**Polynomial Operations**

**Palko Lorand-Arpad**

**30424**

1. **Objective**

Propose, design and implement a system for polynomial processing. Consider the polynomials of one variable and integer coefficients. This application should allow the user to do some specific operations on two different polynomials. The operations implemented are the following: addition, subtraction, multiplication, differentiation, division, etc.

**2. Methodology**

The polynomials will have integer coefficients. This will not cause problems for addition, substraction, multiplication and derivation. The only problem will be at integration, because after integration we will get a floating point number for the coefficient, so it’s very likely to get an error because the coefficient was declared as an integer.

When implementing, we will take in consideration more things : how do we save the polynomials, what kind of data structures we need, how do we get our input, what kind of data structures will be needed and how do we create a structure that will implement the operations clearly. The interface should be practical, so the user would not have difficulties using it.

We are introducing the polynomials in one string : ax^b+cx^d etc. This way a and c will be the coefficients, b and d will be the degrees. This way, we will have to parse the string, divide it into monomials, and then do the operations.

The sum, substraction, product and the derivative of a polynomial will be another polynomial.

**3. Polynomials in general**

Polynomials are those expressions that have variables raised to all sorts of powers and multiplied by all types of numbers. When one is working with polynomials it should be familiar with the terminology used, and few of the words one needs to feel comfortable with are the following:

**Ascending Order**  -  An ordering of terms of a polynomial in which the degree of each term is at least as large as the degree of the preceding term.

**Term** – Elements of the polynomial. Each term consists of the product of a number—called the coefficient of the term

**Binomial**  -  A polynomial with two terms.

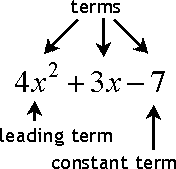
**Common Factor**  -  A factor of all the terms in an expression.

**Degree**  -  *of a monomial* is the sum of the exponents of its variables. *of a polynomial* is the largest of the degrees of its individual terms. Because x=x^1 the grade of the polynomial equals to 1.

**Descending Order**  -  An ordering of terms of a polynomial in which the degree of each term is no larger than the degree of the preceding term.

**Leading Coefficient**  -  The coefficient of the first term of a polynomial written in descending order.

**Monomial**  -  An expression with a single term; a real number, a variable, or the product of real numbers and variables. A monomial without a variable is called *monomial constant* or just *constant*. The grade of a constant is 0.



.

**Polynomial**  -  A monomial or the sum or difference of several monomials. A polynomial in a single indeterminate can be written in the form:

a_n x^n + a_{n-1}x^{n-1} + \dotsb + a_2 x^2 + a_1 x + a_0,

**Trinomial**  -  A polynomial with three terms.

For the example,

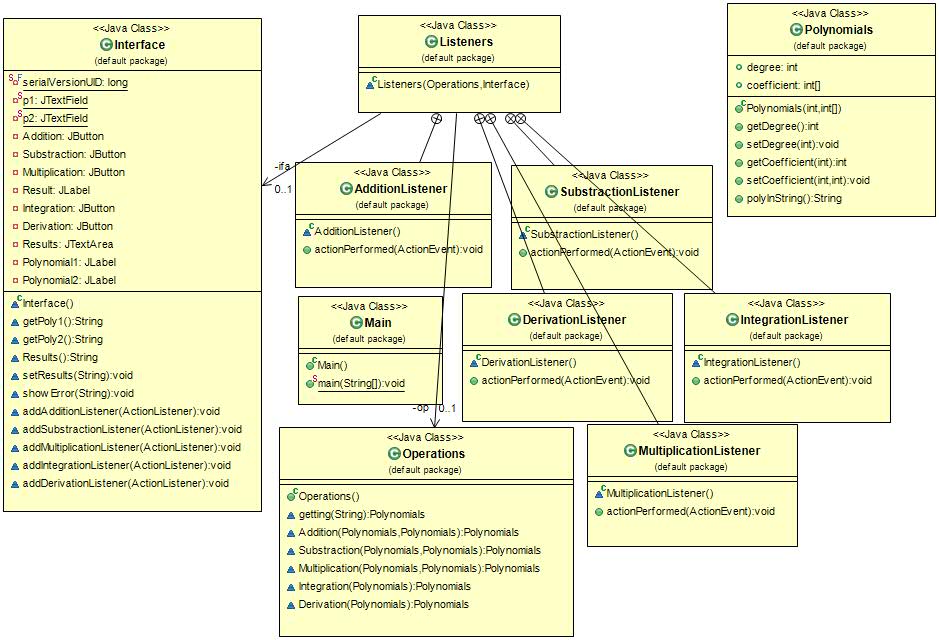
 -5x^2y\,

is a term. It has the coefficient -5, its variables are x and y, the grade of x is two and the grade of y is one. By definition of the grade of a monomial is follows that the grade of the above example is 2 + 1 = 3.

