Programming Techniques

Operations with Polynomials

1st Homework

Student name : Kari Tamás Zsolt

Group : 30424

Abstract

In mathematics , a polynomial is an expression consisting of variables and coefficients that involves only the operations of addition , subtraction , multiplication and non - negative integer exponents . An example of a polynomial with of a single variable , x , is *x*2 − 4*x* + 7 , which is a quadratic polynomial .

Polynomials appear in a wide variety of areas of mathematics and science . For example , they are used to form polynomial equations , which encode a wide range of problems , from elementary word problems to complicated problems in the sciences; they are used to define polynomial functions , which appear in settings ranging from basic chemistry and physics to economics and social science; they are used in calculus and numerical analysis to approximate functions . In advanced mathematics , polynomials are used to construct polynomial rings and algebraic varieties , central concepts in algebra and algebraic geometry .

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

a_n x^n + a_{n-1}x^{n-1} + \dotsb + a_2 x^2 + a_1 x + a_0,

where a_0, \ldots, a_n are numbers and X is a symbol which is called an indeterminate or variable . The symbol X does not represent any valu , although the usual (commutative , distributive) laws valid for arithmetic operations also apply to it .

The same polynomial can be expressed more easily by using a summation notation :

\sum_{i=0}^n a_i x^i

A polynomial can either be zero or can be written as the sum of a finite number of non - zero terms . Each term consists of a product of a number – called the coefficient of the term – and a finite number of unknowns , raised to nonnegative integer powers . The exponent on an indeterminate in a term is called the degree of that indeterminate in that term ; the degree of term is the sum of the degrees of the unknowns in that term , and the degree of a polynomial is the largest degree of any one term with a nonzero coefficient . The degree of an indeterminate without a written exponent is one . A term with no indeterminate is called respectively constant term and constant polynomial ; the degree of a constant term and of a nonzero constant polynomial is 0 .

Objectives

The goal of this project is to develop a Polynomial Calculator . The capabilities of this app include the basic arithmetic operations ( addition , subtraction , multiplication and division ) , as well as some operations that are specific to polynomials , namely differentiation and integration . The user is able to introduce new polynomials , view them , update them and subsequently perform the operations if the given inputs are correct . The result is shown separately . The communication between the human user and the application is done through a graphical user interface ( GUI ) , which provides a visual aid and provides clear instructions on how to use the application . The application can be used as a tool in any scientific field which deals with polynomials and a number of operations are performed on them that would be hard or slow to compute in head .

Example of working

Our program reads 2 polynomials from the interface. To make the explanation easier, consider that our polynomials are: P1 = x ^ 4 + 2 \* x ^ 3 + 3 \* x ^ 2 + 4 \* x + 3 and

P2 = x ^ 5 + 2 \* x ^ 4 + 3 \* x ^ 3 +4 \* x ^ 2 + 3 \* x + 2.

After reading the polynomials we can choose between several operations. There are operations which require both of the polynomials, but there are some which require only one.

* Operations which require both of the polynomials :
* Adding is done by pressing the add button ; P1 + P2 = x ^ 5 + 3 x ^ 4 + 5 x ^ 3 + 7 x ^ 2 + 7 x + 5 ;
* Subtracting P2 from P1 is done by pressing the sub button ; P2 - P1 = - x ^ 5 – x ^ 4 – x ^ 3 – x ^ 2 + x + 1 ;
* Multiplication of P1 and P2 is done by pressing the mul button ;

P1 \* P2 = x ^ 9 + 4 x ^ 8 + 10 x ^ 7 + 20 x ^ 6 + 31 x ^ 5 + 38 x ^ 4 + 38 x ^ 3 + 30 x ^ 2 + 17 x + 6 ;

* Division of P2 by P1 (in the app we have p1 by p2, but here that is not possible) is done by pressing the div button; P2 / P1 = x ;
* Operations with one polynomial (I use P1 to illustrate the operations)
* Differentiate the polynomial is done by pressing the der button ; P1 ‘ = 4 x ^ 3 + 6 x ^ 2 + 6 x + 4 ;
* Integrate the polynomial is done by pressing the integ button ;

Integral(P1) = 0.2 \* x ^ 5 + 0.5 \* x ^ 4 + 1 \* x ^ 3 + 2 \* x ^ 2 + 3 \* x ;

Problem analysis

As the objective already shows , this application will mostly deal with polynomials , so the first thing to figure out is how to model the instances of polynomials . These instances will be later used in the operations .

The graphical user interface is another major component , it should be easy to use , and it should provide a terminal for introducing polynomials that will be saved as instances . There should be some fields that display the introduced polynomials also the resulting polynomial after the operations . The user needs to be able to select the operation of choice and apply that operation . It should provide a nice visual and functional aspect for users that are not familiar with the application .

