Documentation

Laboratory – Assignment 1

**Polynomials Calculator**

Student: Oltean Andrei- Petru

Group: 30421\_2

***Table of contents***:

[1. Objective of the project 3](#_Toc95297885)

[2. Dimensions of the problem 3](#_Toc95297886)

[3.Design 3](#_Toc95297887)

[4. Implementation 3](#_Toc95297888)

[5. Results 3](#_Toc95297889)

[6. Conclusions 3](#_Toc95297890)

[7. Bibliography 3](#_Toc95297891)

1. ***Objectives:***

The objective of this project is to create a polynomial calculator using OOP specifications in the Java language having a graphical interface that takes two polynomials as input and print the result of different operations that can be executed on the input data.

This main objective can be broken down into secondary objectives:

1. Identify a method of interpreting and storing the input into a more manageable form to work with.
2. Create the methods of executing the different requested operations.
3. Designing an UI for the calculator.
4. Implementing the abovementioned objectives.
5. Testing the calculator.
6. ***Dimensions of the problem:***
7. *Analyzing the problem:*

A polynomial is an [expression](https://en.wikipedia.org/wiki/Expression_(mathematics)) that can be built from [constants](https://en.wikipedia.org/wiki/Constant_(mathematics)) and symbols called indeterminates or variables by means of [addition](https://en.wikipedia.org/wiki/Addition), [multiplication](https://en.wikipedia.org/wiki/Multiplication) and [exponentiation](https://en.wikipedia.org/wiki/Exponentiation) to a [non-negative integer](https://en.wikipedia.org/wiki/Non-negative_integer) power.

A polynomial in a single indeterminate x can always be written (or rewritten) in the form

where  are constants and is the indeterminate. The word "indeterminate" means that  represents no particular value, although any value may be substituted for it.

The polynomial consists of a list of certain terms, also called monomials, for example   is a monomial. The coefficient is 3, the indeterminate is and the degree is 2. Forming a sum of several terms produces a polynomial, like the following one: , which has three terms with different exponents:  the first is degree two, the second is degree one, and the third is degree zero.

An alternative representation for polynomials consists of a sequence of ordered pairs:

Each ordered pair corresponds to the term of the polynomial. An ordered pair is composed of the coefficient of the i-th term and its index representing the exponent i.

For example, the polynomial can be represented by the sequence {(3,2), (2,1), (1,0)}. The sequence contains the (coefficient, exponent) pair of each monomial from the given polynomial.

So, we can say that polynomial is composed of one or more monomials with i in the range [0, n] where n is the degree of the polynomial.

This way of representing polynomials can be used to perform the most common operations on polynomials: addition, subtraction, multiplication, division, differentiation, and integration.

1. *Modelling the problem:*

The user will be able to use the functions of the calculator by introducing in the interface, two polynomials. He/she will have the possibility to choose a specific operation to perform, such as:

- Addition of two polynomials

- Subtraction of two polynomials

- Multiplication of two polynomials

- Division of two polynomials

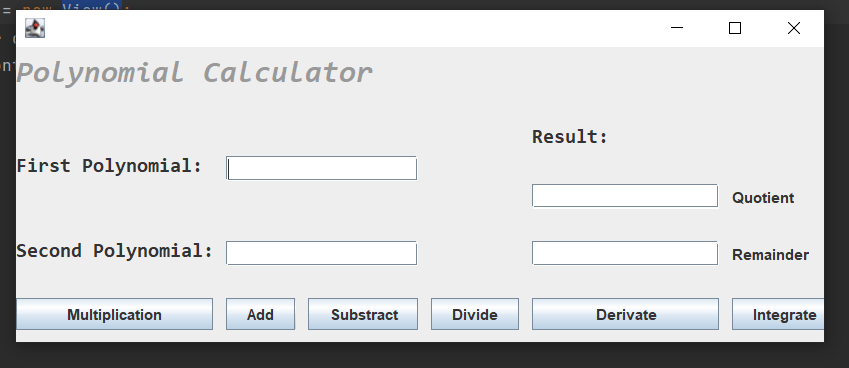
- Differentiation of a polynomial

- Integration of a polynomial

The result of the chosen operation will be displayed in the interface. The user will also have the option to set the current result as one of the operands for the following operation, but he/she must fetch again the new operand, if the previous result is not a proper polynomial, the application will throw an error box.

