MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
CATEDRA CALCULATOARE

TEMA

la disciplina

Tehnici de programare fundamentale

Calculator Polinomial

Borbei Raul

An academic :2020 – 2021

Grupa: 30225

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE
UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
CATEDRA CALCULATOARE

CUPRINS

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE
UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
CATEDRA CALCULATOARE

1. Obiectivul temei

Obiectiv principal

Obiectivul principal al acestei teme de laborator este de a propune, proiecta și implementa un sistem care sa faciliteze operațiile pe polinoame cu o singura variabila și care au coeficienți întregi. În urma rezolvării acestor cerințe va rezulta un calculator de polinoame cu ajutorul căruia sa se poată efectua diferite operații pe polinoame precum: adunare, scădere, înmulțire, derivare.

Obiective secundare

Obiectiv Secundar	Descriere	Capitol
Dezvoltarea de use case-uri si scenarii	Modul de utilizare si funcționalitatea aplicației	2
Alegerea structurilor de date	Structurile de date folosite pentru a duce la capăt obiectivul principal	3.1
Împărțirea pe clase	Folosirea unui MVC (Model – View – Controller) pentru crearea unui GUI (Graphic User Interface), a claselor Monom, Polinom etc.	3.2
Dezvoltarea algoritmilor	Vor fi descrise structurile de date necesare pentru atingerea obiectivului principal, schema UML (diagrama de clase) precum si algoritmii folosiți pentru realizarea operațiilor.	4
Implementarea soluției	Vor fi descrise fiecare clasa cu câmpurile si metodele importante precum si descrierea interfeței utilizator;	4.1
Testare	Vor fi descrise câteva scenarii de testare a operațiilor pe polinoame, folosind interfața grafica	5

2. Analiza problemei, modelare scenarii, cazuri de utilizare

Pentru a utiliza programul utilizatorul trebuie sa introducă in Text Fieldurile din interfață grafica polinoamele asupra cărora dorește sa efectueze operații. Operațiile posibile sunt: adunarea celor doua polinoame, scăderea celui de al doilea din primul, înmulțirea polinoamelor, catul împărțirii celor doua, restul împărțirii, derivarea primului polinom și integrarea lui.

Polinoamele vor fi introduse sub forma a*x^b, unde a și b sunt numere întregi, a fiind coeficientul lui x și b puterea sa. Polinomul poate fi introdus indiferent de ordinea puterilor atâta timp cat este respectata forma. Programul va afișa mereu polinomul rezultat sortat descrescător după puterile lui x. Pentru a extrage din stringul de intrare coeficienții și exponenții am folosit Regex.

3. Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfețe, relații, packages, algoritmi, interfața utilizator)

3.1. Structuri de date

Structura de date principala folosita este ArrayList de Monoame. Un polinom este astfel format dintr-o lista de monoame.

