

DOCUMENTATIE

TEMA *1*

NUME STUDENT: Bodogae George Stefan
GRUPA: 30424

CUPRINS

DOCUMENTATIE.....	1
TEMA 1.....	1
CUPRINS.....	2
1. Main Objective	3
2. Analysis, modeling, scenarios, use cases	3
3. Design.....	Error! Bookmark not defined.
4. Implementation	5
5. Results.....	8
6. Conclusions.....	12
7. Bibliography.....	12

1. Main Objective

The main goal of the project is to implement in java a polynomial calculator which can add, subtract, multiply, divide, integrate and derivate polynomials. The implementation should be done in an OOP way by using the MVC model to organize the classes into packages. The program should have an easy to use GUI and error detection. Testing should be done with Junit and the test results shown in the documentation.

2. Analysis, modeling, scenarios, use cases

Polynomials are expressions of a variable which can be raised at different powers. Polynomials can be split into monomials, which are the terms of the polynomial. A monomial is basically a polynomial with only one term. It is also called a power product and it is a product of powers of nonnegative integer exponents. Polynomials can be used for many applications, but in this one the focus will be on the following operations: addition, subtraction, multiplication, division, integration and derivation.

The addition of polynomials can be done by grouping all the terms together and then reordering and combining terms which have the same exponents.

Ex. $P1=2x^2+3x+1$; $P2=3x+2$; then $P1+P2=2x^2+6x+3$

The subtraction is done similarly to the addition, but this time we add all the monomials from the second polynomial with reversed coefficient sign to the second polynomial.

Ex. $P1=2x^2+3x+1$; $P2=3x+2$; then $P1-P2=2x^2-1$

The multiplication operation is done by taking the polynomial sums of monomials and multiplying each monomial from the first polynomial with each monomial from the second polynomial. After the multiplication is done it is a good practice to reorder the monomials with respect to their exponents and then adding the terms of the same exponent

Ex. $P1=2x^2+3x+1$; $P2=3x+2$; then $P1*P2 = 6x^3+4x^2+9x^2+6x+3x+2$

Then by reordering the powers and adding the terms which have the same exponent we get:

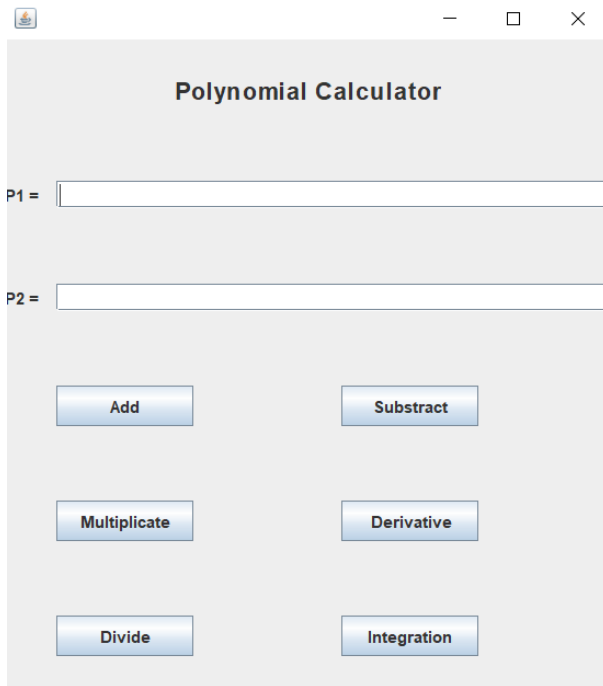
$P1*P2=6x^3+13x^2+9x+2$

The division is done by dividing one polynomial by another polynomial of the same or lower degree (greatest power monomial). This operation yields two results: the quotient and the remainder which are of type polynomial.

Ex. $P1=2x^2+3x+1$; $P2=3x+2$; then $P1/P2=2/3x+5/9$, remainder = $-1/9$

The interface consists of 6 buttons for the operations and 2 formatted text fields for the input polynomials. The results will be displayed in the first formatted text field “P1”. The operations which use only one input polynomial will take the first one and the result of the operations will also be displayed there. For exiting the application, the user will simply press the “exit” situated at the top right corner of the window.

