Documentatie de descriere solutie

Tehnici de Programare

Polynomial Calculator

Coblisan George, grupa 30225

An academic: 2020 – 2021

Abstract

Acest document urmareste descrierea programului, solutia abordata, tehnicile de programare folosite in elaborarea aplicatiei, utilizarea aplicatiei, analiza problemei, rezultate.

Cuprins

1. Cerinte functionale

Aceasta aplicatie dezvolta un calculator polinomial bazat pe calculul a doua polinoame sau transformarea unuia sau a doua polinoame si contine 6 operatii cu polinoame: adunarea a doua polinoame, scaderea a doua polinoame, inmultirea a doua polinoame, impartirea a doua polinoame, derivarea unui polinom si integrarea unui polinom.

E de dorit ca aplicatia sa aiba o interfara usor de utilizat si prietenoasa astfel incat orice tip de utilizator sa o foloseasca cu usurinta si cu drag.

2. Constrangeri de implementare

Din punctul meu de vedere, constrangerile fundamentale ale acestei aplicatii sunt urmatoarele:

- Descompunerea polinomului in monoame si folosirea de 2 clase Polinom si Monom
- Folosirea colectiilor din java (List) in schimbul utilizarii array-urilor in cat mai multe cazuri posibile
- Folosirea buclei foreach in schimbul buclei clasice (for int i=0...)
- Implementarea claselor sa contima maxim 300 de linii (cu exceptia claselor UI) si a metodelor sa contina maxim 30 de linii (cu anumite exceptii)
- Folosirea de „regular expressions and pattern matching” pentru extragerea coeficientilor polinomului
- Folosirea unui sistem de testare (Junit) pentru o mai buna validare a operatiilor implementate si bineinteles o verificare mai rapida

3. Obiectivul aplicatiei

Principalul obiectiv al acestei aplicatii este ca interfata sa fie usor de folosit, prietenoasa si sa functioneze corect alaturi de toate cele 6 operatii ale calculatorului polinomial. Totodata, este esential ca aplicatia sa dezvolte o arhitectura Model – View – Controller pentru o mai buna organizare a codului scris si o eleganta a acestuia, dar si pentru a opera cu solutii moderne si demne de un programator.

