

**FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE**

**CATEDRA CALCULATOARE**

**Tehnici de programare**

**Documentatie Tema1**

Calculator de polinoame

Muresan Ioana Danina

Grupa 30226

Cuprins:

1.Obiectivul temei

2.Analiza problemei,modelare,scenarii,cazuri de utilizare

3.Proiectare

4.Implementare

5.Rezultate

6.Concluzii

7.Bibliografie

**1.Obiectivul temei**

Obiectivul acestei teme este realizarea unui calculator de polinoame .

Calculatorul de polinoame poate realiza urmatoarele operatii de baza : adunarea a doua polinoame , scaderea a doua polinoame , inmultirea a doua polinoame , impartirea a doua polinoame , derivarea unui polinom si integrarea unui polinom. Fiecare dintre aceste polinoame vor fi introduse de catre utilizator din interfata grafica , de unde acesta poate alege operatia pe care doreste sa o efectueze pentru cele doua polinoame introduse anterior . De asemenea , rezulatul operatiei alese de utilizator va fi afisat in interfata grafica .Pentru implementarea ideii alese s-a folosit o structură de tip Model View Controller, în care modelul este reprezentat de descrierea monomului, ca și componentă a polinomului și a funcționalității lor, view-ul este reprezentat de interfața grafica pentru utilizatori și controllerul face legătura intre acestea două.

**2.Analiza problemei,modelare,scenarii,cazuri de utilizare**

Diagram

Description automatically generated

* Analiza:

Un polinom reprezinta o expresie formata din mai multe monoame care , la randul lor , sunt construite cu ajutorul unor coeficienti si exponenti. Cel mai important lucru in acest proiect este implementarea operatiilor polinomului si pentru acest lucru trebuie sa folosim cativa algoritmi ( adunare , scadere , inmultire , impartire , derivare , integrare). Presupunem că valorile puterilor monoamelor vor fi mereu întregi, fără a fi nevoie de validarea acestui aspect, de asemenea atunci când rezultatul operatiei va fi 0 căsuţa de reultat va aparea goala.

* Modelare:

Pentru a putea realiza cele 6 operatii care au fost propuse, este necesar in primul rand sa gasim o metoda de a lucra mai concret cu un polinom care, atunci cand este introdus de la tastatura este mai intai un string, dar pentru a-l adapta problemei noastre il vom transforma intr-o lista de monoame, fiecare monom fiind alcatuit dintr-un coeficient si un exponent.In cazul tuturor operatiilor,cu exceptia integrarii,coeficientii sunt numere intregi.La integrare,putem avea coeficienti reali,deoarece impartirea dintre coeficient si putere nu este egala, tot timpul,cu un numar intreg.

Rezultatul operatiei alese va fi afisat tot pe interfata grafica,intr-un camp separate,intitulat “rezultat”.

* Scenarii:

Pentru ca programul sa ruleze corect, exista un set de reguli pe care utilizatorul trebuie sa il indeplineasca atunci cand introduce un polinom:

* Necunoscuta tuturor polinoamelor va fi ‘x’,nu ‘X’ ;
* Fiecare termen ( monom ) va fi de tipul : ‘coeficientx^exponent’ ;
* Termenii vor fi delimitati de semnele ‘+’ sau ‘-‘ ;
* Nu se va pune spatiu intre termenii polinomului , sau intre semnele ‘ + ‘ sau ‘ – ‘ .
* Pentru primul monom din polinom,daca coeficientul este >0,nu este nevoie sa se puna ‘+’,dar nici nu e gresit daca se pune
* Cazuri de utilizare:

Atunci cand utilizatorul apasa butonul de ‘Run’ se deschide interfata grafica. Aceasta interfata este una simpla si usor de utilizat. Este alcatuita din 6 butoane aferente fiecare operatii ( ‘Addition’ , ‘Substraction’ , ‘Multiplication’, ‘Division’ , ‘ Derivation’ , ‘Integration’ ) si 3 casete de text. Primele 2 casete de text deasupra carora scrie ‘Polinom 1=:’ si ‘Polinom 2=’ sunt destinate introducerii celor 2 polinoame asupra carora se vor realiza operatiile iar cea de a treia caseta de text deasupra careia scrie ‘ Rezultat:’ este destinata afisarii rezultatului unei anumite operatii. Un avantaj al acestui calculator de polinoame este faptul ca utilizatorul poate efectua mai multe tipuri de operatii asupra acelorasi polinoame fara a fi nevoit sa le rescrie .

