



DOCUMENTATIE

CALCULATOR DE POLINOAME

STUDENT : TEREZ RAZVAN DAN

GRUPA : 30221

AN UNIVERSITAR : 2020-2021

CUPRINS

1. OBIECTIVUL TEMEI
2. ANALIZA PROBLEMEI, MODELARE, SCENARIU, CAZURI DE UTILIZARE
3. PROIECTARE (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator)
4. IMPLEMENTARE
5. REZULTATE
6. CONCLUZII
7. BIBLIOGRAFIE

1. OBIECTIVUL TEMEI

Obiectivul acestei teme este de a implementa un calculator de polinoame cu interfata grafica. Prin utilizarea acestui calculator de polinoame, vom avea posibilitatea de a introduce doua polinoame, iar apoi putem obtine rezultatul uneia dintre operatiile urmatoare, asupra polinoamelor : adunare, scadere, inmultire, impartire, derivare si integrare. Pentru a realiza acest calculator, obiectivele de care m-am folosit au fost : crearea unei clase de monoame, in care sa reprezint monoamele din care va fi format un polinom, crearea clasei de polinoame, in care sa formez polinomul din mai multe monoame, si sa efectuez operatiile care se pot aplica asupra polinoamelor, dar si crearea unei clase, in care sa implementez codul pentru interfata grafica, cu ajutorul careia utilizatorul sa poata efectua operatii asupra polinoamelor. Aceste obiective au fost descrise in capitolul de proiectare.

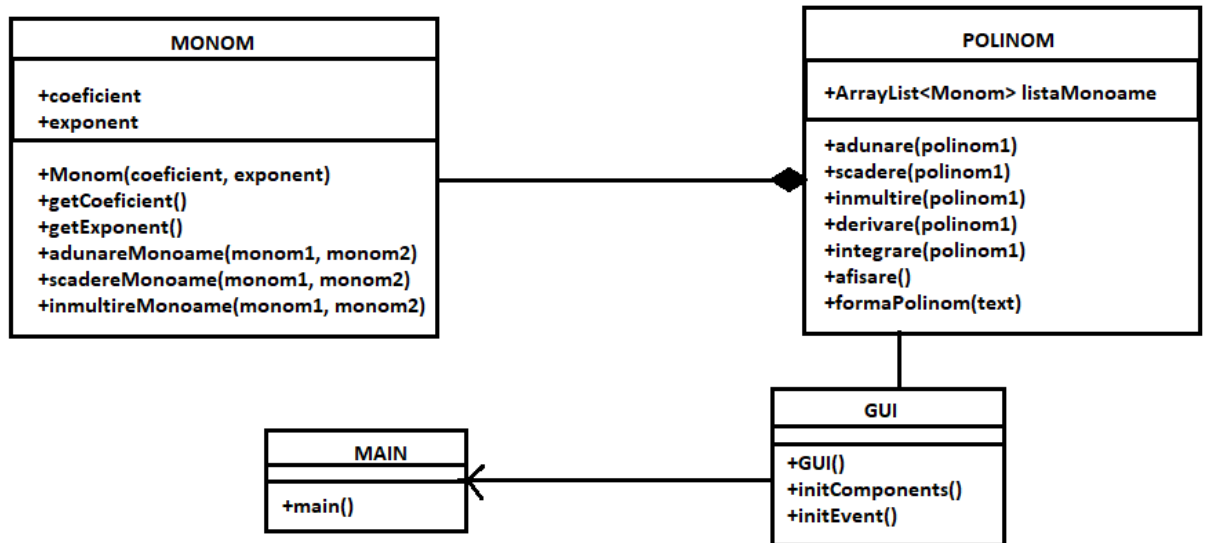
2. ANALIZA PROBLEMEI, MODELARE, SCENARIU, CAZURI DE UTILIZARE

Pentru a reprezenta polinomul, am considerat ca pot forma acest polinom din mai multe monoame, reprezentate prin coeficienti si exponenti intregi. Exponentul unei variabile dintr-un monom este egal cu gradul acelei variabile in acel monom. Pentru ca $x^1 = x$, gradul unei variabile fara exponent este unu. Un monom fara variabile se numeste monom constant, sau doar constanta. Gradul unui termen constant este 0. Coeficientul unui monom poate fi orice numar, inclusiv fractii, numere irationale sau negative. Un polinom construit cu o singura variabila se numeste univariat. In mod normal, un polinom ar trebui sa aiba termenii de grad mai mare inaintea celor cu grad mai mic.