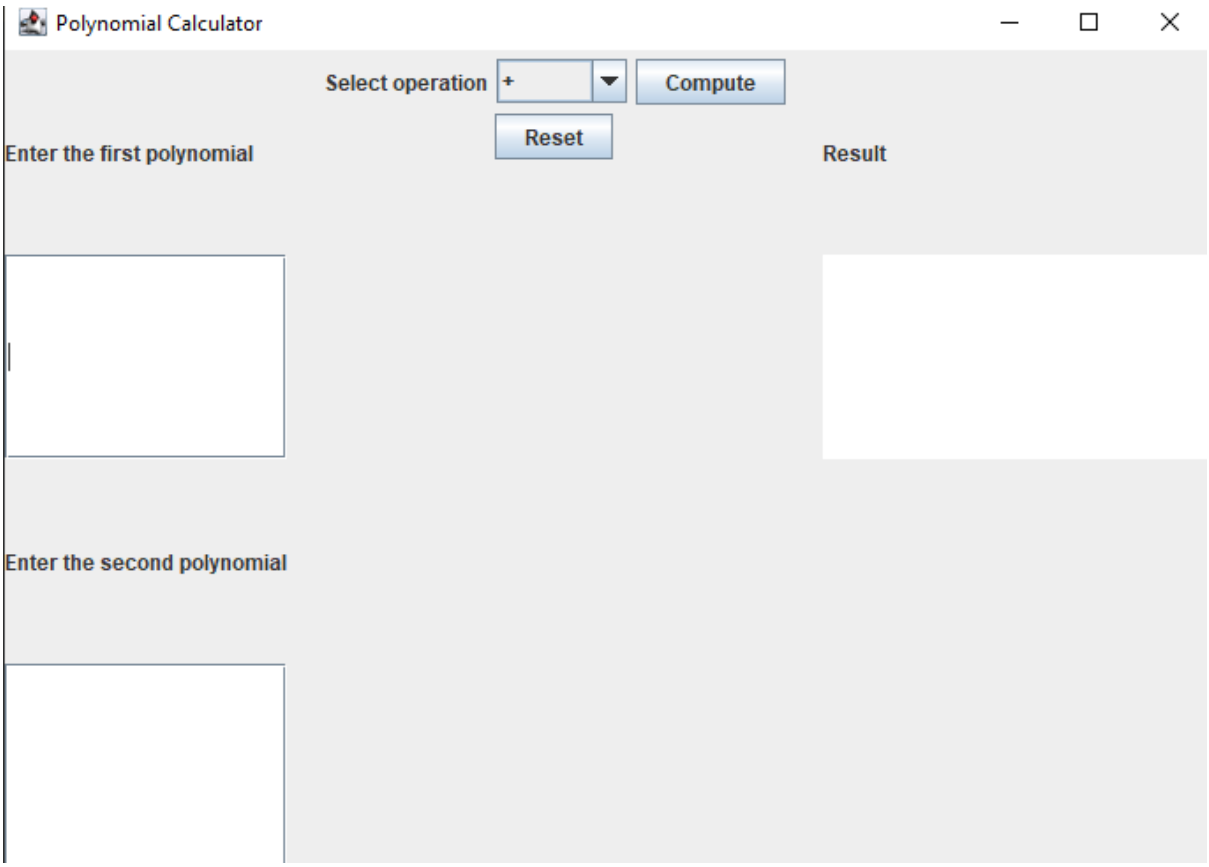
Bineinteles ca un alt obiectiv important este ca toate operatiile sa fie functionale pe toate cazurile posibile si sa informeze utilizatorul in cazul in care a introdus date de intrare invalide sau nu a introdus toate datele necesare pentru functionarea calculatorului cu un mesaj de eroare si un mesaj informativ.

Consider ca obiectivul pentru dezvoltator este sa analizeze toate cazurile posibile si sa adapteze problema la nivel de cod in cel mai simplu mod posibil deoarece acest mediu de dezvoltare ofera o multitudine de avantaje si o infinitate de moduri de a aborda problema, iar pentru utilizator sa inteleaga modul de functionare al aplicatiei si sa faca diferenta daca rezultatul afisat este cel asteptat sau nu.

4. Utilizarea aplicatiei

Meniul aplicatiei este unul primitor, simplu de folosit pentru orice tip de utilizator contine doua text field-uri pentru introducerea polinoamelor, un combo box (o lista) care contine toate cele 6 operatii, un buton pentru efectuarea operatiei, un buton pentru resetarea calculatorului si un text field pentru afisarea rezultatului obtinut in urma efectuarii calculului dorit.

Poza urmatoare reprezinta meniul/interfata calculatorului alaturi de toate functionalitatile precizate mai sus:



In continuare voi prezenta pasii care trebuie efectuati de catre utilizator pentru folosirea in mod corect a aplicatiei.

- Introducerea a doua polinoame chiar si pentru operatiile de derivare si integrare deoarece am considerat ca este mai eficient sa operez 2 polinoame si in acele cazuri pentru ca utilizatorul sa faca mult mai repede operatiile dorite si sa nu ramana un textfield gol.
- Polinoamele trebuie sa fie numai si numai de forma: aX^2+bX+c=0. Orice alt text introdus va genera un mesaj de eroare in text field-ul Result si va informa utilizatorul cu privire la eroare si forma corecta care trebuie introdusa.
- Pentru a selecta operatia dorita se face click pe sageata din dreptul mesajului „Select Operation”, apoi pentru efectuarea calculului si afisarea acestuia in text field-ul Result se face click pe butonul „Compute”.
- Totul este gata, iar daca utilizatorul doreste sa efectueze si alte calcule fara a opri aplicatia si a reporni-o poate face un simplu click pe butonul „Reset” si totul este pregatit pentru a incepe din nou pasii descrisi anterior.