**3.Proiectarea ( decizii de proiectare , diagrama UML , structuri de date , proiectare clase , interfete , relatii , packages , algoritmi , interfata utilizator )**

Diagrama UML a proiectului:

Text

Description automatically generated with low confidence

Pentru a respecta paradigmele POO,programul a fost impartit in mai multe pachete cu nume suggestive care contin diferite clasa:

* Pachetul view contine clasa PolView;
* Pachetul model contine clasele Monom,Polinom si PolPatternMatching;
* Pachetul controller contine clasa PolController;
* Pachetul test contine clasa TestJunit;

A screenshot of a phone

Description automatically generated with medium confidence

Interfata grafica cuprinde urmatoarele elemente:

* Frame -“rama” in care se adauga toate elementele de care avem nevoie pentru buna functionare a programului . E practice o fereastra care , dupa utilizare , apasand butonul ‘X’ din dreapta sus se poate inchide ( Exit\_on\_close ) ;
* Butoane - 6 butoane cu diferite functionalitati

-multiplicationButton -realizeaza inmultirea a doua polinoame

-substractionButton-realizeaza scaderea a doua polinoame

-divisionButton-realizeaza impartirea a doua polinoame

-addButton -realizeaza adunarea a doua polinoame

-derivateButton- realizeaza derivarea polinomului din prima casuta(polinom1)

-integrationButton-realizeaza integrarea polinomului din prima casuta(polinom1)

* TextField - spatii dreptunghiulare in care se pot introduce date de la tastaura. In proiectul prezentat exista 2 textField-uri:

“firstPoli”-aici se va introduce primul polinom

“secondPoli”-aici se va introduce al doilea polinom

* TestArea - spatii dreptunghiulare in care se pot introduce date de la tastaura,dar il folosesc pentru a afisa rezultatul fara a se putea introduce date de la tastatura

“resultText”-aici se va scrie rezultatul operatiei alese

* Label - eticheta,care poate fi titlul,o informatie,o indicatie etc. ,care poate ajuta utilizatorul sa foloseasca programul. In cazul meu e folosit pentru a afisa informatii aflate deasupra casetelor de text: “Polinom 1=”, ”Polinom 2=” , ”Rezultat: ” .

**4.Implementare**

In cadrul proiectului au fost implementate cele 6 operatii cerute si anume: adunarea a doua polinoame , scaderea a doua polinoame , inmultirea a doua polinoame ,impartirea a doua polinoame , derivarea unui polinom si integrarea unui polinom .

Clasele existente sunt:

-**Clasa PolController** unde este instantiate interfata grafica si de unde se ruleaza aplicatia ;

Pentru fiecare buton,am implementat ActionListener() pentru a realiza operatia specificata.

Instantiez 2 polinoame,apeland constructorul fara argumente. Iau in doua stringuri diferite, textul introdus in TextField-urile din view ,apeland metoda getText().

Apelez apoi metoda getMonoms() din clasa PolPatternMatching,pentru fiecare polinom.

In final,apelez operatia si scriu in TextArea rezultatul.

-**Clasa Monom** : aceasta contine 3 atribute( degree , coefficient si flag). De asemenea , clasa contine doua constructore, metode de getter si setter pentru fiecare atribut, dar si doua metode de toString() care returneaza un String, pentru a afisa ca si “frumos” monomul. Singura diferenta intre metoda toString() si toString2() este ca toString2() afiseaza coeficientii ca si numere de tip float,iar toString() afiseaza coeficientii ca numere de tip int.

