Computer Science 250

Project #3: A Rational Number Class

**Due:** Sec. 01: Wed. Dec. 13, 4:00 p.m.

 Sec. 02: Wed. Dec. 13, 5:00 p.m.

**No late programs will be accepted.**

A rational number is a number that can be represented as the quotient of two integers. Write a C++ class that can be used to represent rational numbers.

Your class should meet the following specifications:

* Rational numbers should be represented by two ints.
* It should have three constructors: A default constructor that initializes the object to 0, a constructor that takes a single int parameter N and initializes the object to the rational number N/1, and a constructor that takes two ints as parameters, representing the numerator and denominator of the rational. The third constructor should fail gracefully if the class user attempts to create a rational object with 0 as the denominator.
* It should have input and output functions with the following signatures and meanings:

/\* Read a string of the form "n/d" from the keyboard, and

 return a new rational object representing this value.

 Assume that both n and d are legal int values, and that

 there is no space between the n, the '/' and the d. \*/

Rational input();

/\* Write the object to the display in the form n/d. The

 displayed rational will be in simplest form (that

 is, neither n nor d has a common factor). If the

 rational is negative, the minus sign should appear

 before the numerator.

void output() const;

* Add the following operators as members of the class: +, -, \*, /, ^. +, -, \*, and / perform the usual arithmetic operations. In addition, overload the insertion and extraction operators (<< and >>) as friends of your class.
	+ The ^ operator is a binary operator that raises the rational number (the left operand) to an integer power (the right operand).
	+ The insertion operator should print the rational in simplest form.
	+ If the user requests an undefined operation (division by zero or 0^0), your class should fail gracefully.
* Follow the usual C++ conventions when writing your class, including:
	+ Place the class definition in the file "rational.h" and its implementation in "rational.cc".
	+ Use reference parameters, const parameters, const functions, and const return values, as appropriate.
	+ Anything that is not part of the public interface should be in the private section of the class declaration.

**What to turn in:** When you are ready to submit your class, print out a hard copy of the two files. Then, create a tarball containing these two files. No other files should be included in the tarball. Email me a copy as an email attachment. The subject of your email should be CS 250 – Project 3 - <your last name, your first name>.

**Think before you submit. When you send me your electronic submission, make sure it is the version of your class that you wish to have graded.** If you make multiple electronic submissions, I will deduct 5% of the total possible points from your final grade for each submission after the first.