**DOCUMENTATION**

**1st Assignment**

Student Name: Nablsiy Ahmad

Group: 30228

CUPRINS

1. Objective of the assignment ……………………………………………...3

2. Analyzing the problem …………………………………………………..3

3. Designing…………………………………………………………………4

4. Implementation…………………………………………………………...5

5. Result……………………………………………………………………..7

6. Conclusion………………………………………………………………..7

7. Bibliography……………………………………………………………...7

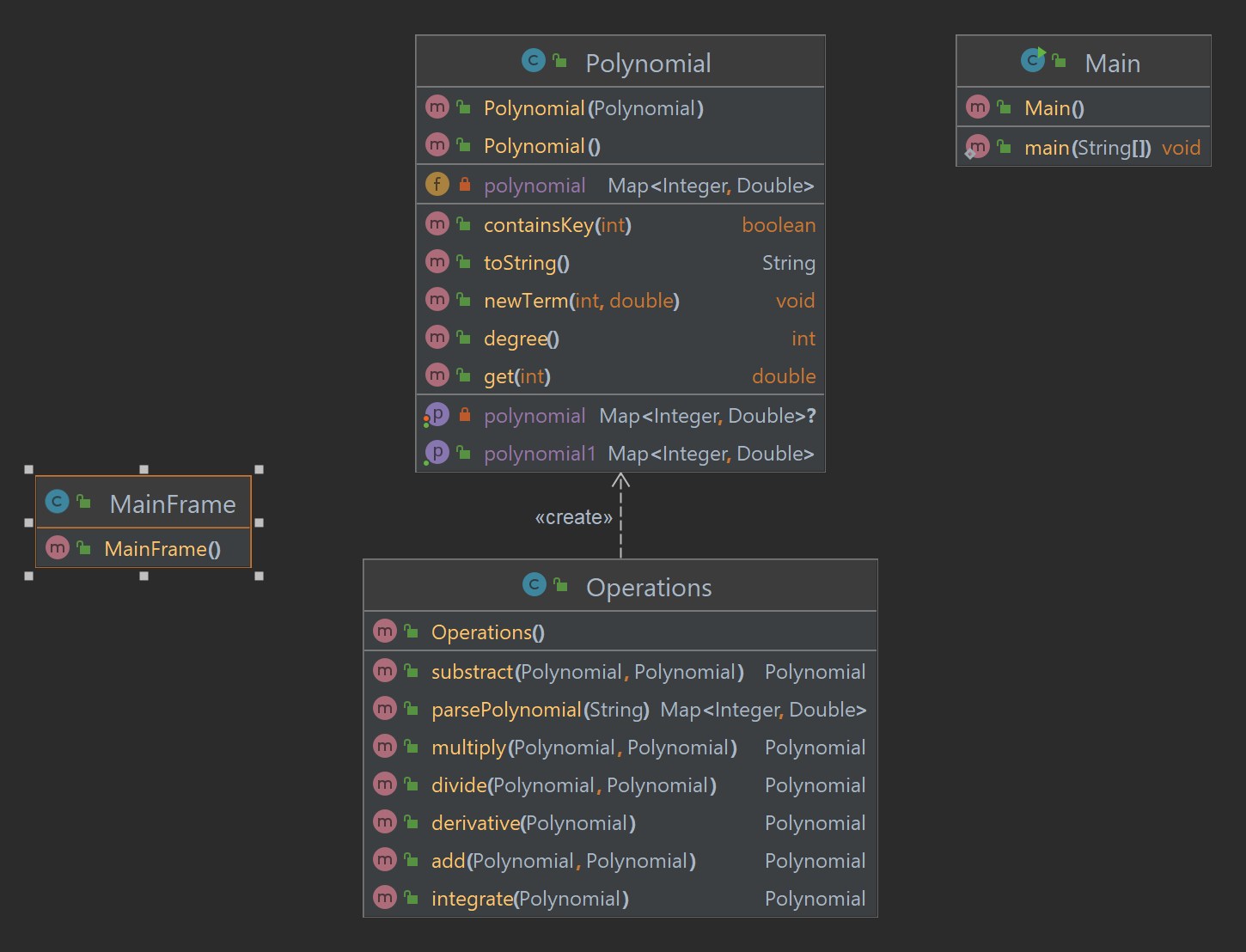
* **Main objective of this assignment:**

It is to implement in JAVA language a polynomial calculator which has a suitable interface where the user can introduce the two polynomials by typing on the keyboard and then selecting an operation to be applied on the polynomials, like(sum, sub, multiply, divide, derivate, integrate).

* **Sub-objectives:**

1. Analyzing the problem and identifying the requirements:
   1. *functional requirements:*

* the calculator should allow to user to introduce two polynomials.
* The calculator should allow the user to choose the operation applied on these polynomials.
* The calculator should be able to apply the sum operation on the introduced polynomials and give the result.
* The calculator should be able to subtract one polynomial from another and give the result.
* The calculator should be able to multiply the two polynomials together and give the result.
* The calculator should be able to apply the division operation on the introduced polynomials and give the result.
* The calculator should be able to integrate a polynomial.
* The calculator should be able to derivate a polynomial.
  1. *non-functional requirements:*
* The calculator should be easy for the user to use.
* The calculator should have a clear and interactive interface.
* The calculator should display an error message once the user don’t introduce a correct format polynomial.
* The calculator should be precise and perform the arithmetic operations.
* **Designing:**

****

After understanding the project and analyzing how we should build this polynomial calculator, I found out that first of all, in order to design this calculator we need to create a class which constructs the polynomials, a class which builds the operations applied on the polynomials, a class that creates the suitable interface for our calculator and definitely the Main class where we run the whole project.

