Technical University of Cluj-Napoca

Programming Techniques

Assignment 1

**Polynomial Calculator**



Teacher: Ioan Salomie

Teacher Assistant: Viorica Chifu

Student: Stioiu Denis-Adrian

Group:30424

1. Project Objective

To design and implement a polynomial calculator with a dedicated graphical interface through which the user can enter polynomials, select the operation to be performed: addition, subtraction, multiplication, division, derivative, integration, and to display the obtained result.

1. Problem analysis, modelling, scenarios, use case

The application can be used by students and high school students to solve elementary algebraic problems with polynomials. The application will be used:

|  |  |
| --- | --- |
| Step | Description |
| 1 | Insert a polynomial in the first JTextField |
| 2(except for derivation and integration) | Insert a polynomial in the second JTextField |
| 3 | Compute the chosen operation by pressing a button |
| 4 | The result will be displayed using a JLabel |

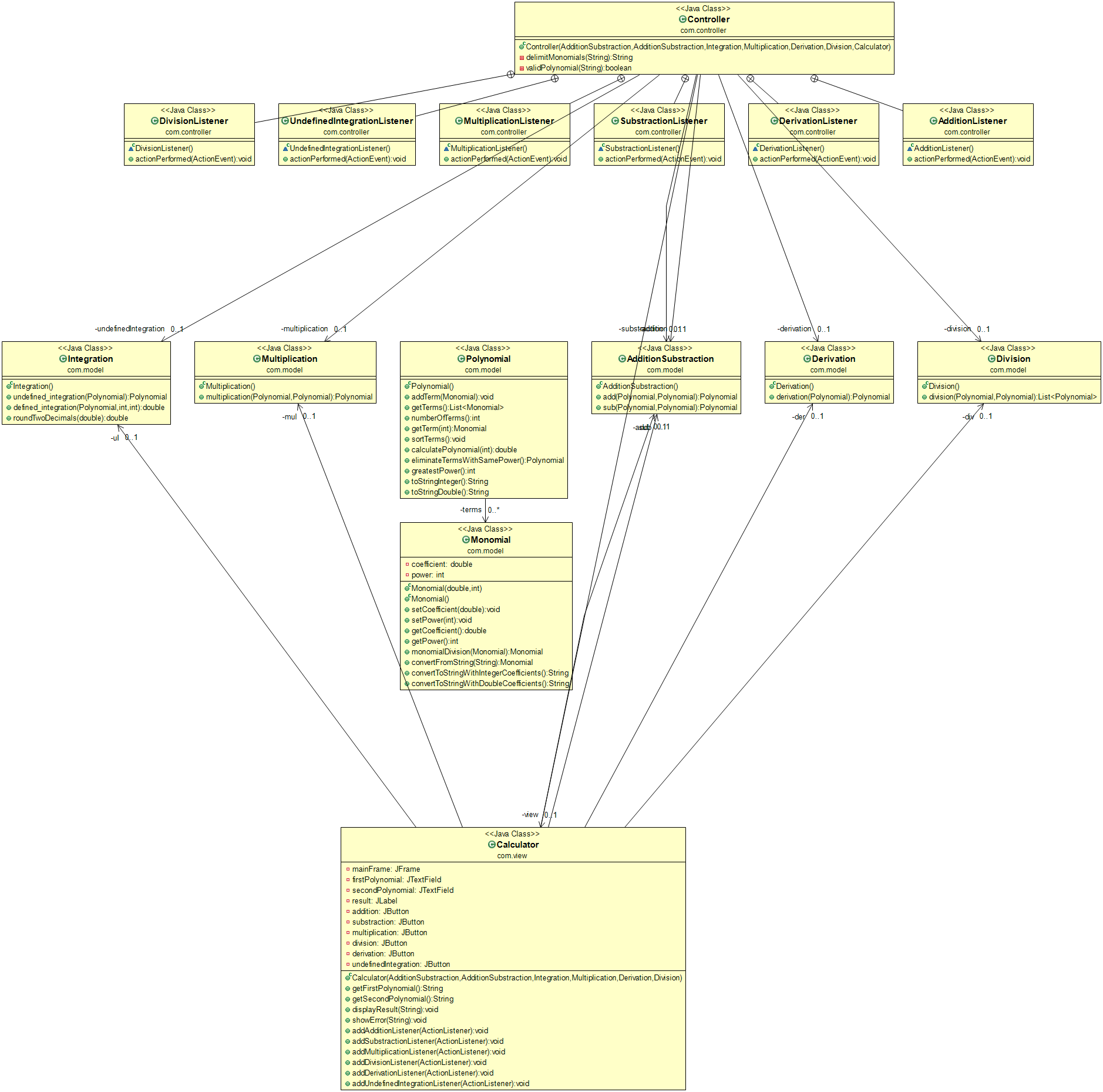
1. Design

For designing the application I chose the MVC(Model View Controller) architectural pattern. For that I used three packages: Model, where I stored the operations, Monomial and Polynomial classes, View, where I designed the interface of the application, and Controller, where I linked Model and View. In the class Main I implemented all the three packages. The MVC is a software pattern that has as the main advantages: enabling logical grouping of related actions on a controller together. It is easy to modify due to the separation of responsibilities, the future development or modification becoming easier.

