IT 179

Fall, 2021

Program 3

Linked Lists

**Objective:** Practice with linked lists

You are to develop a Java program that adds polynomials. The polynomials will be represented in your program as singly linked lists of terms. All of the polynomials will be in terms of the variable *x*.

You will write four Java classes.

**The Term Class**

Term represents a term in a polynomial. It will have two private instance variables (data fields) named coefficient and exponent. For example, 4x6 would be represented by a Term object with coefficient 4 and exponent 6. In terms of methods, this class needs

* getters for each instance variable
* a constructor that takes parameters for both instance variables
* a toString that returns the Term in the form: 4x^6
* an addTerm method that takes a Term as a parameter and returns a Term that is the sum of the calling object and the parameter. If the sum of the two Terms cannot be expressed as a single Term (because the exponents don’t match), it returns null.

**The Polynomial Class**

Polynomial is the central class of your program, representing the actual polynomial expressions. It will include the following elements:

* A private inner class Node with two instance variables termData of type Term and next of type Node. This class functions as the node class for the singly linked list representing your polynomial and should have the necessary functionality for that.
* Two private instance variables of type Node: termsHead and termsTail. These will be null if the polynomial has no terms. Otherwise, they will point to the first and last terms of the polynomial, respectively.
* The following public methods:
  + a default constructor
  + a constructor that accepts a String representing a polynomial in the form 4x^6 + 2x^3 + 1x^2 and initializes the object to represent that polynomial. We are simplifying the problem by guaranteeing spaces between terms and operators and explicitly representing coefficients of 1 (so x2 is 1x^2 rather than x^2). Note that subtraction in the polynomial will result in a negative coefficient: 3x^2 - 2 will have two terms, one with coefficient 3 and exponent 2 and one with coefficient -2 and exponent 0. There will be no negative exponents.
  + an addTermToPolynomial that adds a Term to the end of the list of terms.
  + a toString that returns a String representing the polynomial in the form shown in the sample output below.
  + an addPolynomial that accepts a Polynomial object as a parameter and returns a Polynomial object representing the sum of the calling object and the parameter. Your new polynomial must have the terms in the standard order (largest exponent to smallest) and must not include terms with coefficients of 0.
* You may add private methods as desired to improve the design of the program.

**Your fourth class** will include main and will use the other classes to write a program that allows users to enter pairs of polynomials to be added.

**Sample program run (user input in bold)**

Welcome to the Polynomial Addition Program.

Please enter your first polynomial:

**4x^3 + 2x – 7**

Please enter the second polynomial:

**-10x^4 + 2x^3 - 2x + 3**

The sum is:

-10x^4 + 6x^3 –4

Would you like to add two more polynomials?  **y**

Please enter your first polynomial:

**-6x^3 - 5x^2 + 4**

Please enter the second polynomial:

**2x^3 - 7x^2 + 3x – 4**

The sum is:

-4x^3 -12x^2 + 3x

Would you like to add two more polynomials?**n**

Thank you for using the Polynomial Addition Program.

**Notes**

* Good program design is expected. Consider using private methods to break up longer methods into smaller pieces of functionality. Make sure everything has appropriate access control (private or public).
* Good commenting is expected with correct Javadoc style comments.
* Make sure that your program runs as specified on the sample data as well as other test data which you should create yourself.
* Grading criteria:
  + MainClass(30%)
  + Term Class (25%)
  + Polynomial class with the inner class (45%)
* Deductions
* Late (-10% per calendar day)
* (-40) Syntax Errors
* (-30) Runtime Errors
* (-10) Style and Organization
* Reading material related to the assignment:
  + From Data Structures (IT179 book):
    - Chapter 1
    - Chapter 2

**Submission**

Zip your .java files and submit the .zip file to the Program 3 assignment on ReggieNet.

* + Email submissions will not be accepted
  + Follow the late policy in the syllabus
  + Corrupt files, empty files, invalid format files will result a zero