Programming Practice: Rational Number Operator Overloading

We have designed and implemented rational numbers using C programming language. In this assignment, you will define and implement a rational number project using separate compilation of C++ programming language. Suppose a rational number p/q is represented by two relatively prime integers p and q, where q is a non-zero positive integer. In the case of p/0, set this number to rational number 0/1. If rational number p/q that p and q are not relatively prime, simplify p/q by dividing both p and q by their greatest common division (GCD). If p/q is a negative rational number, simplify the rational number as p<0 and q>0. When printing a rational number, if the denominator is 1, print the numerator only. You will write a C++ project of a class named Rational with the following requirements:

1. Define a default constructor and a copy constructor. The default constructor may take 0, 1, or 2 arguments, say, (p, q), with default value 0/1, p/1, and p/q, respectively. The copy constructor takes a rational number as its argument and copy the value of the argument.
2. Use overloaded operator member functions and friend functions to define and implement arithmetic operations "a+b" (addition), "a-b" (subtraction), "a*b" (multiplication), "a/b" (division), and "|a|" (absolute value), and comparing relations "a==b" (equal to), "a!=b" (not equal to), "a<b" (less than), "a>b" (greater than), "a<=b" (less than or equal to), and "a>=b" (greater than or equal to), where each of a and b is either an integer or a rational number, but not both of them are integers. Note that the two operands **cannot** be integers at the same time.
3. Use overloaded operator member functions to define and implement assignment statement of rational numbers and its variations, including, "<variable>=<expression>", "<variable>+=<expression>", "<variable>-=<expression>", "<variable>*=<expression>", and "<variable>/=<expression>". The left-hand-side <variable> is a rational reference and the right-hand-side <expression> is either an integer object or a rational object.
4. Overload I/O stream operators "<<" and ">>" to enable output and input of rational numbers.

In the main function int main() declare three rational variables a, b, and c, and three integer variables d, e, and f. Input values of these variables from the console. Declare rational variable g without initial value. Perform the following operations:

1. Evaluate and print the results of expressions -a, a+b, |a+b|, a+d, d+a, a-b, |a-b|, a-d, d-a, a*b, |a*b|, a*d, d*a, a/b, |a/b|, a/d, and d/a,
2. Evaluate and print the results of relations a==b, a==e, f==c, a!=b, a!=e, f!=c, a<b, a<e, f<c, a<=b, a<=e, f<=c, a>b, a>e, f>c, a>=b, a>=e, and f>=c.
3. Evaluate and print the results of expressios a*c/d+e*b-d/b-a, a*c/(d+e*b)-d/(b-a), and a*((c/(d+e)*b)-d/b)
4. Execute statement **if** (a>=b) **then** g=(d+e)*f **else** g=(d-e)/f; and print the value of g.
5. Execute statement **if** (c*d==b*e) **then** g*=b/d+c/e **else** g/=b*d+c*e; and print the value of g.
6. Evaluate and print the result of expression max(f-a/b, c+e*b)-min(a+c/e, b*d).

Euclidean algorithm for computing the great common divisor of two integers m and n is given below for your reference:

```
// GCD: greatest common divisor
int gcd(int m, int n) {
   m = abs(m);
   n = abs(n);
```

```
        if (m % n == 0) return n;
        else return gcd(n, m % n);
}
```

Using "cout <<" to print boolean value true/false, you need to include <iomanip>. Declare flag as a variable of type bool. Assign a boolean value to flag and then output the value with statement "cout << boolalpha << flag;" (reference boolalpha).

Execution Example: (next page)

命令提示符

```
D:\>rational_overloading
Enter rational number a (in the form of: p/q): 5/3
**** Value of a: 5/3
Enter rational number b (in the form of: p/q): 3/-5
**** Value of b: -3/5
Enter rational number c (in the form of: p/q): -7/4
**** Value of c: -7/4
Enter integer number d: -5
**** Value of d: -5
Enter integer numbere: -4
**** Value of e: -4
Enter integer number f: 6
**** Value of f: 6
====================================
**** Expression -a = -5/3
**** Expression a+b = 16/15
**** Expression |a+b| = 16/15
**** Expression a+d = -10/3
**** Expression d+a = -10/3
**** Expression a-b = 34/15
**** Expression |a-b| = 34/15
**** Expression a-d = 20/3
**** Expression d-a = -20/3
**** Expression a*b = -1
**** Expression |a*b| = 1
**** Expression a*d = -25/3
**** Expression d*a = -25/3
**** Expression a/b = -25/9
**** Expression |a/b| = 25/9
**** Expression a/d = -1/3
**** Expression d/a = -3
====================================
**** Relation a==b is false.
**** Relation a==e is false.
**** Relation f==c is false.
**** Relation a!=b is true.
**** Relation a!=e is true.
**** Relation f!=c is true.
**** Relation a<b  is false.
**** Relation a<e  is false.
**** Relation f<c is false.
**** Relation a<=b is false.
**** Relation a<=e is false.
**** Relation f<=c is false.
**** Relation a>b  is true.
**** Relation a>e  is true.
**** Relation f>c is true.
**** Relation a>=b is true.
**** Relation a>=e is true.
**** Relation f>=c is true.
====================================
**** Expression a*c/d+e*b-d/b-a = -421/60
**** Expression a*c/(d+e*b)-d/(b-a) = -2875/2652
**** Expression a*((c/(d+e)*b)-d/b)-a = -63/4
====================================
Statement: if (a>=b) then g=(d+e)*f else g=(d-e)/f;
**** The new value of object g: -54
====================================
Statement: if (c*d==b*e) then g*=b/d+c/e else g/=b*d+c*e;
**** The new value of object g: -27/5
====================================
**** Expression max(f-a/b, c+e*b)-min(a+c/e, b*d) = 961/144
====================================

D:\>
```