Object-Oriented Programming

Lab # 8

**Important Instructions:**

* You are required to create a **multi-file project** to accomplish each task.
* Indent your code properly.
* Use meaningful variable and function names. Follow the naming conventions.
* Make sure that there are no **dangling pointers** and **memory leaks** in your program.

**Task #1**

Implement the class **MyString** whose declaration is given below:

**class MyString**

**{**

**private:**

**char \*str; *// Pointer to the char array that holds the string***

**int strLength; *// Variable to store the length of the string***

**public:**

***// Default constructor to initialize the string to empty string***

**MyString();**

***// Overloaded constructor***

**MyString(const char \*);**

***// Copy constructor***

**MyString(const MyString&);**

***// Overloaded assignment operator***

**const MyString& operator = (const MyString&);**

***// Display the string on screen***

**void display ();**

***// Returns the length (# of characters excluding the null terminator) of the string***

**int getLength ();**

***// Destructor***

**~MyString();**

***// Overload the relational operators to allow comparison of two MyString objects***

**bool operator == (const MyString&) const;**

**bool operator != (const MyString&) const;**

**bool operator <= (const MyString&) const;**

**bool operator < (const MyString&) const;**

**bool operator >= (const MyString&) const;**

**bool operator > (const MyString&) const;**

***// Concatenating two MyString objects. Returns the new MyString***

**MyString operator + (const MyString&) const;**

***// Concatenates another MyString with the current MyString***

**const MyString& operator += (const MyString&);**

**};**

**Task #2**

Construct a class named **Fraction** containing two integer data members named **num** and **den**, used to store the numerator and denominator of a fraction having the form num/den. The four arithmetic operations on fractions are defined by the following rules:

Addition: 

Subtraction: 

Multiplication: 

Division: 

1. Your class should have a parameterized constructor which will receive two integer values and store them in **num** and **den** respectively. The default values of these two arguments should be 1.
2. Implement a member function **setFraction(int,int)** of the Fraction class which sets the values of num and den to the values passed as arguments into this function.
3. Implement a member function **display()** to display a Fraction on screen in the form num/den. For example, if num is 3 and den is 4 the fraction should be displayed as **3/4**.
4. Overload the **+** operator for adding two Fractions and returning the result.
5. Overload the **–** operator for subtracting two Fractions and returning the result.
6. Overload the **\*** operator for multiplying two Fractions and returning the result.
7. Overload the **/** operator for dividing two Fractions and returning the result.
8. Implement a member function **convertToDouble()** of the Fraction class which returns the value of the Fraction as a double value. For example, if **num** is 3 and **den** is 4, then this function should return 0.75.
9. Write a driver program (main function) which should demonstrate all of the above functions of the Fraction class.
10. Modify your Fraction class so that the above four arithmetic operators can be used to add, subtract, multiply, and divide a Fraction with an integer value.
11. Overload the **+= –= \*= /=** operators for adding, subtracting, multiplying, and dividing a Fraction with another Fraction.
12. Modify the Fraction class so that these four operators (**+= –= \*= /=**) can be used to add, subtract, multiply, and divide a Fraction with an integer value.
13. Implement a member function **reduceToLowestTerms()** of the Fraction class which reduces the Fraction to its lowest form (see Page 262 in Lafore’s book). Use this function in the previously implemented member functions of the Fraction class, so that the Fraction resulting from each arithmetic operation is in its lowest form.
14. Implement the copy constructor for the Fraction class.
15. Overload the assignment operator for the Fraction class.
16. Overload the **==** operator to determine whether two Fractions are equal or not.
17. Overload the **<** operator to determine whether one Fraction is smaller than another Fraction.
18. Overload the **>** operator to determine whether one Fraction is greater than another Fraction.
19. Write a driver program to test the working of all member functions of the Fraction class.