# **Lab 4 Recursion**

# **Week beginning 2nd October 2023**

**Recursion**

**Use JUnit for all the methods which have a return value. Otherwise test by hand in** main() **method.**

1(a). Step through the code for Factorial method in debug mode and see all the method calls and the methods returning.

Note: For the purpose of debugging here, easier to call factorial() from main() rather use JUnit.

(b) Use a very large value as argument for factorial() so that the method gives a java.lang.StackOverFlow error.

2. Test the code for recursive version of size() method of CP3LinkedList as given in slides.

3. (a) Write a recursive method to find the sum of all values in an array.

Put the method in a class called MyArrays.

The header for the method is:

public static int sum(int [] values)

Hint: sum the partial array containing all but the last element. Then add the last element to this sum.

Hint: Use a public method that calls a private recursive helper method.

4. Write a recursive version of binary search method. See code for binarySearch method in BinarySearch folder

Hint: Use a public method that calls a private recursive helper method.

5. CP3LinkedList has a print() method that outputs the data on the list. Write a recursive version of this method. It will call the following helper method:

private void printSub(Node head)

Hint: If head is null, then there’s nothing to output. Otherwise, output the head.data before recursively moving on to head.next.

Difficult: Then consider how you could change the printSub method to output the nodes in reverse order.

6. The Fibonacci sequence used in Mathematics is defined as follows:

F(0) = 0

F(1) = 1

F(n) = F(n-1) + F(n-2) where n > 1

i.e. a recursive definition.

Write a recursive and a non-recursive method to compute this function.

Debug the recursive to see the different method calls.

Can you explain why recursion is very inefficient here?

**Question**: For all the methods above, consider whether an iterative or recursive method is more efficient. Compare the time taken to run the iterative and recursive methods.