5. Proiectare

Etapa de proiectare a aplicatiei consta in algoritmii din spate si modul de gandire a efectuarii tuturor operatiilor. In prima faza am implementat clasele Polinom si Monom care sunt cele mai importante clase al intregului program si definesc modul de lucru pentru calcule. Clasa Monom este cea mai

importanta si contine un constructor care initializeaza coeficientul si puterea unui monom, de fapt monomul reprezinta un coeficient al unui polinom si este vital si esential pentru realizarea tutoror operatiilor. Clasa Polinom contine o lista de monoame in care se adauga toti monomii unui polinom, respectiv se sterg si totodata putem afla numarul de monoame al unui polinom, dar si sa prelucram un anumit monom al acestuia.

In continuare voi prezenta cum functioneaza fiecare operatie in parte:

- **Operatia de adunare:** algoritmul parcurge numarul de monoame al celor doua polinoame si compara puterile fiecarui polinom pe rand, in cazul in care puterile sunt egale se va crea un monom nou in care se aduna coeficientii celor 2 monoame anterioare si in cazul in care puterile nu sunt egale se va crea un monom nou cu acelasi coeficient si putere, insa in ordine descrescatoare a puterilor pentru a afisa polinomul final sub forma corecta. La fiecare pas se adauga monomul in lista de monoame a polinomului rezultat.
- **Operatia de scadere:** algoritmul parcurge numarul de monoame al celor doua polinoame si compara puterile fiecarui polinom pe rand, in cazul in care puterile sunt egale se va crea un monom nou in care se scad coeficientii celor 2 monoame anterioare si in cazul in care puterile nu sunt egale se va crea un monom nou cu acelasi coeficient si putere, insa in ordine descrescatoare a puterilor pentru a afisa polinomul final sub forma corecta. La fiecare pas se adauga monomul in lista de monoame a polinomului rezultat.
- **Operatia de inmultire:** algoritmul parcurge pe rand fiecare monom al primului polinom si inmulteste coeficientul cu coeficientul fiecarui monom din al doilea polinom. Aceeasi procedura, se adauga monomul creat la fiecare pas in polinomul rezultat si la final se sorteaza polinomul rezultat, mai exact se aduna monomii cu puteri egale si se ordoneaza in ordine descrescatoare a puterilor.
- **Operatia de impartire:** algoritmul este identic cu cel clasic de impartire pe foaie. Algoritmul functioneaza cat timp puterea primului monom din polinomul 1 este mai mare decat puterea primului monom din polinomul 2, adica cat timp mai avem ce imparti si nu am ajuns la rest. In continuare se va imparti mereu coeficientul primului monom din polinomul 1 cu coeficientul fiecarui monom din al doilea polinom (PE RAND dupa fiecare scadere) si monomul rezultat reprezinta un monom din polinomul final, dupa care se inmulteste monomul rezultat cu tot polinomul al doilea si rezultatul obtinut va fi scazut din polinomul initial. Cum am spus la prima conditie, acesti pasi se repeta pana cand ajungem la un rest sau la 0.
- **Operatia de derivare**: algoritmul parcurge fiecare monom al unui polinom si inmuleste coeficientul cu puterea, dupa care scade puterea cu o unitate. Cred ca este cel mai simplu algoritm atat de inteles, cat si de scris.
- **Operatia de integrare:** algoritmul parcurge fiecare monom al unui polinom si imparteste coeficientul cu puterea plus 1, dupa care aduna puterea cu o unitate. Si acest algoritm este unul la fel de simplu.

6. Implementare

In acest capitol voi prezenta fiecare clasa in parte alaturi de metodele acestora.

- **Polinom**

Aceasta clasa contine o lista de monoame si 4 metode standard: adaugarea unui monom, stergerea unui monom, returnarea numarului de monoame si returnarea monomului de pe o anumita pozitie.

