**Homework 1**

***Programming techniques***

**Polynomial calculator**

*Student:*  Ungureanu Florin - Catalin

*Grupa:* 30424

*Professor:* Ioan Salomie

*Advisor:*Pop Cristina

*Contents*

1. Main objective .................................................................................... 1

2. Problem analysis, modeling, scenarios, using scenario .................... 1

*2.1. Problem analysis........................................................................ 1*

*2.2. Modeling..................................................................................... 2*

*2.3. Scenario ...................................................................................... 5*

3. Design ................................................................................................ 5

*3.1. UML diagram ......................................................................... 6*

*3.2. Class design .................................................................. 7*

*3.3. Algorithms ..................................................................................10*

*3.4. Graphical User Interface............................................................ 10*

4. Implementation and testing .....................................................................11

5. Results, conclusions and ulterior development....................................... 12

6. Bibliography ...........................................................................................12

1. **Main objective**

**The problem to be solved** : „Propose,design and implement a system for polynomial processing.Consider the polynomials of one variable and integer coefficients*”.*

**The objective of** the homework is the implementation and testing of algorithms specific to polynoms. These implementation must follow the OOP paradigm and to use Java for doing so . As an IDE of choice ,*Eclipse proved itself the most enhancing in terms of productivity.*

The operations I chosed to implement are the following : *addition*, , *multiplication*, *division*, *derivation* , calculating the primitive and integrating on an interval [a,b].

**2.Problem analysis, modelare, scenarios, using scenarios**

* 1. **Problem analysis**

For solving the problem we need know what are we going to use, how do we want to receive the inputs , how we build the structure of the polynomials, and also how do we do the operation designed to be efficient and to use as little additional space as possible. We need to take in consideration a practical user interface .

I choose to define the polynomial as an ordered list of monomial, each monom having a specific coefficient and exponet.

Because the operation of division and integration has as result a polynomial with non integer (double ) coefficient I had to introduce the monomial with real coefficient .

Therefore I had the need to implement a monomial with a coefficient both integer and double.Because they share the same type of exponent reap the benefit of using an abstract class which will ensure the usage of an optimal number of methods .

Also we need to think about how we are going to introduce the polynomial from the UI .

There are different number of operants in each case . For the addition,subtraction ,multiplication and division there are two operands while for calculating the derivative and integrating , there is only one operands.All these aspecs were taken care of in the user interface . The inputs must follow a certain format otherwise it won’t work.

**Modeling**

Modeling means showing the process of the polynomial operations

Mathematically , a polynomial is an expression made-up from one or more variables and constants. Those constans under-pin the monomials , which are composed by one

Coefficient ( double or integer ) and a power which is compolsory non negative integer.

If in it there are monomials with the same power those are added and form a single monomials.

The polynomial expression is : P(x) = a0x0 + a1x1 + a2x2 + .... + anxn ,

Implementarea operatiilor pe polinoame:

* + 1. ***Addition***

Taking two polynomials P1 (x) si P2 (x) and a polynomial S (x) depicting the sum of the polynomials :

P1 (x) = a0 x0 + a1 x1 + a2 x2 + ... + an xn , where n is the degree of P1 (x)

P2 (x) = b0 x0 + b1 x1 + b2 x2 + ... + bm xm , where m is the degree of P2 (x)

We have the following cases in respect to the values of the degrees :

1. The degrees are equal (n = m) :

S (x) = (a0 + b0) x0 + (a1 + b1) x1 + (a2 + b2) x2 + ... + (an + bn) xn

1. If the degree m > n

S (x) = (a0 + b0) x0 + (a1 + b1) x1 + ... + (an + bn) xn + bn+1 xn+1 + ... + bm xm

1. The degree m < n

S (x) = (a0 + b0) x0 + (a1 + b1) x1 + ... + (am + bm) xm + am+1 xm+1 + ... + an xn

* + 1. ***Subtraction***

Considering two polynomials P1 (x) si P2 (x) and a polynomial D (x) as a result of subtraction of the first two polynoms :

P1 (x) = a0 x0 + a1 x1 + a2 x2 + ... + an xn , where n is the degree of P1 (x)

P2 (x) = b0 x0 + b1 x1 + b2 x2 + ... + bm xm ,where n is the degree of P1 (x)

We have the following cases in respect to the values of the degrees :

1. The degrees are equal (n = m) :

D (x) = (a0 - b0) x0 + (a1 - b1) x1 + (a2 - b2) x2 + ... + (an - bn) xn

1. If the degree m > n

D (x) = (a0 - b0) x0 + (a1 - b1) x1 + ... + (an - bn) xn - bn+1 xn+1 + ... - bm xm

