DOCUMENTATION

POLYNOMIAL CALCULATOR

ASSIGNMENT_1

STUDENT NAME: Maris Radu-Ioan
GROUP: 30421

CONTENTS

1. Assignment Objective 3
2. Problem Analysis, Modeling, Scenarios, Use Cases 3
3. Design 4
4. Implementation 5
5. Results 6
6. Conclusions 7
7. Bibliography 7

1. Assignment Objective

The Main Objective:

The assignment requirements are to design a polynomial calculator with an interactable graphical interface through which the user can insert polynomials, select the mathematical operation that is whished to be performed and view the result.

Sub-Objectives:

- Analyze the problem and identify the requirements.
- Make a design for the Polynomial calculator.
- Divide into smaller problems (addition, subtraction, etc.)
- Implement the smaller problems mentioned.
- Test the application.

2. Problem Analysis, Modeling, Scenarios, Use Cases

The application is meant for users that want to realize operations with polynomials. The operations that a user can solve are:

- Addition of 2 polynomials
- Subtraction of 2 polynomials
- Multiplication
- Division
- Derivative of a polynomial
- Integral of a polynomial

Successful scenario:

This case is met when the user's input is valid. This means that the polynomial that he inputs respects the following structure:

sign_coefficient_x_^_power

(where "_" is meant for visualization, representing a mergeing)

The Sign, coefficient and power are optional for input, and evem without writing them the input is consider valid.

Sign is either "+" or "-".

Coefficient is a double variable, meaning that you can write rational numbers as coefficient.

Power is a integer variable, meaning that the power is a natural number.

When writing more monoms in the polynomial, the user should write + or – without space between them. In this case, when 2 monomes that have the same power, the last inputed monome will be stored in the TreeMap

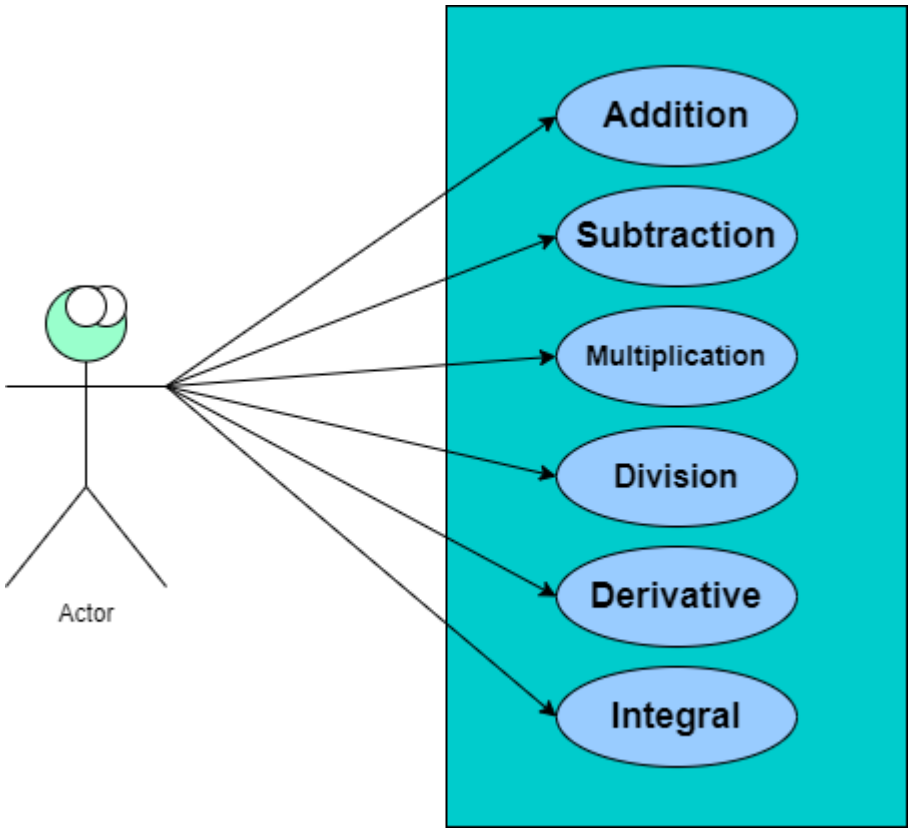
Examples of valid inputs:

| x | 2x^3 | 4x | x^7 | 5x+7x^3 |

Unsuccessful scenario:

When the input is not valid, considering the formula mentioned above, the imput is not recognized by the calculator.
Exemples of an invalid input are:

4*x^3	4f	xx

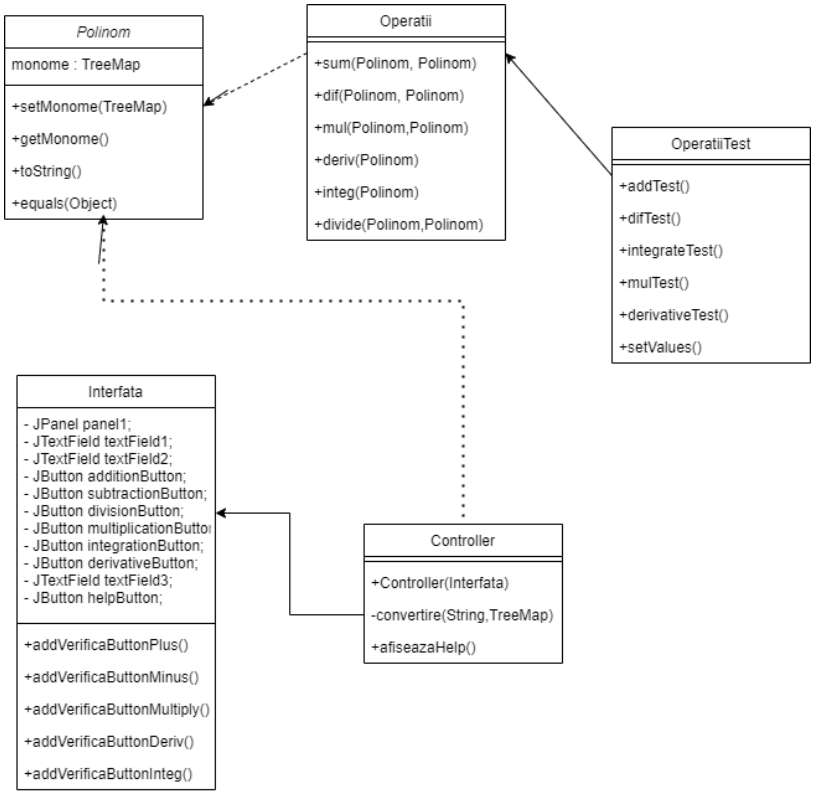


Use case diagram

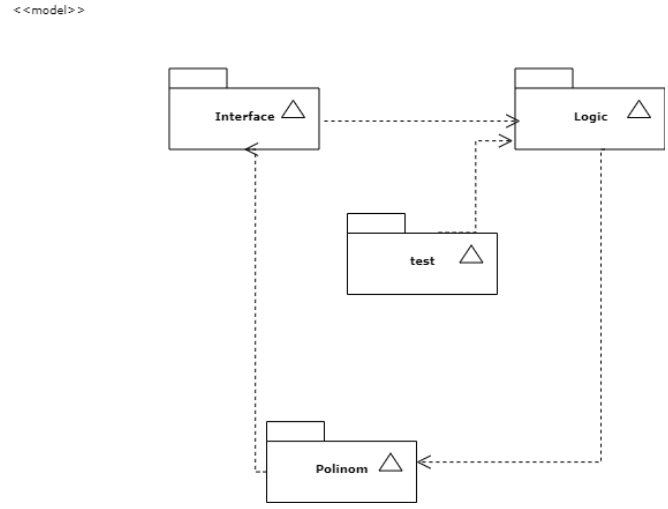
3. Design

I used an OOP design in implementing the solution, by distributing them in classes and methods. For the implementation I used 5 main classes that I distributed in 4 packages.

