DOCUMENTATION

ASSIGMENT NUMBER 1

Jimborean Oana Elena

Group 30422

CONTENTS

1. Objectives………………………………………………………………3
2. Analysis………………………………………………………………...3
3. Design…………………………………………………………………..7
4. Implementation…………………………………………………………8
5. Results………………………………………………………………….11
6. Conlcusions…………………………………………………………….12
7. Bibliography……………………………………………………………13

1) Objectives

* Main objective:

-To design and implement a polynomial calculator with a dedicated graphical interface through which the user can insert polynomials, select the mathematical operation to be performed and view the result

* Sub-objectives:

-Analyze the problem and identify requirements

- Design the polynomial calculator

- Implement the polynomial calculator

- Test the polynomial calculator

2) Analysis

a) Analyzing the problem

A polynomial is a mathematical expression consisting of variables, coefficients, and the operations of addition, subtraction, multiplication, and non-negative integer exponents.

A polynomial in a single indeterminate x can always be written (or rewritten) in the form



where a0,a1,a2, …, an are constants and x is the indeterminate. The exponent on an indeterminate in a term is called the degree of that indeterminate in that term; the degree of the term is the sum of the degrees of the indeterminates in that term, and the degree of a polynomial is the largest degree of any term with nonzero coefficient.

A monomial is a polynomial expression that contains variables and a coefficient, and does not contain addition or subtraction. Monomials are often called terms if they are a part of a larger polynomial.

Polynomials represent numbers, and as such, any mathematical operation can be performed on polynomials just as they are done on numbers.

b) Modelling the problem

The user can introduce two polynomials and chose which operation they want the calculator to execute. The operations that can be done on the polynomials are:

-Addition of two polynomials: is done by adding the coefficients of the monomials with the same degree

-Subtraction of two polynomials: is done by subtracting the coefficients of the monomials with the same degree

-Multiplication of two polynomials: is done by multiplying each term in one polynomial by each term in the other polynomial using the distributive law and add the powers of the same variables using the exponent rule

