Calculator de polinoame

Documentație

Nume: Moș Robert

Grupa: 30228

Cuprins:

1. Scopul problemei
2. Studiul problemei
3. Structura problemei
4. Implementarea
   1. Clase
   2. Metode
   3. GUI
   4. Diagrame UML
5. Testarea problemei
6. Concluzii
7. Bibliografie
8. Scopul problemei

Scopul acestei teme de laborator este implementarea unui calculator de polinoame, care aduna, scade, înmulțește, împarte, derivează și integreaza două polinoame. Prin implementarea acestui calculator de polinoame reducem timpul pierdut pentru calcularea acestora. Pentru folosirea mai ușoară a programului este implementată o interfață grafică pentru utilizatorul care folosește această aplicație.

1. Studiul problemei

Un polinom este o expresie cu mai mulți termeni care poate fi construită din constante si varibile. Polinomul la rândul său este alcătuit din monoame care contin un coeficient inmultit cu o variabilă, fiecare variabilă poate avea un exponent. Un monom care nu are o variabilă se numește monom constant. Pentru studiul problemei, se analizează în detaliu problema pentru toate cazurile de operații, deoarece acestea diferă pentru fiecare caz, spre exemplu la adunare se adună doar coeficienții a căror monoame au același grad, în cazul în care un monom dintr-un polinom nu are pereche în celălalt polinom se va scrie asa cum este, pe același caz merge si scăderea. De asemenea, pentru înmulțire se înmulțește fiecare monom din primul polinom cu fiecare monom din cel de-al doilea polinom, apoi se adună. Un alt caz este la derivare unde fiecare monom al fiecărui polinom este derivate, gradul fiecărui monom se înmulțește cu coeficientul si se scade gradul. Împărtim problema în mai multe subprobleme. Ne gândim la structura unui polinom, acesta este format din mai multe monoame. Fiecare monom are la rândul său un grad și un coeficient. Există mai multe moduri de rezolvare a problemei pentru fiecare caz, spre exemplu pentru adunare sau scadere se pot sorta monoamele intr-un vector iar cele vecine care au acelasi grad să se adune sau să se scadă. Metoda folosită de mine este alta, am parcurs fiecare polinom si dacă aveau același grad le-am adunat sau le-am scăzut, dupa caz. La rularea programului, utilizatorul va avea o interfață care contine două casete text în care se vor scrie polinoamele si o casetă text pentru afișarea rezultatului. De asemenea fereastra va contine și butoane pentru adunare, scădere si înmulțire. Utilizatorul va trebui să introducă polinoamele de mână în casetele text apoi să selecteze un buton corespunzător operației pe care vrea să o facă. Dacă utilizatorul dorește să schimbe operația făcută, după afișarea rezultatului in caseta text acesta poate să selecteze alt buton de la altă operație pentru a afișa rezultatul altei operații.

1. Structura problemei

Pachetele Java ajută la organizarea mai multor module și grupează clase și interfețe. Eu am împarțit problema in două package-uri pentru a avea un program mai ordonat și pentru a se întelege mai bine despre ce este vorba. Pentru interfața grafică am făcut un package care contine clasele Control, Model si View. Pentru partea tehnica a proiectului am facut un package numit DataModels unde am pozitionat clasele Polinom, Monom si Operatii.

1. Implementarea
   1. Clase

Pentru aceată problemă s-au făcut mai multe clase:

Clasa Polinom: În clasa Polinom se adaugă monoamele la polinom

Clasa Monom: În această clasă se creează structura monomului și se face afișarea lui ca string

Clasa Operații: În clasa Operații se află fiecare operație făcută pe polinoame, adunare, scădere, inmultire, derivare.

Clasa App: În clasa App se dau comenzile pentru executarea programului, se spune ce să se execute.

Clasa Main: În această clasă se va afișa fereastra in care se vor introduce polinoamele pentru calculul acestora

Clasa Control: Conține un listener pentru adunare, scadere si inmultire.

Clasa Model: În această clasă se transformă termenii din string-uri în double sau int după caz

Clasa View: În această clasă se vor creea panel-urile si butoanele, se va seta fontul scrisului, tipul scrisului si se vor creea butoanele.

