CS 255 – Fall 2021 - University of North Alabama

Project 2: Build Your Own Class - Polynomial

Total Project Source Code Due Date: Tuesday, October 28, 2021 at 11:59 pm.

Late Policy: You may submit a project late for up to 24 hours after the deadline with an automatic 20% reduction from the total possible points. You will have a 30-minute grace period to account for any time discrepancies. No work will be accepted after the late date.

Submission: Upload file (poly.cpp) to Canvas, for this project.

Compiler: I will compile your project in Dev C++.

Problem: You will create the implementation of a Polynomial data type.

Specifics: I will provide the Polynomial class specification and a driver program to test your Polynomial class. It is your job to fill in the methods/functions I have specified.

Data Representation:

Your Polynomial will be represented as a 21-term integer array. The index to the array represents the degree. The value at a given index is the coefficient for that degree. Since there are only 21 terms, your highest index will be 20 and your lowest 0. Therefore, your highest exponent will be 20 and your lowest 0. Let's assume we wanted to store the following polynomial: $x^{20} + 3x^{16} + 15x^4 + 2x^3 + 16x^2 + 5$. In the instance of such a Poly, our terms array would look like the following:

examplePoly.terms

Degree	*20*	*19*	*18*	*17*	*16*	*15*	*14*	*13*	*12*	*11*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*
Coeff	**1**	**0**	**0**	**0**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**15**	**2**	**16**	**0**	**5**

Operations Upon the Data:

1. Constructor: All coefficients should be initialized to zero.

2. Input: Input Polynomials in the Following format
   <Cx^D [+Cx^D]>

   Example: The Polynomial listed above would be

   <1x^20 + 3x^16 + 15x^4 + 2x^3 + 16x^2 + 5x^0>

   You may assume all Poly's I input will be in this format.

3. Output: Output Polynomials in the same format as the input.
   Bonus: You may obtain bonus points for printing the term with degree 0 without the x^0.

4. Reset: All coefficients should reset to zero.

5. Evaluate: Given x, evaluate the polynomial.

6. Addition operator: Return the sum of the two given polynomials.

7. Subtraction operator: Return the difference of the two given polynomials.

8. Addition assignment operator: Add the polynomial in the parameter to the calling object.
   Hint: If your Addition operator is overloaded correctly, this is a one-liner.

9. Derivative: Take the derivative of the calling object. The terms array will be changed after this method is called.

I will provide the poly.h and a driver, you will implement the class. You may create a driver to test this program. However, you do not have to submit it.

Comments:

Comment Header: You should have a header for each file that contains your name, the name of the file, the description of the file (for this project, you will include the project description), the course number, and the due date.

Throughout: Comment major segments of code. Comment any identifiers that do not clearly identify a variable or constant. Comment formulas. Comment any outside sources.

Functions: Use the format listed in the design plan.

Other notes:

Do not use global variables.

Keep data members private.

Do not change the Poly class without prior permission.

As applicable in any project, you should design your code in such a way that it is reusable. Make appropriate use of functions and reuse existing code where possible for your driver.