1. If the degree n > m

D (x) = (a0 - b0) x0 + (a1 - b1) x1 + ... + (am - bm) xm + am+1 xm+1 + ... + an xn

* + 1. ***Multiplication***

Taken two polynomials P1 (x) si P2 (x) and the polynomial Prod (x) as a product of the two polynomials we got :

P1 (x) = a0 x0 + a1 x1 + a2 x2 + ... + an xn , where n is the degree of P1 (x)

P2 (x) = b0 x0 + b1 x1 + b2 x2 + ... + bm xm , where is the degree P2 (x)

Prod (x) = a0 b0 x0 + (a0 b1 + a1 b0 ) x1 + (a0 b2 + a1 b1 + a2 b0) x2 ***+*** ... + (a0 br + a1 br-1 + a2 br-2 + ... + ar b0) xr + .... + ( .... ) xm+n

* + 1. ***Division***

If we consider two polynomials P1 (x) si P2 (x) and a polynomial C (x) depicting the result of the quotient and R (x) as being the reminder of the divison :

P1 (x) = a0 x0 + a1 x1 + a2 x2 + ... + an xn , where n is the degree of P1 (x)

P2 (x) = b0 x0 + b1 x1 + b2 x2 + ... + bm xm , where m is the degree of P2(x)

After the division P1 (x) / P2 (x), the result must obey the following rule (rule of division with a remainder )

P1 (x) = P2 (x) \* C (x) + R (x). In the case where the degree of the diviser is greater than the degree of the divident , the quotiend will take the result 0 and the reminder will be equal to the divident. This algorithm is much complex than the others . We first set the value of of the rest to the value of the divident P1(x) , then repeat the cycle till the degree of the reminder is greater or equal than the degree of the divisor

The loop is :

* Succesive divison of monomial of the highest degree , the result will a part of the polynomial C(x)
* The multiplication of the resulted monomial with the integer polynomial P2(x)
* Subtract the result from the rest(reminder) , so the degree will decrement

At the end from this loop the polynomial C(x) from the concatination of the resulted monomial at eatch cycle of the loop, and R(x) is obtain through succesive modification of it by subtracting .

* + 1. ***Derivation***

The derivation is applied to a single polynomial . This is made possible by loop through monomials and transform the degree and the coefficiend according to the formula .

P (x) = a0 x0 + a1 x1 + a2 x2 + ... + an xn, wher n is the degree of P1 (x)

P’ (x) = a1 x0 + 2 a2 x1 + ... + n an xn-1, the degree of the derivative will be n-1

The mathematical formula tha underpin the alghorithm is :

,

Where c is the coefficient of the monomial and g is the degree of it .

After the derivation of the polynomial the result will be another polynomial will a small degree ,at which the coeffiecients are different.

* + 1. ***Integration***

Integrations alike the derivative is for a single polynomial.

The modification of monomials is made according to the formula

, where c is the coefficient of the monomials and g the degree of it .

Because the division is present we need a monomial which can accept double coefficient .