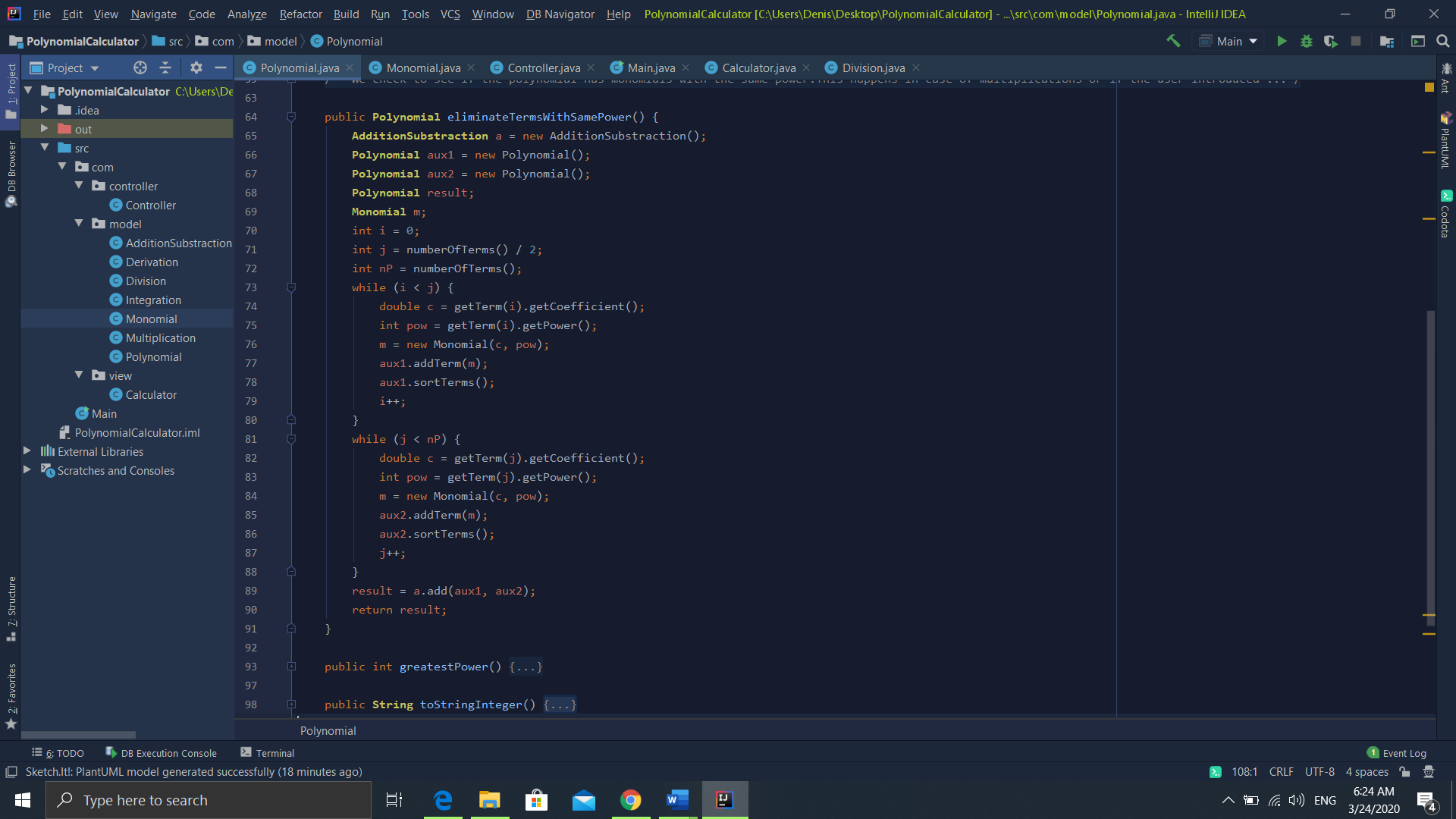
The application has been written in Java programming language, the IntelliJ environment. It was developed using class based programming, in which inheritance is achieved by defining classes of objects, as opposed to the objects themselves. It uses inheritance for code reuse and extensibility in the form of either the classes or prototypes.

The UML diagram is a diagram based on the UML(Unified Modeling Language), is a modern approach to modeling and documenting software. It is based on diagrammatic representation of software components. The top compartment contains the name of the class. It is printed in bold and centered, and the first lette is capitalized. The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase. The bottom compartment contains the operations the class can execute. To specify the visibility of a class member, the following notations must be placed before the member’s name:+ for public, - for private, # protected and ~ for package.

The UML diagrams for the classes:



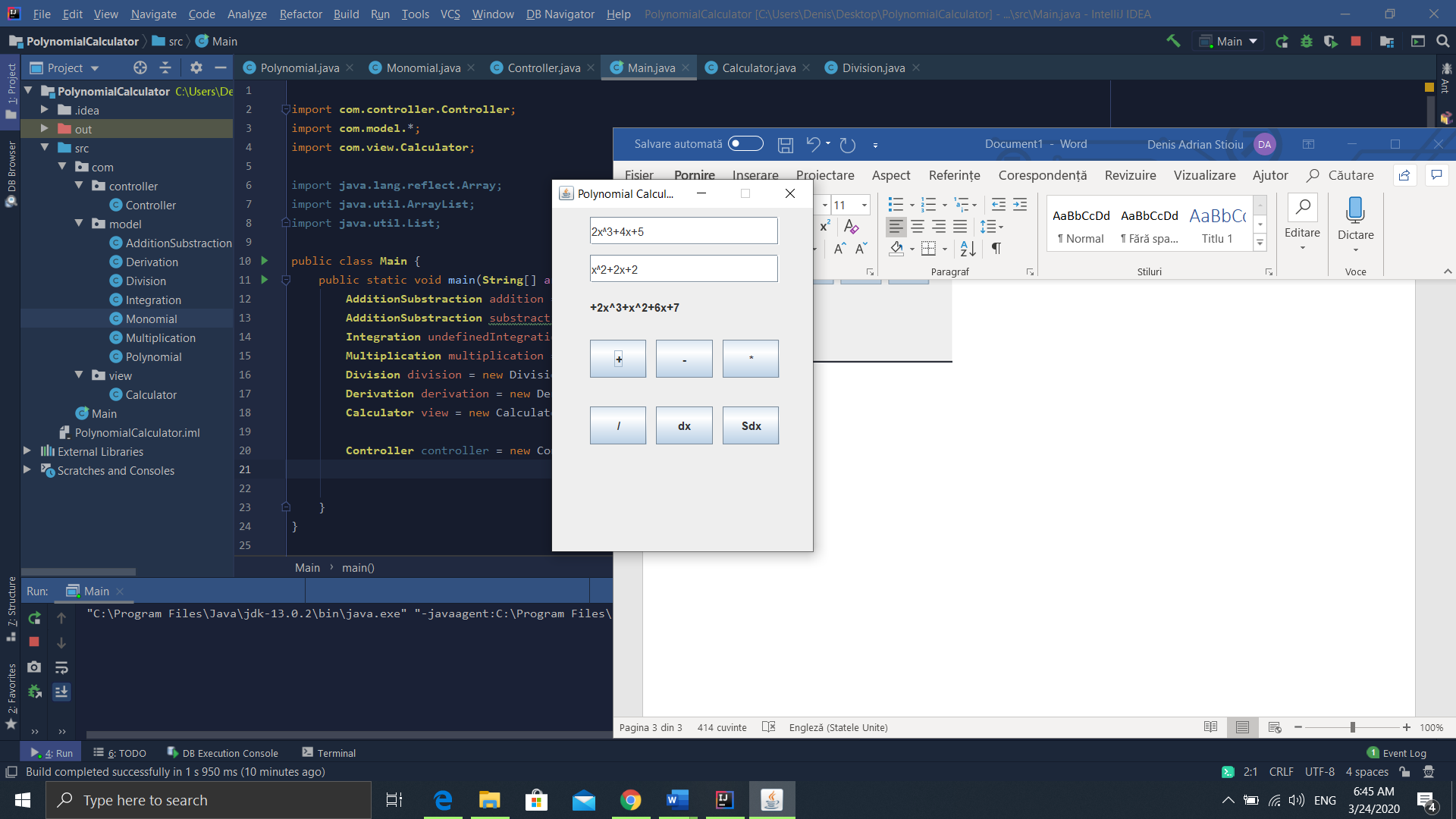
I used lists on two occasions. Firstly, when declaring the Polynomial as a list of monomials, and secondly, for the division operation, when I stored the quotient and the remainder inside a list of polynomials.



