**Polynomial processing**

**Stoica Razvan Cosmin**

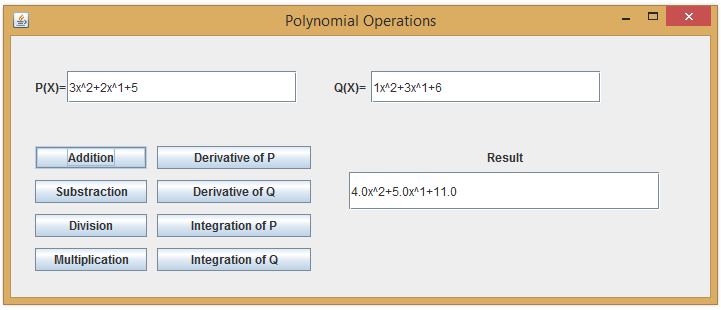
**Group 30424**

1. **Homework objective**

Propose, design and implement a system for polynomial processing, which should do the following operations: multiplication, division, addition, subtraction, derivation, integration. It should also contain a graphic interface & Junit tests.

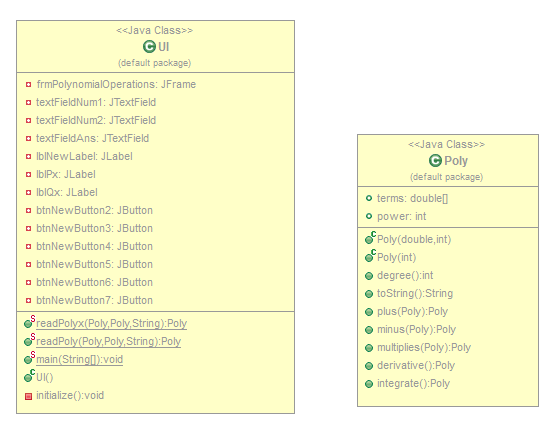
1. **Problem analysis**

The program should be as user friendly as possibly. So there should be a button for every operation, also the user should be able to introduce a polynomial in the following format: ax^2+bx^1+c. In the case in which the user introduces something wrong, there should be displayed and error, and an example of how he/she should write the polynomial.



1. **Design**

I decided to use an array of doubles (for the integration & division) to store the polynomial terms, and an integer to store the power (or the degree). In the Poly.java Class we have 2 constructors which are used to create monoms. For example: 2x^3 in terms of code will be Poly(2,3), which will give an array of terms={0,0,2,0} and will set the power=2. The other constructor is used only by the integration operation and it sets the array length based on the only the degree. For example: P(3) will give an array of terms length a+1 and set the power to 3. The degree method gets the degree of the polynomial. The toString() method is overwritten in order to display the polynomials in the correct form. In the code we have an example in order to show how it works, but the idea is that when you want to display the polynomial P, you will display it in the ax^2+bx^1+c form. The program has the following operations: plus, minus, multiplies, derivative, integrate. The plus method, has as an argument a polynomial b which will be added to polynomial a. The way the algorithm works is it will create another polynomial c which will get initially a terms array of length of the maximum of a and b that will be filled with 0’s. We use 2 loops in order to add the terms of the polynomial a and then b. After that we compute the degree using the degree method in order to get the final degree which may have changed if a term from b = -term from a and we return the polynomial c. The minus method works almost the second loop we subtract the terms from b. The multiplies method creates a polynomial c with the length of a+b powers filled with 0’s. Then we put at the we have 2 for loops in order to multiply each term, which we will add at the position i+j, and then we will compute the degree and return the polynomial c. **ex**: a=2x^2+1 ; b=3x^2+2 ;a.terms{1,0,2} power=2; b.terms{2,0,3} power=2 ;=> c.terms={0,0,0,0,0}; Step 1 in for (i=0) j=0,1,2; c[0]=1\*2+0 -> c[0]=2 | c[1]=1\*0+0 | c[2]=1\*3+0; Step 2 in for (i=1); j=0,1,2 -> c[1]=0+0\*2=0 | c[2]=3+0=3 | c[3]=0+3\*0=0; Step 3 in for (i=2); j=0,1,2 -> c[2]=3+2\*2=7 | c[3]=0+0\*2=0 | c[4]=0+2\*3=6; => c.terms={2,0,7,0,6} -> 6x^4+7x^2+2. For the derivative method we create a polynomial c with the power=the power of the given polynomial – 1. Which will create an array of size power filled with 0. Then we add at position I in a for from 0 to power-1, the term i+1 from the given polynomial multiplied by i+1. Then we will return the polynomial c, **ex:** a=2x^2+1 ; b=3x^2+2 -> a.terms{1,0,2} power=2 -> b.terms{2,0,3} power=2, in case of a-> c={0,0} power = 1 c[0]=1\*0=0 c[1]=2\*2=4 c={4,0} -> 4x^1, in case of b -> c={0,0} power =1 c[0]=1\*0=0 c[1]=2\*3=6 c={0,6} => c=6x^1. For the integrate method we create again a polynomial c, but this time of power+1, and we put at position i in a for loop from 1 to power+1 the terms from the position i-1 divided by the current position, then we return the polynomial c, **ex:** a=2x^2+1 ; b=3x^2+2 ; power=2 a.terms{1,0,2} power=2; b.terms{2,0,3} power=2; c.terms{0,0,0,0}, c[1]=1/1; c[2]=0/2; c[3]=2/3; return c={0,1,0,2/3}.



UML DIAGRAM

The user interface is pretty simple, and it contains as seen from the UML diagram, 6 buttons each doing different operations (addition, multiplication etc.), 3 labels (P(X),Q(X) and Result), and 3 text fields which are used to write the polynomials and display the result. If we write a different format then ax^2+bx^1+c, we will get the following pop-up message "Try to enter a valid number!\n For example: 3x^2+2x^1+1". I should also mention that the readPoly method is used to interpret the text given by the user, so for the given example: 3x^2+2x^1+1, it will actually read Poly(3,2)+Poly(2,1)+Poly(1,0). The way it works is: firstly in the string that we read we replace – with +-, then we split it into a parts (which is a string array) for example if we read the string -3x^2+5x^1+2, the parts array will be equal to {-3x^2,5x^1,2} , then we use a for loop of the size of the parts array and we split the parts with “x” and “^” so we get the numbers in another array named str2 (for i=0, str2 will be {3,2}) and we create a new monom with the split numbers P(3,2) and we add it to the polynomial P; there is also a case for the case in which the term has power 0, in which we just add 0 for de degree when creating the monom. Used packages (for the user interface): java.awt.EventQueue, javax.swing.JFrame, javax.swing.JButton, javax.swing.JOptionPane, java.awt.event.ActionListener, java.awt.event.ActionEvent, javax.swing.JLabel, javax.swing.JTextField.

1. **Implementation and testing**

The operations work correctly if they are written in the expected format, if not the program will throw an exception (the user will see a pop-up message that will warn him/her if he did something wrong).

1. **Results**

All the given operations work correctly except for division which I didn’t manage to implement. But for the user interface I displayed the 2 polynomials separated by the “/” sign in order to make it user friendly and to keep the design intact. Also I didn’t use the proposed OOP implementation & didn’t manage to do the JUnit Testing part.

1. **Conclusion and further developments**

As a conclusion the project needs a lot of improvements in order to be optimal and bug free, and as for further developments I think that adding the Junit tests and division operation are the most important features missing.