* 1. Metode

Metodele folosite în clasa Monom:

public Monom(Integer grad, Double coeficient) {  
 this.grad = grad;  
 this.coeficient = coeficient;  
}

În metoda monom se atribuie o valoare pentru grad si pentru coeficient

public String toString() {  
 String afisare = "";  
  
  
 if (grad == 0) {  
 if (coeficient != 0) {  
 if (coeficient < 0) afisare = "" + (int) coeficient;  
 else afisare = "+" + (int) coeficient;  
 } else afisare = "";  
 } else {  
 if (coeficient == 0) afisare = "";  
 else {  
 if (coeficient < 0) afisare = "" + (int) coeficient + "X^" + grad;  
 else afisare = "+" + (int) coeficient + "X^" + grad;  
 }  
 }  
  
 return afisare;  
}

Afisează ca string fiecare monom în funcție de caz, dacă are gradul egal cu 0 si coeficientul diferit de 0 se verifică dacă este mai mica decât 0 se face afișarea specifică pentru acest caz. Dacă gradul este diferit de 0 se verifică pentru coeficient dacă este 0, mai mic decât 0 sau mai mare și se face afișarea corespunzătoare a monomului.

Metode folosite în clasa Polinom:

public String toString() {  
 String str = "";  
 for (Monom m : Pol) {  
 str = str + m.toString();  
 }  
 return str;  
}

Returnează ca string fiecare monom al polinomului

Metode folosite în clasa Operații:

public Polinom aduna() {  
 Polinom polinomRezultat = new Polinom();  
 //array de monoame care are fiecare monom al fiecarui polinom  
 ArrayList<Monom> elementeP1 = P1.getPolinom();  
 ArrayList<Monom> elementeP2 = P2.getPolinom();  
  
 //parcurge monoamele fiecarui polinom  
 for (Monom m1 : elementeP1) {  
  
 for (Monom m2 : elementeP2) {  
 if (m1.getGrad() == m2.getGrad()) {  
  
  
 Monom m = new Monom(m1.getGrad(), m1.getCoeficient() + m2.getCoeficient());  
 polinomRezultat.adauga(m);  
 }  
  
 }  
 }  
 //daca se gaseste in primul corespondent si in al doilea nu va adauga monomul  
 for (Monom m1 : elementeP1) {  
 int ok = 0;  
 for (Monom m2 : elementeP2) {  
 if (m1.getGrad() == m2.getGrad()) {  
 ok = 1;  
 }  
  
 }  
 if (ok == 0) {  
 polinomRezultat.adauga(m1);  
 }  
 }  
 for (Monom m2 : elementeP2) {  
 int ok = 0;  
 for (Monom m1 : elementeP1) {  
 if (m2.getGrad() == m1.getGrad()) {  
 ok = 1;  
 }  
  
 }  
 if (ok == 0) {  
 polinomRezultat.adauga(m2);  
 }  
 }  
 Collections.*sort*(polinomRezultat.getPolinom()); //sorteaza in functie de grad monoamele  
 return polinomRezultat;  
}

Metoda de adunare adună două polinoame. Parcurge fiecare polinom și verifică dacă gradele sunt la fel, dacă sunt la fel se adună coeficienții pentru fiecare monom și adaugă rezultatul în polinomul rezultat. Dacă într-un polinom există un monom care are un grad care nu se găsește în celălalt polinom cu ajutorul unei variabile se verifică acest lucru pentru fiecare polinom si dacă este adevărat se va pune monomul în polinomul rezultat, după care la final se va sorta polinomul cu o funcție de sort.

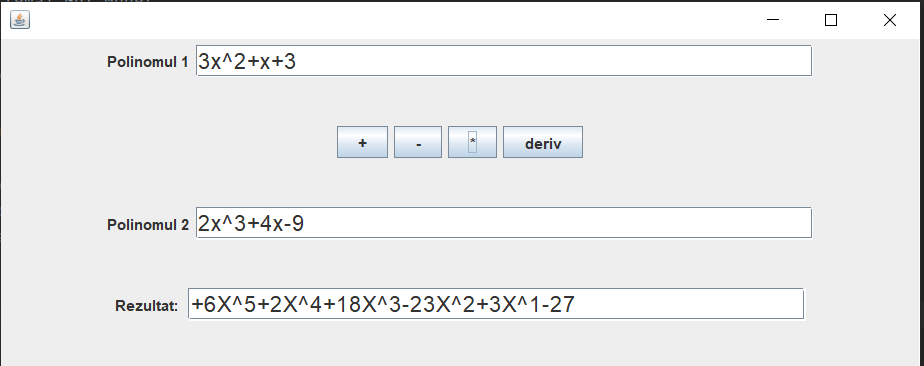
Operația de scădere funcționează aproape pe același principiu ca si operația de adunare: se face un arraylist de monoame apoi se parcurge fiecare monom pe rand din ambele polinoame, se verifică dacă cele două monoame au același grad, dacă au acelasi grad se va contrui un nou monom unde se va pune rezultatul scaderii celor două monoame. In cazult în care un monom cu un anumit grad din primul polinom nu are un monom corespondent de același grad în cel de-al doilea polinom se va adăuga la polinomul rezultat, pentru a face acest lucru am folosit un contor care initial este pus pe 0 care arată că nu are corespondent in cel de-al doilea polinom, după care se verifică dacă gradele sunt egale iar dacă nu sunt se va pune monomul in polinomul rezultat. Același lucru se face si pentru cel de-al doilea polinom, se verifică dacă are corespondent în primul, dar in acest caz, dacă nu are corespondent va fi scris cu semnul – in polinomul rezultat. La finalul metodei se face sortarea polinomului rezultat cu functia sort.