I chose the following algorithm to arrange the polynomial after multiplication, when there are more terms with the same power.

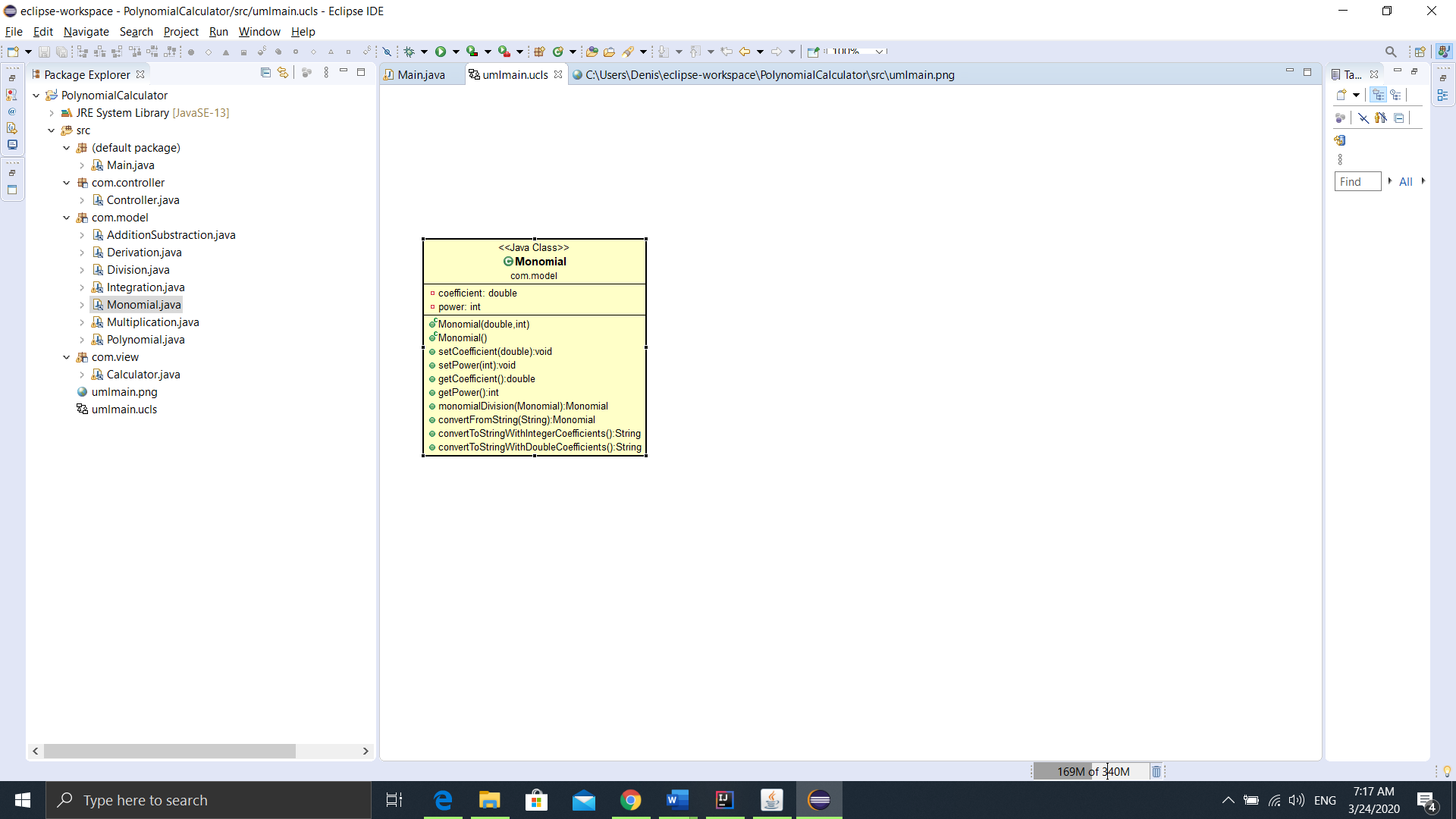
As we can see from the picture above, I used two auxiliary polynomials , where I stored half of the the monomials in each one. Then, I added the two halves, and I stored the new result in an updated Polynomial which I returned.

