*Homework 1:*

*Polynomial Calculator*

***Programing Techniques***

* *Rus Rares Tudor*
* *Group 30422*

*Contents:*

1. *Requirements . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ..3*
2. *Problem Analysis. . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .3*
3. *Use Cases . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4*
4. *Projection . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .5*
5. *Implementation . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5*
6. *Results . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8*
7. *Conclusions . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10*
8. *Bibliography . . . . . . . . . . .. . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . 10*

*--Requirements :*

The main objective of this assignment is to design and implement a system for polynomial processing. Considering the polynomials of one variable and having integer coefficients. To achieve such a goal, we have to implement all the possible operations on the polynomials and those are addition, subtraction, multiplication, division, derivation and integration. In order to successfully fulfil this task we have to understand what a polynomial is, so here is the definition:

A polynomial is an expression that can be build from constants and symbols called indeterminate or variables by means of addition, multiplication, and exponentiation to a non-negative integer power. Two such expressions that may be transformed, one top the other, by applying the usual properties of commutativity, associativity, and distributivism of addition and multiplication are considered as defining the same polynomial.{\displaystyle \sum \_{k=0}^{n}a\_{k}x^{k}}

A polynomial in a single indeterminate can always be written in the form:

P(x) = anxn + an-1xn-1 + … + a2x2 + a1x + a0,

Where an, …, a0 are constants and x is the indeterminate. The word “indeterminate” means that x represents no particular value, although any value may be substituted for it. The mapping that associates the result of this substitution to the substituted value is a function, called a polynomial function.

*--Problem Analysis:*

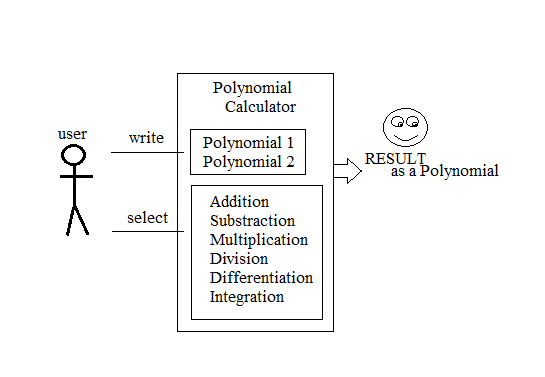
Abstraction in its main sense is a conceptual process where general rules and concepts are derived from the usage and classification of specific examples, literal signifiers, first principles, or other methods.

Conceptual abstractions may be formed by filtering the information content of a concept or an observable phenomenon, selecting only the aspects which are relevant for a particular subjectively valued purpose. For example, abstracting a leather soccer ball to the more general idea of a ball selects only the information on general ball attributes and behavior, excluding, but not eliminating, the other phenomenal and cognitive characteristics of that particular ball. In a type–token distinction, a type (e.g., a 'ball') is more abstract than its tokens.

As presented above, a polynomial is nothing more than a collection of components of the form anxn. Each component of this form is called a monomial. Furthermore, each monomial has two components: coefficient and power. The power of the monomial must be a non-negative integer and the coefficient can be any real number.

So in order to make our task easier we will define a polynomial as a list of monomials, and we will make all the polynomial operations(addition, subtraction, multiplication, division, integration and differentiation) using the same operations but implemented on the monomial.

*--Use Cases:*



A user is required to write two polynomials and select an operation in order to interact with the system (application). From that point, the system will decide if the inputs, provided by the user are correct or not, and will perform the selected operation and will provide the correct result to the user

Perform Addition/ Subtraction/ Multiplication/ Division of two polynomials:

- *User Main SUCCES scenario*:

-User launches application successfully.

-User provides first polynomial by typing a string in the first text field provided by the application

-User provides second polynomial by typing a string in the second text field provided by the application.

-User presses the “Addition”/ “Subtraction”/ “Multiplication”/ “Division” button from the given list of buttons.

-The first provided string is checked and successfully transformed into a polynomial object.

-The second provided string is checked and successfully transformed into a polynomial object.

-The addition is performed successfully with no exceptions or errors.

-The result is displayed in a label provided by the application.

