 buttons for the numbers (0 - 9) and 4 buttons for the basic arithmetic operations. Also, you’ll need to have a text field to display the result of the calculation. In addition, the calculator needs to have a memory functions (two buttons) to save and recall one value even after closing the program.