Sabou Cezara
Computer Science, year 2
group 30421

Polynomial Processing System

1.Objectives

1.1Main objective

The main objective of this project is to create a Java Application that could provide the user the ability to process polynomials in the following ways: add, subtract, multiply, divide, integrate and differentiate.

It should be able to give the correct result in the Graphical User Interface

1.2 Secondary objectives

The secondary objectives derive from the main objective:

1. The ability to store the polynomial in a data structure : the Polynomial is composed of an ArrayList of Monomials;
2. Being able to convert the given input into corresponding adequate candidates for processing;
3. Processing the polynomials: add, subtract, multiply, divide, integrate, differentiate;
4. Implementing a GUI(Graphical User Interface)

2. Problem Analysis

Polynomial processing helps a wide range of users, from computer advanced individuals to simple high school students. Therefore, the best option would be to create a GUI that not only is easy to use, but also offers correctness and precision.

I will describe the functionality of the system in the following paragraphs.

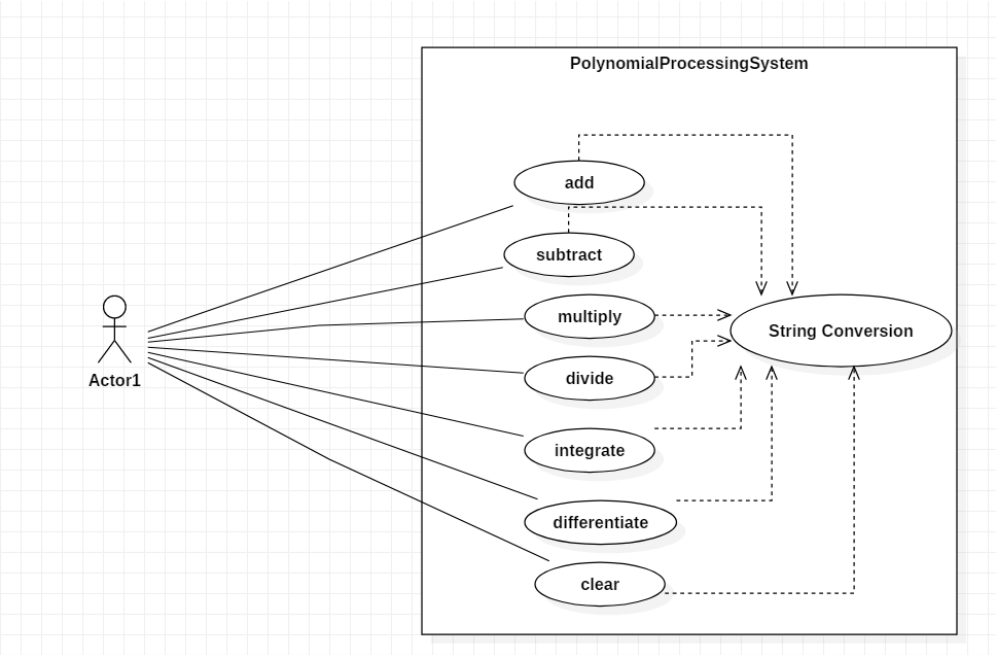
Before having to work with the polynomials, we first need to obtain them. They should be given as input in the two formatted text fields : firstPolynomial and secondPolynomial. They will be processed right after one of the buttons will be pressed.

The polynomial needs to be split into actual monomials, therefore the user needs to take into account that trying to give as input something different than what is given as example in the application, will not give the answer desired, instead it will generate an exception for invalid input.

Also, whenever addition, subtraction, multiplication and division are used, two polynomials are required. The other two: integration and differentiation will only take into account the values present in the text field corresponding to the first polynomial.

After correctly inserting the polynomials, the user only needs to press the button that represents the operation desired.

Sabou Cezara
Computer Science, year 2
group 30421



The use case diagram presents :

Use case title: process polynomial system

- Summary: this use case allows the user to process polynomials
- Preconditions: polynomials correctly inserted
- Main success scenario:
    - User gives the correct polynomial input
    - The Polynomial processing system efficiently checks the validity of the input
    - The user presses the button that represents the operation desired
    - The operation is done successfully
    - The result is displayed in the result text field
- Alternative sequences:
    - The input is not valid
    - The second polynomial has a higher degree than the first
    - Scenario returns to first step