1. Implementation
2. Calculator class

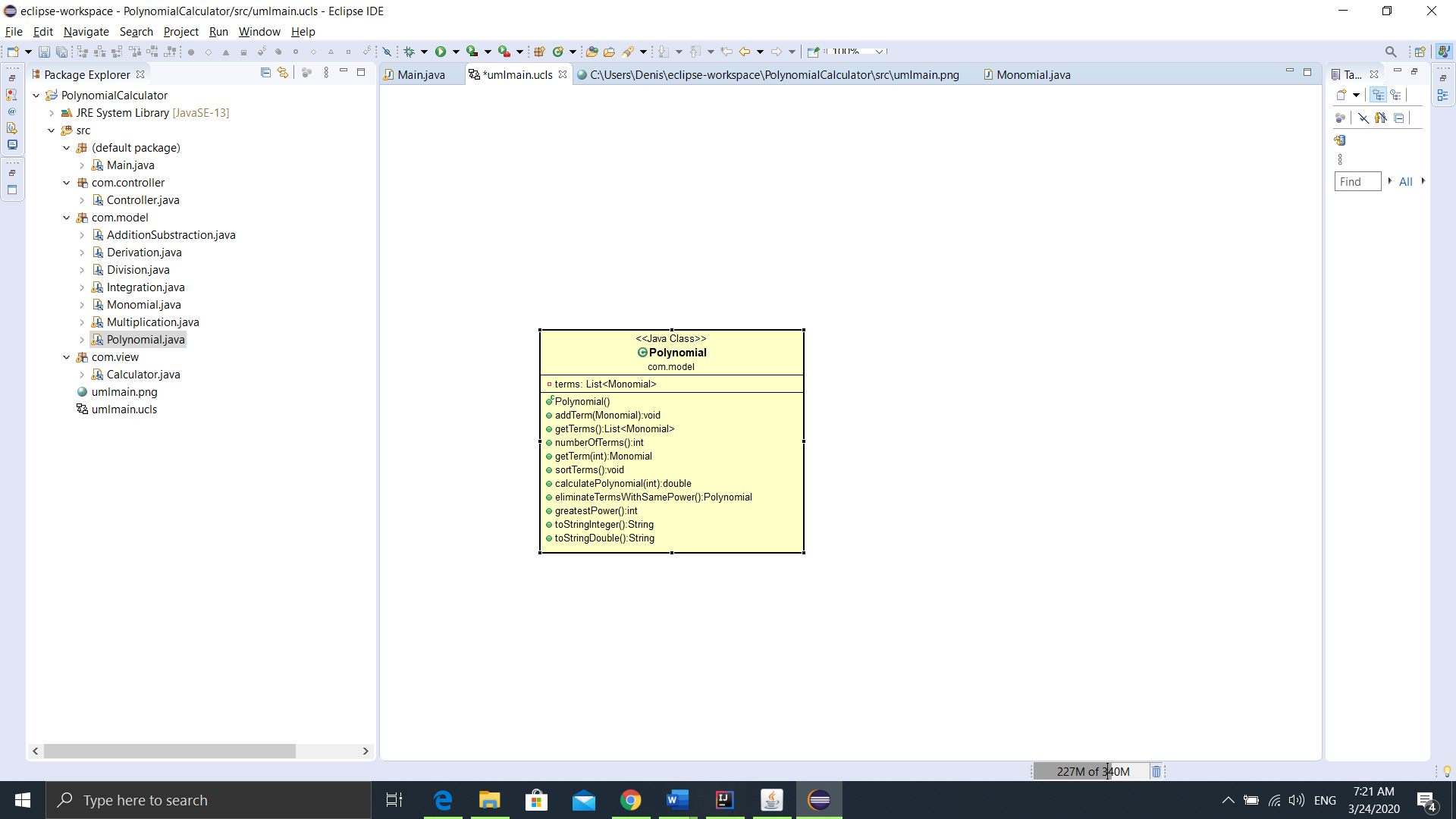
 It is situated in the view package and it contains the interface of the project. The interface consists of a JFrame which contains 2 JTextFields, 6 buttons and a JLabel. The first JtextField is for the the first polynomial, and it will be the one from where we will take the value for derivation and integration. The six buttons are used for each operation( + for Addition, - for Subtraction, \* for multiplication, / for division, dx for derivation, Sdx for integration). When a button is pressed the operation is executed after the inputs are checked. The result is displayed in the JLabel under the second JTextField. The picture on the left is an example of a performed operation, addition. I chose to implent a user friendly interface, in which the instructions can be easily deduced by looking at the JFrame, allowing the application to be used almost as a normal pocket computer.

Below are presented the model classes. The model is the central component of the pattern. It is the application’s dynamic data structure, independent of the user interface. It directly manages the data, logic and rules of the application. The inputs are received from the controller, which performs interactions on the data model objects.