O metoda importanta este „PtoString” care converteste un polinom (lista de monoame) intr-

un string si il pregateste pentru afisare sub forma caracteristica a polinomului: aX^2+bX+c=0.

O alta metoda importanta este „make" care primeste ca parametru un array list de tipul String, care este de fapt polinomul scris sub forma sa, si analizeaza toti coeficientii, puterile, urmand sa se creeze monomi care vor fi adaugati in polinomul returnat.

Metoda „check" analizeaza polinomul scris sub forma de string intr-un array list caracter cu caracter si ne spune daca este scris sub forma valida sau nu.

Metoda „sort" sorteaza polinomul rezultat dupa operatia de inmultire si aduna monomii cu puteri egale.

- **Monom**

Clasa Monom contine un constructor care initializeaza monomul propriu zis cu coeficientul si puterea sa, 2 metode care returneaza coeficientul si puterea si o metoda pentru operatia de impartire care primeste ca arguement un monom si returneaza un monom care va avea coeficientul impartit la coeficientul monomului dat si puterea scazuta cu cea a monomului dat.

- **View**

Aceasta clasa reprezinta interfata grafica a acestei aplicatii si este formata din 3 text area (2 pentru introducerea polinoamelor si unul pentru afisarea rezultatului), un combo box pentru selectarea operatiei si 2 butoane pentru calculare si reset.

Contine 2 metode pentru aplicarea de actionlistener pe butonul de reset si de calculare, o metoda „reset" pentru a goli toate cele 3 text area, o metoda „setError" pentru afisarea unui mesaj de eroare in cazul in care utilizatorul nu a introdus corect polinoamele si 2 metode pentru citirea datelor din cele 2 text area in care se introduc polinoamele.

- **Controller**

Clasa Controller face legatura intre clasa View si clasa Model, preia informatiile din interfata si le transmite in model unde se vor efectua toate operatiile.

Contine un constructor care instantiaza clasele interne, clase care implementeaza ActionListener si reprezinta pe rand fiecare operatie necesara pentru calculator. Aceste clase prelucreaza informatia citita din View numai daca operatia a fost selectata corect, verifica daca este introdusa dupa standardul corect si apoi apeleaza metoda din Model pentru efectuarea calculului.

- **Model**

Aceasta clasa contine o metoda pentru resetarea interfetei si 6 metode pentru efectuarea operatiilor si o metoda care trimite array list-ul rezultat spre View pentru afisare. Cele 6 metode prelucreaza mai intai cele 2 Array List-uri date ca si parametru si le transforma in polinom cu ajutorul functiei din clasa Polinom, dupa care efectueaza algoritmii specifici pentru calculul operatiei, algoritmi care au fost prezentati in capitolul anterior.

