DOCUMENTATION

ASSIGNMENT 1

STUDENT NAME: Teodora Huluban
GROUP: 30222

CONTENTS

1. Assignment Objective

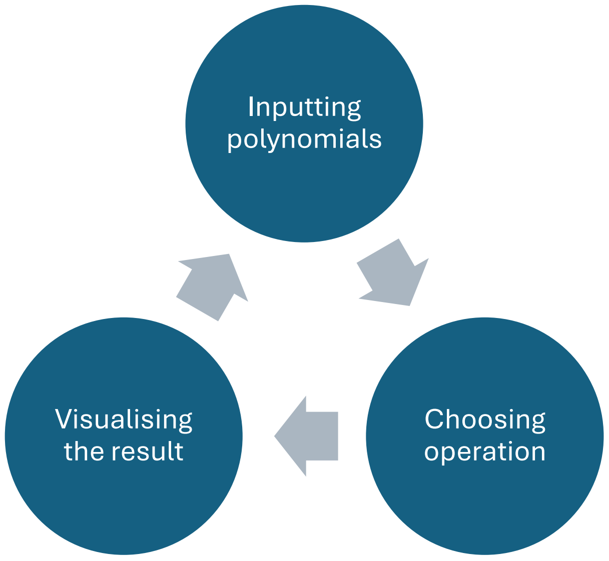
The main objective is to implement a polynomial calculator capable of performing six operations: addition, subtraction, multiplication, division, differentiation, integration; with a dedicated graphical interface through which the user can input the polynomials, select the operation that will be performed, and visualize the results.

The following sub-objectives were acquired in order to achieve the main objective:

- Analyze the problem and identify requirements
  - an analyze on the problem was done, and requirements were identified (detailed at point "2. Problem Analysis…")
- Design the polynomial calculator
  - a package/class diagram was realized for designing the project (detailed at point "3.Design")
- Implement the polynomial calculator
  - four classes were implemented: one for modeling the polynomial, one for operations, one for converting to polynomial, one for graphical interface(detailed at point "4.Implementation")
- Test the polynomial calculator
  - JUnit testing was done (detailed at point "5. Results")

2. Problem Analysis, Modeling, Scenarios, Use Cases

The polynomial calculator should be capable of performing the six operations needed (addition, subtraction, multiplication, division, differentiation, integration) and requires a graphical interface that lets the user input one or two polynomials and can choose from the six different operations. The results will also be displayed through the graphical interface. The system has to be able to validate the input, and a message of error should be displayed if the input is not a valid polynomial.



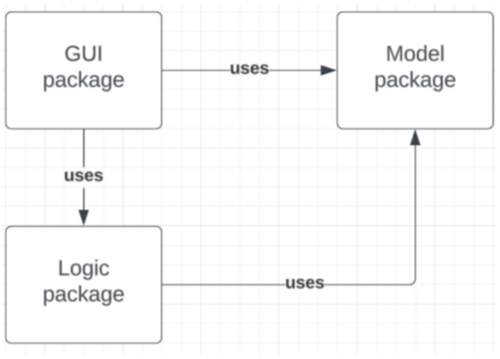
3. Design

The project was structured in three packages:

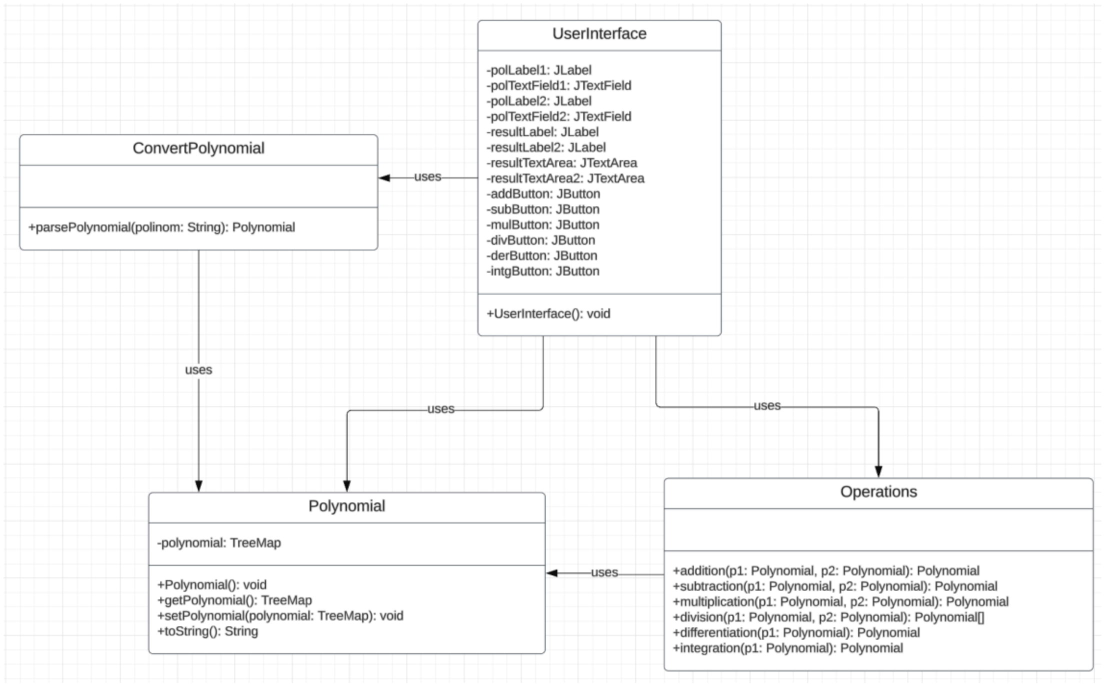
- GUI package containing the UserInterface class - manages the interaction between the user and the calculator
- Model package containing the Polynomial class - manages the modeling of the polynomial
- Logic package containing the Operations class, and the ConvertPolynomial class - holding the methods which performs the operations and the method which converts a string to a polynomial
- Test package containing the OperationsTest class - runs JUnit testing for verifying the operations

For modeling the polynomial, I used TreeMap structure, where the power was stored as the key, and the coefficient as the value

Package Diagram:



Class Diagram:


UserInterface
-polLabel1: JLabel
-polTextField1: JTextField
-polLabel2: JLabel
-polTextField2: JTextField
-resultLabel: JLabel
-resultLabel2: JLabel
-resultTextArea: JTextArea
-resultTextArea2: JTextArea
-addButton: JButton
-subButton: JButton
-mulButton: JButton
-divButton: JButton
-derButton: JButton
-intgButton: JButton
+UserInterface(): void
ConvertPolynomial
+parsePolynomial(polinom: String): Polynomial
uses
uses
uses
uses
Polynomial
-polynomial: TreeMap
+Polynomial(): void
+getPolynomial(): TreeMap
+setPolynomial(polynomial: TreeMap): void
+toString(): String
uses
Operations
+addition(p1: Polynomial, p2: Polynomial): Polynomial
+subtraction(p1: Polynomial, p2: Polynomial): Polynomial
+multiplication(p1: Polynomial, p2: Polynomial): Polynomial
+division(p1: Polynomial, p2: Polynomial): Polynomial[]
+differentiation(p1: Polynomial): Polynomial
+integration(p1: Polynomial): Polynomial

4. Implementation

Polynomial Class:
- models the polynomial through a TreeMap where the power is stored as the key and the coefficient as the value
- contains the toString method which transforms the polynomial into a string

ConvertPolynomial Class:
- contains the parsePolynomial method witch transforms a string into a polynomial. It is used in the graphical interface, for mapping the inputted string to a TreeMap

Operations Class:
- contains six methods, each representing an operation
- addition method: returns the addition between two polynomials sent as arguments
- subtraction method: returns the subtraction between two polynomials sent as arguments
- multiplication method: returns the product of two polynomials sent as arguments
- division method: returns the quotient and the remainder from the division between two polynomials sent as arguments
- differentiation method: returns the derivative of the polynomial sent as argument
- integration method: returns the integral of the polynomial sent as argument

UserInterface Class:
- represents the graphical interface through which the user interacts with the polynomial calculator
- contains two text fields for inputting the polynomials, two text areas for outputting the results, the second one is used for the remainder from the division, or for differentiating/integrating both polynomials; six buttons, one for each operation and action listener for each of them

5. Results

All the manual-input tests done until now have passed. In addition, JUnit testing was done, two for each operation, and two for parsing the polynomial, all tests being successful.

6. Conclusions

To conclude, the polynomial calculator provides a range of essential functionalities for polynomial manipulation and calculation. The implementation is done using well-structured classes and methods, allowing for easy usage and flexibility across various use cases.

The Map interface was found to be very useful for this project, especially for modeling the polynomials, because it is known that the only important parts in representing a polynomial are the power and the coefficient of each monomial.

From this point on, more features can be added: floating point coefficients for the input, the possibility for the input to be more flexible, other operations could be performed, and the graphical interface could be improved further.

7. Bibliography

https://docs.oracle.com/javase/tutorial/uiswing/index.html

https://www.baeldung.com/junit-5