CS19001: Programming and Data Structures Laboratory

Assignment No. 6A (Characters and Strings)

Date: 13-September-2019

Problem Statement:

In the city of Wonderland, once the Mad-Hatter gifted Alice one enigma which can convert any text-string to a garbled message and vice-versa. Alice found that the enigma runs a wonderful program and has a well-defined logic to do such string conversations. The rule that the enigma follows, according to Alice's observation, is given below. The enigma usually take the 8-bit binary representation of each character in a text-string and concatenate all these bits to make an intermediate representation of the whole text-string. Now, it considers each string of six (6) bits in the intermediate representation (starting from left-end towards the right) and map it according to the following rules:

```
 0 => A,  1 => B, ..., 25 => Z    (Capital Letters)
26 => a, 27 => b, ..., 51 => z    (Small Letters)
52 => 0, 53 => 1, ..., 61 => 9    (Numericals)
62 => +, 63 => -                  (Two-Symbols)
```

With the above mapping, the enigma can encrypt (and also reverse-map) strings. For instance,

- the text-string `IIT Kharagpur` gets encrypted into `SUlUIEtoYXJhZ3B1cC`; whereas
- the encrypted string `QWxpY2UgaW4gV29uZGVybGFuZA` gets reverse-mapped to `Alice in Wonderland`.

Your task is to write a C-program that enigma uses to do the following:

- The program takes a text-string as input and encrypt the text-string as per the above rule with only `A`...`Z`, `a`...`z`, `0`...`9`, `+` and `-` characters.
- The program takes another (different) encrypted-string (having only `A`...`Z`, `a`...`z`, `0`...`9`, `+` and `-` characters in it) and reverse-map it back to the text-string.

Though, the text-string and the encrypted-string (through this reverse-map) is not always invertible (why?).

Example Inputs/Outputs:

Sample-1:

```
Enter Original String: IIT Kharagpur
The Encrypted String: SUlUIEtoYXJhZ3B1cC

Enter Encrypted String: QWxpY2UgaW4gV29uZGVybGFuZA
The Original String: Alice in Wonderland
```

Sample-2:

```
Enter Original String: cross this both black buck greater einstein six thirty012345
The Encrypted String: Y3Jvc3MgdGhpcyBib3RoIGJsYWNrIGJ1Y2sgZ3JlYXRlciBlaW5zdGVpbiBzaXggdGhpcnR5MDEyMzQ1

Enter Encrypted String: Y3Jvc3MgdGhpcyBib3RoIGJsYWNrIGJ1Y2sgZ3JlYXRlciBlaW5zdGVpbiBzaXggdGhpcnR5MDEyMzQ1
The Original String: cross this both black buck greater einstein six thirty012345
```

Submit a single C source file. Do not use global/static variables.

CS19001: Programming and Data Structures Laboratory

Assignment No. 6B (Characters and Strings)

Date: 13-September-2019

Problem Statement:

In the city of Wonderland, the numeric system followed was still *Roman*. When Alice was learning this Roman numeric system from her elder sister, Lorina, she came to know that, I, V, X, L, C, D, M are the only characters used to represent numbers in this system and they corresponds to decimal values, $1, 5, 10, 50, 100, 500, 1000$. Any positive number can be constructed using these characters, for example $XIV = 14$. To interpret the decimal value corresponding to a Roman number, the rules that are followed are given in the following.

- For two consecutive Roman characters, if the first character is of higher (or equal) value than the next character, then the added values contribute to the decimal number. For example, $VI = 5 + 1 = 6$.
- For two consecutive characters, if the second character is of higher value than the first character, then the value that contribute to the decimal number is generated by subtracting the value of the first Roman character from the value of the second Roman character. For example, $IX = 10 - 1 = 9$.

Following this scheme of Roman to decimal number conversion, XIV will be interpreted as, $XIV = X[10] + IV[4] = 14$. *Note that, in Roman number system,* 0 *(zero) can never be expressed. Why?*

Your task is to develop a C-program that takes as input a Roman number (as a string), and provide as output the equivalent Decimal number (as long unsigned integer) along with the conversion breakup mechanism.

Example Inputs/Outputs:

Sample-1:

```
Enter Roman String: XIV
Decimal Conversion Breakup: X [10] + IV [4]
Equivalent Decimal Number : 14
```

Sample-2:

```
Enter Roman String: MDCCLXVII
Decimal Conversion Breakup: M [1000] + D [500] + C [100] + C [100] + L [50] + X [10] + V [5] + I [1] + I [1]
Equivalent Decimal Number : 1767
```

Sample-3:

```
Enter Roman String: XCIX
Decimal Conversion Breakup: XC [90] + IX [9]
Equivalent Decimal Number : 99
```

Submit a single C source file. Do not use global/static variables.