Din cerintele acestui proiect, am reusit sa implementez o interfata grafica pentru utilizatori, operatiile de adunare, scadere si inmultire, dar am folosit si incapsularea, asa cum era specificat in cerinta. De asemenea, am folosit liste, dar si bucle foreach in loc de for.

3. PROIECTARE

DIAGRAMA UML :



Pentru proiectarea codului, am folosit 4 clase(Main, Monom, Polinom, GUI), cu ajutorul carora voi putea obtine realizarea proiectului.

Clasa Monom contine 2 attribute : coefficient si exponent, care sunt sugestive pentru coefficientul si gradul monomului. Am adaugat 2 metode getter : getCoefficient si getExponent, cu ajutorul carora voi putea obtine valoarea coefficientului si a exponentului monomului. Apoi, pentru operatia de adunare, am compus functia adunareMonoame, care are rolul de a aduna 2 monoame(adunam coefficientii, iar exponentul ramane acelasi), pentru operatia de scadere am realizat functia scadereMonoame(se scand coefficientii, iar exponentul ramane acelasi), iar metoda inmultireMonoame am folosit-o pentru inmultirea a doua monoame(inmultim coefficientii si adunam exponentii).

Clasa polinom are ca si atribut o lista de monoame, numita listaMonoame, in care se gasesc monoamele cu ajutorul carora vom forma polinomul. Ca si metode gasim metoda adunare(), de tip Polinom, care aduna 2 polinoame, metoda scadere() care scade 2 polinoame, metoda inmultire() care inmulteste 2 polinoame, metoda derivare() cu ajutorul careia obtinem derivate unui polinom, dar si clasa integrare() prin care se obtine rezultatul integralei dintr-un polinom. De asemenea avem si metoda afisare(), care rezolva problema afisarii polinomului, sub forma sa proprie. In aceasta clasa gasim si metoda formaPolinom(), unde este definit tiparul, dupa care ar trebui sa fie introduse polinoamele. Daca nu se respecta acest tipar, vom primi o eroare.

Clasa GUI este o clasa cu ajutorul careia am implementat interfata grafica a proiectului. Aici gasim ca si metode initComponents, in care initializam casutele de text si butoanele in interfata, sub forma dorita, dar si initEvent, care are rolul de a lega componentele intre ele, pentru a vedea rezultate atunci cand le utilizam.

Clasa Main contine un obiect de tipul GUI, cu ajutorul caruia setam vizibilitatea interfetei, pentru a o putea genera atunci cand dam run

INTERFATA GRAFICA :

The image shows a graphical user interface (GUI) for polynomial operations. It has a light gray background and a window-like border with three small circles in the top-left corner. The interface is divided into three main sections:

- INTRODUCETI PRIMUL POLINOM**: A text input field containing the polynomial $4x^2+6x^1-1x^0$.
- Operators**: A row of five buttons with the symbols $+$, $-$, $*$, $'$, and $|$.
- INTRODUCETI AL 2-LEA POLINOM**: A text input field containing the polynomial $4x^4+2x^2-2x^1$.
- REZULTATUL OPERATIEI**: A text input field containing the result $4x^4+6x^2+4x-1$.