-**Clasa Polinom** contine:

-o lista de monoame (un polinom este definit ca sim ai multe monoame puse impreuna)

-un constructor fara parametrii si un constructor care are ca si argument,lista de monoame.

-metode de setter si getter pentru lista de monoame

-o metoda intitulata formatPolinom() care returneaza un String, respectiv polinomul compus din lista de moname . Se foloseste metoda toString() din clasa Monom.

-o metoda intitulata formatPolinom2() care returneaza un String, respectiv polinomul compus din lista de moname . Se foloseste metoda toString2() din clasa Monom.

-o metoda intitulata polinomSort(Polinom p) care sorteaza lista de monoame a polinomului in odine descrescatoare a puterii.

-implementarea metodei de adunare a doua polinoame:

Parcurg initial lista monoamelor celor doua polinoame,deja sortate prin metoda polinomSort(..), si cand gasesc doua monaome cu acelasi grad,creez un nou monom unde pun suma celor doua moanoame(gradul ramane neschimbat iar coeficientii se aduna).Monomul il adaug intr-o lista de monoame a unui nou Polinom create,numit result.De asemenea, fiecare monom are un flag care poate fi 0 sau 1. Daca este 0,atunci monomul respectiv nu a fost deja adunat.In schimb,daca monomul are flagul 1,a fost deja adunat si il las in pace. Schimb flagul in 1 dupa ce pot aduna doua monoame. La final,mai parcurg o data lista de moname a fiecarui polinom,iar daca gasesc monoame cu flagul 0,le adaug nemodificate in lista de monoame a polinomului rezultat. La final,sortez polinomul rezultat,pentru a fi afisat in ordine descrescatoare a puterilor monoamelor listei de monoame,Aceasta metoda returneaza polinomul rezultat.

-implementarea metodei de scadere a doua polinoame:

Parcurg initial lista de monoame a celui de al doilea polinom si inmultesc fiecare coeficient cu (-1). Apoi folosesc metoda de adunare a doua polinoame deja implementata, si adun polinomul 1 nemodificat cu polinomul 2, innmultit cu -1. La final,returez un nou polinom care reprezinta diferenta celor doua polinoame initiale.

-implementarea metodei de inmultire a doua polinoame:

Parcurg initial lista monoamelor celor doua polinoame,creez un nou monom unde pun inmultirea a fiecarei doua monoame din cele doua polinoame. La inmultirea a doua monoame, coeficientii se inmultesc iar gradele se aduna. Monomul rezultat il adaug in lista de monoame a unui nou polinom, polinomRezultat.

In momentul de fata, polinomul rezultat contine inmultirea celor doua polinoame, dar pot exista mai multe monoame cu acelasi grad. De exemplu, “2x^3+5x^2+2x^2-3x^0” poate fi un polinom rezultat. In mod normal, “5x^2” se aduna cu “2x^2” iar polinomul final va fi “2x^3+7x^2-3x^0”. Pentru a realiza acest lucru, mai parcurg o data lista de monoame a polinomului rezultat iar cand gasesc doua monoame cu grad egal, le adun, respective le scad, iar apoi pe al doilea monom il elimin din lista. La final,returnez polinomul rezultat.

-implementarea functiei de derivare a unui polinom

Parcurg lista de monoame a polinomului iar pentru fiecare monom, scad din grad 1 iar coeficientul initial se inmulteste cu gradul initial. Monomul rezultat il pun in lista de monoame a polinomului rezultat. (x^n ) ’ = n \* x^(n-1)

-implementarea functiei de integrare a unui polinom

Parcurg lista de monoame a polinomului iar pentru fiecare monom, cresc gradul cu 1 iar coeficientul initial se imparte cu gradul nou. Monomul rezultat il pun in lista de monoame a polinomului rezultat.

Am aplicat formula matematica : (x^n) = ( 1/(n+1 ) \* x^(n+1)

-implementarea functiei de impartire a doua polinoame

Se ordoneaza polinoamele p1 si p2 dupa puterile descrescatoare ale lui x.

Se face impartirea polinomului de grad mai mare la polinomul de grad mai mic.