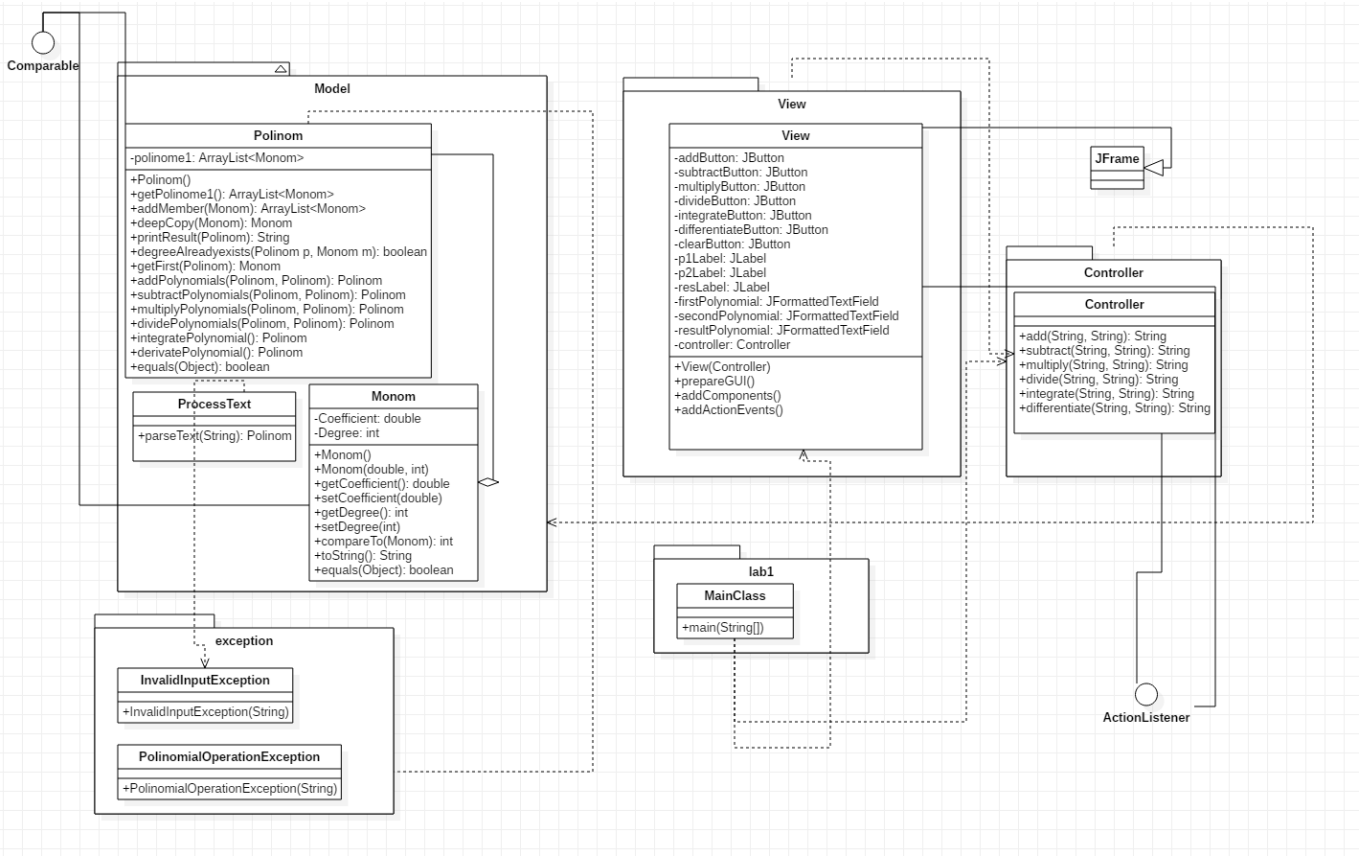
3.Design

Sabou Cezara
Computer Science, year 2
group 30421

The structure of the application is based on 5 different packages, some of them being structured in a MVC pattern:

1. The Model is the one which contains the two classes Polynomial and Monomial, and also the ProcessText class:
    - The Monom Class is in fact a very simple and easy to use class. It holds the coefficient and the degree of every polynomial member and is able to return the string version of the monomial.
    - The Polinom Class uses the Monom class and also it is described not only by the attributes, but also by all the methods implementing exactly the functionalities we need.
    - The textProcess just splits the string into Monomials and returns a new Polynomial with the given values.
2. The View is responsible for directly interacting with the user, but also helping the controller pick the correct commands adapted to the user's needs.
3. The Controller links the model and the View, managing the interactions.
4. The exception class was simply created to group the exceptions and separate them from the rest of the project.
5. The Main class is meant to only initiate the application, and it is not as important.

