Assignment 5

Assignment Instructions:

- Complete exercises 1, 2, 3 and 4 in C/C++. These exercises require recursive functions to be written. There are no marks for iterative implementations.
- Complete the remaining exercises using Java.
- Each exercise is worth the same marks (10 marks each).
- Aim to make your code as concise and logical as possible and make sure code is properly indented.
- Exercises 1-4 can be uploaded as a single C++/text file.
- Upload separate Java files for the remaining exercises (four .java files to be uploaded, as specified in the instructions below). No zip files accepted.
- It's recommended that you test your code by calling your functions/classes from your main() method, but no code written in main() will be graded.
- Do not use any library functions, unless the question specifies that you can.

This assignment is due 6 pm ~~Friday 20th March~~ Monday 23rd March. No late submissions accepted.

Q1. Write a **recursive** function in C++

```
void printNumberedChars(string str, int i=0) { ... }
```

which prints each character of the given string `str` on a new line, with each line numbered 1, 2, 3, ... For example calling the function with `printNumberedChars("hello");` should output the following:

```
1. h
2. e
3. l
4. l
5. o
```

Q2. Write a **recursive** function in C++

```
int sumArray(const int* arr, unsigned int size) { ... }
```

which takes an array of integers, of the specified size, and returns the sum of the numbers in the array.

Q3. Write a **recursive** function in C++

```
long lowestPrimeFactor(long N, long i=2) { ... }
```

that returns the lowest prime factor of the number N. Note the following cases:

- If N is less than 2, then return 1
- If N is prime, then return N
- Otherwise, return the lowest prime factor of N

Note: It is suggested that the default value of the parameter `i` should be 2. This default may be different depending on your implementation.

Q4. Write a **recursive** function in C++

```
void printPrimeFactors(long N) { ... }
```

that prints out all prime factors of N in order of largest to smallest.

For example, calling your method from main() with

```
printPrimeFactors(1289531243);
```

should print out the following to the output console: