Source File: ~/1337/58/lab58.(C|CPP|cpp|c++|cc|cxx|cp)
Input: Under control of main function
Output: Under control of main function
Value: 3

The purpose of this lab assignment is to extend the Rational abstract data type from Lab 57. In this assignment you will add functions for performing arithmetic with rational numbers. The header file for this assignment is shown in Figure 1.

```
#ifndef LAB58_H
#define LAB58_H

#include <lab57.h>

using namespace std;

// Function lcm returns the least common multiple of u & v
int lcm(int u, int v);

// Function rationalAdd returns augend + addend.  The result is
// returned in reduced form.
Rational rationalAdd(const Rational& augend, const Rational& addend);

// Function rationalAdditiveInverse returns the additive inverse.
// The additive inverse, or opposite, of a number a is the number
// that, when added to a, yields zero. Thus, the fraction a/b is
// returned as -a/b.
Rational rationalAdditiveInverse(const Rational *rational);

// Function rationalSubtract returns minuend - subtrahend.  The result
// is returned in reduced form.
Rational rationalSubtract(const Rational& minuend, const Rational& subtrahend);

// Function rationalMultiply returns multiplier x multiplicand.  The
// result is returned in reduced form.
Rational rationalMultiply(const Rational& multiplier, const Rational& multiplicand);

// Function rationalMultiplicativeInverse returns the multiplicative inverse.
// The multiplicative inverse, or reciprocal, of a number a is the
// number that, when multiplied by a, yields the multiplicative
// identity, 1.  The multiplicative inverse of the fraction a/b is b/a.
Rational rationalMultiplicativeInverse(const Rational *rational);

// Function rationalDivide returns dividend / divisor.  The result is
// returned in reduced form.
Rational rationalDivide(const Rational& dividend, const Rational& divisor);

#endif
```

Figure 1. /usr/local/1337/include/lab58.h

The least common multiple of two integers u and v, written $\mathrm{lcm}(u, v)$, is the smallest non-negative integer that is a multiple of (i.e., evenly divisible by) both u and v; and $\mathrm{lcm}(0, 0) = 0$. For non-zero values of u and v, define the least common multiple as

$$\mathrm{lcm}(u, v) = \frac{uv}{\gcd(u, v)}$$

where $\gcd(u, v)$ is the greatest common divisor of u and v.

A sample `main` function for testing these functions is shown in Figure 2. A sample execution sequence is shown in Figure 3. To use the `Makefile` as distributed in class, add a target of `lab58` to `targets2srcfileswithlibrary`.

```
#include <lab58.h>
#include <cstdlib>

using namespace std;

int main()
{
  Rational first, second, result;
  string operators = "+-*/";
  string::size_type i;

  while (rationalInput(rationalInput(cin, first), second))
  {
    if (rationalValid(first) && rationalValid(second))
    {
      cout << "The additive inverse of ";
      rationalOutput(cout, first);
      cout << " is ";
      rationalOutput(cout, rationalAdditiveInverse(&first));
      if (first.numerator != 0)
      {
        cout << " and the multiplicative inverse of ";
        rationalOutput(cout, first);
        cout << " is ";
        rationalOutput(cout, rationalMultiplicativeInverse(&first));
      }
      cout << endl;

      cout << "The additive inverse of ";
      rationalOutput(cout, second);
      cout << " is ";
      rationalOutput(cout, rationalAdditiveInverse(&second));
      if (second.numerator != 0)
      {
        cout << " and the multiplicative inverse of ";
        rationalOutput(cout, second);
        cout << " is ";
        rationalOutput(cout, rationalMultiplicativeInverse(&second));
      }
```

Figure 2. `/usr/local/1337/src/lab58main.C` (Part 1 of 2)

```
      cout << endl;

      for (i = 0; i < operators.length(); ++i)
      {
        rationalOutput(cout, first);
        cout << ’ ’ << operators[i] << ’ ’;
        rationalOutput(cout, second);
        cout << " = ";
        if (operators[i] == ’/’ && second.numerator == 0)
          cout << "Division by zero is not allowed";
        else
        {
          switch (operators[i])
          {
            case ’+’: result = rationalAdd(first, second); break;
            case ’-’: result = rationalSubtract(first, second); break;
            case ’*’: result = rationalMultiply(first, second); break;
            case ’/’: result = rationalDivide(first, second); break;
            default:  cerr << "Error.  Unknown operator.  Exiting." << endl;
                      exit(EXIT_FAILURE);
          }
          rationalOutput(cout, result);
        }
        cout << endl;
      }
    }
    else
    {
      cerr << "Error. Exiting." << endl;
      exit(EXIT_FAILURE);
    }
  }

  return EXIT_SUCCESS;
}
```