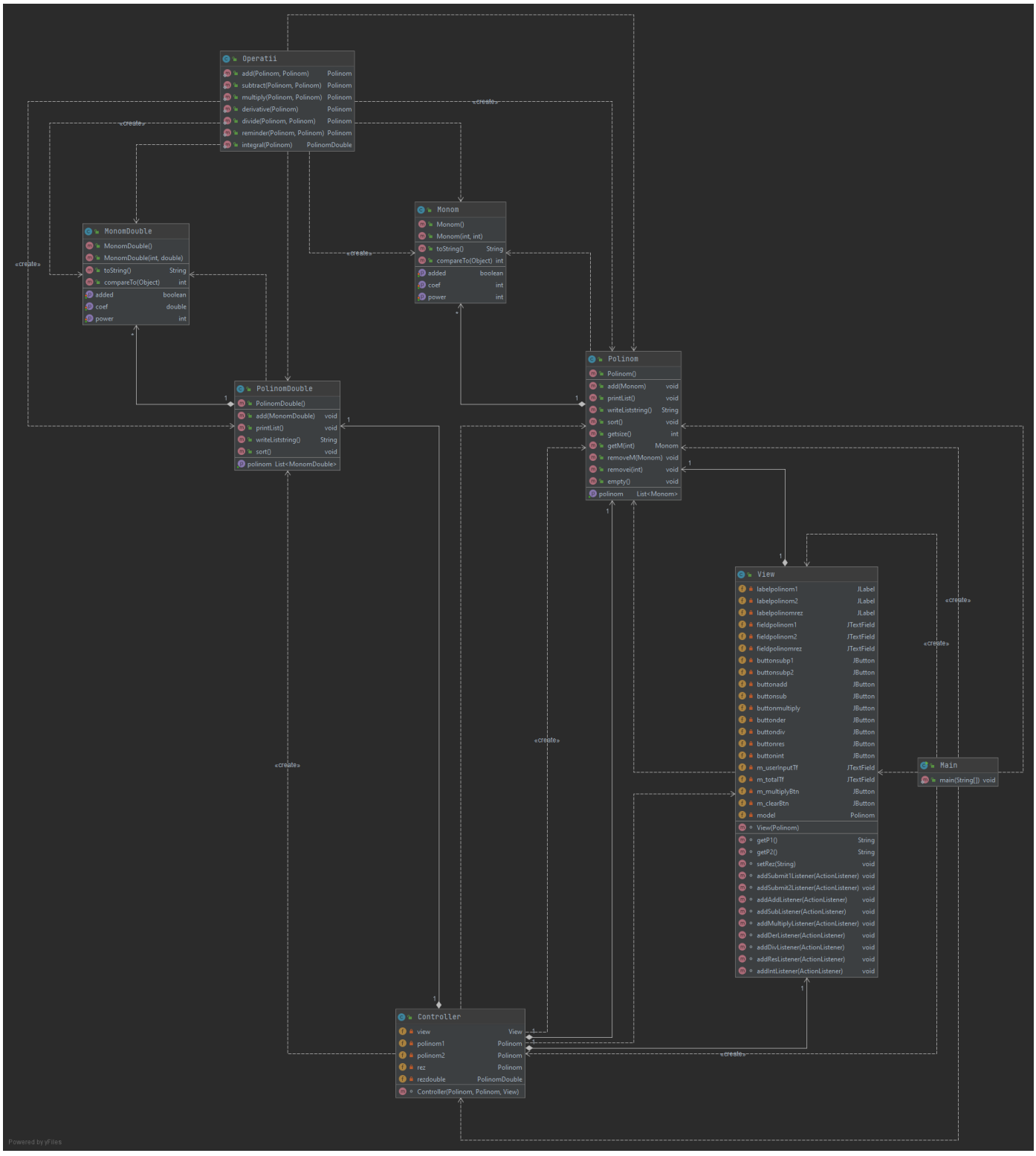
Un monom este format dintr-un coeficient, un exponent și o variabila având astfel forma ax^b, unde a este coeficientul, b exponentul și x variabila. Monomul este reprezentat în Java printr-o clasa separata care conține o variabila pentru coeficient și una pentru exponent si în plus o variabila de tip boolean care sa indice daca monomul a fost procesat sau nu(este folosita pentru operația de adunare și pentru cea de scădere).

3.2. Diagrama UML

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
CATEDRA CALCULATOARE



Clasa Monom

În clasa Monom este cea mai de jos clasa din proiect și clasa cu care am început crearea proiectului. În ea se retine coeficientul și exponentul unui monom.

Clasa Polinom

Clasa polinom conține o lista de monoame și o grămadă de metode necesare pentru a putea efectua operații cu ele precum: print(printează lista), add(adaugă un element în lista), remove(șterge un element din lista).

Clasa PolinomDouble și clasa MonomDouble

Sunt în mare parte la fel ca și clasele polinom și monom doar ca în loc sa conțină variabile de tipul int ele conțin variabile de tip double și sunt folosite la operația de integrare și la cea de împărțire cu rest, deoarece aceste operații pot returna valori cu virgula.

Clasa Operații

Clasa Operații este o clasa în care rețin doar metode pentru efectuarea calculelor cu polinoame. Metodele au nume intuitive precum: add pentru adunare, substract pentru scădere, divide pentru împărțire etc.

Clasa View

Clasa view este o clasa din modelul MCV. Aceasta clasa se ocupa de aspectul interfeței grafice.

Clasa Controller

Face parte tot din MCV și face legătura intre model(în cazul meu polinomul) și view. În aceasta clasa se găsesc metodele pentru listener și transformarea din string în monom cu ajutorul Regex.