*- User Main FAILED scenario*:

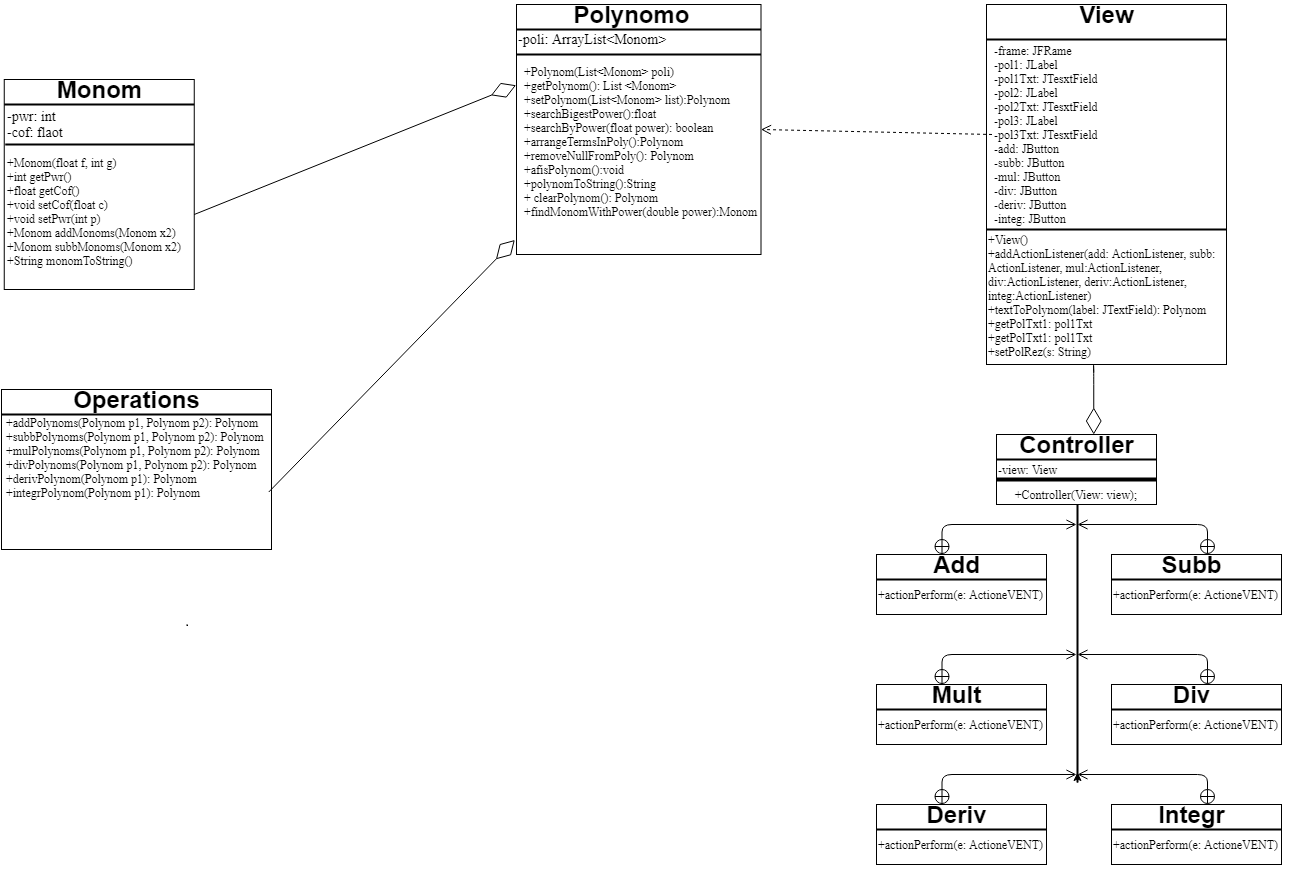
-User can’t launches application successfully.

-User provides wrong inputs for the first or for the second polynomial.

-The first/ second provided string is checked and unsuccessfully transformed into a polynomial object.

-The result is not displayed in a label provided by the application.

*--Projection:*



*--Implementation:*

In the following, we will take a closer look at the role and the implementation of each class with its attributes, constructors and methods.

*- Monom.java*

The attributes of this class are “pwr” , which is an integer and it represents the power of the monom and “cof” , which is a double and represents the coefficient of the monom object. Both variables are declared private for security purposes and they can be accessed outside the Monom class only by using the getter methods “getPwr()” and “getCof()”.

The constructor of this class takes as parameter two numbers of double and integer value and sets them as attributes to the monom object(as its coefficient respectively as its power)

The methods addMonoms, subbMonoms are both methods used to compute the mathematical addition, subtraction, multiplication, division, differentiation and integration, which were all described above. Not to forget that when adding or subtracting two polynomials there may be the case where in the second polynomial there exists no monom with the power of some other monom from the first polynomial, in this case I choose to add the monom with zero.