-Division of two polynomials

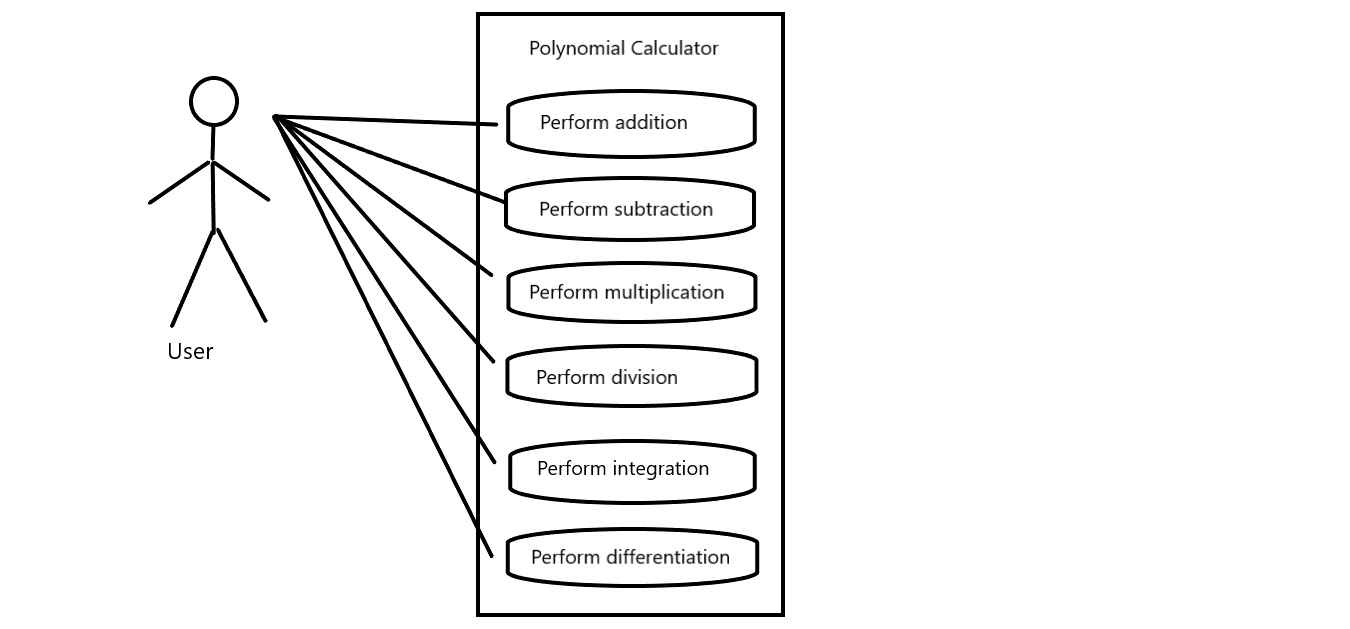
-Differentiation of a polynomial

-Integration of a polynomial

Then, the result should be displayed on the interface. If the polynomial introduced is not proper, the application will show an error message.

c) Scenarios and use cases

Use case diagram:



**Use Case**: add polynomials

Primary Actor: user

Main Success Scenario:

1. The user inserts the 2 polynomials in the graphical user interface.

2. The user selects the “addition” operation

3. The polynomial calculator performs the addition of the two polynomials and displays the result.

Alternative Sequence: Incorrect polynomials

- The user inserts incorrect polynomials (e.g. with 2 or more variables)

- The scenario returns to step 1

- If one of the fields is empty, the calculator interprets it as being 0

**Use Case**: subtract polynomials

Primary Actor: user

Main Success Scenario:

1. The user inserts the 2 polynomials in the graphical user interface.

2. The user selects the “subtract” operation

3. The polynomial calculator performs the subtraction of the two polynomials and displays the result.

Alternative Sequence: Incorrect polynomials

- The user inserts incorrect polynomials (e.g. with 2 or more variables)

- The scenario returns to step 1

- If one of the fields is empty, the calculator interprets it as being 0

**Use Case**: multiply polynomials

Primary Actor: user

Main Success Scenario:

1. The user inserts the 2 polynomials in the graphical user interface.

2. The user selects the “multiply” operation

3. The polynomial calculator performs the multiplication of the two polynomials and displays the result.

Alternative Sequence: Incorrect polynomials

- The user inserts incorrect polynomials (e.g. with 2 or more variables)

- The scenario returns to step 1

- If one of the fields is empty, the calculator interprets it as being 0 and the result of the multiplication with 0 will be 0

**Use Case**: divide polynomials

Primary Actor: user

Main Success Scenario:

1. The user inserts the 2 polynomials in the graphical user interface.

2. The user selects the “divide” operation

3. The polynomial calculator performs the division of the two polynomials and displays the result.

Alternative Sequence: Incorrect polynomials

- The user inserts incorrect polynomials (e.g. with 2 or more variables)

- The scenario returns to step 1

- If the first field is empty, the calculator interprets it as being 0 and the result of the multiplication with 0 will be 0

- If the second field is empty, the calculator interprets it as being 0 and it will generate an error

**Use Case**: differentiate polynomials

Primary Actor: user

Main Success Scenario:

1. The user inserts a polynomial in the graphical user interface.

2. The user selects the “differentiate” operation

3. The polynomial calculator performs the differentiation of the two polynomials and displays the result.

Alternative Sequence: Incorrect polynomials

- The user inserts incorrect polynomials (e.g. with 2 or more variables)

- The scenario returns to step 1

**Use Case**: integrate polynomials

Primary Actor: user

Main Success Scenario:

1. The user inserts a polynomial in the graphical user interface.

2. The user selects the “integrate” operation

3. The polynomial calculator performs the integration of the two polynomials and displays the result.

Alternative Sequence: Incorrect polynomials

- The user inserts incorrect polynomials (e.g. with 2 or more variables)