4. IMPLEMENTARE

CLASA MONOM :

Cu ajutorul acestei clase am format monoamele care alcatuiesc un polinom, dar am facut si operatii asupra lor, cum ar fi adunarea, scaderea si inmultirea. Trebuie specificat faptul ca, pentru a putea realiza adunarea si scaderea a doua monoame, acestea trebuie sa aiba acelasi grad, deci pentru monoame de grad diferit, metodele de adunare si scadere nu sunt bune. In ceea ce priveste operatia de inmultire, aici nu conteaza daca monoamele au acelasi grad sau grad diferit, deoarece exponentii celor doua monoame se vor aduna.

```

package Polinom;

public class Monom {
    public int coefficient;
    public int exponent;

    public Monom(int coefficient, int exponent)
    {
        this.coefficient=coefficient;
        this.exponent=exponent;
    }

    public int getCoefficient() { return coefficient; }

    public int getExponent() { return exponent; }
}

```

Aici am creat clasa polinom, unde am adaugat si un constructor, in care am initializat coefficientul si exponentul monomului. De asemenea, am mai generat si 2 getter pentru coeficient si exponent.

```

public Monom adunareMonoame(Monom monom1, Monom monom2)
{
    int coefficient1 = monom1.getCoefficient();
    int exponent1 = monom1.getExponent();
    int coefficient2 = monom2.getCoefficient();
    int exponent2 = monom2.getExponent();
    Monom monom = new Monom( coefficient: coefficient1 + coefficient2, exponent1);
    return monom;
}

```

In poze de mai sus, am creat metoda adunareMonom, in care voi realiza operatia de adunare pentru doua polinoame. Voi aduna coeficientii celor doua polinoame, iar exponentul ramane acelasi, deoarece monoamele au acelasi grad.

```

public Monom scadereMonoame (Monom monom1, Monom monom2){
    int coefficient1 = monom1.getCoeficient();
    int exponent1 = monom1.getExponent();
    int coefficient2 = monom2.getCoeficient();
    int exponent2 = monom2.getExponent();
    Monom monom= new Monom( coefficient: coefficient1 - coefficient2,exponent1);
    return monom;
}

```

De asemenea, aceasta clasa a fost creata cu scopul de a face diferenta dintre doua monoame. Aici vom scadea coefficientii celor doua monoame, iar exponentul va ramane acelasi, deoarece ambele monoame au aceeasi putere.

```

public Monom inmultireMonoame (Monom monom1, Monom monom2)
{
    int coefficient1 = monom1.getCoeficient();
    int exponent1 = monom1.getExponent();
    int coefficient2 = monom2.getCoeficient();
    int exponent2 = monom2.getExponent();
    Monom monom= new Monom( coefficient: coefficient1 * coefficient2, exponent: exponent1 + exponent2);
    return monom;
}

```

Aceasta clasa am realizat-o pentru a face inmultirea a doua monoame. Ideea din aceasta metoda este de a crea un alt monom, care sa aiba coefficientul egal cu inmultirea coefficientilor celor doua monoame si adunarea exponentilor celor doua monoame.

CLASA POLINOM :

Prin crearea acestei clase, am dorit sa formez polinoamele, dar si sa efectuez operatii asupra lor. Am declarat ca atribut o lista de monoame, care vor alcatui polinomul. Apoi am creat metodele pentru adunare, scadere, inmultire, derivare si integrare. In ceea ce priveste adunarea si scaderea, in aceste metode voi verifica daca polinoamele au monoame cu acelasi grad, iar daca este adevarat, voi efectua operatia dorita, altfel, daca un monom din primul polinom nu are acelasi grad cu niciunul dintre monoamele din cel de-al doilea polinom, acesta se va scrie pur si simplu in rezultatul final, fara a se aduna/scadea cu un alt monom. De asemenea, una dintre metodele principale este si cea de afisare, prin care am dorit sa specific forma in care polinomul, care rezulta din cele doua polinoame, va trebui sa fie afisat. De exemplu pentru cazul in care un monom are gradul 1, va trebui sa apara in fereastra doar x, in loc de x^1 , deoarece ridicarea monomului la puterea 1 nu are sens, deoarece se obtine acelasi rezultat.

```

package Polinom;

import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Polinom {
    List<Monom> listaMonoame= new ArrayList<>();

    public List<Monom> getListaMonoame() { return listaMonoame; }

    public void setListaMonoame(List<Monom> listaMonoame) { this.listaMonoame = listaMonoame; }
}