- **Teste**

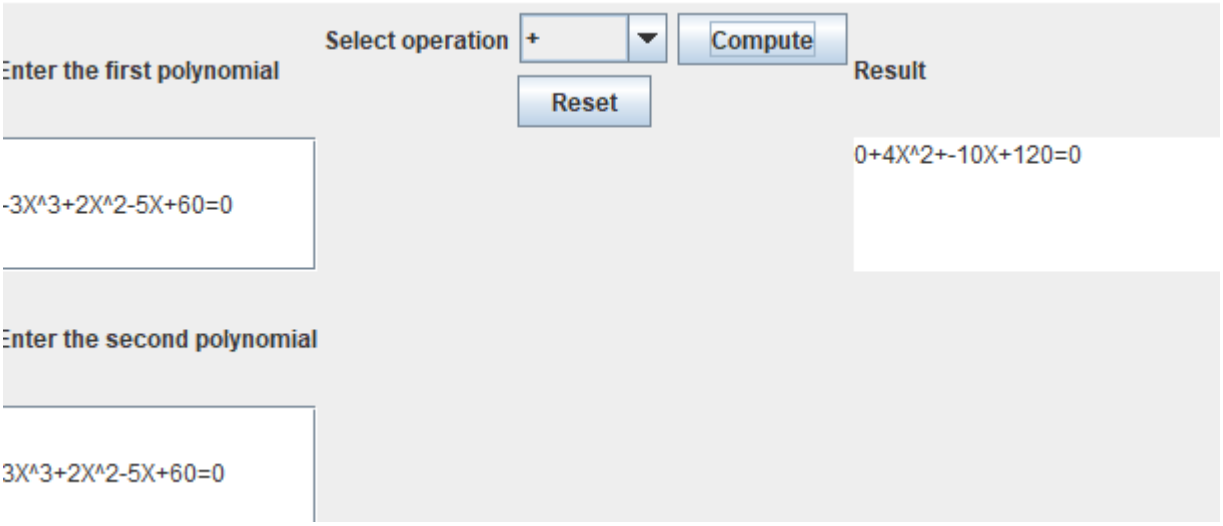
Aceasta este clasa de test care implementeaza Junit avand mai multe metode @Test pentru fiecare operatie in parte, metode care verifica corectitudinea algoritmilor pe diferite inputuri si ofera feedback despre cate operatii s-au efectuat cu succes si ce probleme au aparut.

- **Main**

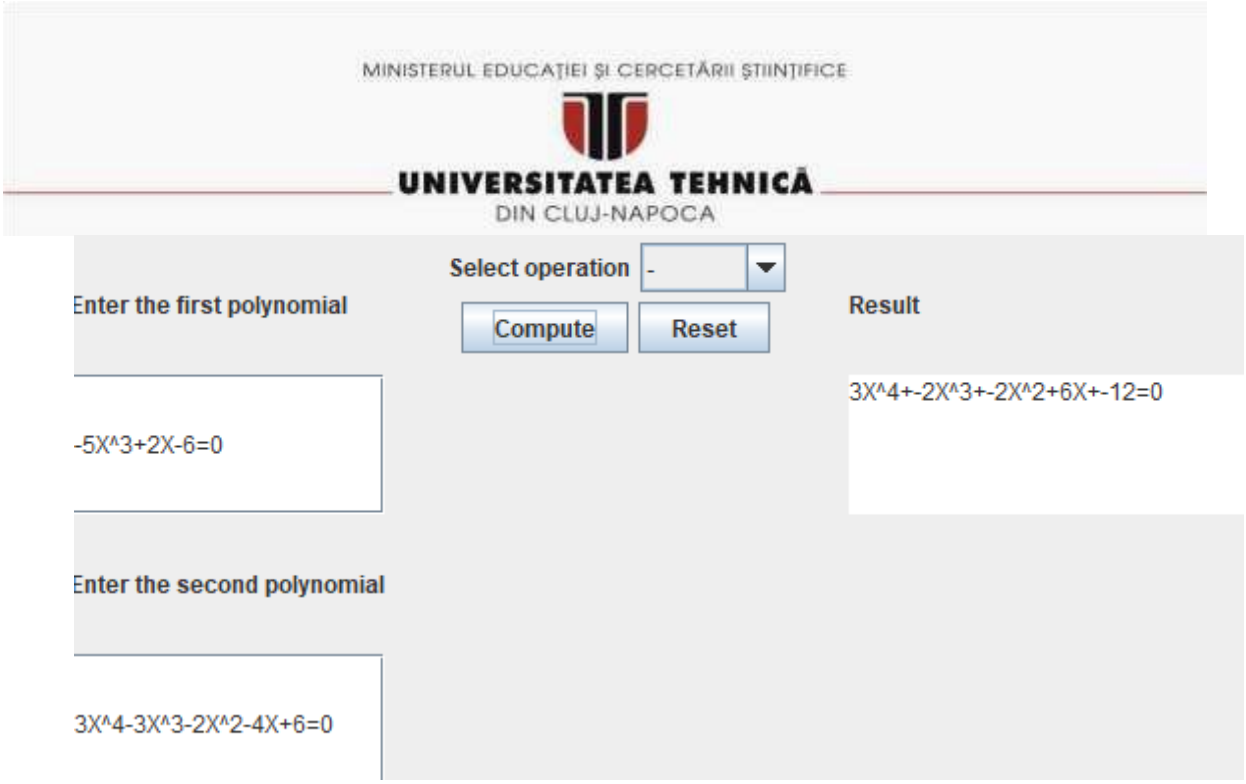
Aceasta clasa este necesara pentru executia programului si apeleaza cele 3 clase: Model, View si Controller.

