***Polynomial Calculator***

***Fundamental Programming Techniques***

***Student :Soponar Darius***

***Group: 30424***

***Specialization: CTI ENGLEZA, second year***

Contents

[*The Task Objective* 2](#_Toc131085563)

[*Analise the Task* 2](#_Toc131085564)

[*Projection* 4](#_Toc131085565)

[*Implementation* 4](#_Toc131085566)

[*Rezults* 8](#_Toc131085567)

[Conclusion 11](#_Toc131085568)

[*Bibliography* 11](#_Toc131085569)

***The Task Objective***

The main objective of this task was to implement a polynomial calculator.

To be able to fulfill this objective, it was necessary to divide it into several secondary objectives, namely:



* The implementation of the polynomials from the real world into classes
* Creation of regular expressions and templates for generating polynomials from a string received as input.
* Implementation of addition
* Implementation of subtraction
* Implementation of multiplication
* Implementation of division
* Implementation of integration
* Implementation of derivation
* Creation of a graphic interface for the user of the application

***Analise the Task***

In order to understand the problem well, we must appeal to the basic knowledge accumulated in mathematics, thus we will define the following notions:

In mathematics, a polynomial is an expression constructed from one or more variables and constants, using only the operations of addition, subtraction, multiplication, and raising to constant positive whole powers.

Polynomials are built from terms called monomials, which are made up of a constant (called a coefficient) multiplied by one or more variables. Each variable can have a constant positive integer exponent. The exponent of a variable in a monomial is equal to the degree of that variable in that monomial.

A monomial without variables is called a constant monomial, or just a constant. The degree of a constant term is 0. The coefficient of a monomial can be any number, including fractions, irrational or negative numbers. A polynomial constructed with only one variable is called univariate

Properties of a polynomial are:

* The sum of two polynomials is also a polynomial
* The difference of two polynomials is also a sum, in which one of the polynomials changes the sign of its coefficients
* The product of two polynomials is also a polynomial
* Dividing two polynomials will result in two other polynomials called remainder
* The derivative of a polynomial is also a polynomial
* The integration of a polynomial is still a polynomial

***Designe***

Use (use-case)

To use the application, the user has at his disposal the application's graphic interface (GUI). When running the application, the user will see a selection button with 6 names, each representing one of the 6 characteristic operations of polynomials. When selecting any name and pressing the “calculate” button the application will return the result.

Choosing an operation that requires 2 polynomials (addition, subtraction, multiplication, division) 2 fields will appear on the screen in which the user will enter the two polynomials according to a certain pattern, for operations who use only one polynomial will write in only one of the textboxes.

***Projection***

Design decisions:

To design this application, I chose to use the Model-View-Controller design pattern in which the project is divided into 3 large packages:

* Models: here will enter the classes that model polynomials
* Views: here we will only have classes related to the graphic interface, a frame of the application is created and the position of the object in this frame is chosen
* Controllers: the application controllers will be filled here, both those that decide what will happen when the user interacts with the application's GUI, as well as the application's main controller, the one that opens the application's frame.
* UML Diagram:A picture containing diagram

  Description automatically generated

***Implementation***

Package Models:

Polinom Class:

In this class I represented a real life polynomial. I choose to represent this polynomial throw a TreeMap with an Integer variable that is the exponent and a Double variable that is the coefficient.

The Polinom class has a treemap variable, setters and getters as well as a constructor.

private TreeMap<Integer, Double> poly = new TreeMap<>();

public Polinom(TreeMap<Integer, Double> poly) {  
 this.poly = poly;  
}

PolinomService class:

In this class I created all the methods used for the implementation of the mathematical calculus of the calculator

public static Polinom parsing(String str)

* This method takes as parameter a string
* It returns a Polynomial type variable that contains a TreeMap with the exponents and coefficients of the Polynomial written as a string.

public static String reverseparsing(Polinom map)

* This method takes as parameter a Polinom
* It returns the Polynomial written back as a string

public static Polinom addition(Polinom polinom1, Polinom polinom2) {

* This method takes as parameters two Polinom type inputs
* It returns the sum of this two Polynomials as a single Polinom type variable

public static Polinom subtraction(Polinom polinom1, Polinom polinom2) {

* This method takes as parameters two Polinom type inputs
* It returns the difference of this two Polynomials as a single Polinom type variable

public static Polinom multiplication(Polinom polinom1, Polinom polinom2) {

* This method takes as parameters two Polinom type inputs
* As for the two methods above we declare another Polinom and we give it the value of one of the two parameters and then we execute the given operation with the other one.

public static Polinom derivation(Polinom polinom) {

* This method takes only one Polinom as parameter
* It returns the derivative form of the polynomial given as a parameter

public static Polinom integration(Polinom polinom) {

* This method takes only one Polinom as parameter
* It returns the Integrated form of the polynomial given as a parameter

View Package:

The View class:

This class represents the transposition of the GUI of the application into code. All the components with which the user will interact while using the application are described here.

The components used in this class are:

* 4JTextFile type variables, two for the polynomials used as input, one for the result and one for the rest from division
* 4 JLabels for the label of each JText
* A JcomboBox for the selection
* A JButton for the calculation button that generates the result

All components are declared as private variables of this class and have created getters and stters to be used further in the controller package. Buttons are also created with methods to add ActionListeners, and these ActionListeners are described in the controllers package. Also in this class, the size of the application frame is set, as well as the elements that make up this frame, as well as their positioning in the frame. The frame is set so that it cannot be resized, because this is how I considered this size to be sufficient.

**The Controller Package:**

ViewController Class:

This class is meant to describe what will happen when the user interacts with the application, when they press application buttons, or enter data, and so on.

He can perform the following actions:

* If the user selects “Addition” from the JComboBox and presses calculate then in the JTextField of result the result of tis operation will appear.
* If the user selects “Subtraction” from the JComboBox and presses calculate then in the JTextField of result the result of tis operation will appear.
* If the user selects “Multiplication” from the JComboBox and presses calculate then in the JTextField of result the result of tis operation will appear.
* If the user selects “Division” from the JComboBox and presses calculate then in the JTextField of result the result of tis operation will appear.
* If the user selects “Integration” from the JComboBox and presses calculate then in the JTextField of result the result of tis operation will appear.
* If the user selects “Derivation” from the JComboBox and presses calculate then in the JTextField of result the result of tis operation will appear.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

I think that in this topic I learned to work much better with Java Treemaps and I learned to use regex expressions, which I have not used at all until now. As development possibilities, I could list the following: improving the design (by choosing more suitable colors, by stylizing the buttons), transforming the application from a desktop application into a web application, linking the application to a database (as the user to be able to see what operations he has performed in the past, a short history), or you could add a window in which the operation is performed step by step (the user would not only see the result, he would also see the steps that must be followed to reaches the desired result), another improvement that I could add to the project in the future would be to achieve simplification in the case of integration, if possible.

Bibliography

<https://stackoverflow.com/questions/52824866/java-polynomial-multiplication>

<https://www.w3docs.com/snippets/java/printing-hashmap-in-java.html#:~:text=To%20print%20the%20contents%20of,print%20the%20keys%20and%20values.&text=Alternatively%2C%20you%20can%20use%20the,the%20values%20for%20each%20key>

<https://www.geeksforgeeks.org/hashmap-getordefaultkey-defaultvalue-method-in-java-with-examples/>