Another method is “monomToString()” which is used to convert the computed monom into a type that can be used in order to show the result to the user, and this type is string. This method adds beside of the coefficient and of the power of the monom an additional “X”, representing the unknown variable and the sign “^” which is placed before the power of the monom. If the power is zero, then it will show the coefficient next to the syntax “X^0” in didactical purpose.

*- Polynom.java*

The only field of this class is a list, “poli” which holds an array of the monoms constructing the polynom object. This field is declared private so it can be accessed outside the class only by using the getter function “getPolynom()”.

The constructor of this class takes as parameter a list of monoms and attributes it to the monom list of the polynom object when instantiated, so it creates a polynom object with an equation corresponding to the given list.

In this class we will observe a lot of functions created for making the operations on the polynomials more easy, so in the following are presented those methods. “searchBigestPower()”, as it’s name suggests this is a method which finds in the array of Monoms the biggest power. “searchByPower(power)” is a method that returns a Boolean, true if the power(given as a float) is in any monoms in the array list. The next method is “removeNullFromPoly()” which removes any null monom in the given list. “arrangeTermsInPoly()” is a method used to order the elements in the list of the polynom object having the biggest power as the first element and the smallest as the last element, here we used some methods from “java.util.Comparator”. “polynomToString” uses a method from “Monom” class in order to convert a given polynom object into a string. The last method met in this class is “findMonomWithPower(power)” it does what it’s name suggests, returning from a list of monoms the monom which has the power equals with “power” given as a parameter.

- Operations.java

In this class I implemented all the possible operations on 2 polynomials and those are: addition, subtraction, multiplication, division, derivation, integration. For addition and subtraction I used the same algorithm: I saved in an integer variable the biggest power in the both polynomials and until that variable is equals to zero I was adding/subtraction the values from the two polynomials. For multiplication I used the basic algorithm as follows: multiplying the first element from the first polynomial with all the elements from the second polynomial, and adding to an empty list, then that list was added to the result and so on for all the elements in the first polynomial. Integration and Derivation uses the same algorithm, the one known from mathematics learned in the eleventh and twelve grade. The hardest operation that had to be implemented was division.

-View.java

This is the class which builds the aspect of the user interface. The components used were described above at the design part. This class has two methods. One is used as action listeners for the buttons which are related to the mathematical operations, buttons which are implemented in the controller. The other one is used in order to convert the string from the text labels into polynomial objects. In order to do that I let the introduced monom to be of the previously described fixed form which is sign , coefficient, x, ^, power. This form is used by this converting method because it creates monoms by taking the values of the second and fifth position of the monom string and puts them into the new created monom. It returns a list of all the introduced monoms so basically a polynom.

-Controller.java

In this class is done the linking between the buttons created in the view class and the mathematical operations created in the polynom class. In order to do that, the extracted string from the text fields is converted into polynoms and then the needed operation is performed on them and finally the resulted polynom is converted back to a string and set as result in the view declared label for the result in so that the user can see it as well.

Design

We can divide the project in two major parts: the logic of the application and the user interface. They both must run flawless in order for the application to work correctly. The design follows a Model-View-Controller pattern.

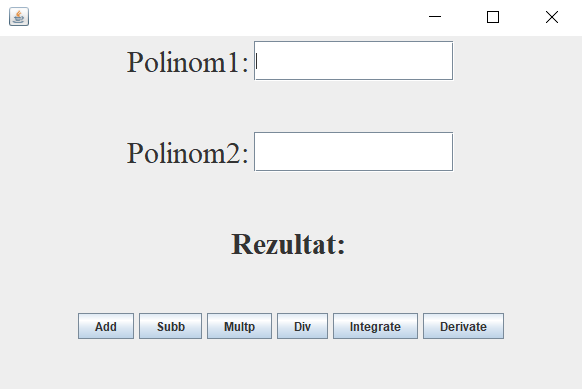
-The logic consists of the approach used(described above), and of the implementation(the classes with their attributes and methods), which glues all the parts of the MVC model together.

-The GUI part was implemented according to the MVC pattern by splitting the written code in three big parts.

-The Model contains the algorithms which run “in the back”, to which the user has no access.

-The view is composed of two text fields, where the user should introduce the polynomials, a label which will show the result of the operations and six buttons which each correspond to one of the six operations which the user can perform on the introduced polynomials.

-The controller is composed of six action listeners, one for each button, where each one is used in order to make the connection between the view and the model, so in other words to link the button pressed by the user to the desired part in the code.