- The scenario returns to step 1

**Functional requirements:**

- The polynomial calculator should allow users to insert polynomials

- The polynomial calculator should allow users to select the mathematical operation

- The polynomial calculator should add/subtract/multiply/divide two polynomials

- The polynomial calculator should perform differentiation and integration on polynomials

- The polynomial calculator should let the user delete characters that they write

- The polynomial calculator should allow user to exit from the polynomial calculator

**Non-Functional requirements:**

- The polynomial calculator should be intuitive and easy to use by the user

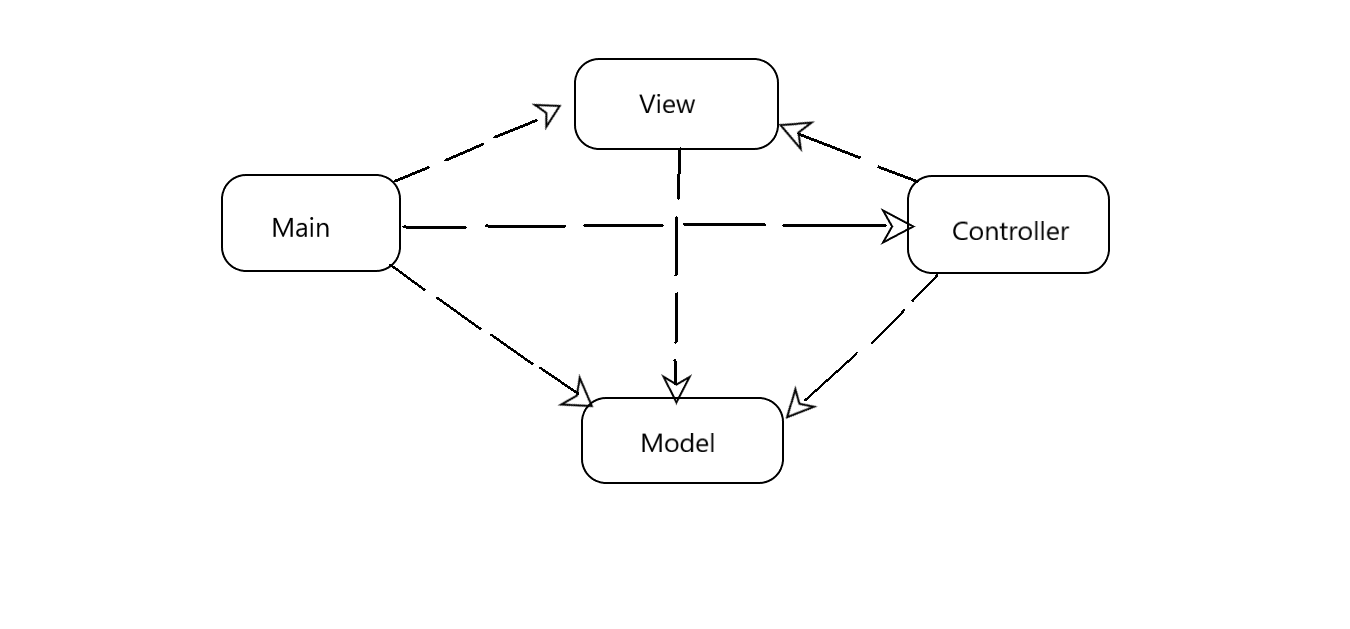
- The interface should be easy to read and understand, users can easily navigate it