7. Rezultate asteptate de utilizator

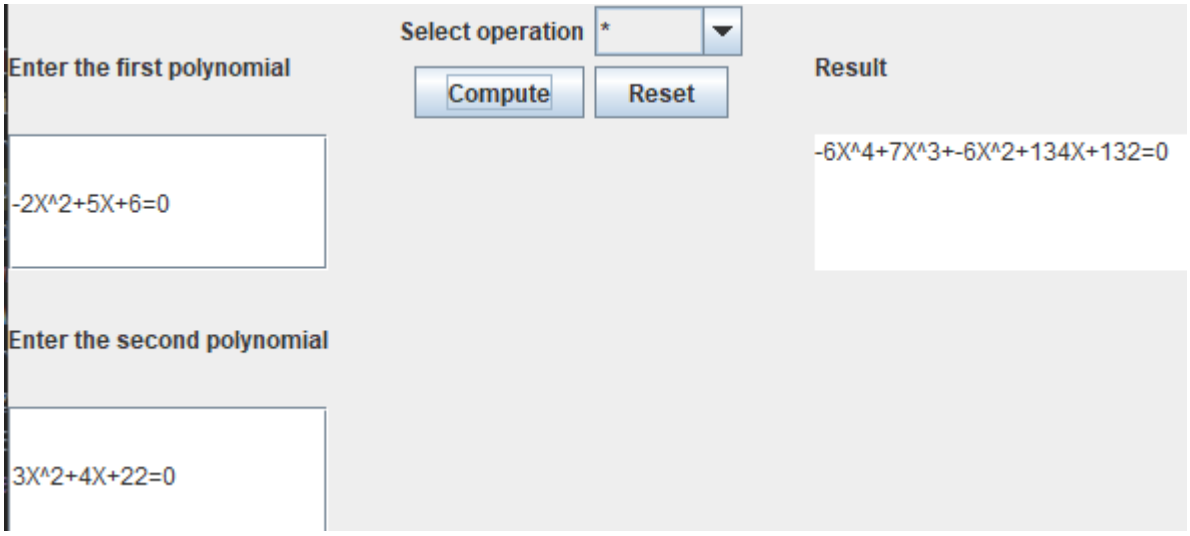
Pentru adunarea urmatoarelor doua polinoame ne asteptam sa obtinem un calcul corect, fiind 0 pe prima pozitie si coeficient cu minus unde se aduna doua numere negative.