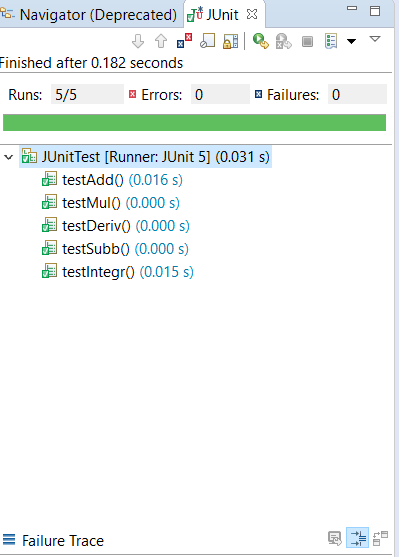
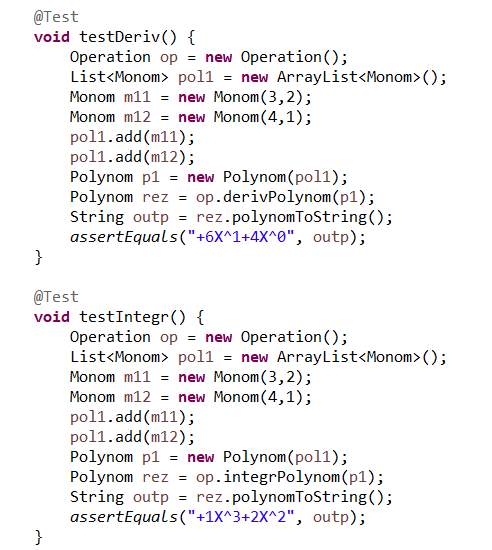
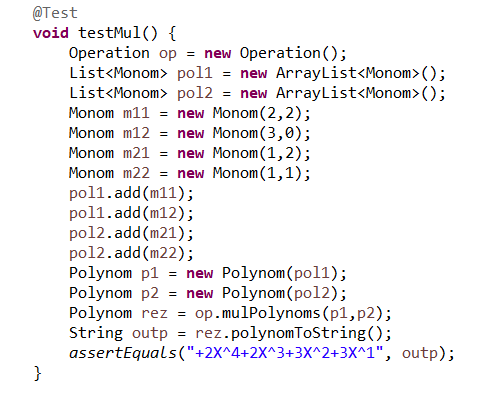
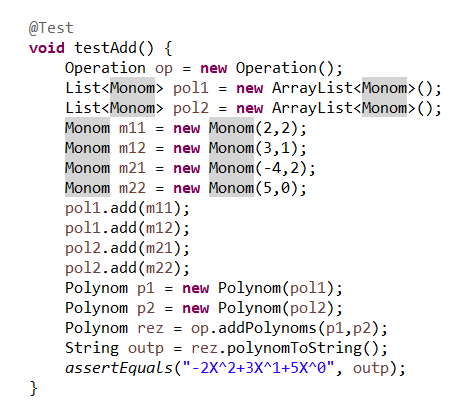
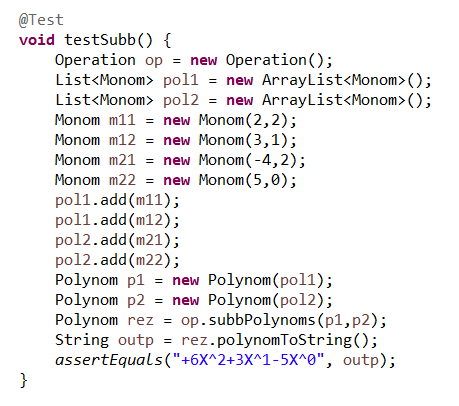


Here is an screenshot from the

main frame of the interface:

*--Results:*

In my project I used the “JUnitTest” for testing if the operations implemented on the polynomials. In the following I attached some screenshots from my project in order to exemplify those tests.



And the results of the tests are:

*--Conclusion:*

Java offers the real possibility that most programs can be written in a type-safe language. ... It extends Java with a mechanism for parametric polymorphism, which allows the definition and implementation of generic abstractions. The paper gives a complete design for the extended language

I personally, learned and better understood how to use classes and modifiers (which sets the class as public or private), modifiers are also used for variables and methods

What’s concerning the further developments, I have some ideas to adjust the interface to make it more attractive because in our days this is what an user is looking for, he or she has no concerns about how the application works or why it takes so much memory. Regarding to possible updates of the application, I have in mind to implement an algorithm which finds the roots of the polynomial and returns a graphical schema and also to increase performance so the application can work faster.

*--Bibliography:*

- <https://www.geeksforgeeks.org/>

- <https://www.wikipedia.org/>

- <https://stackoverflow.com/>