4. Implementare

4.1.Metode

Metode din clasa Monom

În afara de getter, setter și 2 constructori în clasa Monom se mai găsesc încă 2 metode. O metoda toString și o metoda compareTo. Metoda toString ajuta la afișarea polinomului, iar metoda compareTo este necesara pentru a putea sorta Polinomul.

```
public String toString( )
{
    if( coef == 0 && power == 0 )
        return "";

    if( coef == 0 )
        return "";

    if( power == 0 && coef > 0 )
        return "+" + coef ;

    if( power == 0 && coef < 0 )
        return  coef + "";

    if( coef == 1 && power>1 )
        return "+X^"+power;
    if( coef == -1 && power>1 )
        return "-X^"+power;

    if( coef == 1 && power == 1 )
        return "+X";

    if( power > 0 && coef > 0 )
        return "+" + coef + "X^" + power;
    else
        return coef + "X^" + power;
}

@Override
public int compareTo(Object comparemon) {
    int comparepow=((Monom)comparemon).getPower();
    return comparepow-this.power;
}
```

Metode din clasa Polinom

În clasa Polinom avem mai multe clase care ne ajuta sa manipulam polinoamele. Printre aceste clase amintim clasa add pentru a adaugă un monom în polinomul curent, clasa sort care sortează polinomul, o metoda removei pentru a șterge un element, o metoda writeListstring care transforma polinomul într-un string pentru a putea fi afișat într-un TextField din interfața grafica.

```
public String writeListstring(){
    String s="";
    for(Monom elem:polinom){
        s=s+" "+elem;
    }
    return s;
}
```

Metode din clasa Operații

Clasa operații conține metodele pentru calculul cu polinoame. Toate metodele sunt denumite intuitiv precum: add, substract, multiply etc.

```
public static Polinom multiply(Polinom pol1 , Polinom pol2 ) {
    Monom rezultatMono;
    Polinom rezultatPoli = new Polinom();
    for( Monom m1 : pol1.getPolinom() ) {
        for (Monom m2 : pol2.getPolinom()) {
            rezultatMono = new Monom(m1.getPower()+m2.getPower(),
m1.getCoef() * m2.getCoef());
            Polinom polinom1=new Polinom();
            polinom1.add(rezultatMono);
            rezultatPoli=add(rezultatPoli,polinom1);
        }
    }
    rezultatPoli.sort();
    return rezultatPoli;
}
```

Metoda de înmulțire este destul de simpla. Ea se apelează cu doua polinoame ca și parametrii. Vom avea nevoie de un monom și de un polinom rezultat pe care le și inițializam. Pe urma parcurgem cu doua foreach elementele din polinoame și le înmulțim element cu element prin înmulțirea coeficienților și adunarea puterilor lor.

```
public static PolinomDouble integral(Polinom pol1){
        PolinomDouble rez=new PolinomDouble();

        for( Monom m1 : pol1.getPolinom() )
        {
            MonomDouble monom=new
MonomDouble(m1.getPower()+1,(double)m1.getCoef()/(m1.getPower()+1));
            rez.add(monom);
        }
        rez.sort();
        return rez;}}
```

Pentru metoda de integrare se transmite un singur polinom ca și parametru și se va returna polinomul rezultat după ingerarea sa cate un element pe rând. Pentru integrare se parcurge cu foreach pe rând polinomul și se creează un nou monom, de data aceasta cu coeficienți double, pe care îl integram adăugând unu la putere și și împărțind coeficientul la putere+1.

```
public static Polinom add( Polinom pol1 ,Polinom pol2 )
{ Monom rezultatMono;
    Polinom rezultatPoli = new Polinom( );
    for( Monom m1 : pol1.getPolinom() ){ m1.setAdded(false); }
    for( Monom m2 : pol2.getPolinom() ){ m2.setAdded(false); }
    for( Monom m1 : pol1.getPolinom() )
    { for( Monom m2 : pol2.getPolinom() )
        { if( m1.getPower() == m2.getPower() )
            { m1.setAdded( true );
            m2.setAdded( true );
            rezultatMono = new Monom( m1.getPower( ),m1.getCoef( ) +
m2.getCoef( ) );
            rezultatPoli.add(rezultatMono); } } }
    for( Monom m1 : pol1.getPolinom() )
    { if( m1.isAdded() == false ) { rezultatPoli.add( m1
);m1.setAdded(true);} }
    for( Monom m2 : pol2.getPolinom() )
    { if( m2.isAdded() == false ) { rezultatPoli.add( m2 );m2.setAdded(true);
} }
    rezultatPoli.sort();
    return rezultatPoli;
}
```

Pentru metoda de adunare folosesc și variabila booleana care îmi spune daca monomul a fost sau nu adăugat deja în polinomul rezultat. Pentru ca aceeași variabila o folosesc și la scădere este

necesar sa o setez pe „false” înainte sa încep prelucrarea. După care parcurg polinoamele element cu element și adun monoamele care au aceeași putere după care le adaug în polinomul rezultat (de asemenea aici notez ca am prelucrat monomul setând variabila pe „true”). După acest for parcurg încă o data polinomul1 și pe urma polinomul2, verific daca exista monoame neprelucrate și daca exista le adaug în polinomul rezultat. La final sortez polinomul și îl returnez.

