Source File:	`~/1337/64/lab64.(C|CPP|cpp|c++|cc|cxx|cp)`
Input:	Under control of `main` function
Output:	Under control of `main` function
Value:	2

Create a class called `Rational` for performing various operations with fractions. The specification of the class will be provided. Your task will be to provide the implementation. A main program for testing your implementation will also be provided.

Use `int`eger variables to represent the `private` data of the class—the numerator and the denominator. The implementation should provide two constructors: a default constructor that initializes the numerator to zero and the denominator to one and a second constructor that takes two arguments (the first argument should be stored in the numerator and the second in the denominator). The constructors should **not** store the rational number in reduced form. Additional `public` member functions include:

- "set" functions for setting the numerator and denominator. The `setDenominator` function should check its argument for validity. If the function receives an argument equal to zero (0), the function should set the denominator to one (1).
- "get" functions for getting the numerator and denominator.
- reduction of a `Rational` to lowest terms. Also, `reduce` should modify the denominator of a `Rational` with a zero numerator to be one. Further, a negative `Rational` should ensure that the numerator is negative and the denominator is positive. `Rational`s having both numerator and denominator negative should be modified such that both numerator and denominator are positive.

If u and v are integers, not both zero, we say that their greatest common divisor, $\gcd(u, v)$, is the largest positive integer that evenly divides both u and v. When u and v are both zero, every integer evenly divides zero, so it is convenient to set $\gcd(0, 0) = 0$. When either u or v is zero, define $\gcd(u, 0) = |u|$ and $\gcd(0, v) = |v|$. Provide the implementation of this function as a `private` member of the `Rational` class.

A header file is shown in Figure 1, a sample `main` function for testing your implementation is shown in Figure 2, and a sample execution sequence is shown in Figure 3. To use the `Makefile` as distributed in class, add a target of `lab64` to `targets2srcfiles`.

```
#ifndef LAB64_H
#define LAB64_H

using namespace std;

class Rational
{
 public:
  Rational();                           // default constructor
  Rational(int num, int denom);         // additional constructor
  void setNumerator(int num);           // set numerator to num
  void setDenominator(int denom);       // set denominator to denom
  int  getNumerator() const;            // return numerator
  int  getDenominator() const;          // return denominator
  void reduce();                        // reduce to lowest terms and
                                        //   normalize
```

Figure 1. `/usr/local/1337/include/lab64.h` (Part 1 of 2)

```
 private:
  int numerator;
  int denominator;
  int gcd(int u, int v) const;          // returns the gcd of u and v
};

#endif
```

Figure 1. `/usr/local/1337/include/lab64.h` (Part 2 of 2)

```
#include <lab64.h>
#include <iostream>
#include <cstdlib>

using namespace std;

int main()
{
  int n, d;
  Rational first( 1, -2 ), second( -3, 0 ), third;

  cout << "first = " << first.getNumerator() << '/' << first.getDenominator()
       << endl;
  cout << "second = " << second.getNumerator() << '/'
       << second.getDenominator() << endl;
  cout << "third = " << third.getNumerator() << '/' << third.getDenominator()
       << endl;

  while (cin >> n >> d)
  {
    third.setNumerator(n);
    third.setDenominator(d);
    cout << "Before reduce() third = "
         << third.getNumerator() << '/' << third.getDenominator();
    third.reduce();
    cout << " After reduce() third = "
         << third.getNumerator() << '/' << third.getDenominator() << endl;
  }

  return EXIT_SUCCESS;
}
```

Figure 2. `/usr/local/1337/src/lab64main.C`

```
newuser@csunix ~> cd 1337
newuser@csunix ~/1337> mkdir 64
newuser@csunix ~/1337> cd 64
newuser@csunix ~/1337/64> cp /usr/local/1337/data/64/* .
newuser@csunix ~/1337/64> cp /usr/local/1337/include/lab64.h .
newuser@csunix ~/1337/64> cp /usr/local/1337/src/lab64main.C .
newuser@csunix ~/1337/64> cp /usr/local/1337/src/Makefile .
newuser@csunix ~/1337/64> touch lab64.cpp
newuser@csunix ~/1337/64> # Edit Makefile and lab64.cpp
newuser@csunix ~/1337/64> make lab64
g++ -g -Wall -std=c++11 -c lab64main.C -I/usr/local/1337/include -I.
g++ -g -Wall -std=c++11 -c lab64.cpp -I/usr/local/1337/include -I.
g++ -o lab64 lab64main.o lab64.o -L/usr/local/1337/lib -lm -lbits
newuser@csunix ~/1337/64> cat 01.dat
-3  4  3  4
 3 -4 -3 -4
25 45  8 99
 1  0  2  0
129 6579 1935 249
1331 1651 2301 1079
3 1260 6 198
43 1935 207 6579
5 7 -25 -35
-83 1651 127 -1079
1079 1651
newuser@csunix ~/1337/64> cat 01.dat | ./lab64
first = 1/-2
second = -3/1
third = 0/1
Before reduce() third = -3/4 After reduce() third = -3/4
Before reduce() third = 3/4 After reduce() third = 3/4
Before reduce() third = 3/-4 After reduce() third = -3/4
Before reduce() third = -3/-4 After reduce() third = 3/4
Before reduce() third = 25/45 After reduce() third = 5/9
Before reduce() third = 8/99 After reduce() third = 8/99
Before reduce() third = 1/1 After reduce() third = 1/1
Before reduce() third = 2/1 After reduce() third = 2/1
Before reduce() third = 129/6579 After reduce() third = 1/51
Before reduce() third = 1935/249 After reduce() third = 645/83
Before reduce() third = 1331/1651 After reduce() third = 1331/1651
Before reduce() third = 2301/1079 After reduce() third = 177/83
Before reduce() third = 3/1260 After reduce() third = 1/420
Before reduce() third = 6/198 After reduce() third = 1/33
Before reduce() third = 43/1935 After reduce() third = 1/45
Before reduce() third = 207/6579 After reduce() third = 23/731
Before reduce() third = 5/7 After reduce() third = 5/7
Before reduce() third = -25/-35 After reduce() third = 5/7
Before reduce() third = -83/1651 After reduce() third = -83/1651
Before reduce() third = 127/-1079 After reduce() third = -127/1079
Before reduce() third = 1079/1651 After reduce() third = 83/127
```

Figure 3. Commands to Compile, Link, & Run Lab 64 (Part 1 of 2)

```
newuser@csunix ~/1337/64> cat 01.dat | ./lab64 > my.out
newuser@csunix ~/1337/64> diff 01.out my.out
newuser@csunix ~/1337/64> cat 03.dat | ./lab64 > my.out
newuser@csunix ~/1337/64> diff 03.out my.out
newuser@csunix ~/1337/64>
```

Figure 3. Commands to Compile, Link, & Run Lab 64 (Part 2 of 2)