Modelling

In order to use the aspects of the object oriented programming , the polynomials that are modeled are represented by a class that contains all the information about an object of polynomial type as well all the necessary methods that can be performed on an object of this type .

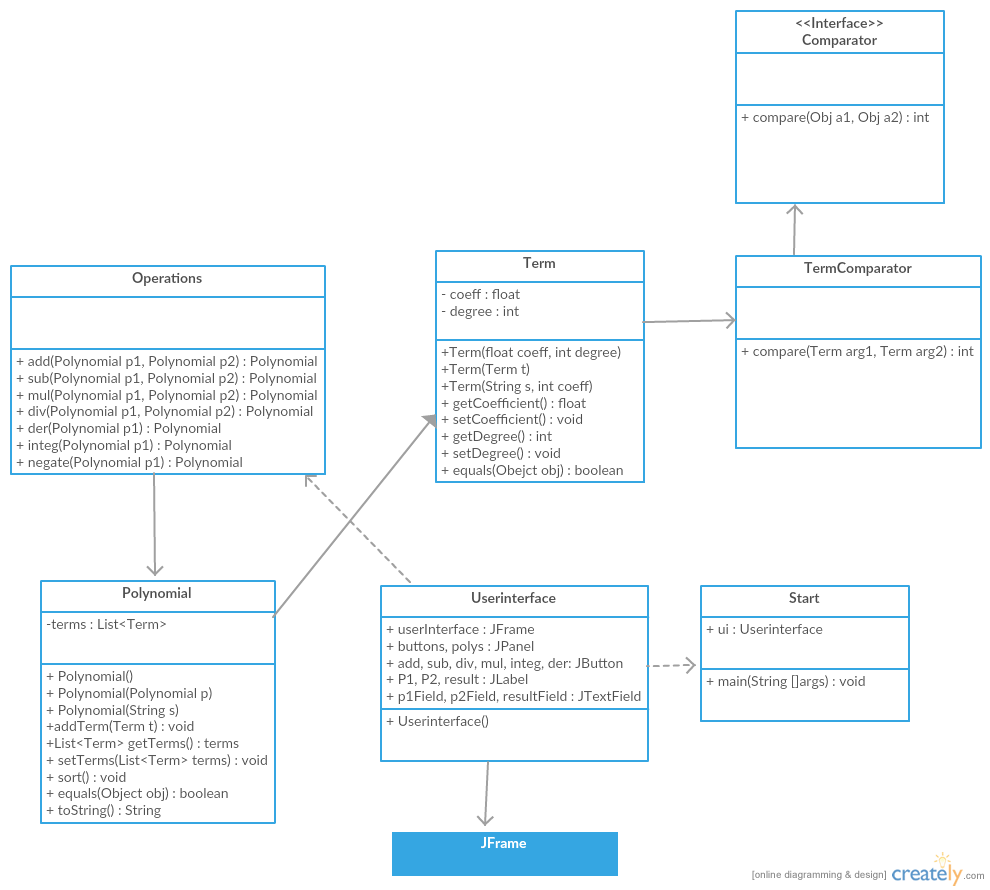
A polynomial will be represented as a list of terms , each term having an integer degree and a real coefficient . This way one will be able to access each term of the polynomial separately .

The other important structural component of the application is the graphical user interface . It was designed to have a main frame containing two panel . The first panel contains two editable text areas , where the user can enter the input polynomials , and a third text area , that is uneditable , for the result . The second panel hold six buttons , one for each operation . We have an add , a sub , a mul , a div , a der and an integ button .

Scenarios and usage cases

In this section the general case of usage is described , presenting possible special cases .

When the app is run , the main frame is shown automatically . There we can see three empty text areas , P1 , P2 and result . The user can enter the input into the two editable fiels , P1 and P2 . The input from the areas is automatically entered into the programs memory , and it doesn’t have to be sorted . After the input values are set , an operation can be performed , by pressing one of the six available buttons . The operations are in order : Addition , Subtraction , Multiplication , Division , Differentiation and Integration . Keep in mind , that the first four operations need an input in both P1 and P2 fields , while differentiation and integration are performed only on P1 . After pressing one button , the result field will be updated to display the result of the chosen operation .



Projecting

This section contains the description of the used classes in greater detail .