I created this UML diagram in order to show the main relationships of the application and to accentuate the MVC pattern structure.



Sabou Cezara
Computer Science, year 2
group 30421

4.Implementation

In the Model package:

4.1 Monomial

The “Monom” class is used to create all the polynomials that the processing system needs. It contains two private attributes: Coefficient, which is a double and Degree, which is of integer type. It also has a constructor which gives the monomial the values it is given in the parameter list.

Besides setters and getters, the monomial contains a “compareTo” method which returns the difference of degree between the current monomial and one given as a parameter.

The “toString()” method is customized to display exactly the form of monomial we will use in the polynomial class.

4.2 Polynomial

The “Polinom” class has as its only attribute a private and final arrayList of Monomials.

An important useful getter, would be getPolinome(), which simply returns the ArrayList. The addMember method is meant to add a new element in the monomial ArrayList, and consequently, the removeMember is meant to remove an existing element from the monomial ArrayList.

The method deepCopy takes as parameter a given monomial and returns a new monomial with the same values of the initial one; this method’s role is to keep the polynomials from changing their first form in case the user wants to compute more than one operation on the same polynomial.

The printResult is intuitive, and also the toString() method; as I mentioned above, it uses the toString() method from the monomial class.

I also created another method to check if I find the degree of a certain monomial already existing in that specific polynomial. This proves useful in the addPolynomials method and also subtractPolynomials.

The getFirst method returns the first element from a given polynome, I use it for testing the degree of two polynomials, in case the user wants to divide them.

We have the addPolynomials method, that puts the first polynomial in the result polynomial and then iterates through the second one: if it finds the same degree, then adds the coefficients, if not, I will end the for each loop. After this step I simply look if in the second polynomial there are any monomials left that were not added.

The subtractPolynomials method is mainly based on the addition method, except the fact that the sign of the coefficient is inversed.

The multiply method at first loops through the 2 polynomials and multiplies them one by one, and looks if the degree exists in the polynomial. If so, it adds the coefficient to the monomial with the same degree.

Sabou Cezara
Computer Science, year 2
group 30421

The divide method is a bit more inetersting, the first thing that I did was to check if the operation is possible, that is if the degree of the first polynomial is greater than the second. After that, I sort the two polynomials and I use a pioneer polynome that contains the element obtained from dividing the first monomial from the first polynomial to the first monomial of the second polynomial. I then multiply that pioneer with the second polynomial by using the multiplyPolynomials stated above, and I subtract the result from the first polynomial. I then keep using the result of the subtraction as the first polynomial until the degree is smaller than the second polynomial. The result will be the polynomial obtained just before stopping this while loop.

The derivatePolynomial method is straightforward: the coefficient becomes the result of the old coefficient multiplied with the degree and the new degree is the old degree decremented with one unit.

The integratePolynomial method is also very similar to the derivate method: it only changes the degree and the coefficient. The degree is simply incremented, and the coefficient becomes the ratio between the initial coefficient and the incremented degree.

4.3 ProcessText

This class only contains a method for parsing the text. It uses regular expression and it is able to split the text into exact monomial expressions. After obtaining it, the monomial is added to the polynomial result. For fining the regular expression the I used pattern matcher, which is easier and faster to use.

In the View package:

4.4 View

The view class is meant to implement the UI. It contains 6 different buttons, each holding the symbol corresponding to the operation the user wishes to compute, except the clear button, which is meant to empty the input textFields and the result textField. I also created three different text fields: in the first one the user should be able to insert the first polynomial, in the second one implicitly the second polynomial and the thitd text field should display the wanted result.

I also implemented 2 kinds of exception that will be thrown when trying to compute the operations: InvalidInputException and PolinomialOperationException. The InvalidInputException is meant to check for the existing monomial patterns, and if the program does not find the same structure as the pattern it will generate an error message asking the user to clear the fields and try to retype the input.

The PolinomialOperationException is generated if the user tries to divide two polyinomials, for which the first polynomial has a smaller degree than the second polynomial. An error message will appear, telling he user to try and insert proper polynomial inputs.