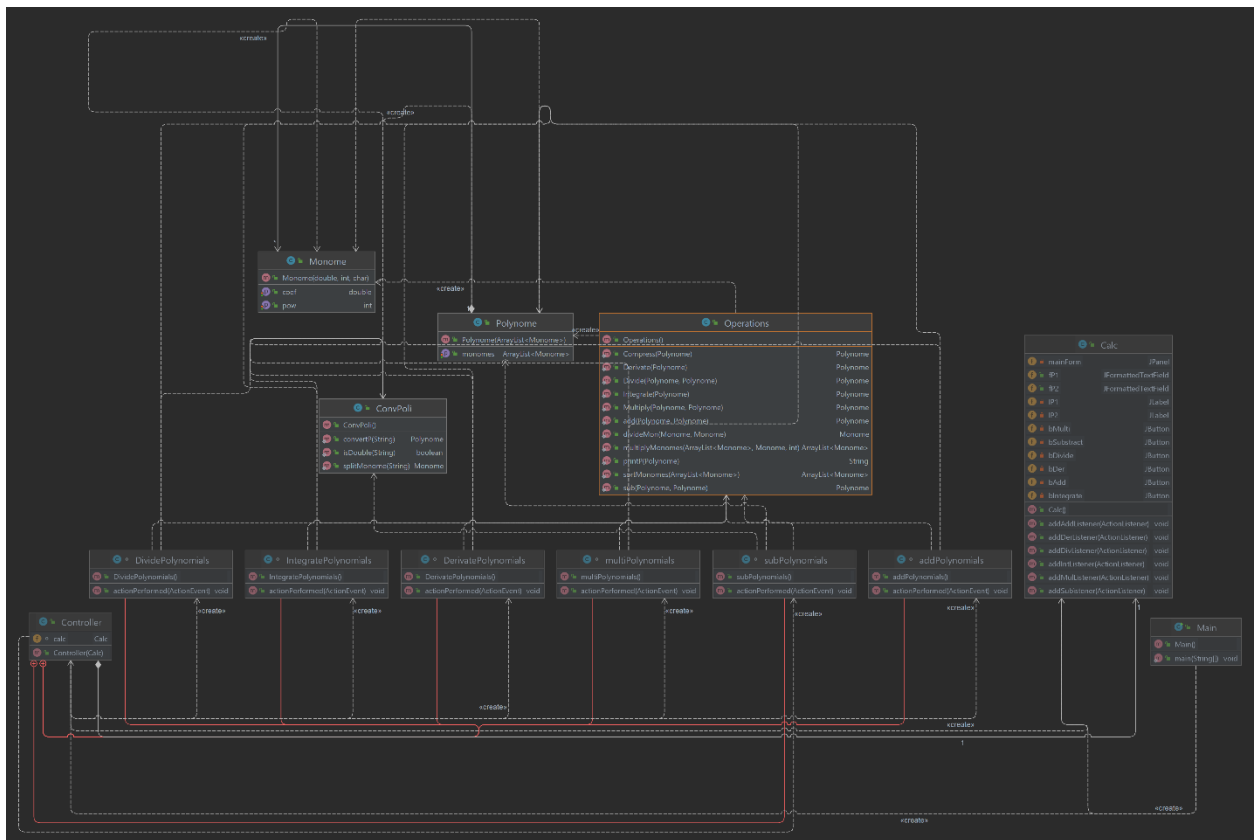
The user will enter one or two polynomials of the type “ $c_1x^n+c_2x^{n-1}...$ ” depending on the operation they want to apply (1 polynomial for the integration/derivation and 2 polynomials for the addition, subtraction, multiplication and division). Then they will right-click on the desired operation and the result will be displayed in the “P1” text field. If the user wishes to use the result polynomial for further operations, they simply leave the result of the past operation as is and press another operation button.