b. Monomial Class

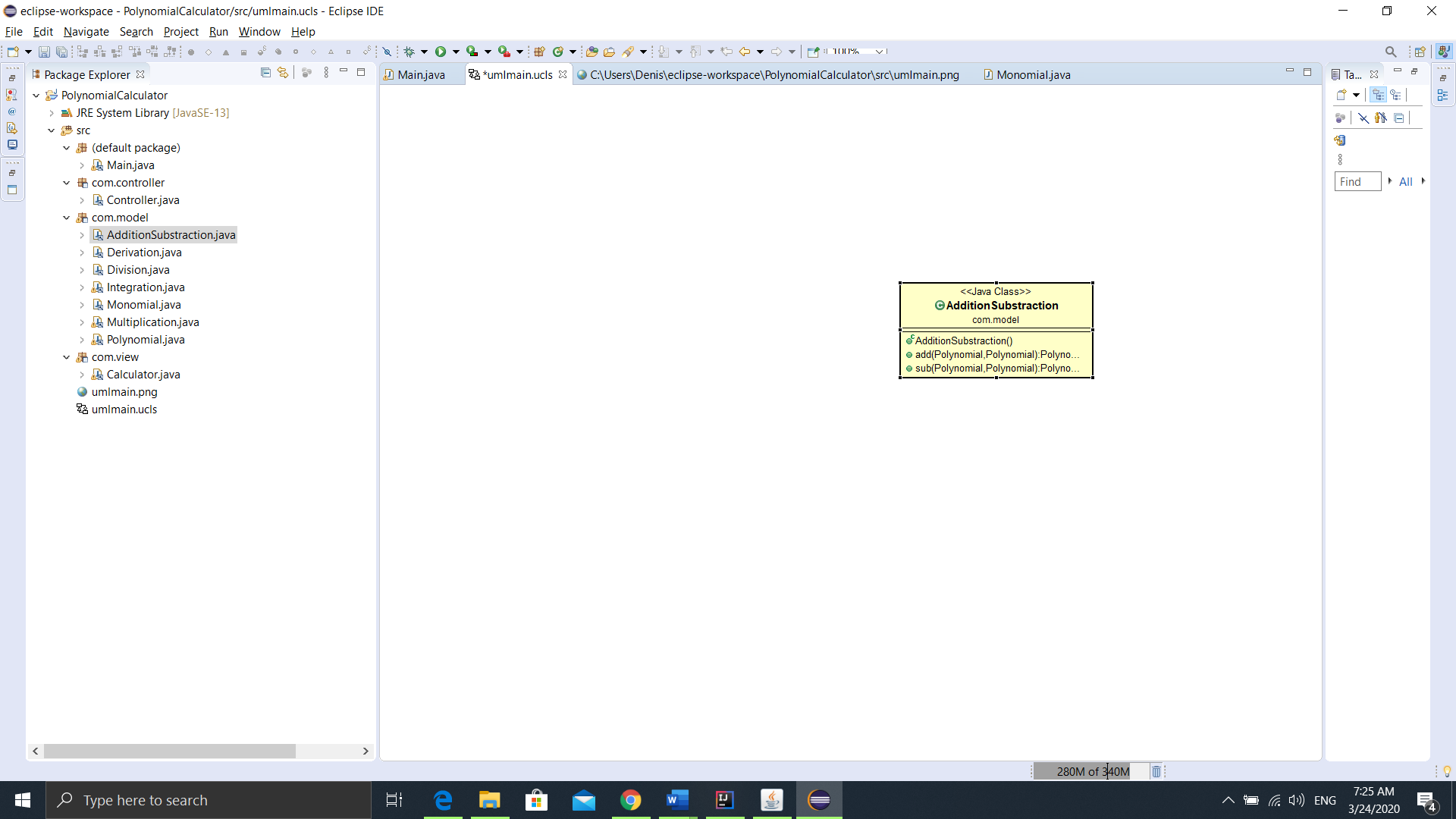
It contains two private fields, the Coefficient and Power. The monomialDivision() method divides two monomials and it is used at calculating the division, in which the divider monomial will have its coefficient set at coefficientDivider / coefficientDivident, and the power set as powerDivider -powerDivident. It also has the method convertFromString() that takes a string parameter. The parameter is then checked using RegEx to see if it can be considered a monomial, in the following formats: sign-digit-x-^-digit, digit-x-^-digit,sign-digit-x,digit-x,sign-digit,digit,sign-x,x,x-^digit,sign-x-^-digit.Then we use a method to print the monomial in Latex format with integer coefficients or with double coefficients.

c.Polynomial Class



It has only one field, the List of monomials. It has a sorting method in order to ordinate the terms in the descending order of their power. The method getTerm() with an int argument is used the get a monomial situated on a specified position. It is used in all the operations. Also there is a calculatePolynomial() that solves the polynomial for a value of x. There are two methods of print, one that prints integer coefficients, and the other one that prints double coefficients for the integration and division operations. addTerm() is a method that inserts a new monomial in the polynomial. It is mostly used when converting a string into monomials, and then inserting them into the polynomial. The method greatestPower it is used to return the term with the greatest power of the polynomial. It is used at the division operation.

d. AdditionSubtraction Class

This class contains the operations for both the addition and subtraction. The only methods of this class are the operations. The addition works on the following algorithm: we add the uncommon power of the monomials into the sum polynomial, and for the common power we add the sum of the coefficients and the power. For the subtraction we multiply by -1 all the coefficients of the second polynomial and the we add them normally using the same algorithm.

Example:

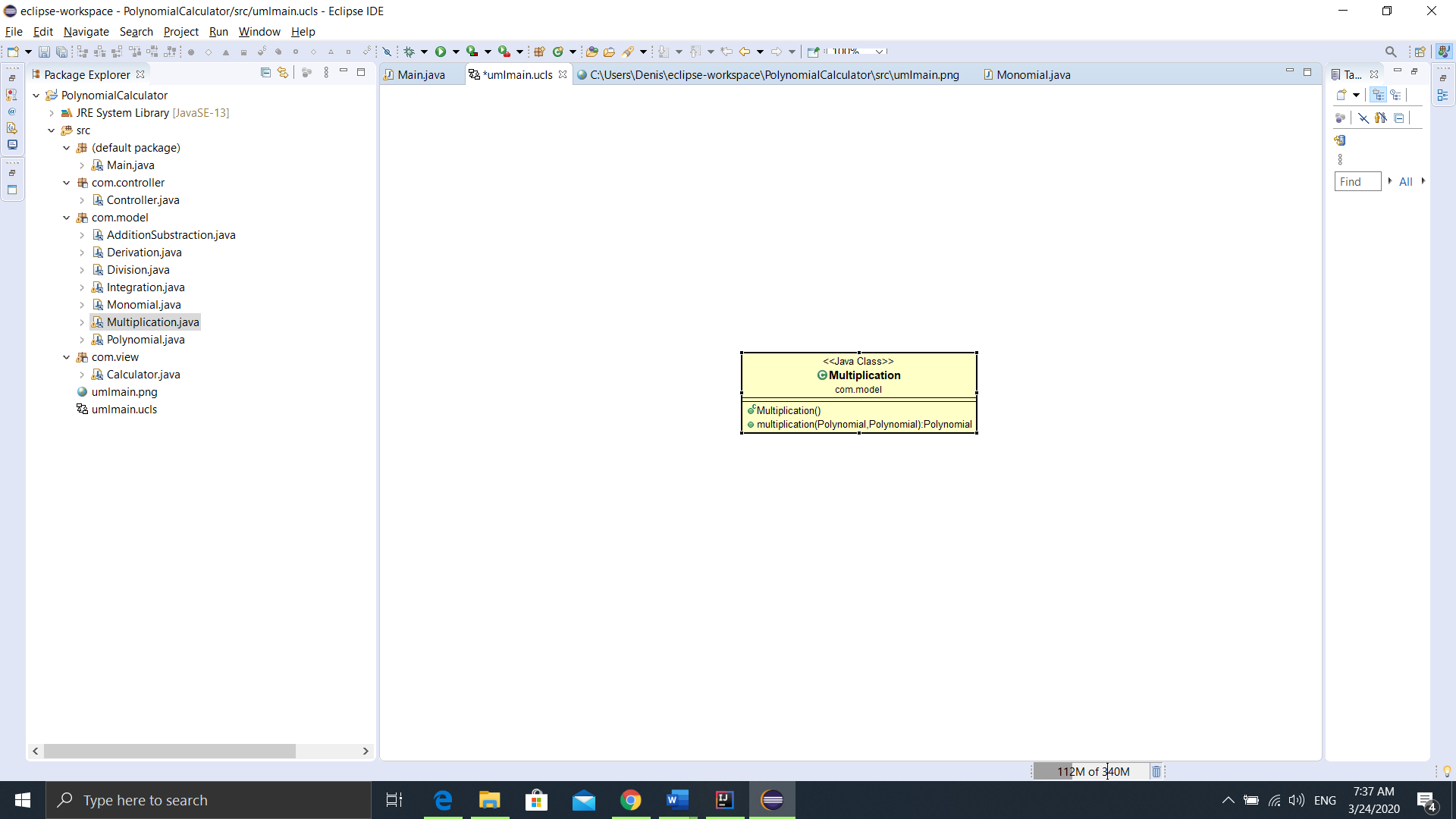
(4x^2 + 3x − 14) − (x^3 − x^2 + 7x + 1) = (4x^2 + 3x − 14) − (x^3 − x^2 + 7x + 1)=