Sabou Cezara
Computer Science, year 2
group 30421

In the Controller package:

4.5 Controller

The implementation of the controller is fairly simple: it implements the methods of addition, subtraction, multiplication, division, integration and differentiation, by having some special methods that take as input the strings that represent the polynomials inserted by the user. The methods parse the text and the result after parsing is exactly in the polynomial form. Then it takes the polynomials and uses the methods described in Polynomial to give the final result, which is then converted to a string using the toString() method from the Polynomial class and returned back to the View, which will display it in the result text field.

In the exception package:

4.6 InvalidInputexception

This method generates an exception if the input is not valid. And is able to display the error message costumized.

4.7PolinomialOperationException

The same as the other method, generates an exception if the division operation cannot take place.

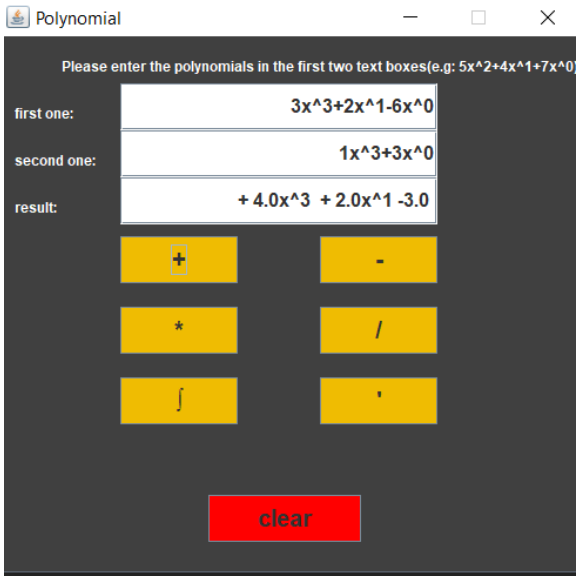
In the lab1 package:

4.8 MainClass

It links the view to the model and the controller. That is the only purpose of the main class.

5.Results

The results are being tested with a JUnit framework. Every method from the following six are tested and checked also for input correctness. The test is based on assertEquals method, which checks if the polynomial is correct and also matches it to the expected result. The match is based on the equals method present in the polynomial class, that refers to the equals method present in the monomial class.



Sabou Cezara
Computer Science, year 2
group 30421

Tested Method	Input	Expected Result	Actual Result	Pass/Fail
addition	- 3x^3+2x^1-6x^0 -1x^3+3x^0	+ 2.0x^1 -3.0	+ 4.0x^3 + 2.0x^1 -3.0	pass
subtraction	- 3x^3+2x^1-6x^0 -1x^3+3x^0	+ 2.0x^1 -9.0	+ 2.0x^3 + 2.0x^1 -9.0	pass
multiplication	- 3x^3+2x^1-6x^0 -1x^3+3x^0	+ 3.0x^6 + 2.0x^4 + 3.0x^3 + 6.0x^1 -18.0	+ 3.0x^6 + 2.0x^4 + 3.0x^3 + 6.0x^1 -18.0	pass
division	- 3x^3+2x^1-6x^0 -1x^2+3x^0	-7.0x^1 -6.0	-7.0x^1 -6.0	pass
integration	3x^3+2x^1-6x^0	+ 0.75x^4 + 1.0x^2 -6.0x^1	+ 0.75x^4 + 1.0x^2 - 6.0x^1	pass
differentiation	3x^3+2x^1-6x^0	+ 9.0x^2 + 2.0	+ 9.0x^2 + 2.0	pass

Sabou Cezara
Computer Science, year 2
group 30421

6.Conclusions

In conclusion, the application is created in OOP style and it serves as a polynomial processing system. A Graphical User Interface is created and easy to use, in order to offer the users an easy access to polynomial processing and to give back fast and correct results.

By developing this application I gained experience in trying to divide the classes into the Model View Controller pattern, understanding its functionalities and obtaining a clearer perspective of the project. I became more familiar to the ArrayList objects and also linking the methods designed by me to the front end.

I am aware that the application is not yet in its final form, and there are some adjustments that need to be done. The input conversion could be improved, as well as the interface. I also believe that there are more efficient ways in which I could implement the methods that compute the polynomial operations. I also would add other methods and options for the user such as: comparing two polynomials, computing the roots of the polynomials and so on.

7.Bibliography

https://www.tutorialspoint.com/java/index.htm

https://www.jetbrains.com/help/idea/maven-support.html

https://ro.wikipedia.org/wiki/Polinom

https://examples.javacodegeeks.com/desktop-java/swing/java-swing-layout-example/

https://docs.oracle.com/javase/tutorial/uiswing/events/intro.html