Polynomial Abstraction

IN C++ 11

Andrew Klenotic & Dylan Stavarz

Polynomial Defined:

In mathematics, a polynomial is an expression consisting of variables and coefficients, that involves only the operations of addition, subtraction, multiplication, and non-negative integer exponents. An example of a polynomial of a single variable, x, is...

$$2x^3+5x^2 - 4x + 7$$

Representation:

For this implementation, Polynomials will be represented as a vector which will contain the coefficients for each term (as an integer) and an integer which will contain the degree of the polynomial (the highest degree or exponent of all the terms). The values in the vector will be stored so that "coefficient[0]" would be the coefficient of the x^0 term. So in the example…

$$P=2x^4+5x^2 - 4x + 7$$

The member p.degree would be 4 and the member p.coefficient would be a vector as follows…

Value	7	-4	5	0	2
Index	0	1	2	3	4

Such that any p.coefficient[n] would be the coefficient of the term of the n^{th} degree. Gaps must be represented by a zero coefficient (as in this case, the coefficient for x^3). Remember that the final term (7) is in actuality $7*x^0$.

Interface:

The Default value of a polynomial when created should be 0, which would be represented as $0*x^0$ or p.degree=0 and p.coefficient[0]=0.

There must be an ability to assign a polynomial from a string representation in the form of Poly1 = “4x^4-2x^2+5x+6” as well as from one polynomial to another as in Poly1=Poly2

Only the arithmetic operators +,-,* need to be implemented but also must allow for a string immediate as in Poly2 = Poly1 * “x^3+2x^2-2x+5” or even Poly1= “4x^4-2x^2+5x+6” + “x^3+2x^2-2x+5”. Poly2 = Poly1 * “5” would be implemented (as $5x^0$) but implementing Poly1 * 5 (an integer) need not be implemented. Boolean operators (==,!=,>=,<=,>,<) must work in an identical fashion.

Results should print through cout as a human readable polynomial expression

cout << Poly1 should yield something like “x^3+2x^2-2x+5”.

Interface (continued)

A function, peval(polynomial,integer) should be implemented that would evaluate a polynomial expression in terms of the variable x. The function should be able to accept polynomial type variables as well as string based immediates. peval("x^3+2x^2-2x+5",2) would return an integer that was the result of evaluating the polynomial "x^3+2x^2-2x+5" with x=2 resulting in (8+8-4+5) or 17.

Additionally, observers should be provided to let the user access the degree of the polynomial, as well as the coefficient of the term of the nth degree. These also should take string based immediates. In the case of the coefficient of the term of the nth degree, should n be greater than the degree of the polynomial, you would return 0 as technically "x^3+2x^2-2x+5" could be read as "$0x^{\infty}+0x^{\infty-1}+\ldots+0x^4+x^3+2x^2-2x+5$". It is not an undefined value.

This implementation will not require division, derivative or other mathematical functions though they can be implemented if you wish.

Operations

Addition/Subtraction: Addition or Subtraction of the Coefficients of the terms of like degree.

P1 -> $x^3 + 4x^2 – 3x + 6$ (Think of -3x as + (-3 * x^1) and +6 as (6 * x^0))

P2 -> $4x^3 + 2x^2 + 1x – 5$ (Again, think of -5 as + (-5 * x^0))

The Result of P1 + P2 would be: $(1+4)x^3 + (4+2)x^2 + ((-3)+1)x^1 + (6+(-5))x^0$

Which Simplifies to $5x^3+6x^2-2x+1$

Subtraction works the same way except that you subtract the coefficients.

Meaning, the Result of P1 - P2 would be: $(1-4)x^3 + (4-2)x^2 + ((-3)-1)x^1 + (6-(-5))x^0$

Which Simplifies to $-3x^3+2x^2-4x+11$

Operations

Multiplication: Each term of one polynomial (P1) is multiplied by each term of the second (P2).

P1 -> $x^3 + 4x^2 – 3x + 6$ (Think of -3x as + ($-3\ x^1$) and +6 as $+6x^0$)

P2 -> $4x^3 + 2x^2 + 1x – 5$ (Again, think of -5 as + ($-5\ x^0$)

The Result of P1 x P2 = $(x^3 * 4x^3) + (x^3 * 2x^2) + (x^3 * x) + (x^3 * -5) + (4x^2 * 4x3) + (4x^2 * 2x^2) + (4x^2 * x) + (4x^2 * -5) + (-3x * 4x^3) + (-3x * 2x^2) + (-3x * x) + (-3x * -5) + (6 * 4x^3) + (6 * 2x^2) + (6 * x) + (6 * -5)$

Which becomes: $4x^6 + 2x^5 + x^4 + (-5)x^3 + 16x^5 + 8x^4 + 4x^3 + (-20)x^2 + (-12)x^4 + (-6)x^3 + (-3)x^2 + 15x + 24x^3 + 12x^2 + 6x + (-30)$ … which simplified becomes: $4x^6+18x^5-3x^4+17x^3-11x^2+21x-30$

Boolean: P1 == P2 Returns TRUE if P1.degree == P2.degree AND P1.coefficient[n] == P2.coefficient[n] for every value of n, FALSE otherwise. P1 != P2 returns TRUE if the degrees or any given coefficients[n] are not equal.

<=,>=,<,> compare only the degrees. P1 < P2 returns P1.degree < P2.degree, etc. etc. P1 < “5” for example would return FALSE, as “5” would be interpreted as $5x^0$, and P1.degree would have to be 0 or greater (there are no negative degrees).