CMPE 50 – Spring 2015, Tarng
Lab #6– Classes II (friends, const functions)

Instructions: Before you leave the lab, you should submit your answers through Canvas->Assignment->Lab6->Submission. Please submit your answers (.cpp files with output embedded and appropriate documentation/comments) even if you couldn't complete/run them.

Exercise 1:
Create a class `Resource`. The class should have:

a) Two private variables `status` and `writeTo` representing integer value either 0 or 1.
b) One default constructor that initializes the `status` and `writeTo` to zero.
c) One single parameterized constructor to initialize the `writeTo` variable.
d) Two constant accessor functions per class that return the values of `status` and `writeTo` separately
e) Two mutator functions per class that set the values of `status` and `writeTo` separately.
f) One output (member) function that outputs the resource status. The output function should be able to print on the screen or an output file.

```
void output(ostream &out_stream);
```

g) A friend function `check_status` that accesses the `status` variables of both classes. If `status` in each resource object is set to "1", display "resource available" else display "resource unavailable".

```
bool check_status(Resource &res1, Resource &res2)
{
   // Define status checking code here.
   // return true if res1 status and res2 status are both 1
   // else return false.
}
```

Exercise 2:
Define a class `Rational` for rational numbers. A rational number is a number that can be represented as the quotient of two integers. For example, 1/2, 3/4, 64/2, and so forth are all rational numbers. (By 1/2, etc., we mean the everyday meaning of the fraction, not the integer division that this expression would produce in a C++ program.) Represent rational numbers as two values of type `int`, one for the numerator and one for the denominator. Declare both member variables as private.

Include a constructor with two arguments that can be used to initialize the member variables of an object to any legitimate values. This class should have the following public member functions:

1. Two functions called `input` and `output`. The `input` takes an argument of type of `istream`. It allows the input from the input stream to both the `numerator` and `denominator` member variables. The output function, having an argument of type `ostream`, will output the rational number in a form 1/2, 15/32, 300/401, and so forth to the output stream. Note that the numerator, the denominator, or both may contain a minus sign, so -1/2, 15/32, and -300/-401 are also possible inputs.
2. Friend functions `add`, `subtract`, `multiply`, and `division`. Each of these friend functions return a `Rational` object and has two input arguments of type `Rational&`. These friend functions should apply correctly to the type `Rational`.
3. Friend functions equal and `less_than`, which both return a `bool` value. They take two `const` arguments of type `Rational&`.

Write a test program to test your class.

(Hint: Two rational numbers a/b and c/d are equal if a*d equals c*b. If b and d are positive rational numbers, a/b is less than c/d provided a*d is less than c*b. You should include a helper function to normalize the values stored so that, after normalization, the denominator is positive and the numerator and denominator are as small as possible. For example, after normalization 4/-8 would be represented the same as -1/2. The following sample code gives a clue on how to find the greatest common denominator GCD of two positive numbers.)

```
int gcd(int n1, int n2)
{
	int temp;

	while (n2 != 0)
	{
		// cout << "n2 = " << n2 << endl;
		temp = n1;
		n1 = n2;
		n2 = temp % n2;
	}

	return n1;
}
```