- Users should easily determine what a feature is and what it can do

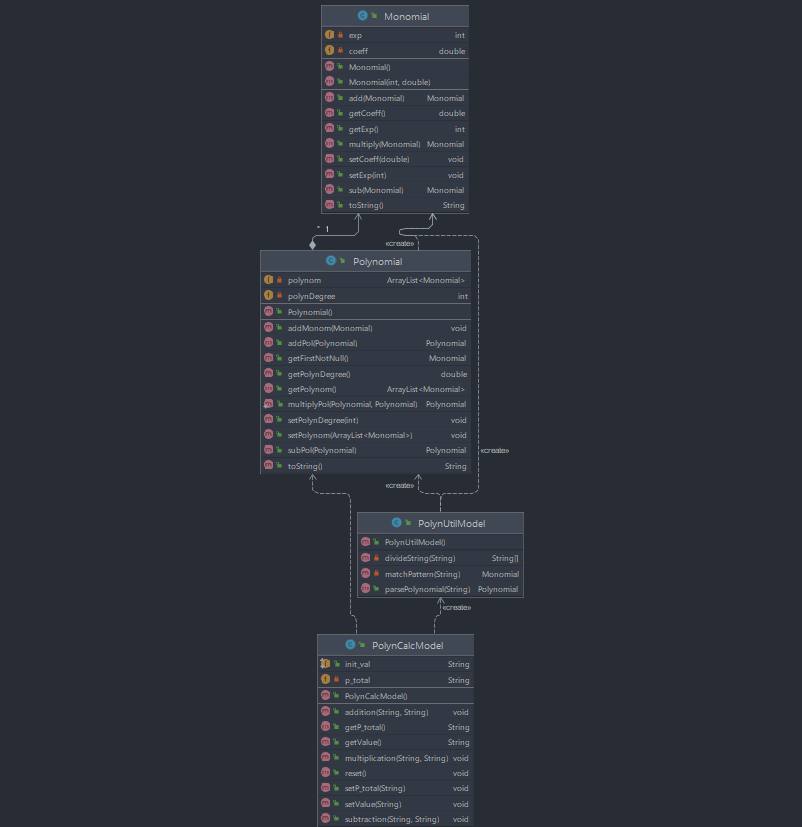
3) Design

My app is structured in 4 packages: View, Controller, Model and Main. In the “main” package I have a class “Main”, from where I launch the application.

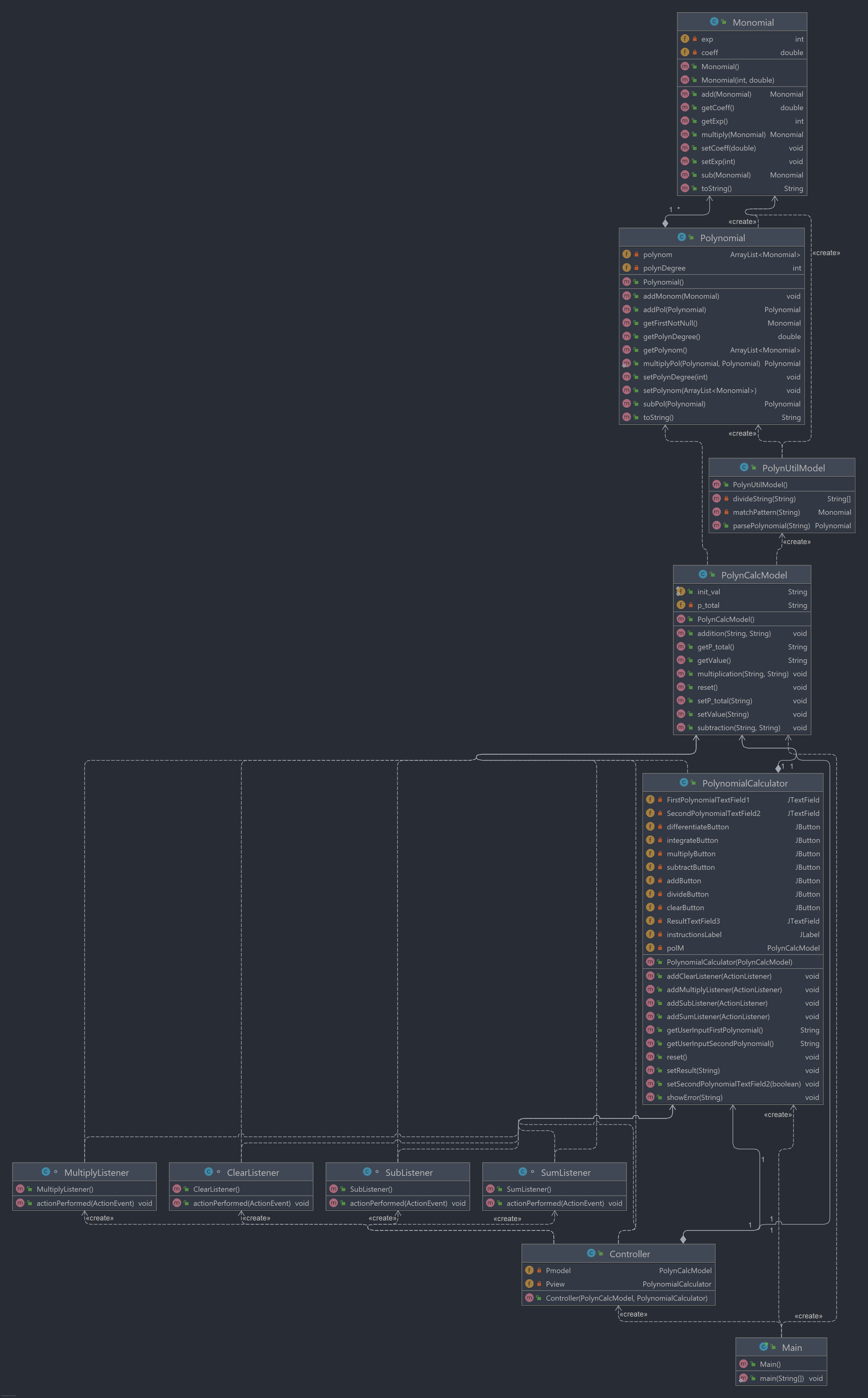
* + Package Diagram:



* + UML Diagram for classes from the package “model”:



* + Class Diagram:



The data structures with which I have been working in this problem are either primitive data types, especially integers and doubles, and ArrayList type object or new created object such as Monomial and Polynomial. I have used ArrayList instead of the classic arrays because I think that they are more efficient from the point of view of memory management.

4) Implementation

As a design pattern, I used the MVC pattern. The model-view-controller pattern proposes three main components or objects to be used in software development:

- Model, which represents the underlying, logical structure of data in a software application and the high-level class associated with it.

- View, which is a collection of classes representing the elements in the user interface

- Controller, which represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

1. In the “model” package I have the next classes: Monomial, Polynomial, PolynCalcModel, PolynCalcUtil.

a.1. Monomial class

This class has two instance variables, an int exponent and a double coefficient.

*Constructors:*

-public Monomial ( int exp, double coeff) : the constructor that initializes the monomials with the transmitted coefficient and exponent

-public Monomial() : default constructor, initializes the degree/power and the coefficient with 0

*Methods:*

-public Monomial add(Monomial m): this method has as a parameter another monomial, having the same degree, and returns a new monomial, that is obtained by adding the coefficient of the instance monomial and the coefficient of the monomial given as parameter, and the degree is the same.

-public Monomial sub(Monomial m): this method has as a parameter another monomial, having the same degree, and returns a new monomial, that is obtained by subtracting the coefficient of the instance monomial and the coefficient of the monomial given as parameter, and the degree is the same.

-public Monomial multiply(Monomial m): this method has as a parameter another monomial, and returns a new monomial, that is obtained by multiplying the coefficient of the instance monomial and the coefficient of the monomial given as parameter, and the degree is the sum of their exponentials.

- public String toString(): it is an overridden method of the classic method in which a monomial is transposed as a String for better legibility.

a.2. Polynomial class

This class has two instance variables, the polynomial’s degree given as integer and a list of monomials as ArrayList<Monomial>.