P (x) = a0 x0 + a1 x1 + a2 x2 + ... + an xn

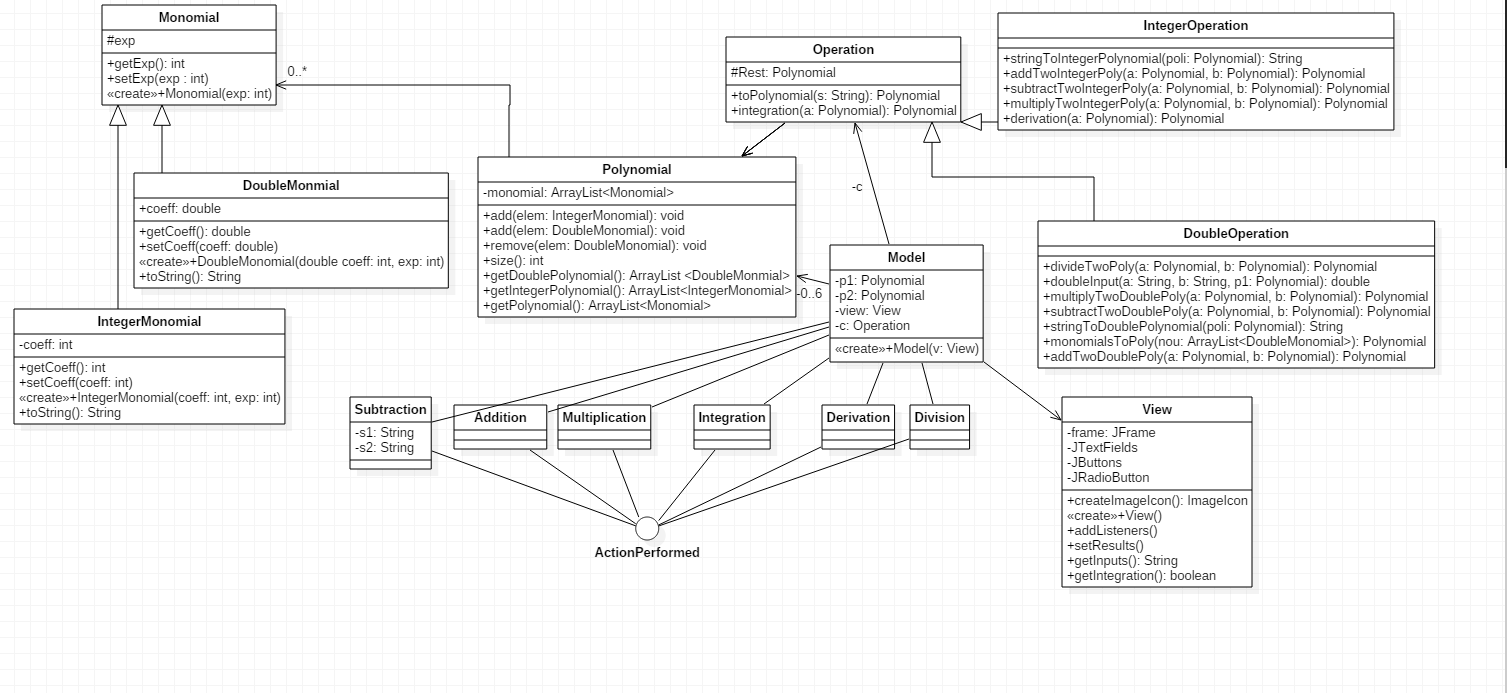
* 1. **Scenarios**

Namely when we integrate the coefficients can no longer be integers this show the necessity for another type of monomials, that with double coefficient.

1. **Design**

I developed the project using the IDE Eclipse and the Java language.The problem was divide in clases and will be presented .The first step on solving the problem was the de making of the UML diagram which underpined the project and then implementing the clases.

* 1. **UML Diagram**



* + 1. ***Monomial class***

The Monom class is an abstract class that will be expanded by the MonomIntreg and MonomReal classes. As an attribute, the Monom class is the degree, which is an entire value. The attribute is specified by the Protected Access Modifier because it must also be visible for its subclass. Only the getExp () method is implemented in this class and is specified by the public access modifier because we need to have access to this method and other project classes. The getExp () method returns the degree of the created monomer.. Therefore, for the display of the monomial we have chosen to call this method from the monom subclasses, which will send as a

* + 1. ***Polynomial class***

The polynomial class has as an attribute the ArrayList madeup by objects of type Monomial .Therefore the relationship between the class Monomial and Polynomial is of aggregation .Clasa Polinom are ca atribut privat un ArrayList compus din obiecte de tipul Monom, astfel creandu-se relatia de agregare intre clasa Monom si clasa Polinom.

* 1. ***Algorithms***

The description of the algorithm used in respect to mathematical poperties of polynomials.

* + 1. ***Addition***

This method return the addition of two polynomials having two argument ,the first and the second polynomial .

The algorithm determine the maximum exponent ,then the monomials of the second are introduced in the first ,then the polynomial is looped sever times equal to the predetermined maximal exponent .The following steps can be summarized as looping through the lists of monoms till we reach the exponent zero then add the monoms that has the same expoonent and then put it in the resulted polynomial.

* + 1. ***Subtraction***

Analog to the addition , the only difference is that we switch the sign of the second polynomial.

* + 1. ***Multiplication***

The multiplication algorithm consists in multiplying the monomer and saving each result to a new polynomial. It is done by using two "for" loops. Currently, the list of results contains several monos that have the same grade, which is against the classic polynomial. Therefore, this list must be found, and if two monoaments have the same degree, they gather and set the position of one of the monomers with the sum of these monomains, and the other monomer is removed from the list, similar to the addition and subtractor algorithms.