Metode din clasa View

Clasa View conține instrumentele necesare realizării interfeței grafice. Conține un constructor mare în care se creează interfața și metodele care urmează sunt metode pentru a adaugă Listener pe butoane astfel încât acestea sa facă operațiile necesare.

Metode din clasa Controller

Metodele din clasa controller fac legătura intre View și polinom. De asemenea aici se afla și metoda pentru transformarea din String în Polinom folosind Regex.

```
class Submit1Listener implements ActionListener{
    public void actionPerformed(ActionEvent e){
        String poli;
        Monom m;
        polinom1.empty();
        poli=view.getP1();
        String[] coef=poli.split("x\\^\\d+\\+?");
        String[] powers=poli.split("\\-?\\+?\\d+x\\^");
        for(int i=0;i<coef.length;i++){
            m=new
Monom(Integer.parseInt(powers[i+1]),Integer.parseInt(coef[i]));
            polinom1.add(m);
        }
        polinom1.sort();
    }
}
```

Metode din clasa Main

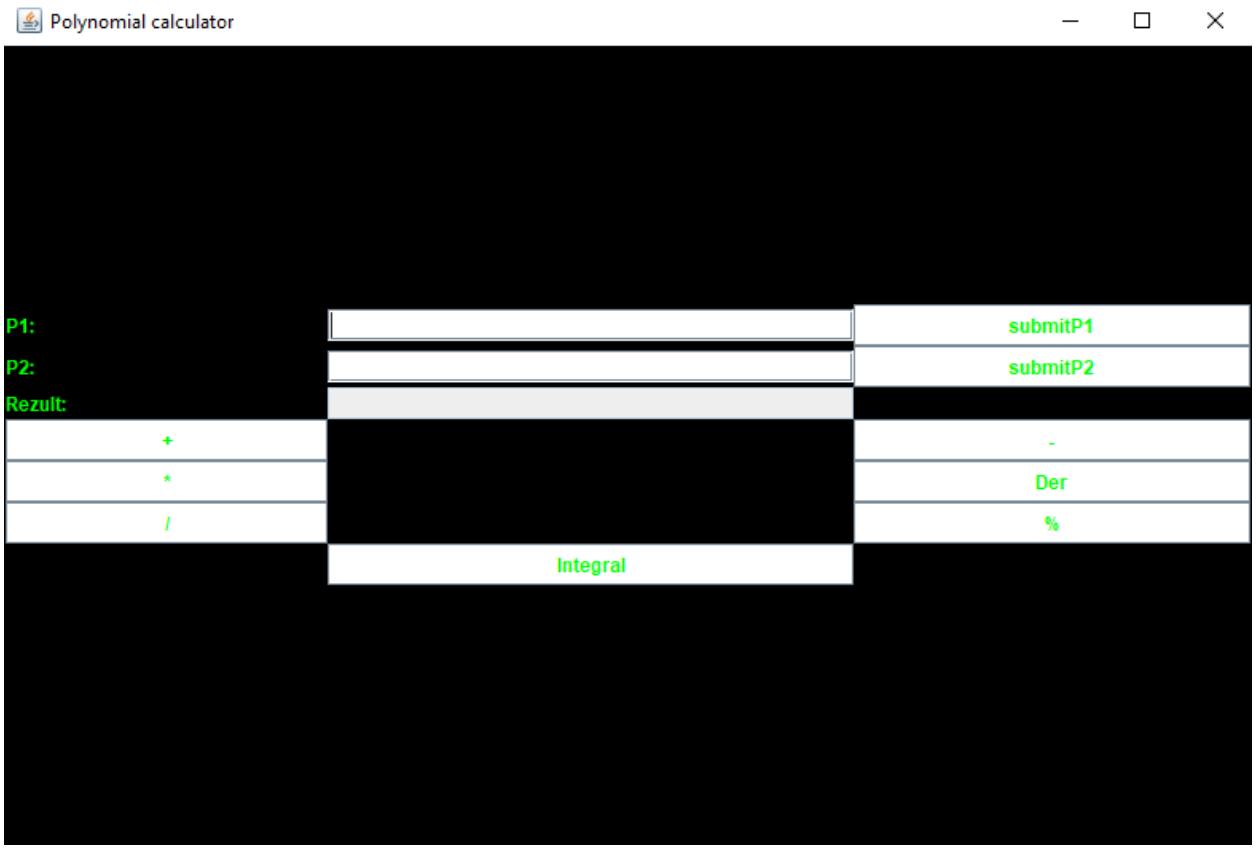
Clasa Main este mica și simpla având în vedere faptul ca tema este făcută pe arhitectura MCV. Prin urmare în clasa Main se apelează doar componentele necesare.

```
public class Main {
        public static void main(String[] args) {

        Polinom pol1=new Polinom();
        Polinom pol2=new Polinom();
        View view=new View();
        Controller controller=new Controller(pol1,pol2,view);
        view.setVisible(true);

    }
}
```