=4x^2 + 3x − 14 − x^3 + x^2 − 7x − 1 = 4x^2 + 3x − 14 − x^3 + x^2 - 7x − 1

=−x^3 + (4x^2 + x^2) + (3x − 7x) + (−14 − 1) = −x^3 + (4x^2 + x^2) + (3x − 7x) + (−14 − 1)

=−x^3 + 5x^2 − 4x − 15

e. Multiplication

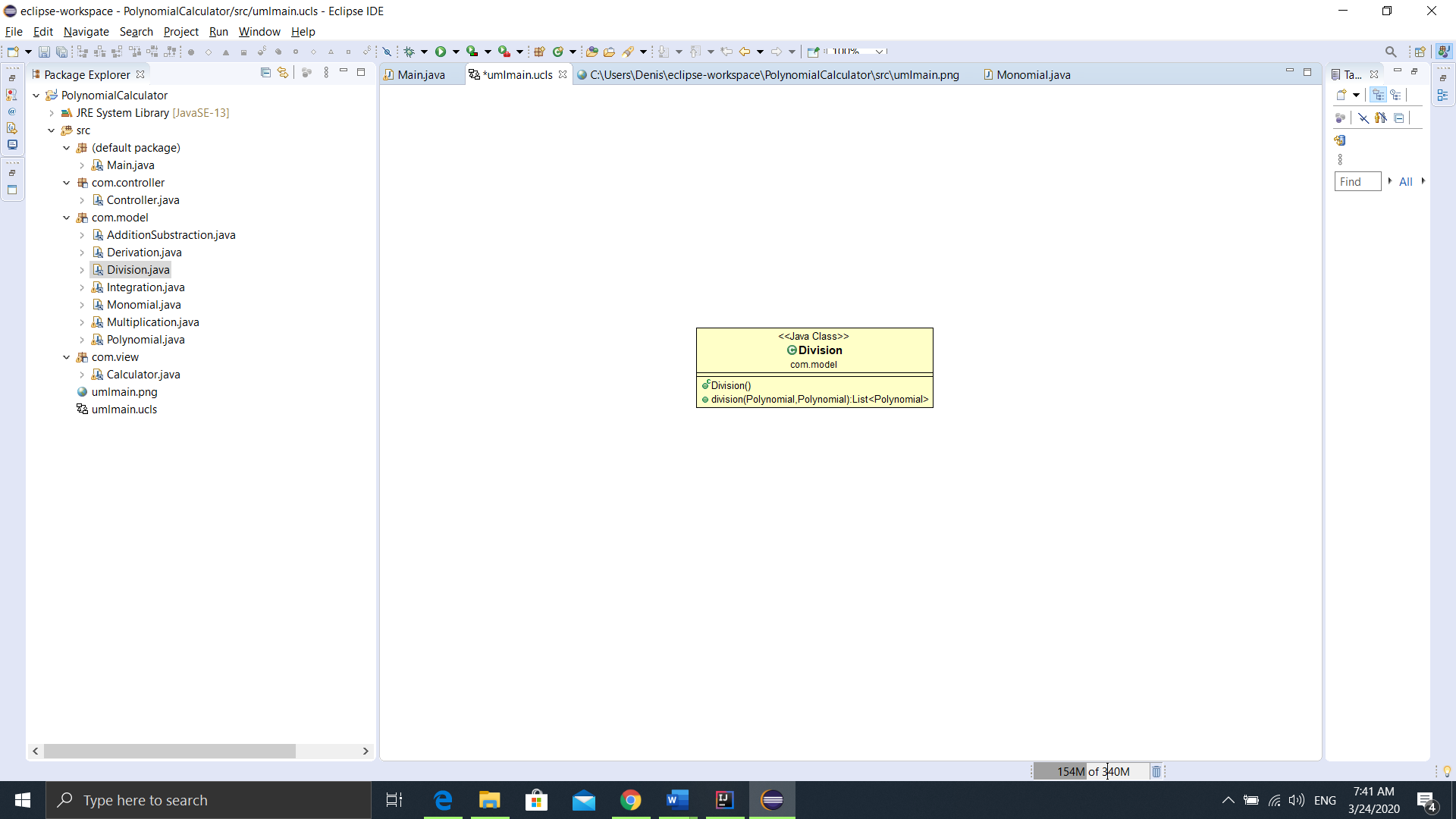


The multiplication algorithm: we multiplicate each monomial of a polynomial with the monomial of the other polynomial. The monomial multiplication means that the new coefficient will the product of the other two while we add the powers of the two monomials. After this step we add the monomials with the same power to eliminate the doubles.

