CS202

Assignment 4: Roman Numeral Calculator

Description

For this assignment, you will implement operators to compute roman numerals, the class you will construct can be seen below

```
class romanNumeral
{
public:
  romanNumeral();
  romanNumeral(string);
  romanNumeral operator+(const romanNumeral&) const;
  romanNumeral operator+(int) const;
  romanNumeral operator-(const romanNumeral&) const;
  romanNumeral operator-(int) const;
  romanNumeral operator*(const romanNumeral&) const;
  romanNumeral operator*(int) const;

  friend romanNumeral operator+(int, const romanNumeral&);
  friend romanNumeral operator-(int, const romanNumeral&);
  friend romanNumeral operator*(int, const romanNumeral&);
  friend ostream& operator<<(ostream&, const romanNumeral&);
  friend istream& operator>>(istream&, romanNumeral&);
private:
  string rNum;
};
```

Description of Members

Each member will contain/perform the following

- `string rNum` contains the roman numeral
- `romanNumeral::romanNumeral()` is the default constructor that sets `rNum` to an empty string
- `romanNumeral::romanNumeral(string str)` is the constructor that will assign `str` to `rNum`
- `romanNumeral romanNumeral::operator+(const romanNumeral& rhs) const` is the addition operator that adds the `rhs` object to the object that calls the operator and a `romanNumeral` object is returned that contains the sum
- `romanNumeral romanNumeral::operator+(int rhs) const` is the addition operator that adds the `rhs` variable (which is an integer) to the object that calls the operator and a `romanNumeral` object is returned that contains the sum

- `romanNumeral romanNumeral::operator-(const romanNumeral& rhs) const` is the subtraction operator that subtracts the `rhs` object from the object that calls the operator and a `romanNumeral` object is returned that contains the difference
- `romanNumeral romanNumeral::operator-(int rhs) const` is the subtraction operator that subtract the `rhs` variable (which is an integer) from the object that calls the operator and a `romanNumeral` object is returned that contains the difference
- `romanNumeral romanNumeral::operator*(const romanNumeral& rhs) const` is the multiplication operator that multiplies the `rhs` object to the object that calls the operator and a `romanNumeral` object is returned that contains the product
- `romanNumeral romanNumeral::operator*(int rhs) const` is the multiplication operator that multiplies the `rhs` variable (which is an integer) to the object that calls the operator and a `romanNumeral` object is returned that contains the product
- `romanNumeral operator+(int lhs, const romanNumeral& rhs)` is a `friend` function that adds an `int` with a `romanNumeral` (in the order where there is an integer, then the operator, then the object), and returns a `romanNumeral` object that contains the sum
- `romanNumeral operator-(int lhs, const romanNumeral& rhs)` is a `friend` function that subtracts an `int` with a `romanNumeral` (in the order where there is an integer, then the operator, then the object), and returns a `romanNumeral` object that contains the difference
- `romanNumeral operator*(int lhs, const romanNumeral& rhs)` is a `friend` function that multiplies an `int` with a `romanNumeral` (in the order where there is an integer, then the operator, then the object), and returns a `romanNumeral` object that contains the product
- `ostream& operator<<(ostream& out, const romanNumeral& rhs)` is the output function that allows the `romanNumeral` object to output like a regular variable, i.e. `cout << obj1;` where `obj1` is a `romanNumeral` object
- `istream& operator>>(istream& in, romanNumeral& rhs)` is the input function that allows the `romanNumeral` object to be read in like a regular variable, i.e. `cin >> obj1;` where `obj1` is a `romanNumeral` object

As always comment your code and do not modify the `romanNumeral` class, you may have extra helper functions declared in your header file and implement them in the class .cpp file but they cannot be members of the class

Contents of Main

You will write your main to test all your functions, my main starts with

```
int main()
{
  romanNumeral r1("XLIX");
  romanNumeral r2("XXXVIII");
  romanNumeral r3;
  romanNumeral r4;
  romanNumeral result;
  int base10Num;

  //YOUR CODE COMES HERE
}
```

I use `result` to store the result after each operator used in main, and I read from the user and insert into `r3` and `r4`

Sample Output

No sample output given for this assignment, it would spoil the fun

```
Jimis-MacBook-Pro:Asst04 VaskoDaGamer$ make
g++ -c main.cpp
g++ -c romanNumeral.cpp
g++ main.o romanNumeral.o
Jimis-MacBook-Pro:Asst04 VaskoDaGamer$ ./a.out

Enter a roman numeral: XIX

Enter a roman numeral: III

Enter a base 10 number: 13

r1 = XLIX
r2 = XXXVIII
r3 = XIX
r4 = III
base10Num = 13

r1 + r3 = LXVIII
r2 + base10Num = LI
r1 * r4 = CXLVII
r3 * 6 = CXIV
r1 - r2 = XI
r1 - base10Num = XXXVI
100 - r1 = LI
2 * r2 = LXXVI
base10Num + r2 = LI

Jimis-MacBook-Pro:Asst04 VaskoDaGamer$
```

Submission

Upload your files: `romanNumeral.h`, `romanNumeral.cpp`, `main.cpp`, and `makefile` onto the moodle site within deadline