# CS1020 Take-home Lab #3

## Exercise #2: Big numbers

(http://www.comp.nus.edu.sg/~cs1020/3_ca/labs.html)

### Objective

Achieve basic understanding of linked lists.

### Task Statement

Chin is an undergraduate in the Department of Mathematics working on his final year project. His research on prime numbers requires the handling of large numbers. However, he discovered that Java primitives like integers do not support large numbers. In fact, the biggest integer that we can store in a variable of type int is $2^{31}$-1 on a 32-bit processor.

As a close friend of Chin, you vaguely recall that this problem can be solved with linked lists.

You will hence develop a way for Chin to input large numbers and output the sum of these numbers.

You are to create a Java class named **Digit** to represent a node. The attributes are the digit itself and reference to the next node in the list.

You will also need to write a Java class named **BigNumber** which contains an **add()** method that takes in two big numbers (both are String objects) and return their sum (also a String object).

*Hint:* Use each node to store a digit. Think about how an addition operation is performed on two numbers.

| Important: You are **not allowed** to use the BigInteger (or equivalent) class, or the LinkedList class provided by Java. Doing so violates the objective of this exercise and hence your program will be deemed incorrect. |
|---|

Some questions to think about when you're done (beyond the scope of this assignment):

1) Can this method be applied to subtraction?
2) What about multiplication and division?

### Input

The first line of the input contains a positive number **L** ($1 \leq L \leq 20$) which determines the number of addition operations to be performed. Each successive line contains two large positive integers **M** and **N** ($0 \leq M+N < \infty$) separated by a space.
(In your program, you should use more descriptive variable names instead of **M** and **N** and follow Java naming convention.)

### Output

The output contains the sum of each **M**+**N**.

**Sample Input #1**

```
1
900139190258102985175475477 185120982701967121286128612
```

**Sample Output #1**

```
10852601729600701064616040 89
```

**Sample Input #2**

```
3
012948190285125891275182612 12861892679161261826211
129619261902681269102611 26 198491289512812961261261
858869126948395893001361261 1298412098691826091286021860126 57
```

**Sample Output #2**

```
1296105217780505253700882 3
1316041747978093987152238 7
1298420687383095575244951873739 18
```

**Sample Input #3**

```
2
659803489238951829581209851928 5 1298591809435943758934759345436
0 0
```

**Sample Output #3**

```
7896626701825462054746857864721
0
```

**Skeleton Program**

The following skeleton is given. Normally, you would have put the **Digit** class and **BigNumber** class (the client) in separate files, but to reduce the number of files submitted, you are to put them in a single file **BigNumbers.java**.

```
/*
 * CS1020 (AY2012/3 Sem2)
 * Lab #3 Ex2
 * Author    :
 * Matric no.:
 * Description of program: (fill in the description below)
 */
import java.util.*;

class Digit {

   // Data attributes
   private int digit;
   private Digit next;

   // Constructors
   // This default constructor has an empty body
   public Digit(){
   }

   public Digit(int digit, Digit next) {
      this.digit = digit;
      this.next = next;
```

```
    }

    // Accessors
    public int getDigit() {
        return digit;
    }

    public Digit getNext() {
        return next;
    }

    public void setDigit(int digit) {
        this.digit = digit;
    }

    public void setNext(Digit next) {
        this.next = next;
    }
}

class BigNumber {
    private Digit head = null;
    private int length = 0;

    // Constructors
    // This default constructor has an empty body
    public BigNumber() {
    }

    // Build a linked list of digits
    public BigNumber(String number) {
        // Fill in
    }

    // Accessors
    public Digit getHead() {
        return head;
    }

    public int getLength() {
        return length;
    }

    // Append a digit to the linked list
    public void add(int digit) {
        // Fill in
    }

    // Sum two big numbers and return its sum
    public static String sum(BigNumber n1, BigNumber n2) {
        // Fill in
    }

    public String toString() {
        // Fill in
    }
    // You may add in other method(s) if you wish
}
```

```
// This is the driver class
class BigNumbers {

   public static void main(String[] args) {
      // Declare a Scanner object to read input

      // Declare the necessary variables

      // Read input and process them accordingly

      // Output the result
      // Ensure your output is in the right format
   }
}
```