*Constructors:*

-public Polynomial() : default constructor

*Methods:*

-public void addMonom(Monomial m): it adds a parameter-transmitted monomial to the list of monomials of the instance polynomial. The method adds the monomial in descending order of the degrees of monomials from the list of monomials.

- public Monomial getFirstNotNull(): it returns the first monomial that has a coefficient not zero from the list of monomials of the instance polynomial.

- public String toString(): it is an overridden method of the classic method in which a polynomial is transposed as a String for better legibility.

-public Polynomial addPol(Polynomial p): it makes the sum between two polynomials and returns the resulting polynomial.

-public Polynomial subPol(Polynomial p): it makes the difference between two polynomials and returns the resulting polynomial.

-public static Polynomial multiplyPol(Polynomial p1, Polynomial p2): it multiplies two polynomials and returns the resulting polynomial.

a.3. PolynUtilModel class

This class has methods that help with the interaction between the graphical interface and the processing system of polynomials.

*Methods:*

- public Polynomial parsePolynomial(String input): it transposes a String into a polynomial defined as in the Polynomial class.

- private String[] divideString(String input): it divides the String into several substrings by the "+" sign, which is, in this case, the delimiter between the monomials.

- private Monomial matchPattern(String s): it verifies the form of a monomial and it is able to identify the coefficient of the monomial and its degree. It returns a monomial with the coefficient and grade identified, after the match case.

a.4. PolynCalcModel class

This class has two attributes: a String for displaying the results and a static final String(set as “”) for reset and initial values.

*Methods:*

- public void addition(String poly1, String poly2): In this method, we call the parsePolynomial method, which transforms a String intro a polynomial and then the addPol method which makes the sum of the polynomials resulting from parsing. The string for the result is put in the p\_total.

- public void subtraction(String Poly1, String Poly2): In this method, we call the parsePolynomial method, which transforms a String intro a polynomial and then the subPol method which makes the difference of the polynomials resulting from parsing. The string for the result is put in the p\_total.

- public void multiplication(String Poly1, String Poly2): In this method, we call the parsePolynomial method, which transforms a String intro a polynomial and then the multiplyPol method which multiplies the polynomials resulting from parsing. The string for the result is put in the p\_total.

1. In the “controller” package I have a single class: “Controller”.

Controller class: it has attributes of a PolynomialCalculator(from view package) object and a PolynCalcModel object.

The constructor for an instance of the Controller adds listeners for the buttons in the graphical interface. Within the class, we also define classes that implement ActionListener and that execute according to the selected operation, certain computational processes.

1. In the “view” package I have a single class: “PolynomialCalculator”.

PolynomialCalculator class: this class extends JFrame and represents the implementation of the graphical interface.

The attributes for this class are:

private JTextField FirstPolynomialTextField1 = new JTextField();

private JTextField SecondPolynomialTextField2 = new JTextField() ;

private JButton differentiateButton = new JButton("Derive");

private JButton integrateButton = new JButton("Integrate");

private JButton multiplyButton = new JButton("Multiply");

private JButton subtractButton = new JButton("Subtract");

private JButton addButton = new JButton("Add");

private JButton divideButton = new JButton("Divide");

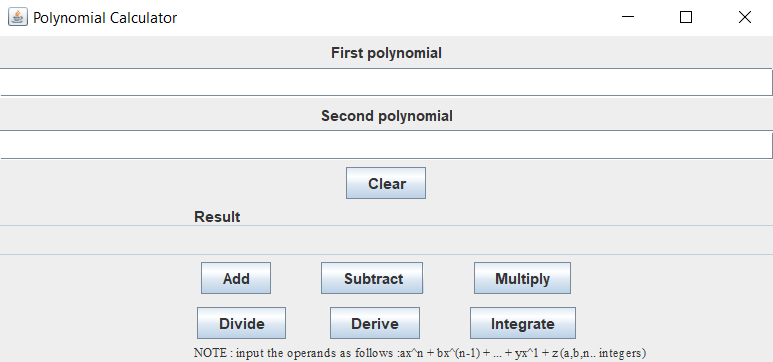
private JButton clearButton = new JButton("Clear");

private JTextField ResultTextField3 = new JTextField();

private JLabel instructionsLabel = new JLabel();