The design of this calculator was made according to my understanding of how this will work out.

It is simple and clear, the operations Class implements methods using the polynomials constructed in the polynomial Class, so without the polynomial class and its methods the operations class cannot be built. The Mainframe Class uses the GUI in order to implement the calculator interface and this class contains buttons the also apply the operation methods from the class operations so there buttons can work out according to the logic implementation in the Operations Class.

The main Class is where we run this calculator by using an object created by the MainFrame Class.

* **Implementaion:**

**The Polynomial Class** is mainly based on a constructor that initializes an empty TreeMap object as the polynomial map.

The polynomial function is stored as a Map from the degree of each term to its corresponding coefficient.

The class has several methods:

*setPolynomial(Map<Integer, Double> polynomial):* Setter method to set the polynomial of the object.

*getPolynomial1():* Getter method to get the polynomial of the object.

*containsKey(int degree):* Returns true if the polynomial contains a term with the specified degree.

*get(int degree):* Returns the coefficient of the term with the specified degree.

*degree():* Returns the highest degree of the polynomial.

*newTerm(int degree, double coefficient):* Adds a new monomial to the polynomial or updates the coefficient of an existing monomial.

*toString():* Returns a string representation of the polynomial in a human-readable format.

*getPolynomial():* Private getter method that returns the polynomial map.

**The Operations class** is a Java class that contains various methods for performing arithmetic operations on polynomials. Here's a breakdown of the different methods:

*add(Polynomial p1, Polynomial p2):* This method takes two Polynomial objects as input and returns their sum as a new Polynomial object.

*substract(Polynomial p1, Polynomial p2):* This method takes two Polynomial objects as input and returns their difference as a new Polynomial object.

*multiply(Polynomial p1, Polynomial p2):* This method takes two Polynomial objects as input and returns their product as a new Polynomial object.

*divide(Polynomial p1, Polynomial p2):* This method takes two Polynomial objects as input and returns their quotient as a new Polynomial object. It uses polynomial long division to perform the division.

*integrate(Polynomial p1):* This method takes a Polynomial object as input and returns its integral as a new Polynomial object.

*derivative(Polynomial p1):* This method takes a Polynomial object as input and returns its derivative as a new Polynomial object.

*parsePolynomial(String polynomialString):* This method takes a String representation of a polynomial as input and returns a Map<Integer, Double> object that represents the polynomial.

The Polynomial class is likely a custom class that contains a Map<Integer, Double> object that represents the polynomial as a mapping between exponents and coefficients. The newTerm(int exponent, double coefficient) method likely adds a new term to the polynomial by updating the map.

The Pattern and Matcher classes from the java.util.regex package are used in the parsePolynomial method to parse a String representation of a polynomial into its constituent terms. The TreeMap class from the java.util package is used to store the resulting polynomial in parsedPolynomial as a sorted map.

**The MainFrame Class** is the GUI application. The application allows the user to enter two polynomials, and then perform various operations on them. The application includes six buttons for performing different operations on the polynomials, and a text area for displaying the result of the operations.

The class extends JFrame and contains several GUI components such as JTextFields, JLabels, and JButtons. It also has two fields named poly1 and poly2 of type Polynomial. These fields will store the polynomial objects created from the user input.

The constructor of the MainFrame class sets up the JFrame properties such as size, title, background color, etc. It then creates and adds the GUI components to the JFrame. The buttons have action listeners added to them that will perform the corresponding operations when clicked.

The operations that can be performed on the polynomials include addition, subtraction, multiplication, division, derivation, and integration. However, the action listeners for these buttons are currently empty, so the actual implementation of these operations is missing.

Overall, the class provides the basic framework for a polynomial calculator GUI application but requires further implementation to be fully functional.

**The Main class** contains the main method which creates a new MainFrame object and sets its size to 700x700 where we run the programm by using this object. Then, the program creates an Operations object and several Polynomial objects in order to apply some arithmetic operations on the polynomials, so i can verify the results and see if the operations are working correctly or no.

* **RESULT:**
* **Conclusion:**

Implementing an application by using JAVA makes things more easier for us. An application like the Polynomial Calculator is a very helpful program in the practical life. This app can be devloped more to have more features(like to save the results of each operation for a limit number of operations) and apply more operations(like root, limit x---> ....etc) which will lead us to have more efficient program.

* **Bibliography:**

[**https://dsrl.eu/courses/pt/**](https://dsrl.eu/courses/pt/)

<https://youtu.be/4MeC_tO20Sk>

<https://youtu.be/4sVJuH0NdSw>

<https://youtu.be/3DDLJOExiWM>