```

Aici am realizat clasa pentru Polinom, in care am creat o lista de monoame, care va forma polinomul. De asemenea am generat metoda de setter si getter pentru lista de monoame.

```

public Polinom adunare(Polinom polinom1) {
    Polinom rezultat = new Polinom();
    Monom monom1;
    Monom monom2;
    int monomPol1=0;
    int monomPol2=0;
    Monom monom = new Monom( coefficient: 0, exponent: 0);

    while((this.listaMonoame.size() > monomPol1) && (polinom1.listaMonoame.size() > monomPol2))
    {
        monom1=this.listaMonoame.get(monomPol1);
        monom2=polinom1.listaMonoame.get(monomPol2);

        if (monom1.getExponent() < monom2.getExponent()) {
            rezultat.listaMonoame.add(monom2);
            monomPol2 = monomPol2 + 1;
        }
        else
        if (monom1.getExponent() > monom2.getExponent()) {
            rezultat.listaMonoame.add(monom1);
            monomPol1 = monomPol1 + 1;
        }
        else
        {
            monom=monom.adunareMonoame(monom1, monom2);
            rezultat.listaMonoame.add(monom);
        }
    }
}

```

```

else
if (monom1.getExponent() > monom2.getExponent()) {
    rezultat.listaMonoame.add(monom1);
    monomPol1 = monomPol1 + 1;
}
else
{
    monom=monom.adunareMonoame(monom1, monom2);
    rezultat.listaMonoame.add(monom);
    monomPol1 = monomPol1 + 1;
    monomPol2 = monomPol2 + 1;
}
}

while(this.listaMonoame.size() > monomPol1) {
    monom1=this.listaMonoame.get(monomPol1);
    rezultat.listaMonoame.add(monom1);
    monomPol1 = monomPol1 + 1;
}
while (polinom1.listaMonoame.size() > monomPol2)
{
    monom2=polinom1.listaMonoame.get(monomPol2);
    rezultat.listaMonoame.add(monom2);
    monomPol2++;
}
return rezultat;
}

```

Aceasta metoda a fost creata cu scopul de a face adunarea a doua polinoame. Vom verifica, daca exponentii monoamelor din cele 2 polinoame sunt egale, mai mici sau mai mari, iar in functie de aceste situatii vom face operatia necesara pentru polinoame.

```

public Polinom scadere(Polinom polinom1) {
    Polinom rezultat = new Polinom();
    Monom monom=new Monom( coefficient: 0, exponent: 0);
    Monom monom1;
    Monom monom2;

    int monomPol1=0;
    int monomPol2=0;
    while((this.listaMonoame.size() > monomPol1) && (polinom1.listaMonoame.size() > monomPol2))
    {
        monom1=this.listaMonoame.get(monomPol1);
        monom2=polinom1.listaMonoame.get(monomPol2);

        if (monom1.getExponent() < monom2.getExponent()) {
            monom2.coeficient=-monom2.getCoeficient();
            rezultat.listaMonoame.add(monom2);
            monomPol2 = monomPol2 + 1;
        }
        else
        if (monom1.getExponent() > monom2.getExponent()) {
            rezultat.listaMonoame.add(monom1);
            monomPol1 = monomPol1 + 1;
        }
        else {
            monom=monom.scadereMonoame(monom1, monom2);
            rezultat.listaMonoame.add(monom);
        }
    }
}

```



```

        if (monom1.getExponent() > monom2.getExponent()) {
            rezultat.listaMonoame.add(monom1);
            monomPol1 = monomPol1 + 1;
        }
        else {
            monom=monom.scadereMonoame(monom1, monom2);
            rezultat.listaMonoame.add(monom);
            monomPol1 = monomPol1 + 1;
            monomPol2 = monomPol2 + 1;
        }
    }

    while(this.listaMonoame.size() > monomPol1) {
        monom1=this.listaMonoame.get(monomPol1);
        rezultat.listaMonoame.add(monom1);
        monomPol1 = monomPol1 + 1;
    }

    while (polinom1.listaMonoame.size() > monomPol2)
    {
        monom2=polinom1.listaMonoame.get(monomPol2);
        monom2.coeficient=-monom2.getCoeficient();
        rezultat.listaMonoame.add(monom2);
        monomPol2 = monomPol2 + 1;
    }
    return rezultat;
}