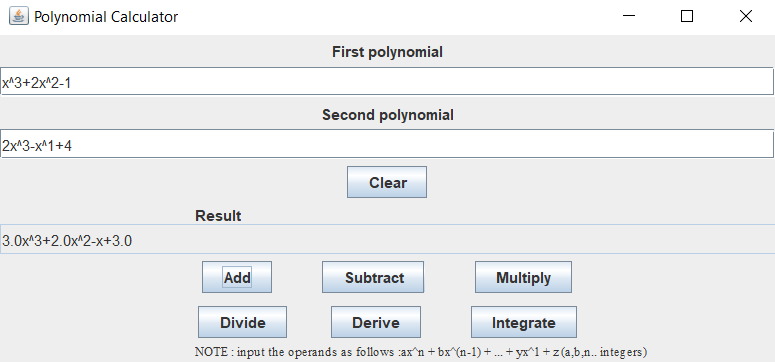
private PolynCalcModel polM;

The interface:

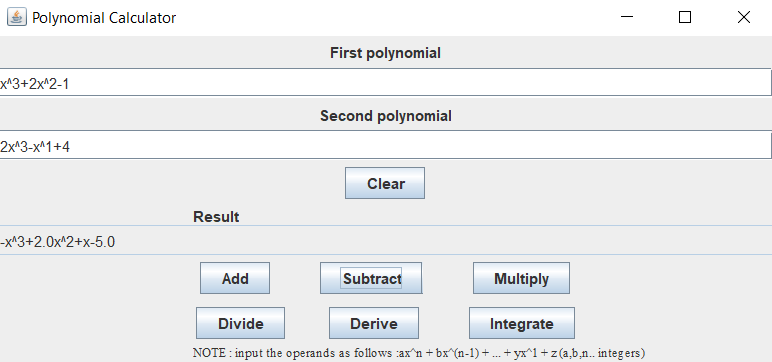


5) Results

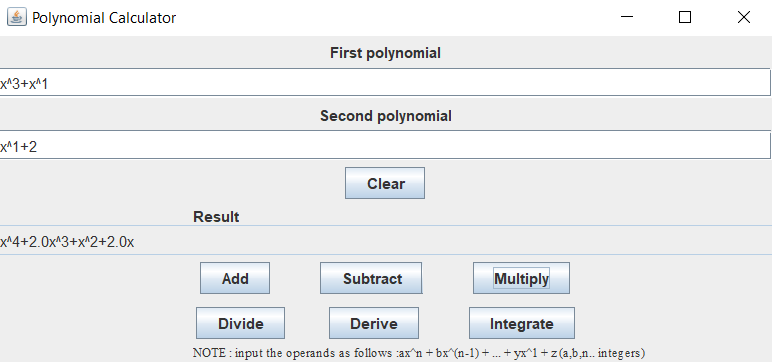
If the user choses addition:



If the user choses subtraction:



If the user choses multiplication:



If the user press “Clear”, the fields for First polynomial, Second Polynomial and Result are emptied.

6) Conclusions

This project helped in having a better understanding of the OOP concepts. Trying to make the code work by yourself and the research for new concepts is a benefit of a better learning and improving your skills.

The project can be improved, because it operates just on addition, subtracting and multiplying of two polynomials.

It is very easy for users to use the calculator, all they have to do is to insert two polynomials and click on the button with a specified operation and then the result will be displayed.

7) Bibliography

<https://dsrl.eu/courses/pt/materials/A1_Support_Presentation.pdf>

<https://stackoverflow.com/questions/44278138/adding-two-polynomials-stored-as-linked-list-in-java>

<https://en.wikipedia.org/wiki/Polynomial>

https://courses.lumenlearning.com/beginalgebra/chapter/4-2-2-adding-and-subtracting-polynomials/