The image shows a screenshot of a Java Swing window titled "Polynomial Calculator". The window has a standard title bar with a small icon on the left and minimize, maximize, and close buttons on the right. The main content area has a light gray background. At the top, the title "Polynomial Calculator" is centered in a bold black font. Below the title, there are two text input fields. The first field is preceded by the label "P1 =" and the second by "P2 =". Both fields are empty. Below the input fields, there are six buttons arranged in a 3x2 grid. The buttons are labeled "Add", "Subtract", "Multiply", "Derivative", "Divide", and "Integration". Each button has a light blue gradient and a thin black border. The "Derivative" button is slightly offset to the right compared to the others in its row.

3. Implementation

Class diagram:



This class diagram shows all the classes with the afferent methods and class variables and also the dependencies that occur between classes. In this case, the dependencies are all create actions, which instantiate the classes, defining objects of that type using the default constructors(For example: Polynomial poli = new Polynomial());

I used an OOP approach for this implementation by designing every class to serve a well defined purpose and also to be able to organize them into packages according to the MVC model. The class diagram from above shows the way I implemented the classes with their methods in order to simplify the flow of the program.

The preferred method of storing monomials was `ArrayList<Monome>` because it is easier to use and more efficient than the Java Array. For the coefficients I used double and int for the power of the monomial. For the input I used String because it is the direct and way of getting input from the user, and then I implemented the ConvPoli Class to convert the String into Polynomial and Monomial type objects for easier and faster processing.

Model, View, Controller(MVC) is a design scheme for logic division of the programming elements such that they exist in an interconnected manner. Though MVC comes from WWW programming, it is very beneficial to work and organize a program in this way because it makes the development process tidier and it allows for easier further implementation by keeping the files organized. It was the first time working with this kind of architecture for me and I am positive that it helped me during the process of development of my application.

I used packages to organize the project's classes with respect to the MVC model(model,view,controller), such that the class "Main" and the class "Controller" are in the "controller" package, "Calc " is in the "view" package and ConvPoli, Monome, Polynome and Operations are in the "model" package.

1. Monome class

The Monome class is used for defining the monomial.

Constructors:

```
public Monome(double coef, int pow)
```

Methods:

```
public double getCoef() //getters and setters for the class variables
public void setCoef()
public int getPow()
public void setPow()
```

2. Polynome class

The Polynome class is used to define the polynomial as an ArrayList of monomials.

```
Constructors:public Polynome(ArrayList<Monome> monomes)
```

Methods:

```
public ArrayList<Monome> getMonomes()
public void setMonomes(ArrayList<Monome> monomes)
//getters and setters for the class variables
```

3.ConvPoli class

This is a class which is used for converting the input strings given by the user into monomials and then object by using regex and then using logic to convert the monomials into a Monome object.

This class only contains static methods so it doesn't have any constructors.

Methods:

```
public static boolean isDouble (String s)
-used to check if a string can be parsed into a double variable
```

```
public static Monome splitMonome(String mon)
-This method is used to split a monomial received through a parameter as a string into a Monome object.
```

```
public static Polynome convertP(String exp)
```

-This method is used to convert strings given as parameters into Polynome objects using regex. The regex pattern that was used is:

```
" ([+-]? [^+-]+) "
```

4.Operations class

This class contains methods for all operations and also for printing

```
public static String printP(Polynome p)
```

This method is used to convert a Polynome object back into a string so it can be displayed as a result.

```
public static Polynome add(Polynome p1, Polynome p2) //addition
public static Polynome sub(Polynome p1, Polynome p2) //subtraction
public static Polynome Multiply(Polynome p1, Polynome p2) //Multiplication
public static Polynome Derivate(Polynome p1) //Derivative
public static Polynome Integrate(Polynome p1) //integration
public static Polynome Divide(Polynome p1, Polynome p2)
```

```
public static ArrayList<Monome> sortMonomes (ArrayList<Monome> m)
```

This method is used to sort an ArrayList<Monome> after the multiplication operation, as the terms need to be sorted descending by the power of the monomials.

```
public static Polynome Compress(Polynome p)
```

This method is used to add the coefficients of monomials when there are multiple monomials with the same power.

This is also used after multiplication.