1. *Different scenarios and use cases:*

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The use cases are strongly connected with the user steps that’s why I tried to design my interface in a user friendly mode and the result is the following:

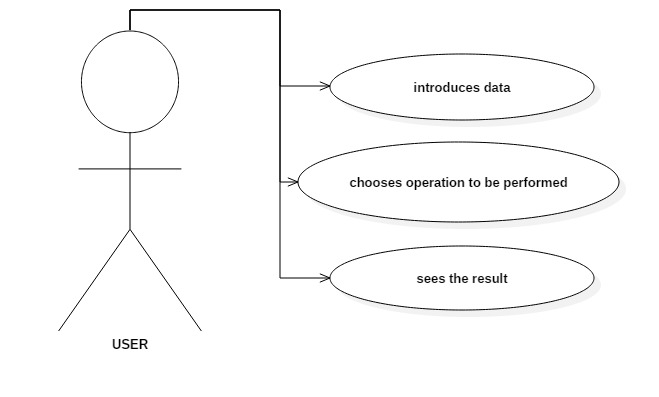


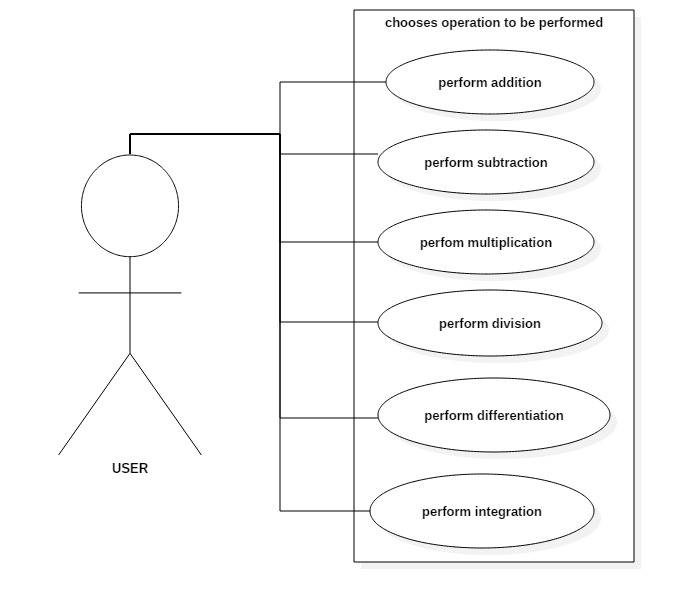
The user will introduce 2 polynomials in the corresponding TextFields, then will select the desired operation. The result will be displayed on the Result TextFields, The user must pay attention to follow the format of the polynomial, which appears in the GUI.

1. ***Design:***

*a)Diagrams:*

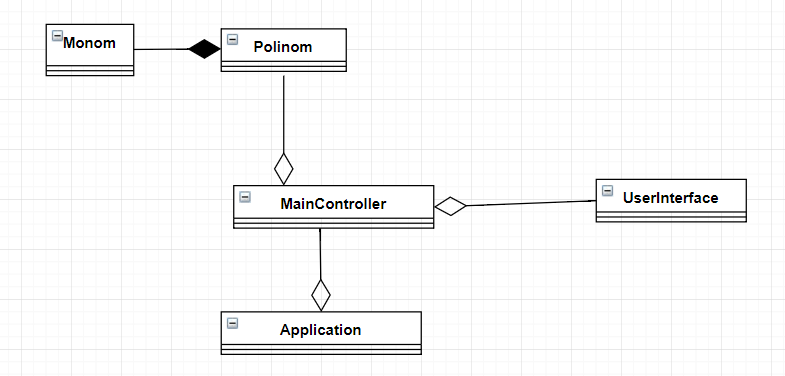
-use case:

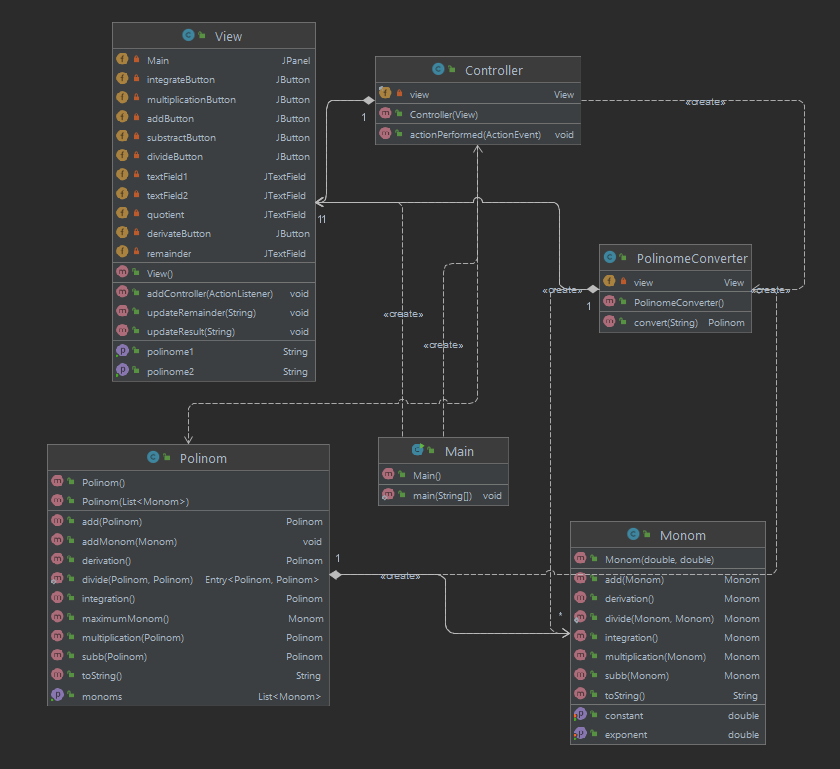




The use case presents the actor, which in our case is the user that interacts with the application. The user can perform several actions on the two chosen polynomials, such as addition, subtraction, multiplication, division, integration and differentiation.

-Class Diagrams:





*b) Data Structures:*

The data structures that I used in the implementation of the project are primitive data types, such as doubles and strings and more complex ones like ArrayList, HashMap type objects and new created objects like Monom or Polinom (which consists in a list of Monomials).

Regarding this, I have decided to use ArrayList instead of the classic arrays because I think that they are more efficient from the point of view of memory management, performance and provide a faster access to their content, also, the size of an ArrayList is not fixed and adding new elements to the list is very easy because we do not have to worry about exceeding the “length” of the list (array).

A better implementation of the project would be to use Java Generics methods and classes, because they enable programmers to specify, with a single method declaration, a set of related methods or, with a single class declaration, a set of related types, respectively

*c) Packages:*

Java packages help in organizing multiple modules and group together related classes and interfaces.

In object-oriented programming development, model-view-controller (MVC) is the name of a methodology or design pattern for successfully and efficiently relating the user interface to underlying data models. The MVC pattern is widely used in program development with programming languages such as Java, Smalltalk, C, and C++.

The MVC pattern has been heralded by many developers as a useful pattern for the reuse of object code and a pattern that allows them to significantly reduce the time it takes to develop applications with user interfaces.

