PIC 10B SPRING 2013 HOMEWORK 2

Mathematical background

A rational function of a single variable x is a function $R(x) = P(x)/Q(x)$, where $P(x)$ and $Q(x)$ are polynomials, $Q(x) \neq 0$. When adding, subtracting, or multiplying two polynomials together, the resulting object is also a polynomial. However, when dividing two polynomials, the result is in general a rational function.

Polynomials P are of the form

$$P(x) = a_0 + a_1 x + a_2 x^2 + \ldots a_k x^k,$$

and rational functions R are of the form

$$R(x) = \frac{P(x)}{Q(x)} = \frac{a_0 + a_1 x + a_2 x^2 + \ldots a_k x^k}{b_0 + b_1 x + b_2 x^2 + \ldots b_m x^m},$$

where $Q(x) \neq 0$.

Assignment

We will be modifying the `Polynomial` class from HW1 to use operator overloading, and adding a new class. Make a `Polynomial` class and a `Rational` class. The `Rational` class will contain two private `Polynomial` variables, representing the numerator and denominator.

Start with the code from HW1. Here is a brief description of *new* functions to include:

- `Rational()` is the default constructor, and should initialize the rational function to 0/1 by default.
- `Rational(Polynomial p)` is the constructor with parameters that sets the private variable for the numerator polynomial equal to `p` and the denominator equal to a `Polynomial` object with value 1.
- `Rational(Polynomial pN, Polynomial pD)` is the constructor with parameters that sets the private variable for numerator equal to pN and private variable for denominator equal to pD.
- `Polynomial::evaluateAt(double x)` will evaluate the Polynomial at the numerical value x.
- `Rational::evaluateAt(double x)` will evaluate the Rational at the numerical value x.

Also please define the following non-member operators:

- $+$: `Polynomial` $\times$ `Polynomial` $\to$ `Polynomial`
- $-$: `Polynomial` $\times$ `Polynomial` $\to$ `Polynomial`
- $*$: `Polynomial` $\times$ `Polynomial` $\to$ `Polynomial`
- $+$: `Rational` $\times$ `Rational` $\to$ `Rational`
- $-$: `Rational` $\times$ `Rational` $\to$ `Rational`
- $*$: `Rational` $\times$ `Rational` $\to$ `Rational`
- $/$: `Rational` $\times$ `Rational` $\to$ `Rational`

Also define the boolean operators `==, !=, <, >` for the `Polynomial` class only. The `==` and `!=` should test whether each element of the coefficient vector is equal to the other. The

Date: April 7, 2013.

`<, >` should compare the degrees of the polynomials, so for example $100+100x < x^2$ because $2 > 1$. Note that the definition of operators $<$ and $>$ implies that they should return false if the degrees of each polynomial are equal.

- ==: `Polynomial` × `Polynomial` → `bool`
- ! =: `Polynomial` × `Polynomial` → `bool`
- <: `Polynomial` × `Polynomial` → `bool`
- >: `Polynomial` × `Polynomial` → `bool`

And define the member operators `++, --, +=, -=, *=` for both the `Polynomial` and `Rational` classes, and in addition the `/=` member operator for `Rational`. These should perform the appropriate mathematical operations. The ++ and −− operators should add 1 and subtract 1, respectively; so for a Polynomial this simply increments the coefficient indexed by 0, and for a Rational this will in effect add or subtract the denominator. The `+=, -=, *=, /=` operators should be applied to an object of the same data type, so a Polynomial += Polynomial or a Rational += Rational.

- `Polynomial++ : (NULL)` → `Polynomial`
- `++Polynomial : (NULL)` → `&Polynomial`
- `Polynomial+ = : Polynomial` → `&Polynomial`
- `Polynomial− = : Polynomial` → `&Polynomial`
- `Polynomial∗ = : Polynomial` → `&Polynomial`
- `Rational++ : (NULL)` → `Rational`
- `++Rational : (NULL)` → `&Rational`
- `Rational+ = : Rational` → `&Rational`
- `Rational− = : Rational` → `&Rational`
- `Rational∗ = : Rational` → `&Rational`
- `Rational/ = : Rational` → `&Rational`

Note: You may find yourself attempting to access private variables in the non-member operator definitions. Feel free to include additional member functions as needed in order for the functions and operators listed on the homework to function properly.

Place your code in a source file labeled *hw2.cpp*. ***If your file is not named this exactly, your homework will not be collected***. As with all programs in this course, your code should contain useful comments. In particular, your name, the date, and a brief description of what the program does should appear at the top of your source file.

What to Turn in

Place in your Submit folder the source file *hw2.cpp* with exactly this name (all lowercase, no spaces). The files will be automatically collected on Friday 4/12/13 at 5:00pm.

Grading		
Correctness	No errors, input/output correct, output presented nicely	5 points
Arithmetic	Correctly performs polynomial and rational arithmetic	5 points
Solution	Code is efficient but easy to follow	5 points
Style	Variable names, comments, indentation	5 points
	TOTAL	20 points

Note on grading: There is an automatic 5 point penalty for any homework that does not compile.

A *sample* of output is below. NOTE! The values input are just an example. When grading the homework the grader will select values to input for the coefficients, so do not just hard-code these coefficients. It is sufficient to use cin and cout for this assignment, there is no need to use getline().

Final Note about the output below. The console will automatically put in newlines if the expression runs over, so don't worry if your return carriages do not match up exactly with the output above. The coefficients need to match exactly, and there should be exactly one line between each of these expression so it is easy for the grader to read.

```
Welcome! Please input the coefficients of the first polynomial, p.
When you are finished, enter -123456789.
1
-2
0
4
-123456789

Your first polynomial is p(x) = 1-2x+4x^3.

Please input the coefficients of the second polynomial, q.
0
-1
5
0
0
0
-3
-123456789

Your second polynomial is q(x) = -x+5x^2-3x^6.

p(x)+q(x) = 1-3x+5x^2+4x^3-3x^6

p(x)-q(x) = 1-x-5x^2+4x^3+3x^6

p(x)*q(x) = -x+7x^2-10x^3-4x^4+20x^5-3x^6+6x^7-12x^9

p(x)/q(x) = 1-2x+4x^3 / -x+5x^2-3x^6

p(x)/q(x)+p(x)*q(x) = 1-2x+1x^2-8^3+45x^4-46x^5-40x^6+106x^7-42x^8+60x^9+24x^10
-120x^11+9x^12-18x^13+36x^15 / -x+5x^2-3x^6

p(x)+1 = 2-2x+4x^3

p(x)+2 = 3-2x+4x^3

(p(x)/q(x)) * (1+x^2-3x^4) = 1-2x+1x^2+2x^3-3x^4+10x^5-12x^7 / -x+5x^2-3x^6

Does p(x) equal q(x)? No.

Is p(x) < q(x)? Yes.

Press any key to continue...
```