* + 1. ***Dividing***

The method of dividing polynomials involves returning a result vector formed by the cat and the remainder of the two polynomials. Because we can only return a single polynomial , the rest can be accessed with the setter. The algorithm consists primarily in initiating the polynomial that will constitute the rest initially formed from the real monomains of monomains in the polynomial separated. Then it enters a "while" cycle that has the condition that the degree of the rest is greater than the degree of the divider. In this cycle, the maximal degree of monolayers is divided. The resulting monoma is added to the polynomial cat. Then the multiplication of this monomer with the dividing polynomial monomers is achieved, and the result is subtracted from the corresponding monomers in the rest of the polynomial. Thus, for each scroll of the "while" cycle, the degree of rest will decrease by at least one unit.

* + 1. ***Derivative***

The algorithm from derivation loop the list of monomials and apply the fromula of derivation for a monomial for each member from the list .

* + 1. ***Integration***

Integration alike deriavtion with the specification that the formula for integration is different and always adding the constant ( + C ) in the case we calculate the primitive and not the definite integral for an interval [ a ,b ] .

* 1. **User interface**

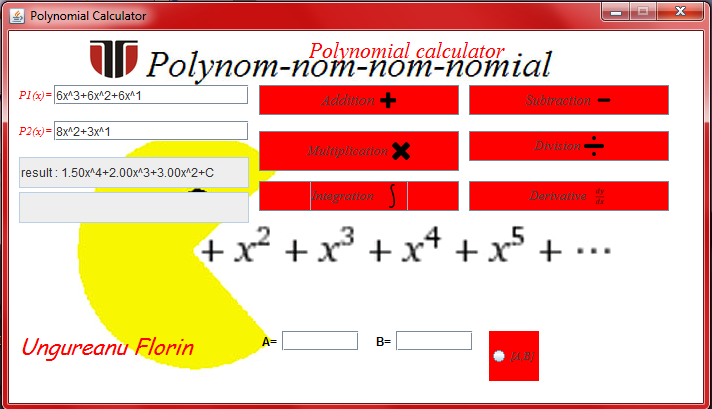
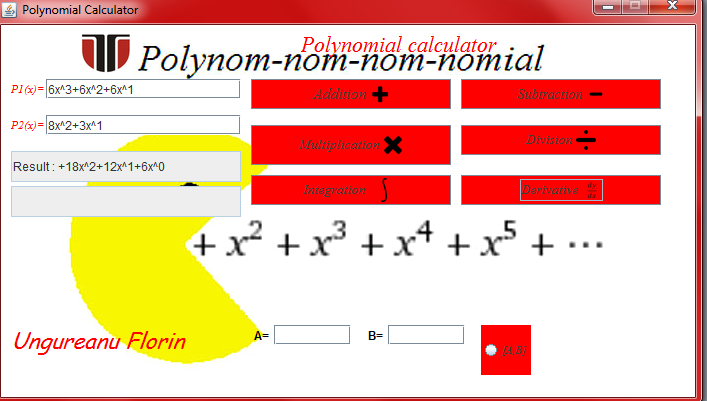
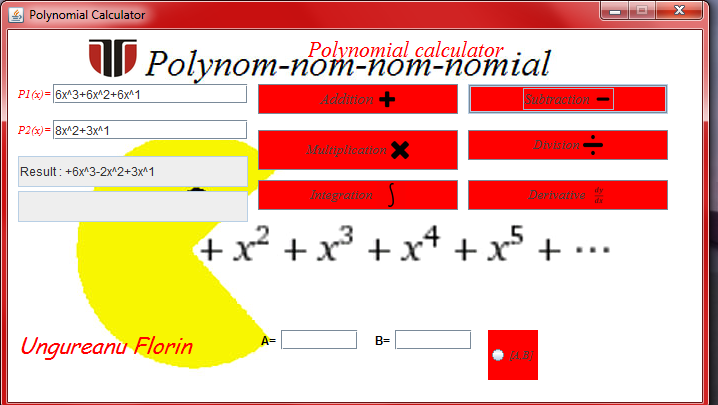
The user interface consists of the interaction between the user and the polynomial processing system. In the constructor of the class are declared and created all the objects that make up the interface and added to a panel. Button actions and text boxes for which actions have been added are also added.

**Implementing and testing**

For a short demonstration :

P1 (x) = 6x3 + 6 x2 + 6 x

P2 (x) = 8x2 + 3



1. **Results, conclusions and ulterior develloptment**

The results obtained from the user's polynomial processing can be seen in the graphical interface and checked by performing the calculations in parallel on paper. Thus, it is easy to note that the solutions obtained by calling Polynomial methods are as correct as possible, which attests the coherence and efficiency of the algorithms implemented for their realization.

1. **Bibliografie**
2. OOP courses– Marius Joldos
3. .[www.stackoverflow.com](http://www.stackoverflow.com)
4. <http://docs.oracle.com/javase/tutorial/>