Rational Class

Problem:

Create a class called Rational for performing arithmetic with fractions.

Use integer variables to represent the private data of the class – the numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it's declared. The constructor should contain default values($\frac{1}{1}$) in case no initializers are provided and should store the fraction in reduced form. For example the fraction $\frac{2}{4}$ would be stored in the object as 1 in the numerator and 2 in the denominator. Provide public operator overloading functions that perform each of the following tasks, and all behave are the same as int type. All the operator should only do with Rational, no need to support with int or other types:

1) Binary Arithmetic:
    a) +
    b) -
    c) *
    d) /
2) Assignment:
    a) =
    b) +=
    c) -=
    d) *=
    e) /=
3) Relational:
    a) >
    b) <
    c) >=
    d) <=
    e) ==
    f) !=
4) Unary Arithmetic:

a) prefix++ , the value of a variable is incremented by 1 before using it.
b) postfix++, the value of a variable is incremented by 1 after using it.
c) prefix--, the value of a variable is decremented by 1 before using it.
d) postfix--, the value of a variable is decremented by 1 after using it.
e) - , negative

5) cout (this is friend function):
   a) It should support the code like this
      i) `cout << rational_var << endl;`
   b) Printing Rational numbers in the form a/b, where a is the numerator and b is the denominator. If a/b is a negative number, the negative sign needs to be in front, for example -3/8 is correct but 3/-8 is wrong.

Input:

Each test contains multiple test cases. The first line contains the number of test cases T. Description of the test cases follows.

Each of the following T lines contains 5 integers C, a, b, c, d. C represents the method to be done.

when C is equal to 1, it will test Binary Arithmetic.
when C is equal to 2, it will test Assignment.
when C is equal to 3, it will test Relational.
when C is equal to 4, it will test Unary Arithmetic.

Please see file main.cpp for details.

Output

If your class is good enough, sample code will do well.

Sample input:

5
1 1 2 3 4
2 4 3 2 1
3 2 5 1 4
4 4 5 6 4
3 1 2 2 4

Sample output:

Operator + 5/4
Operator - -1/4
Operator * 3/8
Operator / 2/3
Operator += 13/3
Operator -= -7/3
Operator *= 8/3
Operator /= 3/8
Operator > 1
Operator < 0
Operator >= 1
Operator <= 0
Operator == 0
Operator != 1
Operator prefix ++ 43/10 9/5 5/2
Operator postfix ++ 23/10 9/5 5/2
Operator prefix -- 3/10 -1/5 1/2
Operator postfix -- 23/10 -1/5 1/2
Operator negative- -4/5 -3/2
Operator > 0
Operator < 0
Operator >= 1
Operator <= 1
Operator == 1
Operator != 0

Some important:

You should use the main.cpp which we gave you, and don’t rewrite any code in main.cpp.

You can rewrite any code in rational.h and rational.cpp, but it must be able to successfully compile with main.cpp.

Others:

we will not use 0 as denominator, and all the numbers could be stored in int type.