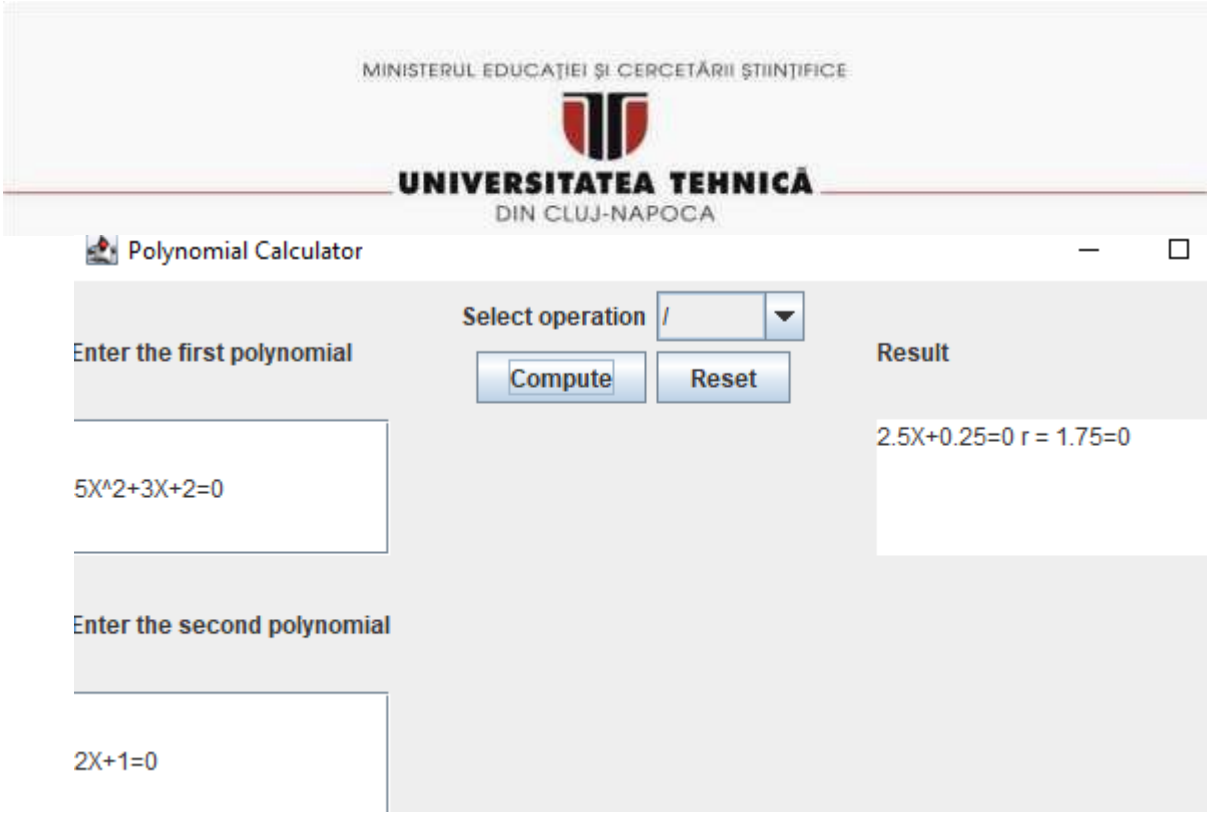
Pentru scaderea urmatoarelor doua polinoame ne asteptam sa obtinem un calcul corect, fiind un polinom de gradul 4, coeficienti cu minus in cazurile care se cuvin.



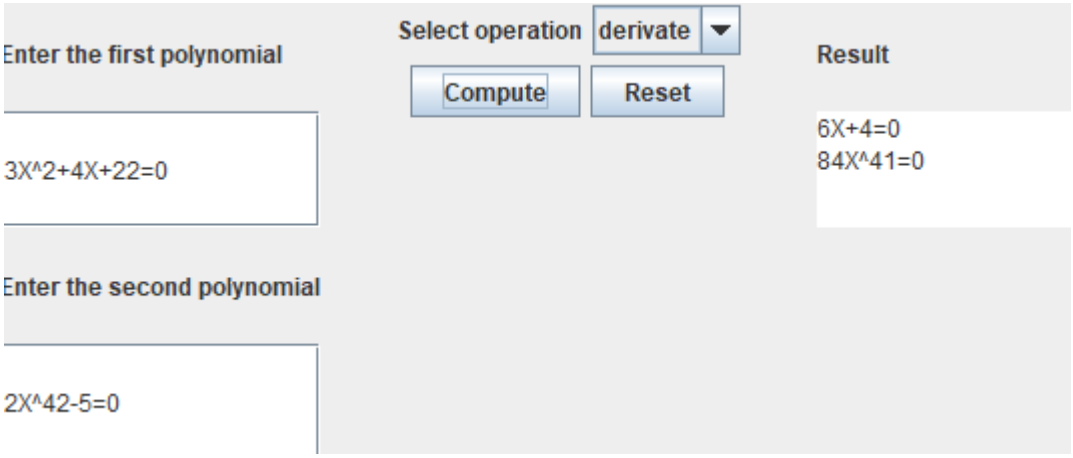
Pentru inmultirea urmatoarelor doua polinoame ne asteptam sa obtinem un calcul corect, fiind un polinom de gradul 4, coeficienti cu minus in cazurile care se cuvin si puterile ordonate in ordine descrescatoare.