```

Aici am realizat metoda pentru operatia de scadere a doua polinoame. La fel ca si la adunare, vom verifica fiecare dintre monoamele polinoamelor, si vom compara exponentii lor. Mai apoi vom forma monoamele pentru polinomul final si le vom introduce in lista.

```

public Polinom inmultire(Polinom polinom1) {
    Monom monom=new Monom( coeficient: 0, exponent: 0);
    Polinom polinom2 =new Polinom();

    for (Monom monom1:this.listaMonoame)
    {
        Polinom p1 =new Polinom();
        for (Monom monom2:polinom1.listaMonoame)
        {
            monom=monom.inmultireMonoame(monom1,monom2);
            p1.listaMonoame.add(monom);
        }
        polinom2=polinom2.adunare(p1);
    }
    return polinom2;
}

```

Aceasta este clasa pentru inmultire a doua polinoame, in care avem ca si atribut un polinom, iar in interiorul clasei am adaugat un atribut monom, dar si un alt polinom. Vom parcurge cu o bucla foreach monoamele din lista de monoame si vom inmulti fiecare monom din polinomul 1 cu celelalte monoame din polinom 2, iar apoi vom aduna toate monoamele rezultate, si va rezulta polinomul final.

```
public Polinom derivare(Polinom polinom1) {
    int monomPol1 = 0;
    Polinom rez = new Polinom();

    while(monomPol1 < polinom1.listaMonoame.size()) {
        Monom monom1 = polinom1.listaMonoame.get(monomPol1);
        monomPol1 = monomPol1 + 1;
        if(monom1.getExponent() != 0) {
            Monom monom2 = new Monom( coefficient: monom1.getCoefficient() * monom1.getExponent(), exponent: monom1.getExponent()-1);
            rez.listaMonoame.add(monom2);
        }
    }
    return rez;
}
```

Aici am realizat metoda pentru derivarea unui polinom, in care avem ca si atribut un polinom, iar in metoda am mai declarant un monom, si un rezultat de tipul polinom, in care vom stoca polinomul rezultat de dupa derivare. Derivarea se realizeaza prin scaderea cu -1 a exponentului si inmultirea coeficientului cu exponentul curent.

```
public Polinom integrare(Polinom polinom1) {
    int monomPol1 = 0;
    Polinom rez = new Polinom();

    while(monomPol1 < polinom1.listaMonoame.size()) {
        Monom monom1 = polinom1.listaMonoame.get(monomPol1);
        monomPol1 = monomPol1 + 1;
        Monom monom2 = new Monom( coefficient: monom1.getCoefficient()/(monom1.getExponent() + 1), exponent: monom1.getExponent() + 1);
        rez.listaMonoame.add(monom2);
    }
    return rez;
}
```

Aceasta metoda realizeaza integrarea unui polinom. La fel ca si la derivare, am declarant attributele polinom1, monomPol1 si rez, in care vom stoca rezultatul final. La integrare, gradul exponentului va creste cu 1, iar coeficientul va fi coeficientul initial/gradul exponentului.

CLASA GUI :

În ceea ce privește interfața grafică a acestui proiect, codul pentru realizarea acesteia este implementat în clasa GUI. În această clasă am setat dimensiunile pentru fereastra principală, dar și pentru componentele care vor apărea în ea, cum ar fi : butonul de adunare, scădere, înmulțire, derivare, integrare, casutele text, în care vom putea scrie cele două polinoame, dar și cea în care se va afișa rezultatul.. Pentru ca aceste lucruri să se realizeze, am creat metodele `initComponent`, în care am inițializat fiecare componentă a ferestrei, dar și metoda `initEvent`, în care am realizat acțiunea butoanelor.

```
package Polinom;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class GUI extends JFrame {
    public GUI(){
        setSize( width: 750, height: 360);
        setLocation(new Point( x: 100, y: 200));
        initComponents();
        initEvent();
    }
}
```