5.Calc class

This class is used to create the gui using java swing. It extends JFrame and contains all the buttons, fields and labels of the mainForm and also the action listeners for the buttons. Each listener points to the Controller Class from the controller package s

The class variables of this class are:

```
private JPanel mainForm;
public JFormattedTextField fp1;
private JFormattedTextField fp2;
private JLabel lp1;
private JLabel lp2;
private JButton bMulti;
private JButton bSubtract;
private JButton bDivide;
private JButton bMod;
private JButton bAdd;
private JButton bExit;
```

and the action listeners:

```

public void addAddListener(ActionListener e) {bAdd.addActionListener(e);}
public void addSubListener(ActionListener e) {bSubtract.addActionListener(e);}
public void addMulListener(ActionListener e) {bMulti.addActionListener(e);}
public void addDivListener(ActionListener e) {bDivide.addActionListener(e);}
public void addIntListener(ActionListener e) {bIntegrate.addActionListener(e);}
}
public void addDerListener(ActionListener e) {bDer.addActionListener(e);}

```

6. Main class

This class only contains the main() function. Here I declared Two objects. One of type Controller which takes control over the functionality of the buttons and an object of type Calc which is the GUI part of the project. This class is the starting point of the application, as the main method is called when the program first runs.

```

public class Main {

    public static void main(String[] args) {
        Calc calc = new Calc();
        Controller controller = new Controller(calc);
        calc.setVisible(true);
    }
}

```

7. Controller class

The controller class is used for controlling the operations and to make the connection between the Calc class and the Operations class. It contains multiple classes that implement java ActionListener each class has the ActionPerformed methods corresponding to each operation from the Operations class. Initially I used the action listeners directly into the Calc class but the MVC model dictates that the methods of the listeners should be in the Controller package as it is more of a part of control than a visual part of the project. The best option was to create a new class named Controller which contains these methods. The classes in the Controller class are:

```

class addPolynomials implements ActionListener
class subPolynomials implements ActionListener
class multiPolynomials implements ActionListener
class DividePolynomials implements ActionListener
class DerivatePolynomials implements ActionListener
class IntegratePolynomials implements ActionListener

```

4. Results

The testing was done using Junit and by hardcoding some Polynomials to do operations with. All the tests passed and below I will list the test classes:

```

public class AddOperationTest
public class SubOperationTest
public class MultiOperationTest

```



```

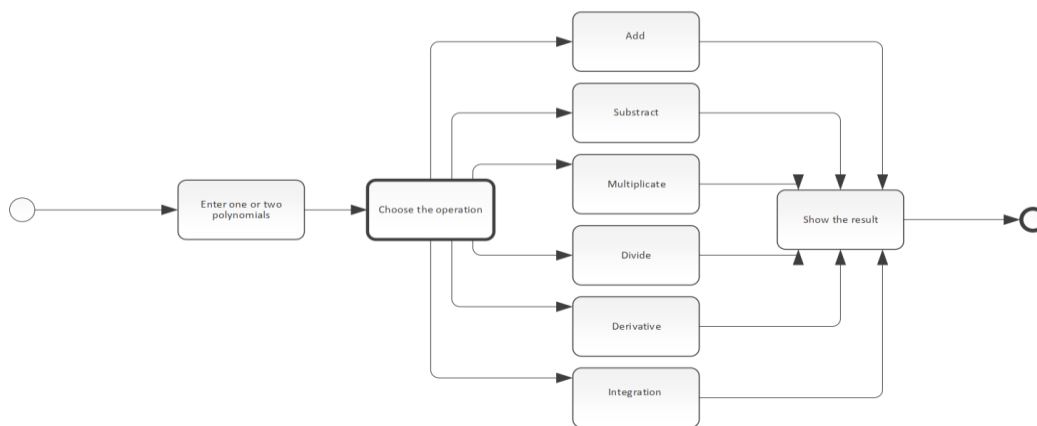
public class IntegrateOperationTest
public class DerivateOperationTest
public class DivideOperationTest