Figure 2. /usr/local/1337/src/lab58main.C (Part 2 of 2)

```
newuser@csunix ~> cd 1337
newuser@csunix ~/1337> mkdir 58
newuser@csunix ~/1337> cd 58
newuser@csunix ~/1337/58> cp /usr/local/1337/data/58/* .
newuser@csunix ~/1337/58> cp /usr/local/1337/include/lab58.h .
newuser@csunix ~/1337/58> cp /usr/local/1337/src/lab58main.C .
newuser@csunix ~/1337/58> cp /usr/local/1337/src/Makefile .
newuser@csunix ~/1337/58> touch lab58.cpp
newuser@csunix ~/1337/58> # Edit Makefile and lab58.cpp
newuser@csunix ~/1337/58> make lab58
g++ -g -Wall -std=c++11 -c lab58main.C -I/usr/local/1337/include -I.
g++ -g -Wall -std=c++11 -c lab58.cpp -I/usr/local/1337/include -I.
g++ -o lab58 lab58main.o lab58.o -L/usr/local/1337/lib -lm -lbits \
-Wl,-whole-archive -llab58 -Wl,-no-whole-archive
newuser@csunix ~/1337/58> cat 01.dat
-3  4 -3  4
-3  4  3  4
 0  5  0  7
 1  1  5  4
25 45  8 99
 3  4 -3  4
 1  4  3  2
 1  4  3 -2
-1 -4 -3 -2
newuser@csunix ~/1337/58> cat 01.dat | ./lab58
The additive inverse of -3/4 is 3/4 and the multiplicative inverse of -3/4 is 4/-3
The additive inverse of -3/4 is 3/4 and the multiplicative inverse of -3/4 is 4/-3
-3/4 + -3/4 = -3/2
-3/4 - -3/4 = 0/1
-3/4 * -3/4 = 9/16
-3/4 / -3/4 = 1/1
The additive inverse of -3/4 is 3/4 and the multiplicative inverse of -3/4 is 4/-3
The additive inverse of 3/4 is -3/4 and the multiplicative inverse of 3/4 is 4/3
-3/4 + 3/4 = 0/1
-3/4 - 3/4 = -3/2
-3/4 * 3/4 = -9/16
-3/4 / 3/4 = -1/1
The additive inverse of 0/5 is 0/5
The additive inverse of 0/7 is 0/7
0/5 + 0/7 = 0/1
0/5 - 0/7 = 0/1
0/5 * 0/7 = 0/1
0/5 / 0/7 = Division by zero is not allowed
The additive inverse of 1/1 is -1/1 and the multiplicative inverse of 1/1 is 1/1
The additive inverse of 5/4 is -5/4 and the multiplicative inverse of 5/4 is 4/5
1/1 + 5/4 = 9/4
1/1 - 5/4 = -1/4
1/1 * 5/4 = 5/4
1/1 / 5/4 = 4/5
```

Figure 3. Commands to Compile, Link, & Run Lab 58 (Part 1 of 2)

```
The additive inverse of 25/45 is -25/45 and the multiplicative inverse of 25/45 is 45/25
The additive inverse of 8/99 is -8/99 and the multiplicative inverse of 8/99 is 99/8
25/45 + 8/99 = 7/11
25/45 - 8/99 = 47/99
25/45 * 8/99 = 40/891
25/45 / 8/99 = 55/8
The additive inverse of 3/4 is -3/4 and the multiplicative inverse of 3/4 is 4/3
The additive inverse of -3/4 is 3/4 and the multiplicative inverse of -3/4 is 4/-3
3/4 + -3/4 = 0/1
3/4 - -3/4 = 3/2
3/4 * -3/4 = -9/16
3/4 / -3/4 = -1/1
The additive inverse of 1/4 is -1/4 and the multiplicative inverse of 1/4 is 4/1
The additive inverse of 3/2 is -3/2 and the multiplicative inverse of 3/2 is 2/3
1/4 + 3/2 = 7/4
1/4 - 3/2 = -5/4
1/4 * 3/2 = 3/8
1/4 / 3/2 = 1/6
The additive inverse of 1/4 is -1/4 and the multiplicative inverse of 1/4 is 4/1
The additive inverse of 3/-2 is -3/-2 and the multiplicative inverse of 3/-2 is -2/3
1/4 + 3/-2 = -5/4
1/4 - 3/-2 = 7/4
1/4 * 3/-2 = -3/8
1/4 / 3/-2 = -1/6
The additive inverse of -1/-4 is 1/-4 and the multiplicative inverse of -1/-4 is -4/-1
The additive inverse of -3/-2 is 3/-2 and the multiplicative inverse of -3/-2 is -2/-3
-1/-4 + -3/-2 = 7/4
-1/-4 - -3/-2 = -5/4
-1/-4 * -3/-2 = 3/8
-1/-4 / -3/-2 = 1/6
newuser@csunix ~/1337/58> cat 01.dat | ./lab58 > my.out
newuser@csunix ~/1337/58> diff 01.out my.out
newuser@csunix ~/1337/58> cat 04.dat | ./lab58 > my.out
newuser@csunix ~/1337/58> diff 04.out my.out
newuser@csunix ~/1337/58>
```

Figure 3. Commands to Compile, Link, & Run Lab 58 (Part 2 of 2)