4.2. GUI



Interfața grafica este una simpla realizata din cod fără folosirea extensiilor de drag&drop.

Interfața este de asemenea și foarte ușor de folosit:

- Se introduce polinomul1 în forma ax^b, unde a și x sunt numere întregi
- Se apasă butonul submitP1
- Se introduce polinomul1 în aceeași forma și se apasă butonul submitP2
- În acest moment programul a memorat cele doua polinoame și utilizatorul poate sa efectueze orice operație pe ele, iar rezultatul operației va apărea în câmpul „Rezultat”.

Operațiile disponibile sunt: adunare(+), scădere(-), înmulțire(*), derivare(Der)- atenție se va deriva mereu primul polinom, catul împărțirii(/), restul împărțirii(%) și ingerare(Integral)- integram primul polinom.

5. Rezultate

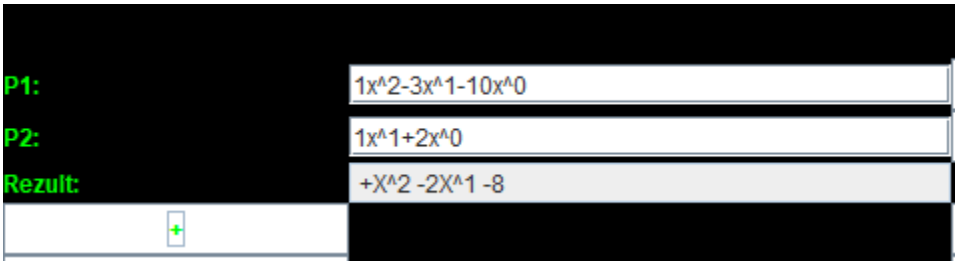
Pentru a verifica programul am testat mai multe polinoame pe care am testat toate operațiile.

Prima data adăugam doua polinoame.

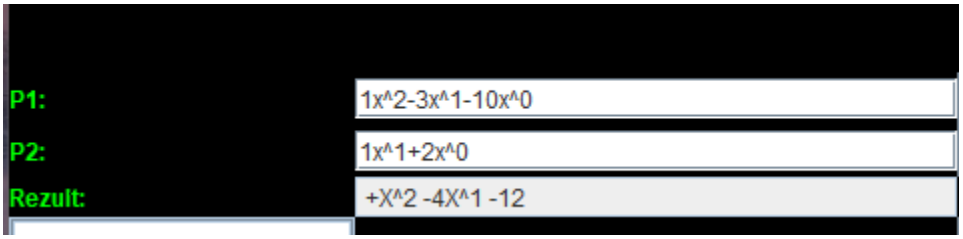
P1: 1x^2-3x^1-10x^0

P2: 1x^1+2x^0

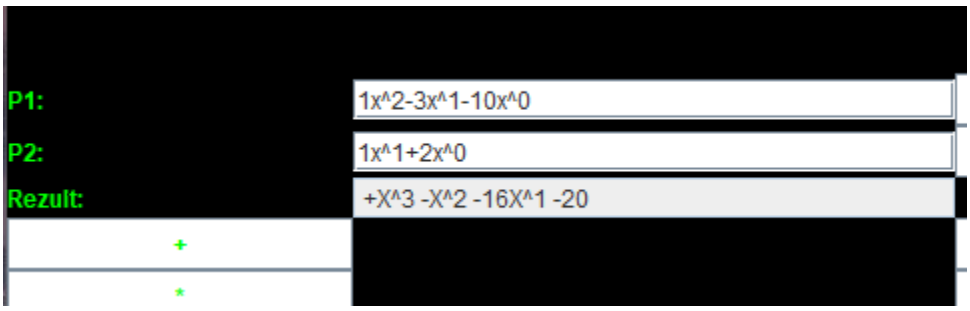
Așteptam la adunare sa obținem: 1x^2-2x^1-8 și rezultatul este:



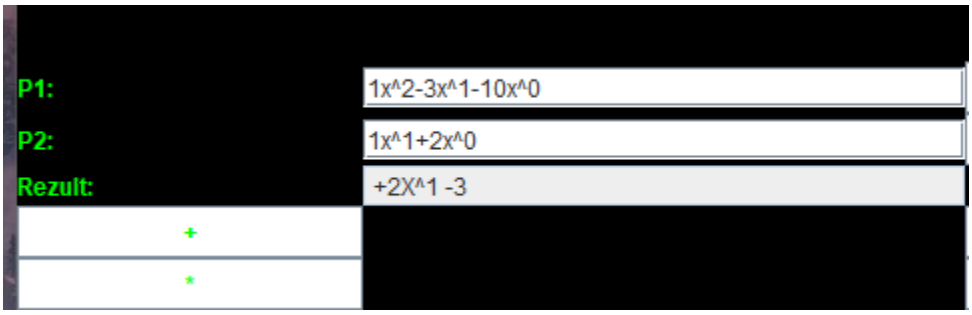
La scadere:x^2-4x^1-12



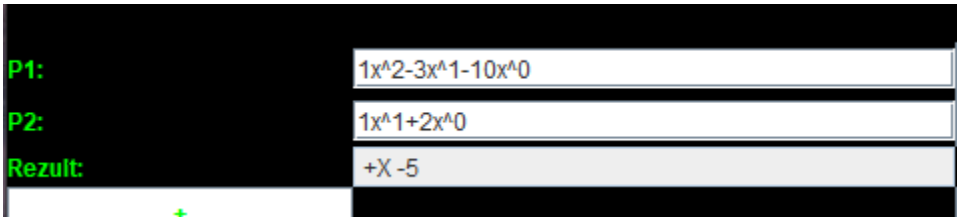
La înmulțire: x^3-x^2-16x-20



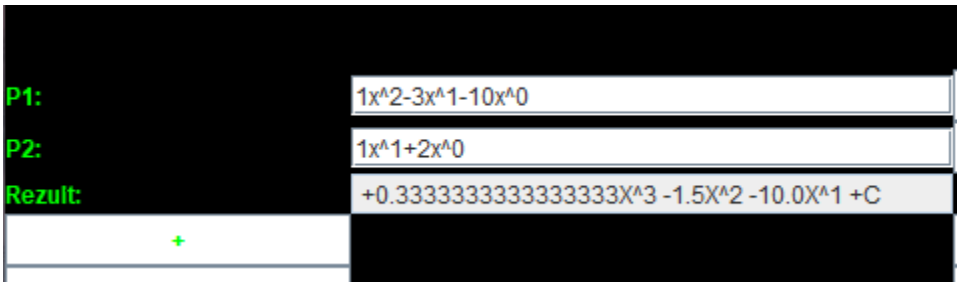
Derivarea primului ne da:2x^1-3



Catul împărțirii celor doua este: x-5 și restul 0, exact cum ne spune și aplicația:



Prin integrarea primului polinom am obține: 1/3x^3-3/2x^2-10x^1+C, dar numărul fiind reprezentat în virgula în loc de 1/3 vom obține 0.(3), însă rezultatul este unul corect.



6. Concluzii

În urma efectuării temei1 m-am familiarizat cu IntelliJ, dat fiind ca eu lucrasem în Eclipse pana acum. De asemenea mi-am readus aminte cum sa proiectez si sa implementez clase în Java, cum se leagă clasele intre ele și ce dependenta au una de alta.

Am învățat sa implementez un GUI pe model MCV într-un proiect mai mare.

Proiectul poate fi dezvoltat pe viitor prin înfrumusețarea interfeței grafice, care în momentul de fata este foarte simpla și basic. De asemenea s-ar putea implementa mai multe Regex-uri pentru polinoamele introduse, astfel încât introducerea polinoamelor sa fie ceva mai permisiva, dat fiind ca în momentul de fata modelul de introducere este unul foarte fix.

7. Bibliografie

https://google.github.io/styleguide/javaguide.html

https://docs.oracle.com/javase/tutorial/uiswing/

https://netbeans.apache.org/kb/docs/java/quickstart-gui.html

https://netbeans.apache.org/kb/docs/java/gui-functionality.html

https://regex101.com/

https://www.mathsisfun.com/algebra/polynomials-division-long.html

https://stackoverflow.com/