DOCUMENTATION

ASSIGNMENT 1

STUDENT NAME: Razvan-Claudiu Cosma
GROUP: 30425_1

CONTENTS

1. Assignment Objective

The main objective of this assignment is to create a polynomial calculator with basic polynomial operations (addition, subtraction, multiplication, division, derivation and integration) in order to familiarize oneself with Java environment, JavaFX for GUI modelling and JUnit for testing.

In order to achieve this objective, multiple steps were required to be followed:

- Analyzing the problem and how it could be solved.
- Designing the UML diagrams: use case, package and class diagrams.
- Creating models and deciding how data will be stored inside.
- Creating methods that are able to perform the required operations on the previously created classes.
- Testing the methods using JUnit to verify their correctness.
- Designing a graphical user interface such that the user could easily interact with the calculator and implementing it using JavaFX.

2. Problem Analysis, Modeling, Scenarios, Use Cases

In order to solve the problem, firstly we need to analyze the requirements. We'll use the MoSCoW technique for this:

- Must have
  - An interface with text fields and the operation selector
  - The ability to extract the polynomials
  - Addition and subtraction
- Should have
  - Multiplication, division, derivation and integration
  - A clear and intuitive display of the result
- Can have
  - A pleasant graphical user interface
- Will not have
  - Support for multiple variable polynomials
  - Support for introducing float coefficients

To be able to fulfill this requirements we create some models to be able to store and manipulate the data given by the user. The Monomial class is a model used to store each monomial, simplify the operations performed in the Polynomial class and to have a better organized code. The Polynomial class stores the polynomials in a TreeMap of monomials such that we can easily work with the stored data and perform the operations.

The calculator has 6 use cases, one for each operation. The scenario is very similar for all the cases. The user writes the polynomials in the text fields (e.g. "2x^3 – 5x + 12") and after that selects the desired operation from the 6 that are available (derivation and integration are performed on the polynomial from the first field). After the desired operation was selected from the choice box, the result will appear in the result text field. If the operation cannot be performed, an appropriate message will be displayed (e.g. for division by 0).

To better understand, a use case diagram is needed.

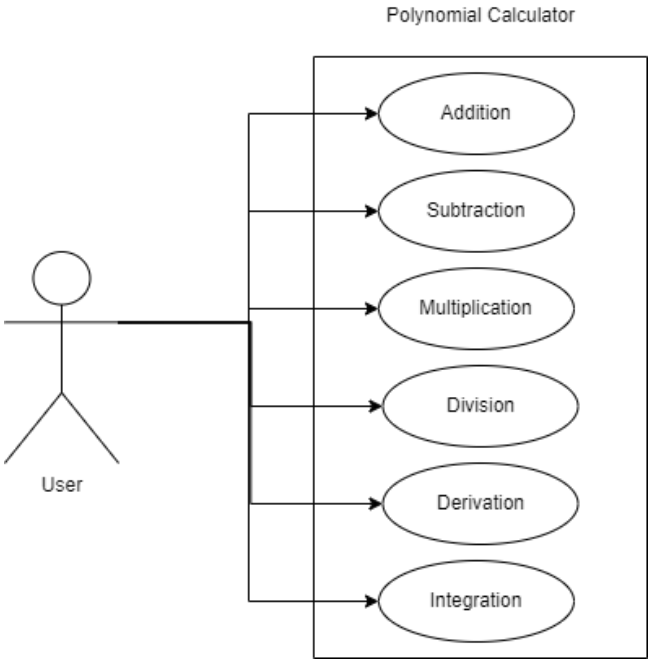


Figure 1. Use case diagram

3. Design

The application was designed respecting the object-oriented paradigm. Encapsulation was used and also appropriate classes were defined. The classes that were designed are the following:

- Calculator - is the main class that sets the scene and loads the app
- CalculatorController - is the class that makes the connection between the GUI and the classes that perform the operations. It takes the information as a string from the text fields and passes it to the Polynomial class to be processed. It receives back a string that is displayed as the result.
- Monomial - is a helper class that implements the reading, operations and display of a monomial in order to be used in the Polynomial class and have a better organized code
- Polynomial – is the class that does all the work. It processes the data given by the controller and has all the algorithms for polynomial operations implemented. It uses TreeMap to store the monomials and for the division, long division algorithm was used.
- PolynomialTest – is the testing class that performs tests on the defined polynomial operations.