 3x^2 - 5x + 4\,.

is a polynomial. It consists of three monomials: the first has grade two, the second has grade one, and the third one has grade of zero. When a polynomial is written in its natural form, it has its terms with higher grades before the one with lower grade.

**4. Design**



The Operations is the module which handles the data of the program. It has no functionality on its own, than handling the data : storing and restoring it, but not making decisions about what to do further with that data.

The Interface is the module whose task is to display data to the user. It does not have any functionality in itself, other than what is needed to transform and lay out the data as needed by the display format. In other words, the Interface acts as an interface between the display and the rest of the program, and doesn't have any functionality related to anything else than displaying the data. It also contains getters and setters for the text fields.

The Listeneris the core module which makes all the decisions. It has all the relevant functionality of the program and interacts with the Operations and the Interface, commanding them and passing data between them as needed. It has a method actionPerformed(), which listens to Interface's button. When the method receives a button press, it changes the state of Operations. It also has methods to get references to op and ifa.

The Interface in my program is responsible for creating a connection between the user and the application, that is creating a user interface in which the user can introduce the polynomials and can click on buttons. The Listeners handles these button clicks and notifies the Operations to execute those operation when a button is pressed.

**Class Interface:** Represents the Graphical user interface of the application and extends the class JFrame which is a component of Swing. By extending this class there is an inheritance relationship between them meaning that the Interface inherits all the attributes and methods of JFrame class. The constructor of the class contains the values needed for initialization of each element. In this class we are using methods to get the polynomial from the textfields and to set the result. When the + button is pressed: Interface detects a mouse click and notifies the Listener by the method ActionListener, that the addition button was pressed. The Listener, in its ActionPerformed methods will execute the operation by calling the Operation’s methods. The object returned by these methods are polynomials.

**Class Polynomials:** Describes an object polynomial of type integer. Its attributes are the grade and coefficients.

-Polynomials(int d, int[] coef) – the constructor of class Polinom that gets as parameter the grade of a polynomial and an array of length grad thats holds the coefficients

-int getDegree() - returns an integer value that is the grade of the polynomial

-void setDegree - gets as parameter the integer value and by using this sets the grade of the current instance

-getCoefficient(i) – method to get access to the coefficient at index i. An integer type of coefficient will be returned

-setCoefficient(int i,int a) – method that has as parameter the index of the coefficient on which we want to make modification and the value by which we want to update the coefficient at i-th position

**-**polyInString()- method that has a very important role by means of interaction between the user and the logic of the application. Depending on the grade of the polynomial and concatenation with variable X this method will generate a string for displaying the polynomial in its classical representation

**Class Operations:** This class implements all the operations of polynomials. Almost every method of this class return a new object of type polynomial that represents the result of the operation.

-getting() - The method gets as parameter the string that was introduced by the user. Initially this method returns a polynomial from a string that was entered in the textfield. First we split up the string by the + sign so there will be an array of strings with the terms that were separated by +. The second for loop will split up these terms by X and ^ sign and each coefficient will be assigned to the index corresponding to its grade.

-Addition() – This method consists of a simple addition between the coefficients of the two polynomials. We start by creating an array of integers of which grade is the maximum between the two polynomials +1 then we add coefficients one by one on each position until we reach the index with the highest grade. The result will be saved in a new array.

-Substraction() – It’s practically identical with addition but before addition we multiply the coefficients of the second polynomial by -1 and then subtraction is done.

-Multiplication() – The grade of the polynomial is calculated which is the addition between the two grades. With the help of two for loops we save on the i+j position the multiplication of the two coefficients.

-Integration() – This method gets as parameter only one polynomial and returns the resulted polynomial after integration. The grade of the polynomial will be the grade of the polynomial that is in the input and each coefficient will be divided with it’s own index in the array +1.

-Derivation() – the method gets as parameter only one polynomial and returns the resulted polynomial after derivation. The grade of the resulted polynomial will be equal to the grade of the entered polynomial -1 and each coefficient is multiplied with its own position in the vector

**Class Listener:** The Listener class has in its constructor the Operations and the Interface and also has methods that perform the operations after a button has been clicked. When an addition is needed, the user input polynomials is requested from the Interface, then call the Operations to add by this polynomial then get the result from the Operations. After all has been done tell the Interface to display the result.

**5. Bibliography**

http://stackoverflow.com/