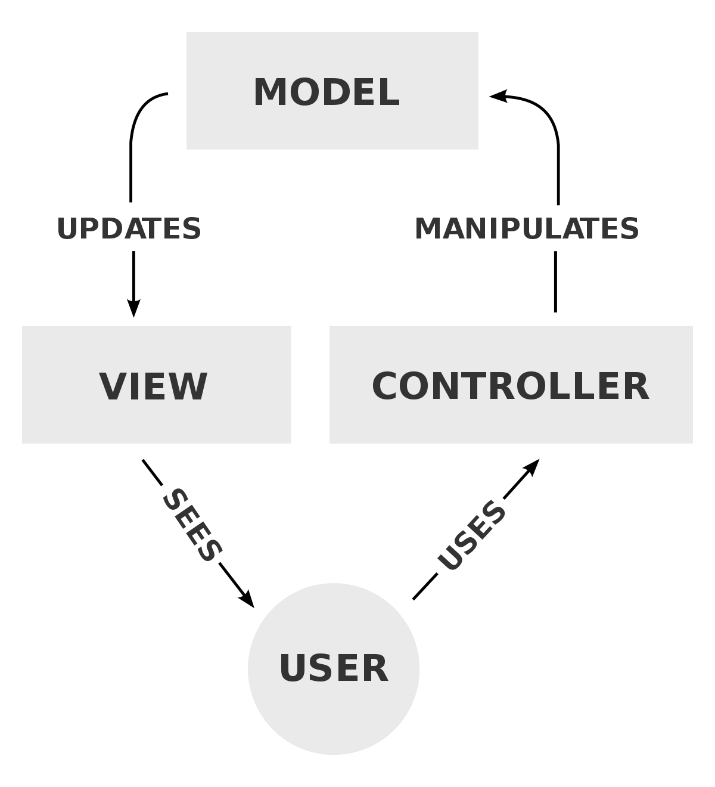
The model-view-controller pattern proposes three main components or objects to be used in software development:

- Model, which represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the user interface.

- View, which is a collection of classes representing the elements in the user interface (all of the things the user can see and respond to on the screen, such as buttons, display boxes, and so forth)

- Controller, which represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

* Application – contains a single class, which contains the customary main() method
* Model – contains the “brain” of the application, the classes which model the problem
  + - * Monom
      * Polinom
      * Polinom Converter
* View – contains a single class which represents the GUI
* Control – it interconnects the model and the view

***4.Implementation****:*

We use a divide and conquer technique, so that our problem will divide in smaller problems, easier to solve and handle. In this programming paradigm we model our problem by some objects and try to solve the problem by sending messages between these objects. Because I followed the MVC architecture, my program consists of 4 parts:

* In the **Model** we have:

1. ***Monom class***

A polynomial is composed by one or more terms, which in Mathematics are called monomials. This class has 2 instance variables, a double coefficient and a double exponent.

*Constructor:* public Monom(double constant, double exponent) that initializes the monomials with adequate coefficient and exponent.

*Methods*: -public double getConstant(): returns the coefficient of the monomial

-public double getExponent(): returns the degree of the monomial

- public void setConstant(double constant): allows us to set the coefficient of the monomial

- public void setExponent(double exponent): *:* allows us to set the degree of a monomial

- public boolean equals(Object other):

- public String toString(): returns one monomial as a string

- public Monom add(Monom other): addition function

- public Monom subb(Monom other): subtraction function

- public Monom multiplication(Monom other): multiplication function

- public Monom derivation(): derivation function

- public Monom integration(): integration function

- public static Monom divide(Monom m1, Monom m2): division function

***b) Polynomial Class***

This class has only an instance variable which consists of a ArrayList<Monom> class that is used to keep the monomials of this polynomial. The monomials are ordered in this list, from the highest to the lowest exponent of the polynomial, in the exact same order as they were introduced.

*Constructor*:- public Polinom(List<Monom> monoms)

*Methods*: - public List<Monom> getMonoms(): get the monoms that form the polynomial

- public String toString():

- public Polinom add(Polinom other):

- public Polinom subb(Polinom other:

- public Polinom multiplication(Polinom other):

- public Polinom derivation():

- public Polinom integration():

- public Monom maximumMonom(): helper function for division

- public static Map.Entry<Polinom, Polinom> divide(Polinom p, Polinom q):

-In the ***View*** (contains the graphical intergace, the user interface):

I implemented the View with the help of the GUI form, in the class we can see all the buttons, labels, textfileds and so on.