```

Operation	Expected Output	Output	Verdict
" $2x^2+3x+2$ " + " $3x^3$ "	" $3x^3+2x^2+3x+2$ "	" $3x^3+2x^2+3x+2$ "	Pass
" $2x^2+3x+2$ " - " $3x^3$ "	" $-3x^3+2x^2+3x+2$ "	" $-3x^3+2x^2+3x+2$ "	Pass
" $2x^2+3x+2$ " * " $3x^3$ "	" $6x^5+9x^4+6x^3$ "	" $6x^5+9x^4+6x^3$ "	Pass
" $2x^2+2x$ " / " x^2+x "	"2"	"2"	Pass
"($2x^2+3x+2$)'"	" $4x+3$ "	" $4x+3$ "	Pass
"Integral($3x^2+2x+2.0$)"	" $x^3+x^2+2.0x$ "	" $x^3+x^2+2.0x$ "	Pass

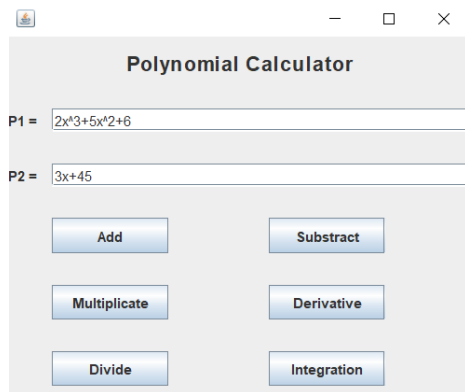
As it can be seen from the results above, all the tests that were done using JUnit were passed with no issues, so the algorithms were working correctly and with no hidden errors.

The next pictures will exemplify how the user would use the polynomial calculator by performing all the possible operations(addition, subtraction, multiplication, division, derivation and integration). The samples of polynomials were chosen at random for the sake of diversity. This simple tutorial should teach the user how to fully use the polynomial calculator with in-use screenshots.



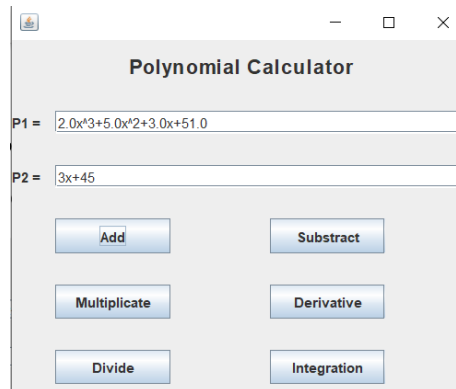
1. Addition example:

Before operation:



A screenshot of a web application titled "Polynomial Calculator". It has two input fields: "P1 =" with the value $2x^3+5x^2+6$ and "P2 =" with the value $3x+45$. Below the inputs are six buttons arranged in a 3x2 grid: "Add", "Subtract", "Multiply", "Derivative", "Divide", and "Integration".

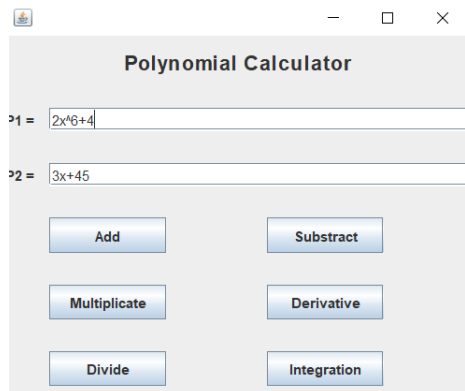
Result shown in P1 text field:



A screenshot of the same "Polynomial Calculator" after the "Add" operation. The "P1 =" field now contains the result $2.0x^3+5.0x^2+3.0x+51.0$, while the "P2 =" field remains $3x+45$. The buttons are the same as in the previous screenshot.

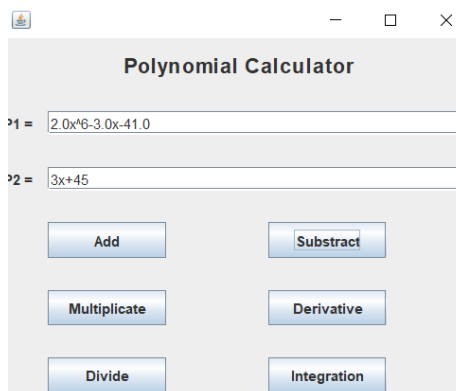
2. Subtraction example:

Before operation:



A screenshot of the "Polynomial Calculator" with "P1 =" set to $2x^6+4$ and "P2 =" set to $3x+45$. The buttons "Add", "Subtract", "Multiply", "Derivative", "Divide", and "Integration" are visible.