The classes used for modeling of the polynomial are : Polynomial and Term and Operations

Term : this class models a term of the polynomial . It has two variables : an integer for the degree and a float for the coefficient . The contructor for this class can be called in three ways : new Term ( float coeff , int degree ) , new Term ( term t ) and new Term ( String s , int coeff ) . The first one creates an Object of type term , getting it’s coefficient and degree from the parameters . The second one creates a new Object of type term identical with its input parameter . The third one is used when reading the polynomials from the text areas from the user interface , and it creates a new Object of type term with degree equal to the number after the ^ symbol from string s , or one , if x isn’t followed by ^ , or 0 , if there is no x , and coefficient that preceeds x .  
This class has getters and setters for its variables , as well as an equals method , that return true is two terms are equal , false otherwise .

Polynomial : this class models a polynomial , using an ArrayList of terms . It has a single variable , a private list of terms . It has three constructors : new Polynomial ( ) , new Polynomial ( Polynomial p ) and new Polynomial ( String s ) . The first one creates and empty polynomial , the second one creates a polynomial equal to its input polynomial . The third constructor is used to read the polynomials from the text areas p1 and p2 . It splits the input string accordint to plus or minus signs . The polynomial class also has a number of methods : List<Term> getTerms , which returns the terms , void addTerm ( Term t ) which adds a new term to the polynomial p if there is no term with the same degree , void sort ( ) which sorts the polynomial in decreasing orderd of the degrees , String toString ( ) which creates a string object representing the polynomial and an equals method , Boolean equals ( Object obj ) which returns true if this polynomial is equal to its input object , false otherwise .

Operations : this class has no constructor , just methods . The seven methods are Polynomial add ( Polynomial p1 , Polynomial p2 ) , Polynomial sub ( Polynomial p1 , Polynomial p2 ) , Polynomial mul ( Polynomial p1 , Polynomial p2 ) , Polynomial div ( Polynomial p1 , Polynomial p2 ) , Polynomial der ( Polynomial p1 ) , Polynomial integ ( Polynomial p1 ) and Polynomial negate ( Polynomial p ) .

The add method adds the two polynomials into a result polynomial . First it creates a new polynomial named result equal to p1 . The it uses the addTerm method from the Polynomial class to add the terms of p2 .

The sub method subtracts p2 from p1 . First it create a new polynomial result equal to p1 , then uses the addTerm method from the Polynomial class with negated coefficients , to add the terms of p2 .

The mul method multiplies p1 and p2 . First it creates a new empty polynomial named result , and starts adding the term by term multiplication of the two inputs .

The der method differentiates p1 . It creates a new empty polynomial named result , and adds terms with coefficient equal to coefficient multiplied by degree , and degree equal to degree-1 .

The integ method integrates p1 . It creates new empty polynomial named result , and adds terms with coefficient equal coefficient over ( degree+1 ) and degree equal to degree+1 .

The div method divides p1 by p2 . It creates a new empty polynomial named finalResult . It checks if p2 is equals to 0 , if yes it throws an exception . It checks if the degree of p2 is greater to p1 , if it is , the result will be 0 . Else it goes into a while cycle that does the division .

These are the classes that help model and provide the necessary methods for performing operations on polynomials .

The following class is part of graphical user interface , and it will be described shortly .

The class Userinterface has many variables : JFrame userInterface , JPanel buttons , polys , JButton add , sub , mul , div , der , integ , JLabel P1 , P2 , result and JTextField p1Field , p2Field , resultField . Its constructor creates the main frame , add the panel and the corresponding elemments to each panel . We also define action listeners to each of the buttons . The Userinterface class has an inner class called Handler , which implements the ActionListener interface . It as one variable , String input , and one method called void actionPerformed(ActionEvent e) . If there was an event , this method determins its source and does the corresponding action . For example if the add button was pressed , it performs an add operation on the two polynomials . At each action , the input string gets the string representation of p1 and creates pol1 with the Polynomial(String s) constructor . It does the same thing for p2 .

The class Start is the one containing the void main ( String [ ] args ) . It only creates the user interface .

Implementation

The development of this application was done using Eclipse Mars software .

Conclusions , further development

During the development of this application I have deepened my knowledge of Object oriented programming in Java , having acquired the skills necessary to design and implement a small program with graphical user interface ( GUI ) and performing some arithmetical operations on some more complex data structures , namely polynomials . This application can be developed further , additional capabilities can be added , such as performing operations on previously obtained results or graphical display of the polynomials using a function plotter , or calculating the value of a polynomial in a specific user given point . There are many possibilities for further development such as polynomial approximation of functions with the help of series known as

Taylor series . They are essential for programming and for expressing the value of any function at a given point , providing a very close approximation .

Bibliography

[http ://en .wikipedia .org/wiki/Polynomial](http://en.wikipedia.org/wiki/Polynomial)

http ://en .wikipedia .org/wiki/Polynomial\_long\_division

http ://stackoverflow .com/