Polynomial Addition

Contributors:

- Rebekah Tippy
- Nunzio Lamartina

Design Explanation:

.

Data Structures Used:

- Term: This is used to keep track of each term's coefficient variable and exponent. It handles accessing these variables as well as printing terms in a readable way, and updating coefficients when terms are added together. Term implements Comparable so that its compareTo method can be used when sorting polynomials. The compareTo method compares Terms based on their variables and exponents.

- DataTransferObject: This is used to transfer data collected in the extractTerm method to the main method where it is used to add new terms. It records the coefficient, variable, exponent and index of the next term, and controls access to them using get methods.

- Polynomial (ArrayList): This an extension of ArrayList used to package term objects. It handles sorting the terms by using the compareTo method, combining polynomials, printing user-friendly polynomials, and combining terms with the same exponent as it adds them.

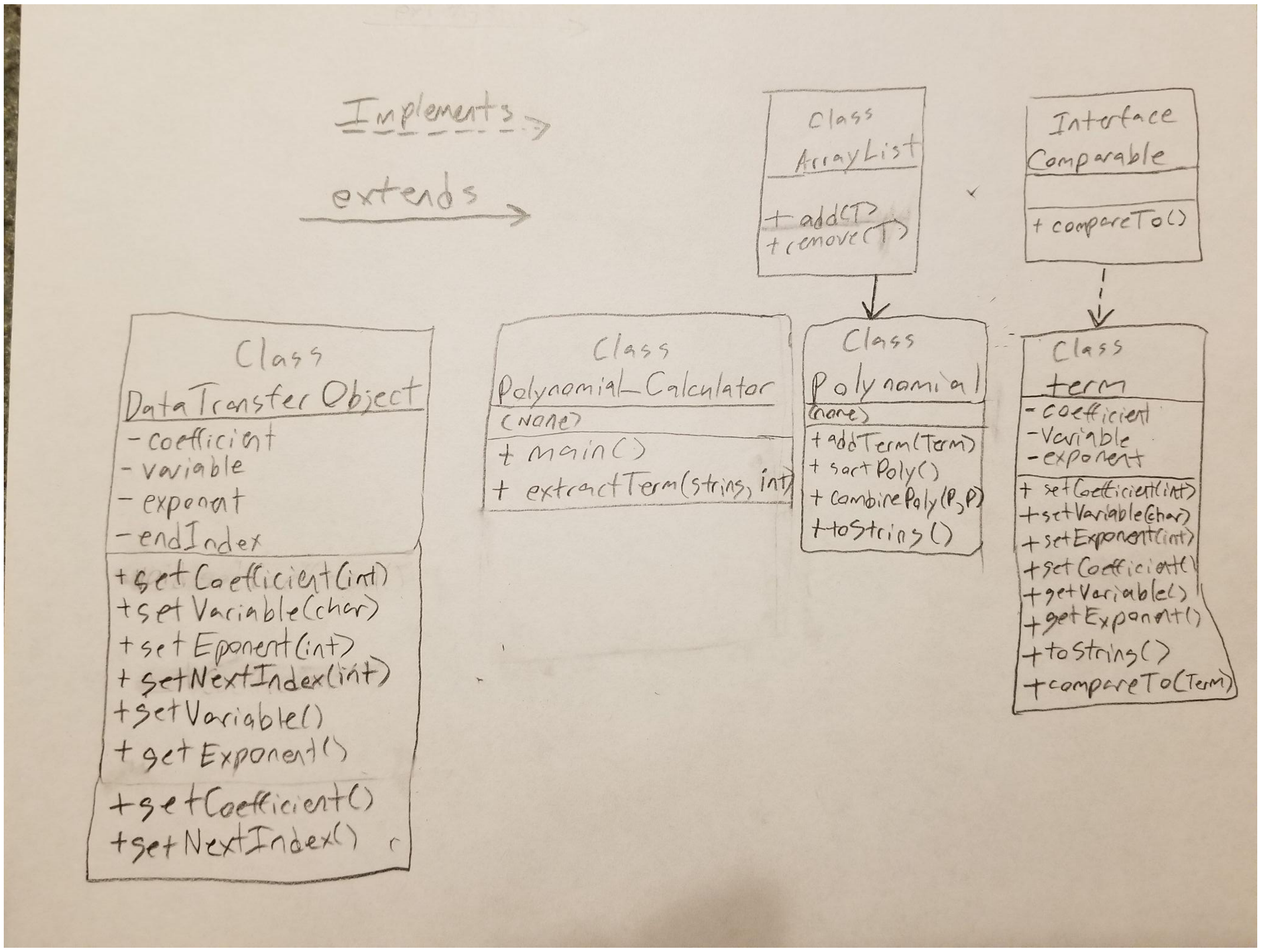
Our Polynomial addition program first presents users with a menu that allows them to enter the first polynomial, enter the second polynomial, add them together, print the result, or quit the program.

When entering polynomials the program prompts the user to enter a polynomial with no spaces, takes that polynomial string and passes it to a method called “extractTerm”, which returns a DataTransferObject. The DTO is then used to create a new Term which is added to the polynomial, using the addTerm method. (This method either combines the new term with a term already in the list which has the same exponent, or adds the term to the end of the list. It also disregards terms with a coefficient of 0.) After the term is added, extractTerm is called again, repopulating the DTO with the next term's data. The program continues to loop, adding the new term and repopulating DTO until the DTO contains an index for the next term that is too large for the polynomial’s length. Once this loop is broken, the sort method sorts the terms in the polynomial by decreasing the exponent.

When adding polynomials together the program calls the combinePoly method passing it both polynomials. This method iterates through the terms in both polynomials adding the ones with like exponents and returns a new polynomial, which is then sorted.

When printing the results the program uses the toString methods to print them in a user-friendly way.

UML Class Diagram:



Test Cases:

Test Case 1:

First Polynomial: 0x^2+x^0-x^10+x^-3+4
Second Polynomial: x-12+x^-12+6+x

Expected Output: -x^10+5+x^-3 + 2x-6+x^-12 = -x^10+2x-1+x^-3+x^-12

Actual Output: -x^10+5+x^-3 + 2x-6+x^-12 = -x^10+2x-1+x^-3+x^-12

Test Case 2:

First Polynomial: -x^1+x-x^5432+1002x^98761
Second Polynomial: x

Expected Output: 1002x^98761-x^5432 + x = 1002x^98761-x^5432+x

Actual Output: 1002x^98761-x^5432 + x = 1002x^98761-x^5432+x

Both test cases have no differences between their expected and actual output.

(note: the actual output uses superscripts)

Group Member Contributions

Rebekah Tippy

- Polynomial_Calculator Class
- Polynomial Class
- DataTransferObject Class
- Project Report

Nunzio

- Term Class
- Polynomial toString method

Future Improvements:

- The combinePoly method has a speed of O(n^2), and could use a faster algorithm if the polynomials were sorted first and then the method was called.

- Prevent the user from printing the results after changing the first or second polynomials without first adding them.