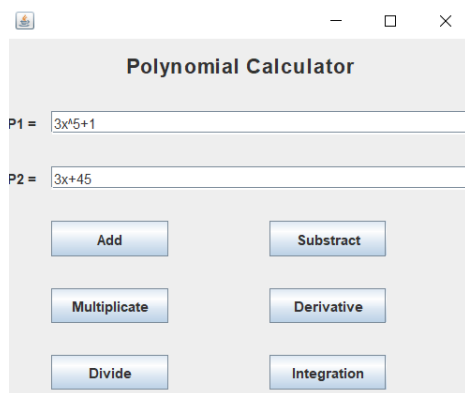
Result shown in P1 text field:



A screenshot of the "Polynomial Calculator" after the "Subtract" operation. The "P1 =" field now shows $2.0x^6-3.0x-41.0$, and "P2 =" remains $3x+45$. The buttons are the same.

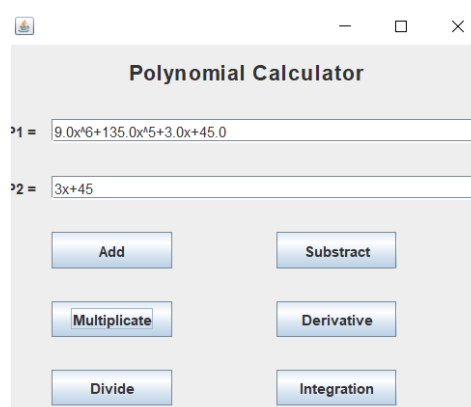
3. Multiplication example:

Before operation:



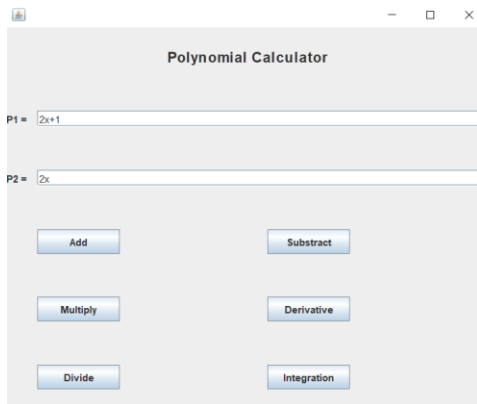
A screenshot of the "Polynomial Calculator" with "P1 =" set to $3x^5+1$ and "P2 =" set to $3x+45$. The buttons "Add", "Subtract", "Multiply", "Derivative", "Divide", and "Integration" are visible.

Result shown in P1 text field:



A screenshot of the "Polynomial Calculator" after the "Multiply" operation. The "P1 =" field now contains the result $9.0x^6+135.0x^5+3.0x+45.0$, and "P2 =" remains $3x+45$. The buttons are the same.

4. Division example:



Polynomial Calculator

P1 = $2x+1$

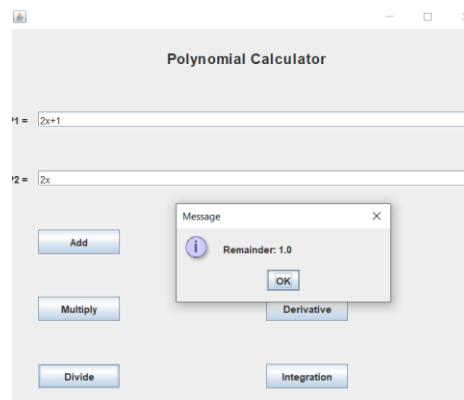
P2 = $2x$

Add Subtract

Multiply Derivative

Divide Integration

Result shown in P1 and remainder shown in popup:



Polynomial Calculator

P1 = $2x+1$

P2 = $2x$

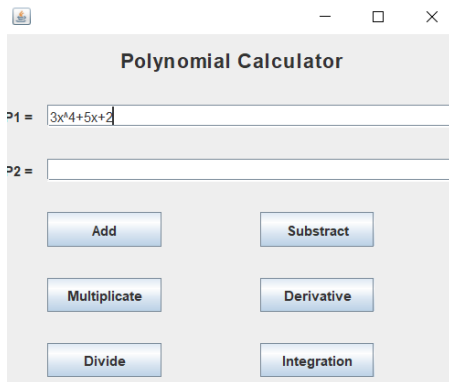
Add Subtract

Multiply Derivative

Divide Integration

Message
i Remainder: 1.0
OK

5. Derivation example:



Polynomial Calculator

P1 = $3x^4+5x+2$

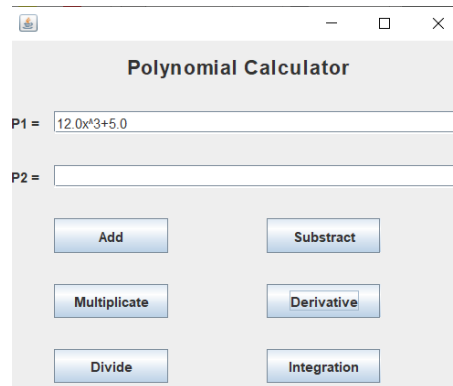
P2 =

Add Subtract

Multiplicate Derivative

Divide Integration

Result shown in P1 text field:



Polynomial Calculator

P1 = $12.0x^3+5.0$

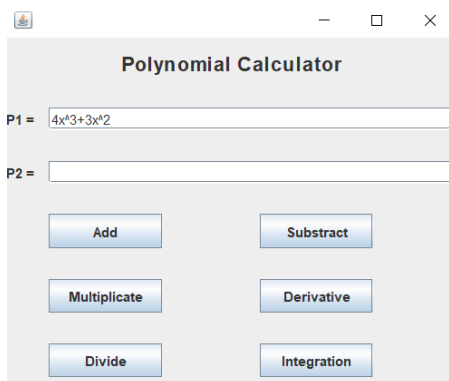
P2 =

Add Subtract

Multiplicate Derivative

Divide Integration

6. Integration example:



Polynomial Calculator

P1 = $4x^3+3x^2$

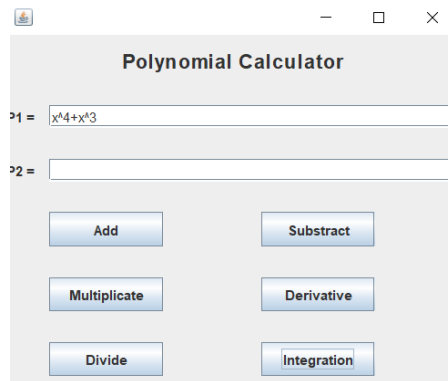
P2 =

Add Subtract

Multiplicate Derivative

Divide Integration

Result shown in P1 text field:



Polynomial Calculator

P1 = x^4+x^3

P2 =

Add Subtract

Multiplicate Derivative

Divide Integration

6. Conclusions

This assignment was very beneficial in expanding my OOP and Java knowledge, the final product is an usable polynomial calculator. The process of programming this calculator was more time consuming than I expected, but the increase in knowledge when dealing with techniques and different structures was worth it. Moreover, there could be implemented a way of doing other operations on polynomials such as polynomial solving for one variable, polynomial system solving operation with two or more variables and this program could also be turned into a learning/quiz platform where the users' knowledge would be tested with polynomial operations and they would enter the answer in a text space. This is a small and simple application, but overall I am pleased with my first polynomial calculator.

7. Bibliography

1. [Stack Overflow - Where Developers Learn, Share, & Build Careers](#)

2. [GeeksforGeeks | A computer science portal for geeks](#)

The first two links were used for understanding the concepts and specifically for learning how to use regex and manipulating data with Java ArrayList<>.

3. [Wikipedia](#)

Wikipedia was used for understanding polynomials and monomials and also the way the operations should be done, especially the long division which was more demanding and had a more complex algorithm.

4. [Iterate faster, innovate together | GitLab](#)

GitLab was used to constantly keep the project updated. This helps the developer to keep the team(or the supervisor in my case) up to date with the changes and current status of the project at all times. This came in really handy when I had to work on the project from different devices and GitLab enabled me to always have a fresh copy of my project at all times.