DOCUMENTATION

ASSIGNMENT *ASSIGNMENT_1*

STUDENT NAME: OLARU SEBASTIAN-TEODOR
GROUP: 30421

CONTENTS

1. Assignment Objective

(i) Main objective of the assignment: Develop a polynomial calculator using Java with OOP principles and a GUI.

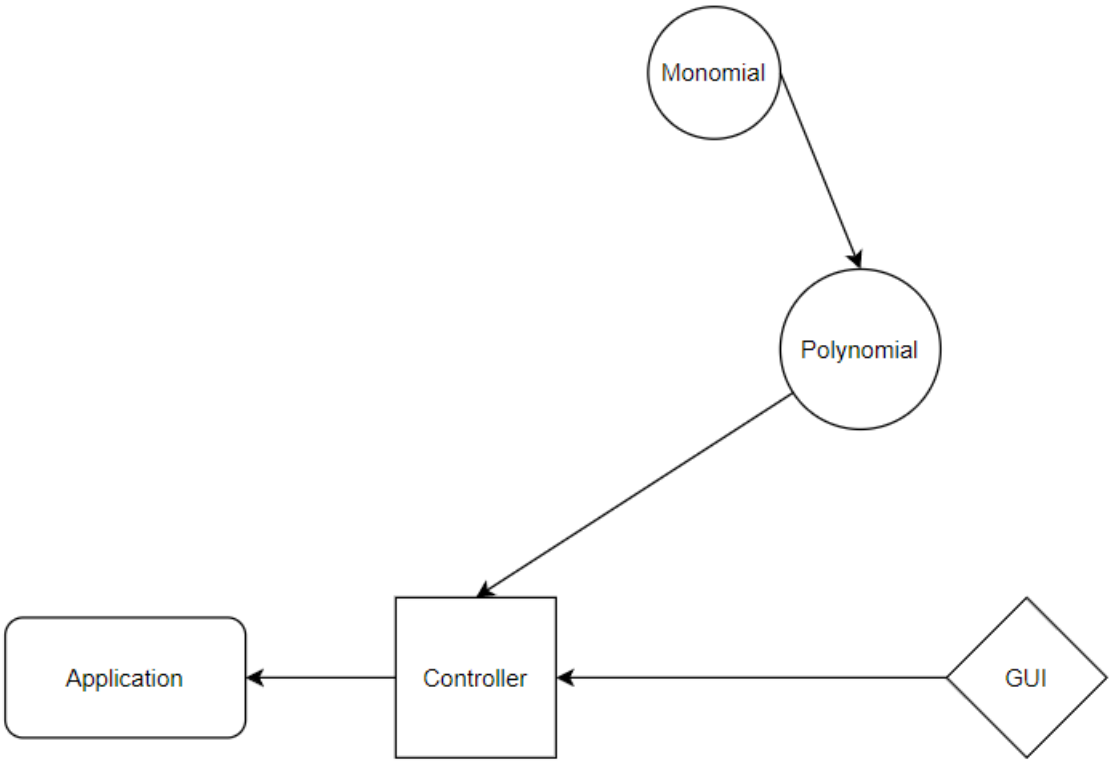
(ii) Sub-objectives:

Sub-objective	Description
Implement OOP principles	Design the polynomial calculator using OOP principles such as encapsulation and defining appropriate classes
Use Map for polynomial modeling	Implement Map for efficient storage and retrieval of polynomial terms
Use foreach instead of for loop	Use the foreach loop instead of the traditional for loop for concise and readable code
Implement a graphical user interface	Implement a GUI using either Java Swing or JavaFX for user input and operation display
Implement polynomial addition and subtraction	Implement methods to perform polynomial addition and subtraction
Keep classes and methods organized	Keep classes and methods to a maximum of 300 and 30 lines, respectively, for organized and readable code
Use Java naming conventions	Follow the Java naming conventions for classes, methods, and variables for consistent and readable code
Provide good quality documentation	Provide clear and concise documentation for the polynomial calculator
Organize source code	Keep the source code organized and easy to read for maintainability and understanding
Implement polynomial multiplication and division	Implement methods to perform polynomial multiplication and division
Implement polynomial differentiation and integration	Implement methods to perform polynomial differentiation and integration
Use regular expressions and Pattern matching	Use regular expressions and pattern matching to extract polynomial coefficients for efficient parsing of user input

Sub-objective	Description
Use Junit for testing	Use Junit for unit testing of the polynomial calculator

Overall, the sub-objectives represent the necessary steps to achieve the main objective of developing a polynomial calculator using Java with OOP principles and a GUI. These steps cover the implementation of the calculator itself, as well as considerations for code organization, documentation, and testing.

2. Problem Analysis, Modeling, Scenarios, Use Cases



1. Input Polynomial Description: The user inputs a polynomial through the GUI. Steps:
   - The user enters the polynomial expression in the input field.
   - The user selects the operation they wish to perform.

- The user presses the "Calculate" button.

2. Perform Addition Description: The calculator adds two polynomials together. Steps:
   - The user inputs two polynomials.
   - The user selects the "Add" operation.
   - The calculator adds the two polynomials together and displays the result in the output field.
3. Perform Subtraction Description: The calculator subtracts one polynomial from another. Steps:
   - The user inputs two polynomials.
   - The user selects the "Subtract" operation.
   - The calculator subtracts one polynomial from the other and displays the result in the output field.
4. Perform Multiplication Description: The calculator multiplies two polynomials together. Steps:
   - The user inputs two polynomials.
   - The user selects the "Multiply" operation.
   - The calculator multiplies the two polynomials together and displays the result in the output field.
5. Perform Division Description: The calculator divides one polynomial by another. Steps:
   - The user inputs two polynomials.
   - The user selects the "Divide" operation.
   - The calculator divides one polynomial by the other and displays the result in the output field.
6. Perform Differentiation Description: The calculator differentiates a polynomial. Steps:
   - The user inputs a polynomial.
   - The user selects the "Differentiate" operation.
   - The calculator differentiates the polynomial and displays the result in the output field.
7. Perform Integration Description: The calculator integrates a polynomial. Steps:
   - The user inputs a polynomial.
   - The user selects the "Integrate" operation.
   - The calculator integrates the polynomial and displays the result in the output field.
8. Display Result Description: The calculator displays the result of the operation in the output field. Steps:
   - The calculator performs the selected operation on the input polynomials.
   - The calculator displays the result of the operation in the output field.
9. Display Error Message Description: The calculator displays an error message for invalid inputs. Steps:
   - If the input polynomials are invalid (e.g. contain non-numeric characters), the calculator displays an error message in the output field.

3. Conclusions

In this assignment, we have designed and implemented a polynomial calculator using Java and OOP principles. The calculator allows users to perform basic arithmetic operations on polynomials, such as addition, subtraction, multiplication, and division, as well as differentiation and integration. We have also implemented a graphical user interface (GUI) using Java Swing.

What we learned:

Through this assignment, we have gained experience in applying OOP design principles, such as encapsulation and defining appropriate classes, to solve a real-world problem. We have also learned how to use Java Swing to create a GUI and handle user input. Additionally, we have improved our skills in implementing algorithms and using data structures, such as the Map data structure to model a polynomial.

Future developments:

In the future, we could further improve the functionality of the polynomial calculator by adding support for more advanced operations, such as finding roots or solving systems of polynomial equations. We could also enhance the GUI by adding more customization options for users, such as the ability to change the color scheme or display options for the output. Additionally, we could implement error handling to prevent invalid input or operations on polynomials that do not conform to the expected format.