Laboratory 7 – Week 9

## Built in Functions and Methods

### 7.1 Introduction

**Firstly, this worksheet *is* one of the worksheets from which your laboratory worksheets portfolio of work will be assessed.**

This laboratory worksheet covers the use of user defined functions, 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.

### 7.2 Preliminaries

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

Unless stated in the text of the exercises, your solutions should have **no side effects** and all of your solutions should handle erroneous input data.

### 7.3 Writing User Defined Functions

There are a number of programming exercises in this worksheet. Some of the functions will need some previous exercises to have been completed. Test all of your programs, for example, you might test your solution for Even/Odd (7.4.1) as follows:

**public** **class** CS1702\_Lab8

{

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

{

**int** i;

**boolean** eo;

String s = "";

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

{

eo = *EvenOdd*(i);

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

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

}

}

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

{

//...

}

}

**7.4 Non-Recursive Functions**

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

### 7.4.1 Even/Odd

Write a function that returns true if an integer (or long) is even and false if the number is odd.

### 7.4.2 Reverse String

Write a function that when passed a character string, returns the string in reverse order.

### 7.4.3 Leap Year

Look at the following web page for the definition of a leap year:

<http://www.timeanddate.com/date/leapyear.html>

Write a function that returns true if a year is a leap year and false otherwise.

### 7.4.4 Days in a Month

For a given a month and year write a program that returns the number of days in that month. Hint: use the function you wrote for calculating whether a year is a leap year or not. You could also use some of the code you wrote for a previous worksheet.

### 7.4.5 Days Alive

Given a person’s date of birth, how many days have they been alive?

### 7.4.6 The Fibonacci Sequence

Read up on Fibonacci numbers at the following website:

<http://mathworld.wolfram.com/FibonacciNumber.html>

Write a program that when given a positive (say *n*) integer displays the first *n* Fibonacci numbers. If *n* is not a positive integer the program should generate an error.

**7.5 A User Defined Function for the Finch Robot**

Write a function/method that takes as input an integer corresponding to a time in milliseconds (ms) and then moves the Finch Robot around randomly for this time counting the number of obstacles it encounters using both obstacle detectors. This number of obstacles is returned by the function/method.

Design your solution using Pseudo-Code. Document (using comments) any assumptions you have made, for example what you have interpreted by “move around randomly”.

### 7.6 Recursive Functions

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

### 7.6.1 Factorial

Develop a function that returns *x*! for a given number *x*, note that this requires a very small change to be made to the recursive function we discussed in the lecture. Make sure you test for all of the valid ranges for *x*. What is the maximum value of *x*! that Java can handle?

### 7.6.2 Fibonacci Numbers

We looked at the Fibonacci numbers in a previous exercise. Write a recursive version that returns the *n*th Fibonacci number (say *Fib*(*n*)). Why are the number of calls to *Fib*(*n*) always odd? Given the value of *Fib*(*n*) (not *n*) can you work out the number of calls?

### 7.6.3 Character Counter

Write a recursive function 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 function returns 5.

### 7.6.4 Palindrome

A palindrome is a word, phrase, verse, or sentence that reads the same backwards or forwards. Write a recursive function that takes as input a character string and returns true or false depending on whether the string is a palindrome or not. Look on the web for some useful test cases (search for *palindrome* on *Google* for example).

### 7.6.5 The McCarthy 91 Function

Implement the following mathematical formulae as a recursive function.



Test the function on integers ranging from -250 to 250. What do you notice about this function?