The Interface is made using Java Swing.
The following are the UML Diagram and Package Diagram.
UML Diagram:



Package Diagram:



4. Implementation

The Interface package contains the Java class Interface that has 3 text field and 7 buttons. 2 of the text fields are used for the input and one for the output. 6 of the 7 of the buttons are meant for executing one of the operations (addition, subtraction, etc.). The 7-th button is the HELP button witch is used to inform the user of the valid method of introducing the polynom.

The Polinom package contains the class Polinom, which is used to create the polynoms using TreeMap. The methods used are setMonome() and getMonome() in order to communicate

with the class, and a method override for equals in order to compare 2 polynoms in the OperatiiTest. The other method implemented is toString(), which converts a polynomial from TreeMap<Integer, Double> to a String which can be displayed in the text field, in the Interface.

The Test package contains the class OperatiiTest, which is meant to test if the operations are correct, using the J Unit Test. It compares the resulted output with what I had manually calculated to be correct.

The Logic package contains 2 classes: Operatii and Controller.

The Operatii class is the one in which all the main operations are implemented.The methods sum() – calculates the sum, dif() – calculates the difference, mul() – calculates the multiplication of 2 polynomials, deriv() – solves the derivative of the first polynomial, integ() – is the integral of the first polynomial and divide() – calculates the division between the 2 inputs. All the methods are static.

The Controller class is responsible for the functionality of the Interface. The method afiseazaHelp() is the one that creates a pop-up in order to inform the user. convertire() method uses Regex in order to read the input correctly and checks for the validity of the input. Also, in this class are defined multiple classes Verifica(). They are responsible for the responsiveness of the buttons.

5. Results

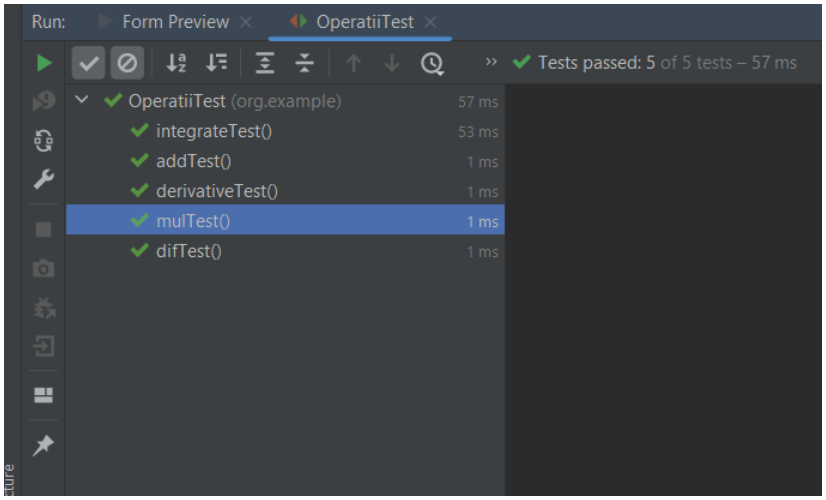
I tested each operation 3 times and they work without any problems.

I used the method setValues() to store 3 polynomials:

X= x + x^2
Y= 3x + x^3
Z= 10x^2+4

The tests that I did were:

addTest() : computes x+y , x+z and y+z
difTest() : computes x-y, y-z and z-x
mulTest(): computes x*y, x*z and y*z
derivativeTest() : computes x', y' and z'
integrateTest() : computes ∫x, ∫y and ∫z



6. Conclusions

I conclusion, I learned how to use Regex (regular expressions) and how to implement them in a java application.

As a future improvement, the calculator could have multiple fields in order to introduce 3 or more polynomials and make operations with all three of them. Also, it can have a more friendly user interface.

7. Bibliography

The references that I consulted during the implementation of the homework are the following:

1. https://docs.oracle.com/javase/tutorial/uiswing/index.html
2. https://www.baeldung.com/junit-5
3. https://google.github.io/styleguide/javaguide.html