The package diagram respects the MVC architecture. For a better understanding of the implementation, here are the package diagram and the class diagram.

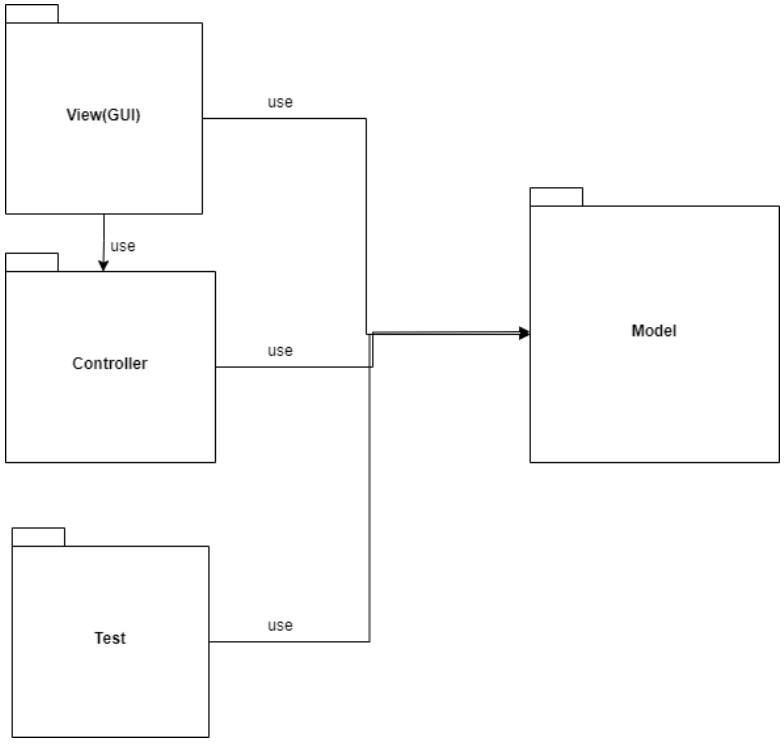


Figure 2. Package diagram

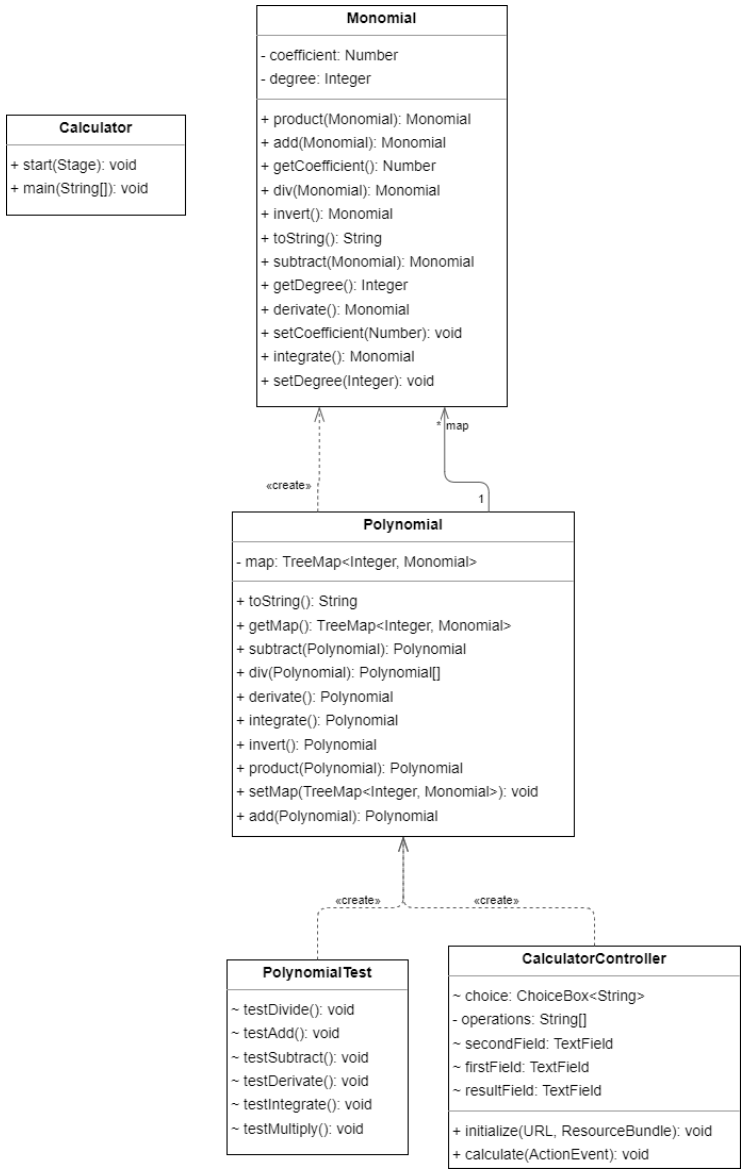


Figure 3. Class diagram

4. Implementation

Calculator

- This class does the setting of the stage and scenes and loads the application.

Monomial

- The monomial class has two fields, one for the coefficient that is saved as a Number and the other for the degree which is saved as an Integer. We also have getters and setters for this fields.
- We have several constructors. One of them is used to simply declare a new monomial with coefficient and degree, while another one is used to declare a monomial from a string using regular expressions and pattern matching.
- The toString method is overridden in order to write the monomial as a string to be understood by the user.
- All the other methods implement the basic operations on monomials.

Polynomial

- The polynomial class has only one field, a TreeMap of monomials in order to be able to store them properly. We also have getter and setter for this field.
- We have several constructors here too. One of them is used to instantiate a polynomial from a monomial and another one is based on the monomial constructor that uses regular expressions. It parses a string and repeatedly call the monomial constructor to match and create new monomials and after that add them into the TreeMap.
