**Documentation**

Assignment 1

Polynomial calculator

Name: Știrb Călin-Alexandru

Group: 30422

**Contents:**

1. Assignment Objective............................................................................................3
2. Problem Analysis, Modeling, Scenariosm, Use Cases..........................................3
3. Deisgn....................................................................................................................4
4. Implementation......................................................................................................7
5. Results....................................................................................................................9
6. Conclusions............................................................................................................10
7. Bibliography...........................................................................................................11
8. **Assignment Objective**

The main objective of this assignment is to familiarize ourselves with a few styles of OO Programing, such as: Maven projects and the Model-View-Controller design. From the perspective of the code, it’s main objective is to implement a functional Polynomial calculator with a dedicated user interface(UI), where the user can insert the polynomials as a string of characters, the way they would on paper.

**Sub-objectives:**

* Implementing the following classes (see the ***design*** section):
* Polynomial: the class which will store our polynomial as a map of integer key and Monomial value
* Monomial: this class is used to store the polynomial’s degree and coefficient
* Operations: this class is used to implement most of our methods
* View: this class represents how the GUI will look
* Controller: this class represents the functionality of the GUI
* Implementing the following methods (see the ***implementation*** section):
* A method which extracts the input strings and transforms them into monomials and then into the actual polynomials.
* A method for each of the operations on polynomials: addition, subtraction, multiplication, derivative and integration.
* A method which takes the resulting polynomial and converts it back into a string to be output on the GUI

1. **Problem analysis, Modeling, Scenarios, Use cases**

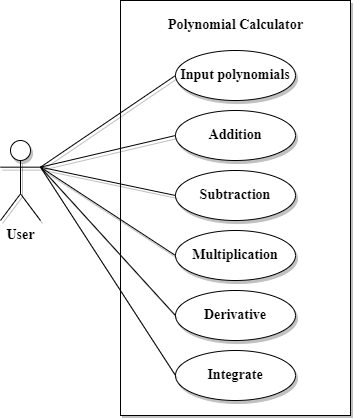
In order for the application to work correctly, the user needs to input a polynomial as they would on paper, like the following example:

Graphical user interface, application

Description automatically generated

* Screenshot taken from the app’s GUI for the insertion of the polynomials

If the user does not input anything, or inputs the polynomials incorrectly, the application will consider the default value for both, which is 1

The primary actor for all of the use cases is the ***user***.

Main success scenario for any of the use cases:

1. The user inserts 2 polyonmials correctly into the GUI
2. The user selects one of the operations in the GUI
3. The user clicks the „compute” button
4. The app performs the selected operation and displays the result

Alternative sequence: Incorrect polynomials

* The user inserts incorrect/empty polynomials
* The app will instead use the default values of 1 for both polynomials

**Functional requirements:**

* The app should allow userts to insert polynomials
* The app should allow users to select the mathematical operation
* The app should have working methods for the following operations: addition, subtraction, multiplication, derivative, integration of polynomials

**Non-functional requirements:**

* The app should be intuitive and easy to use by the user
* The app should not be bloated with unused methods/memory

1. **Design**

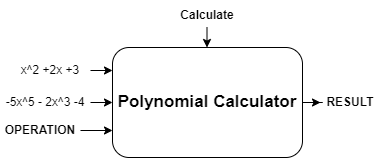
In this section will be presented the design choices taken during the creation of this assignment’s application

* For the implementation of the data model Polynomial we use a hash map to store the classes’ data:

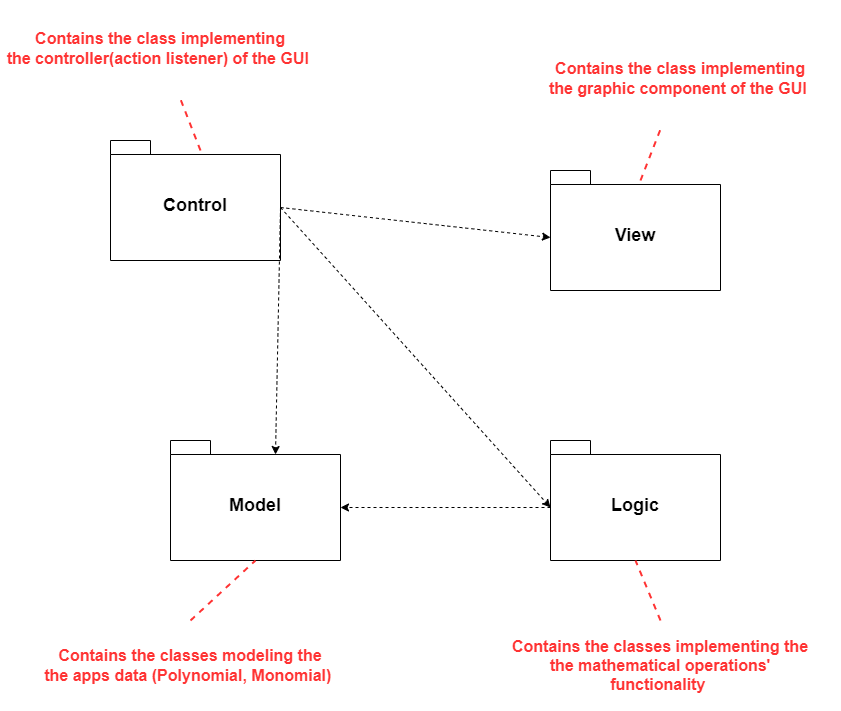
HashMap<Integer,Monomials>

* The declaration above signifies a HashMap with an Integer key where data of type Monomial is stored