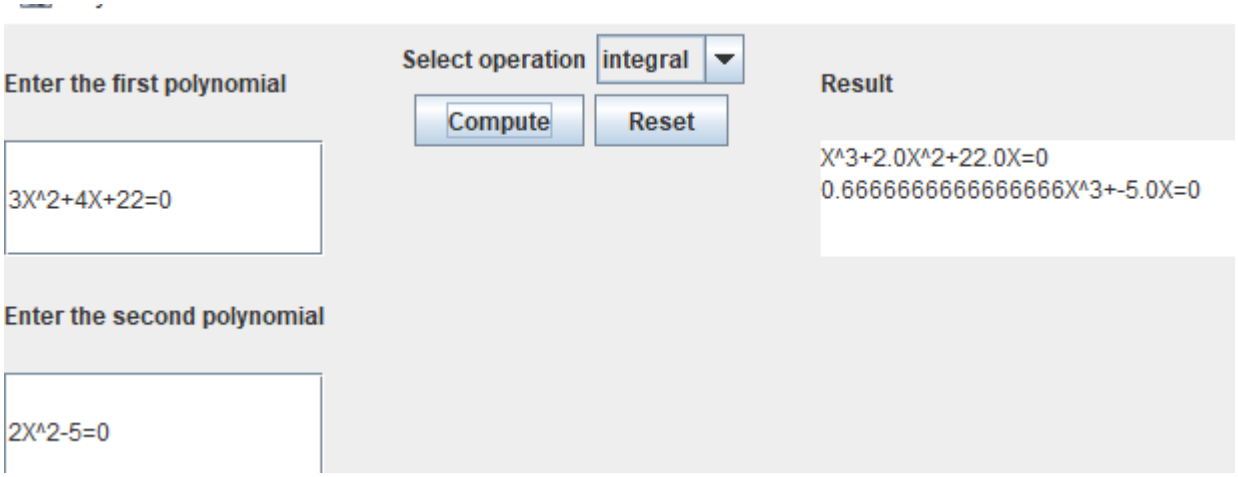
Pentru impartirea urmatoarelor doua polinoame ne asteptam sa obtinem un calcul corect, fiind un polinom de gradul 1 si restul corect.



Pentru derivarea urmatoarelor doua polinoame ne asteptam sa obtinem 2 polinoame.



Pentru derivarea urmatoarelor doua polinoame ne asteptam sa obtinem 2 polinoame.



8. Concluzii si dezvoltare ulterioara

Aplicatia poate fi imbunatatita din multe puncte de vedere, iar cele pe care as fi dorit eu sa le implementez sunt urmatoarele:

- Un sistem de informare pentru utilizator despre cum ar trebui sa foloseasca si sa introduca datele si un sistem care sa ii afiseze pe langa rezultat si modul de calcul al operatiei respective.
- O interfata mai primitoare, cu un altfel de meniu si totodata usor de utilizat.
- Un sistem prin care utilizatorul sa nu mai fie nevoit sa scrie tot polinomul si sa fie atent sa nu greseasca un X sau un ^, prin acest sistem utilizatorul sa aiba optiunea de a selecta coeficientul, puterea si semnul, iar aplicatia sa tina minte si sa formeze in spate polinomul.
- Un istoric al calculelor efectuate de la deschiderea aplicatiei pana la inchiderea acesteia si specificarea operatiilor corecte si gresite.