ES112.02/04

Fall 2003-2004 Term Project

Project #4: "Roman Numeral Calculator"

# 1 The Project

The goal of this project is to write a C program that can act as a very simple Roman numeral calculator. It should read just one operation from the keyboard (one addition, subtraction, division, or multiplication) and output its result, also in Roman numerals.

# 2 Information on Roman Numerals

You have probably already met Roman numerals before, but let us make a quick review. In the days of the Roman Empire, they used sequences of letters to represent numbers. (What we use today are called Arabic numerals.) It was mighty hard to do any mathematics using Roman numerals, so eventually they were abandoned. These days they are only used in numbering things, such as introductory pages of a book, putting dates on buildings and movies, and on the faces of watches and clocks.

## 2.1 Numeral Values

The table below gives the values of the various numerals that are used.

| Numeral | Value |
|---|---|
| I | One |
| V | Five |
| X | Ten |
| L | Fifty |
| C | One Hundred |
| D | Five Hundred |
| M | One Thousand |

## 2.2 Numeral Syntax

There are three rules that must be followed when representing a number using Roman numerals. Here they are:

1. In general, smaller valued numerals follow larger valued numerals. In this case, the values of the numerals are added up to find the value of the represented number.

2. Numerals that are powers of ten (I, X, C, M) can be repeated up to three times in a row. The rest of the numerals (V, L, D) can not be repeated.

3. It is possible to place a smaller numeral before a larger one. In that case, the value of the smaller numeral is subtracted from the value of the larger numeral. But, there are four rules that must be obeyed in this case:

   - The smaller numeral must be a power of ten (I, X, C, M).

- The smaller numeral must be one tenth or one fifth of the larger numeral.
- The smaller numeral must either be the first numeral in the expression, or be preceded by a numeral of at least ten times its value.
- If another numeral follows the larger numeral, it must be smaller than the one that precedes the larger numeral.

## 3 Project Requirements

The running of the program is pretty straightforward. But, error checking and reporting constitutes and important part of this program. The program should ask the user to enter an arithmetic operation in Roman numerals. The user should use '+' for addition, '−' for subtraction, '*' for multiplication, and '/' for division. A sample input from the user may be:

```
Enter an arithmetic operation in Roman numerals: XVI + CX
```

There may or may not be spaces around the operator. However, spaces are not allowed within the numbers. You should check that it is a valid operation. The things you must check for are:

- There must be exactly two Roman numerals, with one operator in between.
- The operator must be one of the operators given.
- The Roman numerals must be valid Roman numerals, according to the rules given here.

Note that only positive integers can be represented using Roman numerals. Zero can not be represented. In the input, zero can not be entered.

With division, you must obviously perform integer division.

Once you have calculated the result, you must print it out using Roman numerals as well. Note that under the rules, you can only represent numbers up to 3999 (MMMCMXCIX) since you are not allowed to repeat M more than three times. If such a result occurs, you should print out "Overflow". If a negative number results (from a subtraction), print its absolute value in Roman numerals, preceded by a minus sign (given it is greater than or equal to -3999). Should an operation result in zero (division, subtraction), just print nothing.

In error detection and reporting, you should be as careful and explicit as possible. You should try to report what rule is being broken and by what portion of the input.