**Enums**

**Recursion**

**Input and Output Files**

Remember to were appropriate Unit Test the methods.

**Enums**

**Exercise 1**

Create an enum class Countries that stores the capital city of several countries. Include the enum in a program that displays each country and its capital city.

**Recursion**

**Exercise 2**

Write a recursive method that finds the sum of the elements of an array of integers.

Approach the problem this way. If the array has only one element, the sum is simply the value of that element. Otherwise if the array has **n** elements, add the first element to the result of a recursive call which will take as a parameter an array of the remaining elements in the array.

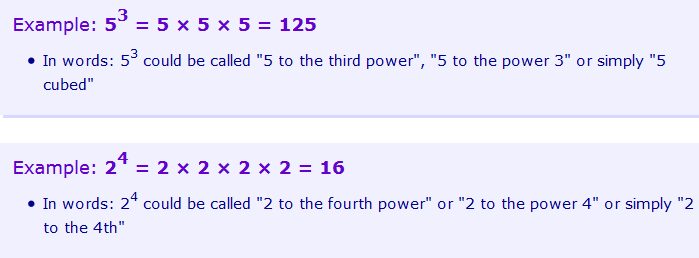
Hint : use **Arrays.copyOfRange(int[], from ,to)**

**Exercise 3**

Create a method that given an array of chars will print the array elements in reverse. Use a recursive call to do this.

**Exercise 4**

Create a recursive method to calculate the nth positive integer power of x.



**Input and Output Files**

**Exercise 5**

Consider again the Caesar cipher introduced in Practical 2 Exercise 3.

Write a program that reads text from a file, encrypts the text using the Caesar cipher and writes the encrypted text to an output file.

**Exercise 6**

**Part 1**

Create a text file in notepad with a number on every line. This will serve as the input of the program, so save this as input.txt. Copy this this to the root of your eclipse project.

The Java SDK provides various classes that allow for the reading / writing and general manipulation of various file types as seen here: <http://docs.oracle.com/javase/tutorial/essential/io/file.html>

Create a class called “**FileLoader**”, which contains a method **readFile**.

readFile should take as a parameter a String that is the name of the file we wish to read. In our case this will always be “input.txt”, but it allows for reuse for other files. Make the method return a list of Integers. Remember that lists in Java do not support primitives (int, double, float, boolean) and so the parent type must be used (Integer, Double, Float, Boolean).

NB: We only need to use the parent type when creating the list, once it has been created we can safely add in primitives.

Within this method read the file “input.txt” line by line, and add each number into a list of Integers. Each time a line is read it will be a string, so remember to convert it to an Integer before adding.

**Part 2**

Once the file has been read, and the list of numbers have been returned, we now want to sort them. As lists are part of the Collections library within Java we can make use of the Collections class that provides common utility methods that allow us to manipulate collections. Use this to sort the Integer list.

**Part 3**

Now that we have a sorted list of numbers, we want to write this to a new file called “output.txt”. In the FileLoader class create a new method called writeFile, and have it take as an input a list of integers, and a string for the name of the file you wish to create. Write to this file the list of numbers each separated with a new line.

Hint: Use the classes File, FileWriter, and BufferedWriter to achieve this.

**Part 4**

Create a test class that will test both of these method calls. To test the reading method create a List of Integers that we expect to be returned from “input.txt” and assert they are equal to what the method actually returns. Write a similar test for the file that we output.

Hint: Use the read file method when trying to test the contents of the new output file written by the test.