*Methods:*

-public void updateResult(String string): updates the result after pressing the button corresponding to an operation.

- public String getPolinome1(): gets the polynomial from the textfield

- public void addController(ActionListener listener): implements the action listener and the conection with the controller.

-in the ***Controller*** we have:

- public Controller(View view)

- public void actionPerformed(ActionEvent e)

4) The Application – contains the Application class which runs the start() method from the MainController in order to “turn on” the application.

*e)Algorithms:*

1. ADDITION

The sum of two polynomials is obtained by adding together the coefficients sharing the same powers of variables so, for example,

|  |  |
| --- | --- |
| (a_2x^2+a_1x+a_0)+(b_1x+b_0)=a_2x^2+(a_1+b_1)x+(a_0+b_0) |  |

and has order less than (in the case of cancellation of leading terms) or equal to the maximum order of the original two polynomials.

1. SUBTRACTION

The subtraction of two polynomials is obtained by subtracting the coefficients sharing the same powers of variables so, for example,



and has order less than (in the case of cancellation of leading terms) or equal to the maximum order of the original two polynomials.

1. INTEGRATION

The algorithm checks whether the exponent is -1. If so, it keeps the coefficient unchanged and prints the integration result as the natural logarithm (ln) function.

Otherwise, it multiplies the coefficient with the exponent and divides it with incremented by one exponent.

4) MULTIPLICATION

The product of two polynomials is obtained by multiplying term by term and combining the results, for example

|  |  |  |  |
| --- | --- | --- | --- |
| (a_2x^2+a_1x+a_0)(b_1x+b_0) | = | a_2x^2(b_1x+b_0)+a_1x(b_1x+b_0)+a_0(b_1x+b_0) |  |
| http://mathworld.wolfram.com/images/equations/Polynomial/Inline9.gif | = | a_2b_1x^3+(a_2b_0+a_1b_1)x^2+(a_1b_0+a_0b_1)x+a_0b_0, |  |

and has order equal to the sum of the orders of the two original polynomials.

1. DIVISION

The division of the two polynomials is used using the basic division polynomial algorithm.

1. DIFFERENTIATION

A very simple algorithm to differentiate a polynomial which is represented by a sequence of ordered pairs was used:

* Drop the ordered pair that has a zero exponent.
* For every other ordered pair, multiply the coefficient by the exponent, and then subtract one from the exponent.

*Graphical User Interface:*

The user can introduce some input data, respectively one or two polynomials and perform on them the desired operations available.

Although the input data has to respect some conditions in order for the application to run correctly, such as:

- Every introduced polynomial is composed by one or more terms, with the exception that, none of the polynomials to be equal with “0” or “0x^0”

- Each term must cover this format [sign][coefficient][x^][exponent] – the square brackets must be ignored, they are used just to make the format more visible and clear.

***5. Results:***

This application was developed and tested only in InteliiJ Idea, but this thing should not affect it’s portability. From the point of view of the used algorithms, they are simple ones, taken from mathematics and implemented in Java.

The application is an user friendly and helpful application to perform basic polynomial operations, as long as the user obeys the input conventions and it is familiar with polynomial operations such as: addition, subtraction, multiplication, division, differentiation and integration.

We wrote the basic operations in the Junit tests and check if it passes to confirm the correctness of our operations.

1. ***Conlusions***

The conclusions that can be drawn from this are plentiful, from the power of regular expressions and their ability to manipulate stings to the importance of planning ahead and having a good plan to start the project and to follow through to the end.

1. ***Bibliogrpahy***

a. Object-Oriented Programming - Lecture Slides of prof. Marius JOLDOS

b. Programming Techniques – Lectures of prof. Cristiana Pop

c. [www.stackoverflow.com](http://www.stackoverflow.com)

d. https://docs.oracle.com

e. [www.wikipedia.org](http://www.wikipedia.org)

f. <http://whatis.techtarget.com/definition/model-view-controller-MVC>

g. https://www.draw.io/ (for diagrams)

h. https://www.geeksforgeeks.org/