Se imparte primul termen a lui p1 la primul termen a lui p2 si se obtine astfel primul termen al catului.

Se inmulteste rezultatul obtinut cu impartitorul si se scade acest produs din deimpartit.Acest calcul ne da primul rest al impartirii.

Se repeta acest procedeu luand primul rest ca deimpartit.

* **Clasa PolView** unde am scris codul pentru realizarea interfetei grafice, cod care cuprinde : modul in care am aranjat frame-ul , etichetele , butoanele , casetele text , atribuindu-le tututor niste coordonate , pentru o vizibilitate mai buna;
* **Clasa PatternMatching** – aici am realizat impartirea polinomului ( care este initial un string ) in monoame , fiecare polinom fiind astfel privit ca o lista de monoame. Metoda getExp ( String a ) afla exponentul din stringul de la intrare si metoda getCoeff ( string m ) aflta coeficientul stringului de la intrare. Folosind aceste doua metode, am creat metoda getMonoms ( String polynom ) care primeste la intrare un polinom sub forma de string si care este transformat intr-o lista de monoame.
* **Clasa TestJunit** unde am facut teste pentru:

adunarea a doua polinoame ;

scaderea a doua polinoame ;

inmultirea a doua polinoame ;

derivarea unui polinom si integrarea unui polinom .

impartirea a doua polinoame;

**5. Rezultate**

Pentru a exemplifica rezulatele obtinute am introdus in interfata , in cele doua casete de text destinate introducerii polinoamelor , urmatoarele doua polinoame:

Polinom 1: 4x^3–2x^1-1x^0

Polinom 2: 12x^3–10x^2-10x^1+1x^0

Rezultatul obtinut in urma adunarii:

Table

Description automatically generated

Rezultatul obtinut in urma scaderii:

Table

Description automatically generated

Rezultatul obtinut in urma inmultirii:

Table

Description automatically generated

Rezultatul obtinut in urma impartirii:

Table

Description automatically generated

Rezultatul obtinut in urma derivarii polinomului 1:

Table

Description automatically generated

Rezultatul obtinut in urma integrarii polinomului 1:

Table

Description automatically generated

Dupa cum se poate observa , toate rezultatele obtinute sunt corecte din punct de vedere matematic . De asemenea , toate rezultatele furnizate de aceasta aplicatie , in urma metodelor implementate au fost verificate si valiadate prin teste Junit ( pachetul Test ) .

**6. Concluzii**

In concluzie , sunt de parere ca acest proiect mi-a aprofundat cunostiintele in tot ce inseamna limbajul Java , implementarea paradigmelor POO , creearea unui program cu o interfata grafica . De asemenea , m-a ajutat sa imi reamintesc tehnicile de programare invatate in semestrul trecut . În același timp, a avut un rol important și în aprofundarea cunoștințelor despre liste. Nu în ultimul rând, a avut un impact pozitiv și asupra gândirii algoritmice și logice, prin implementarea propriu zisă a operatiilor pe polinoame.

Aplicatia realizata este o aplicatie simplista pentru realizarea operatiilor pe polinoame. Ca dezvoltare ulterioară se poate încerca implementarea unei alternative în interfața grafica pentru utilizatori, unde polinomul 1 și polinomul 2 să fie scrise ca șiruri de caractere, iar prelucrarea lor să se bazeze pe o expresie regulata, cu ajutorul careia se pot extrage monoamele prezente. O alta dezvoltare o prezintă regândirea algoritmilor care duc la bun sfârșit operațiile pe polinoame, iar unde mai este loc de ceva mai bun, se poate încerca eficientizarea lor. Complexitatea algoritmilor cu siguranță poate fi îmbunătățită, fiindcă din lipsa timpului suficient au fost abordați niște algoritmi mai simpliști. De asemenea, interfața grafică pentru utilizatori mai poate suferi niște finisaje, astfel încât sa aibă un design mai impunator și modern. Eventual, se pot apela alte limbaje pentru generarea unui nou stil la interfata grafica pentru utilizatori.

7.Bibliografie

1. Wikipedia
2. StackOverflow