- The toString method is overridden here too. It calls repeatedly the toString method from the monomial class for each entry in the TreeMap and appends the strings into a string builder, in order to get a string representing the polynomial.
- All the other methods, except the division, are based only on the monomial operations and call them repeatedly for each monomial in the map in order to perform the computation.
- The division method, uses several polynomial and monomial methods in order to compute the quotient and remainder, following the Long Division Algorithm for polynomials.

CalculatorController

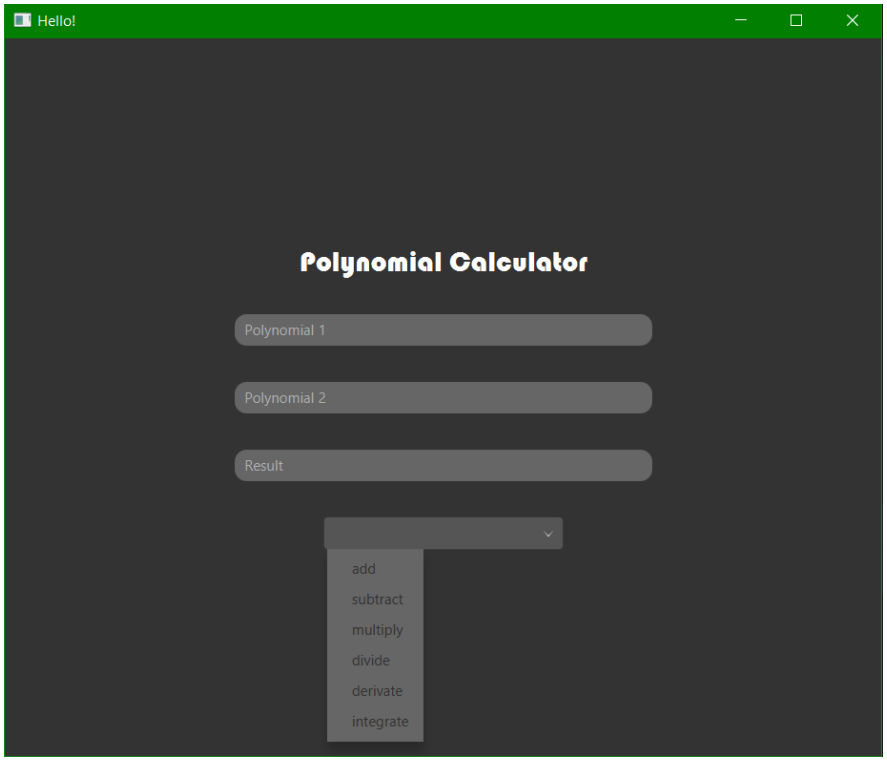
- The calculator controller class is a simple controller class that makes the connection between the GUI and the computation algorithms.
- It has three text fields, one for each polynomial and one for the result, one choice box, to select the desired operation, and one string that is used to store the options for the choice box.
- The class implements initializable and the initialize method is overridden in order to initialize the choice box.
- The class also implements a method called calculate that uses a switch in order to select the operation chosen by the user and call the corresponding method from the polynomial class.