Operația de înmulțire functionează astfel: Se declară 2 arraylist-uri de monoame, se parcurge fiecare monom din primul polinom si se înmulțește fiecare coeficient al acestuia cu coeficientul monomului din cel de-al doilea polinom si se adună gradele fiecăruia. După ce s-a efectuat operația de înmulțire se va sorta polinomul rezultat. Se va creea un nou polinom si se va parcurge polinomul rezultat pentru a verifica dacă monomul are acelasi grad cu monomul de pe urmatoarea pozitie, in caz afirmativ se vor aduna coeficientii si se vor adauga la un polinom rezultat, in caz negativ nu se face operatia de adunare si monomul este adaugat automat in polinomul rezultat.

* 1. GUI

Interfața grafică sau Graphical User Interface este o interfață cu utilizatorul bazată pe un sistem de afișaj ce utilizează elemente grafice. Interfața grafică este numit sistemul de afișaj grafic-vizual pe un ecran, situat funcțional între utilizator și dispozitive electronice. Folosim o interfata grafica cu scopul de a putea fi folosit acest calculator de polinoame si de persoane non-specializate. Pentru realizarea interfeței grafice a acestui calculator de polinoame am folosit Java Swing, pentru că după parerea mea este mai practic și mai ușor de folosit decât alte metode.

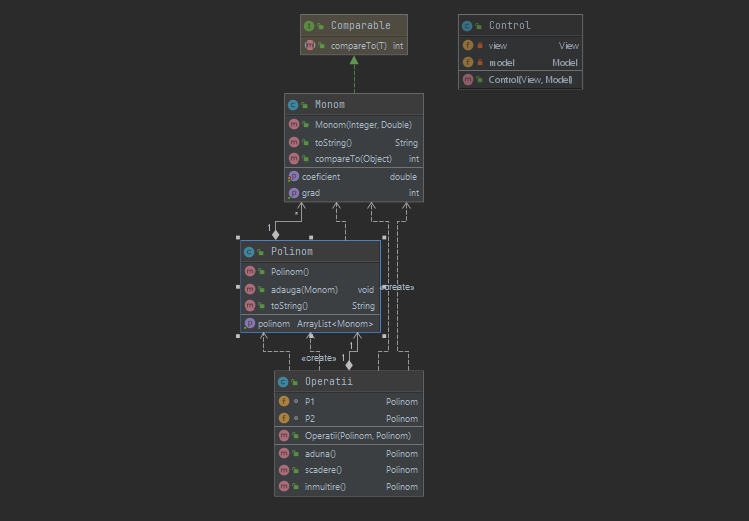
Interfața grafică are următoarele elemente: 2 casete text în care se vor scrie polinoamele pentru a putea fi calculate, o casetă text în care este afisat rezultatul si un buton pentru fiecare operație adunare, scadere, inmultire. După introducerea polinoamelor in locurile speciale pentru acestea la apăsarea unui buton cu operatia se va genera instant rezultatul si va fi afisat in locul pentru acesta.



.

* 1. Diagramele UML

Unified Modeling Language sau UML pe scurt este un limbaj standard pentru descrierea de modele si specificatii pentru software. UML a fost la bază dezvoltat pentru reprezentarea complexității programelor orientate pe obiect, al căror fundament este structurarea programelor pe clase, și instanțele acestora ( numite și obiecte ).



1. Testarea problemei

În cee ce privește testarea operațiilor acestea merg fără nicio problem pentru fiecare din operațiile : adunare, scădere, înmulțire. Pentru testarea problemei se introduce în primul câmp unde scrie „Polinomul1” primul polinom, apoi în cel de-al doilea câmp unde scrie “Polinomul2” se scrie cel de-al doilea polinom. După ce s-a făcut scrierea lor se va apăsa butonul corespunzător operației dorite si rezultatul va fi afișat in câmpul unde scrie “Rezultat”.

1. Concluzii

În concluzie acest proiect m-a ajutat să învaț mai multe lucruri, cu toate că nu este cu totul complet, mai poate fi înbunătățit adăugând și celelalte operații pentru derivare, integrare și împărțire.

1. Bibliografie
2. Wikipedia
3. Youtube

Qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt

Qt qt qt qt qt qt qt qt qt qt qt qt qt q t qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt qt