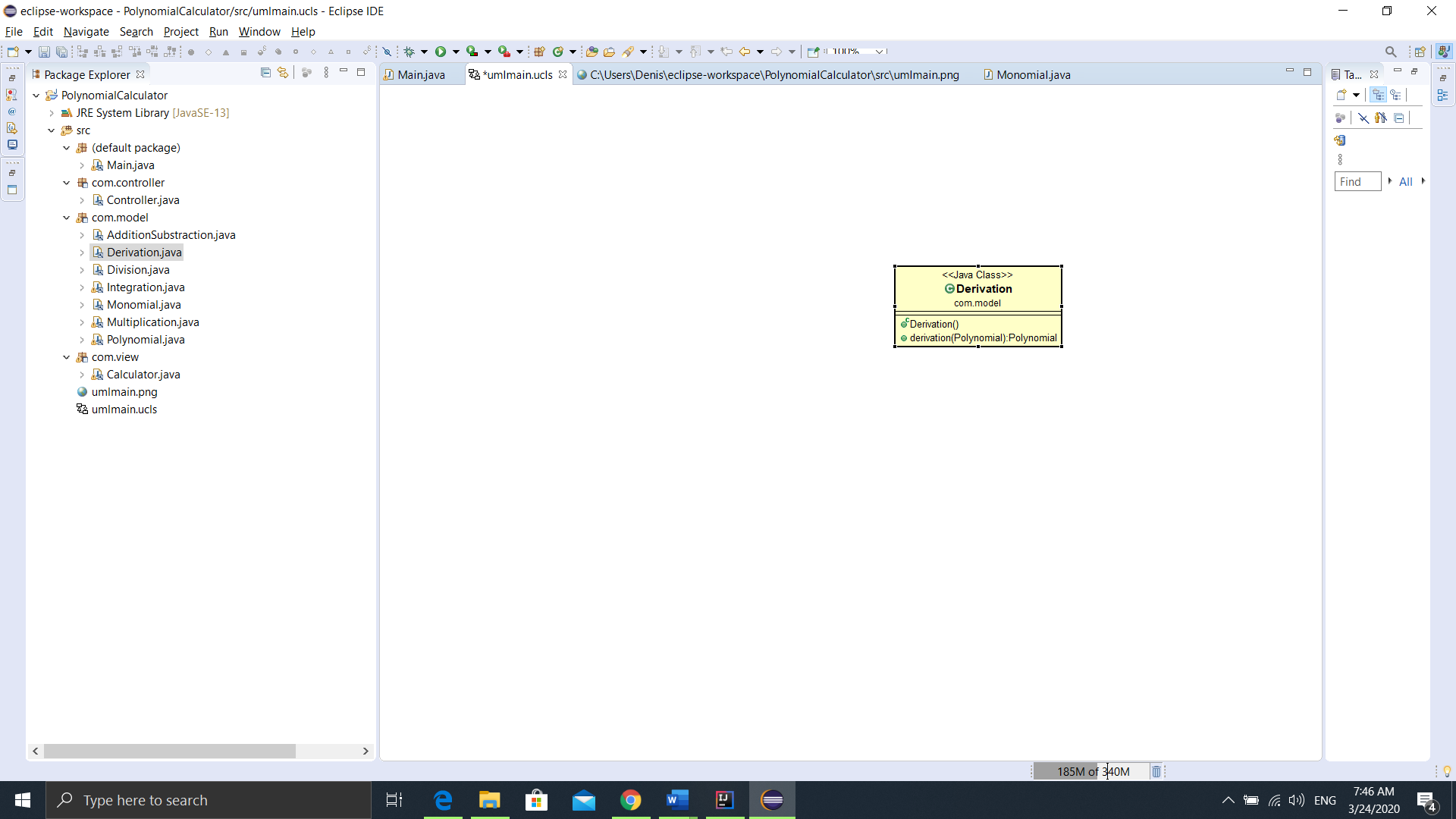
Example:

(x + 5)(x − 5)=x^2 −⧸5x +⧸5x − 25=x^2 − 25=x^2 – 52

f. .Division

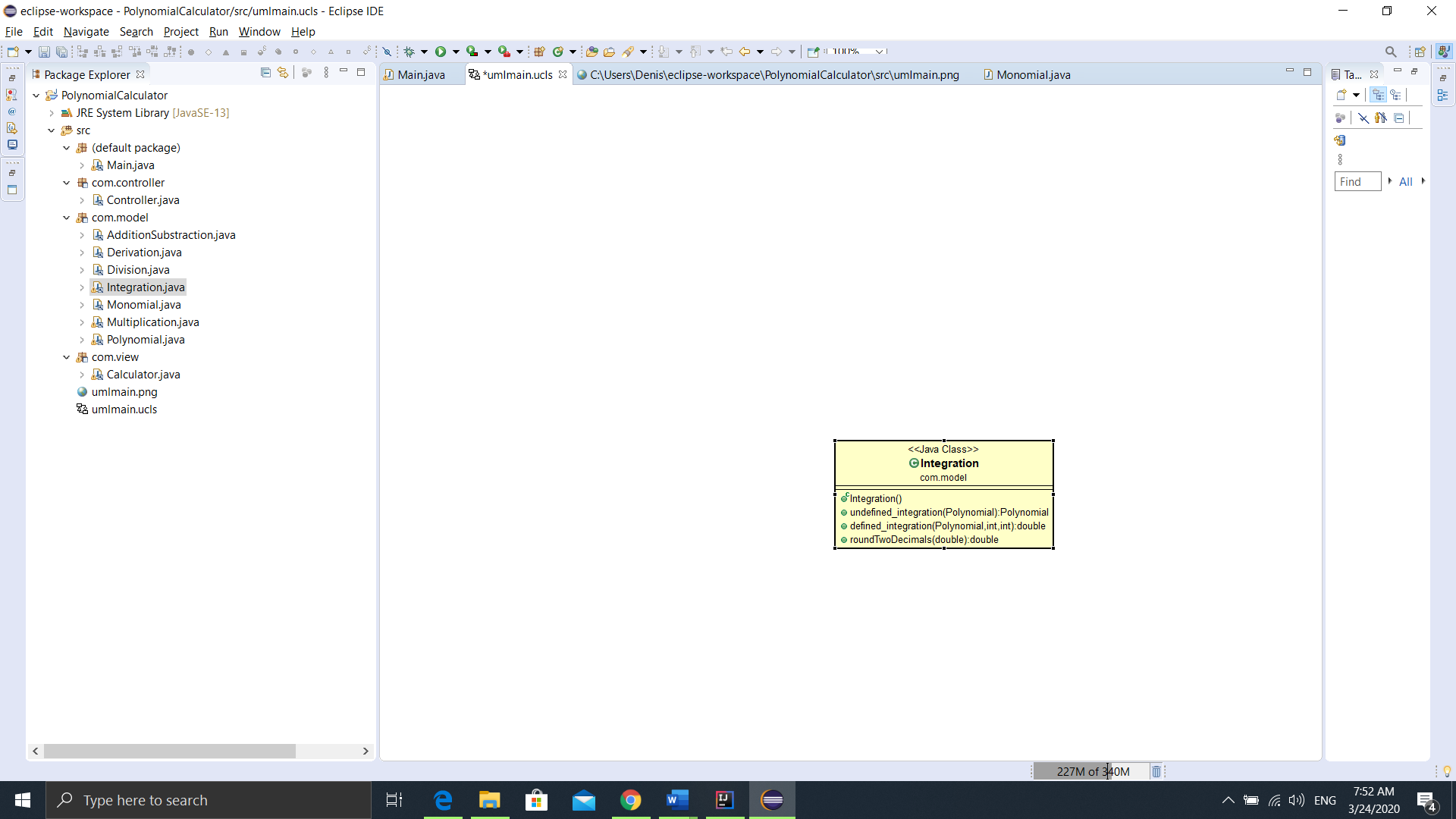
For the Division operation I tried to create an algorithm, but unfortunately it doesn’t work very well. I created a loop that runs as long as the divider’s grade is greater than the dividend’s grade. There, I divided the monomial with the greatest power of the divider with the monomial with the greatest power of the dividend, and I stored the result inside the quotient. I multiplied the quotient with the dividend and the result was subtracted from the divider, and the divider took the value of the difference. When grade of the divider became smaller than the grade of the dividend I stored what remained in the remainder.