PolynomialTest

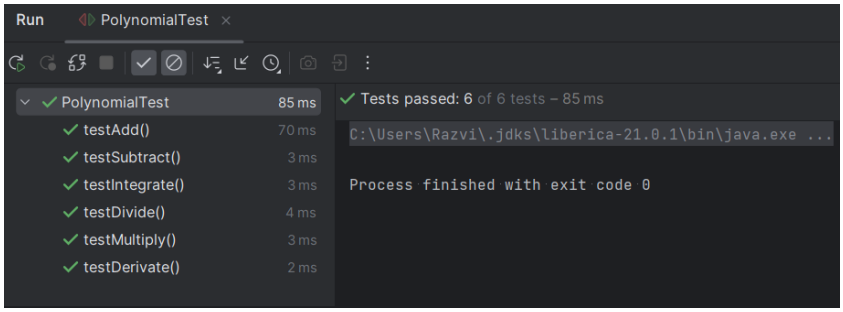
- This is a testing class that verifies the correctness of the algorithms using JUnit tests.
- Each method from this class tests one method from the operations on polynomials.

5. Results

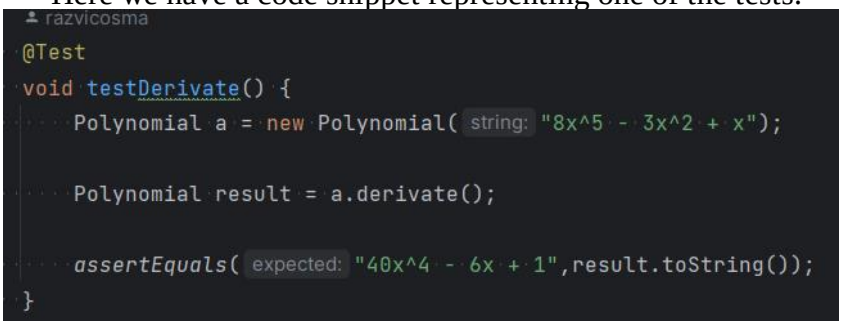
After following all the steps mentioned, the application is a functional polynomial calculator that can be used to perform all basic operations. Here is how the GUI looks like:



Testing was done using JUnit tests as mentioned above. There were six tests, one for each operation performed on polynomials. All the tests were successfully passed, which supports the correctness of the algorithms used.



Here we have a code snippet representing one of the tests.



6. Conclusions

This Java project uses JavaFX for GUI development. It follows object-oriented design principles. It has unit testing, improving the validation. The code is well-structured for readability and maintenance.

This assignment taught me how to do basic operations on polynomials, how to perform unit testing and how to design a graphical user interface using JavaFX. It also taught me how to solve a problem and make diagrams in order to help me structure the code better.

Future work could focus on adding more operations, adding support for float coefficients and multiple variable polynomials, a better GUI, and better performance.

7. Bibliography

1. *Bruce Eckel, Thinking in Java (4th Edition), Publisher: Prentice Hall PTRUpper Saddle River, NJUnited States, ISBN:978-0-13-187248-6 Published:01 December 2005.*
2. *Lectures and Laboratories* - *https://dsrl.eu/courses/pt/*
3. *Polynomials Long Division* - *https://www.mathsisfun.com/algebra/polynomials-division-long.html*
4. *Geeks for Geeks* - https://www.geeksforgeeks.org/