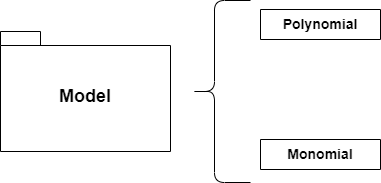
Level 1: Overall system design



* In the above diagram are represented the main inputs and the outputs of the app
* The 2 polynomials represent the user’s written input
* The input „OPERATION” represents the user’s choice of mathematical operation
* The input „Calculate” is a button which tells the app to perform the selected operation on the input polynomials
* The output is the printed result on the GUI

 Level 2: Division into packages

Level 3: Division into classes



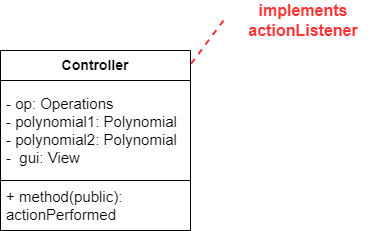
Shape

Description automatically generated

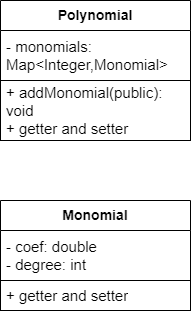
Shape, arrow

Description automatically generatedShape

Description automatically generated

Level 4: Division into routines

A picture containing text

Description automatically generated

Text

Description automatically generated with low confidence

1. **Implementation**
2. **Polynomial class**

* This class contains a single object: private Map<Integer,Monomial>
* It contains 2 constructors:

Graphical user interface, text

Description automatically generated with medium confidence

* Other than getters and setters this class contains only one other implemented method: addMonomial

Text

Description automatically generated

1. **Monomial class**

* This class contains 2 objects:
  + private double coef 🡪the coefficient of the monomial
  + private int degree 🡪 the degree of the monomial;

🡪also the key of the monomial map;

* It does not contain any methods other than getters and setters

1. **Controller class**

* The controller class represents the functional part of the GUI
* This class implements the java Swing interface ActionListener
* This class contains multiple objects:
  + private Operations op 🡪 the operation class implemented for the app
  + private Polynomial polynomial1, polynomial2
  + private static View gui 🡪the GUI that it manages
* It implements a method which builds the input polynomials from the user and computes the selected operation from the GUI:

Text

Description automatically generated

(piece of code from the method)

1. **View class**

* This class implements the graphical elements of the GUI
* It does not implement any methods
* It contains multiple objects from the javax.Swing library
  + public JtextField poly1, poly2 🡪the polynomial input fields
  + public JcomboBox operations 🡪 the drop down operations menu
  + private static Jpanel panel
  + public Jlabel p1, p2, result 🡪the written text on the GUI
  + private static Jframe frame
  + public Jbutton calculateButton 🡪 the button who’s actionListener is implemented in the Controller class

1. **Operations class**

* This class is used to implement all of the main methods used by this app
* It does not contain any class global objects
* The most used/important methods:
  + buildPolynomial 🡪 builds the polynomial from the input string
  + splitToMonomials 🡪 splits the input string into strings of monomials
  + extractData 🡪 this method essentially builds a monomial from a string
  + The mathematical operation methods: add, sub, multiply, derivate, integrate

Text

Description automatically generated

(the addition method with functionality comments)

1. **Results**

* Most methods were tested using the JUnit library and testing enviroment provided by Maven
* Some of the methods were tested by computing the operation by hand on the same input polynomials and compared to the operations done directly in the app itself
* The tests done by JUnit can be run from the Maven project lifecycle tab

Graphical user interface, text

Description automatically generated

(the output on the console after running the test)

Text

Description automatically generated

(this is how the test for the addition function looks)

1. **Conclusions**

In conclusion, this assignment served not only as a way to get a feel for how to fully implement an application but also how to better manage and design an OOP project, and to familiarize oneself with design patterns such as MVC(model-view-controller).

1. **Bibliography**
2. *Java swing tutorials:* [*https://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html*](https://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html)
3. *Alex Lee’s java swing tutorials on youtube:*[*https://www.youtube.com/@alexlorenlee*](https://www.youtube.com/@alexlorenlee)
4. *Geeks for geeks’ tutorials for swing*[*https://www.geeksforgeeks.org/java/?ref=shm*](https://www.geeksforgeeks.org/java/?ref=shm)