g. Derivation



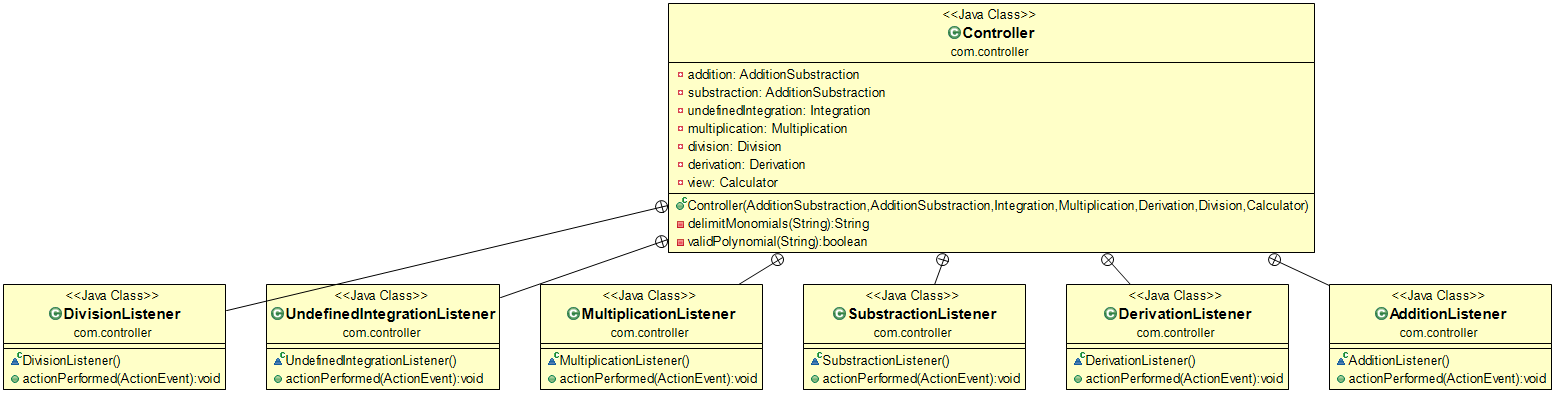
We had to check first if the Polynomial consists only of a constant, in that case the value of the derivation would be ‘0’.Otherwise, we ignore the constants and add inside the derivation polynomial the Monomials with the following fields: coefficient = coefficient\*power and power = power-1.

h. Integration



I implemented two methods: undefined and defined integration. The undefined integration consists of modifying the coefficient, that takes the value coefficient/power+1, and the power that increments by one. I implemented the method roundTwoDecimals in order to round the resulting coefficient. When printing the output of this functionI added +C for the constant that remains after integration.For the defined integration I solved the polynomial that resulted from the undefined integration for 2 different values of x, and then I differenciated them.

i.Controller Class



The controller class is used for accepting input and converting it for the model or view. In this class I check the input to be valid, then perform each polynomial operation corresponding to a button click. If the input is not valid, an exception will be triggered, and an error message will be displayed stating: “Invalid input!”.

j. Main Class

In the Main Class there would be created three methods, Model, View and Controller corresponding to each package.

5.Conclusion

In conclusion, working on this assignment helped me refresh my knowledge about polynomials and operations with them, and also how to transpose them in code. I learned to work with the MVC design pattern, which I find very useful for the future projects. It was the first project where I had to use Regular Expressions to check an input, and it was a challenge to try to figure them out. The application could be optimized and developed further by adding a Plot function that creates the graphic of the Polynomial, and the Division operation could be optimized to work better, in all the possible cases.

6.Bibliography

<https://www.edureka.co/blog/mvc-architecture-in-java/>

<https://www.upgrad.com/blog/mvc-architecture-in-java/>

<https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>

<https://www.mathplanet.com/education/algebra-1/factoring-and-polynomials/special-products-of-polynomials>

<https://www.mathplanet.com/education/algebra-1/factoring-and-polynomials/monomials-and-polynomials>

<https://docs.oracle.com/javase/tutorial/essential/exceptions/throwing.html>

<https://www.jrebel.com/blog/java-regular-expressions-cheat-sheet>

<https://topic.alibabacloud.com/a/use-a-regular-expression-font-colorredregexfont-to-match-polynomials-polynomial-regexpolynomial_1_31_32668196.html>

<https://regex101.com/>

<https://en.wikipedia.org/wiki/Object-oriented_programming>