În această clasă am inițializat fereastra interfeței, având dimensiunile de 750 și 360. Apoi i-am ales o locație, iar metodele `initComponent()` și `initEvent` le-am creat ulterior, iar mai apoi le-am apelat în constructor.

```

private void initComponents(){
    getContentPane().setLayout(null);

    polinom1.setBounds( x: 260, y: 40, width: 240, height: 40);
    getContentPane().add(polinom1);
    polinom2.setBounds( x: 260, y: 180, width: 240, height: 40);
    getContentPane().add(polinom2);
    rezultat.setBounds( x: 160, y: 260, width: 461, height: 40);
    getContentPane().add(rezultat);
    textPol1.setBounds( x: 290, y: 10, width: 400, height: 25);
    getContentPane().add(textPol1);
    textPol2.setBounds( x: 290, y: 150, width: 400, height: 25);
    getContentPane().add(textPol2);
    textRez.setBounds( x: 320, y: 230, width: 200, height: 25);
    getContentPane().add(textRez);
    butonAdunare.setBounds( x: 110, y: 100, width: 70, height: 40);
    getContentPane().add(butonAdunare);
    butonScadere.setBounds( x: 230, y: 100, width: 70, height: 40);
    getContentPane().add(butonScadere);
    butonInmultire.setBounds( x: 340, y: 100, width: 70, height: 40);
    getContentPane().add(butonInmultire);
    butonDerivare.setBounds( x: 450, y: 100, width: 80, height: 40);
    getContentPane().add(butonDerivare);
    butonIntegrare.setBounds( x: 560, y: 100, width: 70, height: 40);
    getContentPane().add(butonIntegrare);
}

```

Aceasta metoda, initComponents() initializeaza fiecare componenta la dimensiunile si pozitia dorita si le plaseaza in fereastra interfetei. De exemplu, aici am initializat casutele text pentru polinom1, polinom2 si rezultat, butoanele pentru operatiile de adunare, scadere, inmultire, derivare si integrare. De asemenea am mai adaugat si cateva texte in care am sugerat text field-ul in care va trebui intrdous primul polinom, text field-ul inc are vom introduce al doilea polinom, respective text field-ul in care va aparea rezultatul dup ace operatia dorita a fost executata cu success, iar forma polinoamelor este una corecta.

```

private void initEvent() {

    this.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            System.exit( status: 1);
        }
    });

    butonAdunare.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            Polinom f1 = new Polinom();
            Polinom pol1 = new Polinom();
            Polinom pol2 = new Polinom();
            pol1 = f1.formaPolinom(polinom1.getText());
            pol2 = f1.formaPolinom(polinom2.getText());
            rezultat.setText(pol1.adunare(pol2).afisare());
        }
    });

    butonScadere.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            Polinom f1 = new Polinom();
            Polinom pol1 = new Polinom();
            Polinom pol2 = new Polinom();
            pol1 = f1.formaPolinom(polinom1.getText());
            pol2 = f1.formaPolinom(polinom2.getText());
            rezultat.setText(pol1.scadere(pol2).afisare());
        }
    });
}

```

In aceste metode am realizat ActionListener, pentru butoanele de adunare si scadere, in care am declarant doua polinoame si un polinom f1, care va reprezenta forma sub care polinoamele ar trebui introduce in casutele text. De asemenea, in pol1 am preluat polinomul care a fost scris in caseta text polinom1, in pol2 am preluat polinomul care a fost scris in caseta polinom2, iar in rezultat am adaugat adunarea/scaderea dintre cele doua polinoame din casutele text.

```

        butonInmultire.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                Polinom f1 = new Polinom();
                Polinom pol1 = new Polinom();
                Polinom pol2 = new Polinom();
                pol1 = f1.formaPolinom(polinom1.getText());
                pol2 = f1.formaPolinom(polinom2.getText());
                rezultat.setText(pol1.inmultire(pol2).afisare());
            }
        });

        butonDerivare.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Polinom f1 = new Polinom();
                Polinom pol1 = new Polinom();
                Polinom pol2 = new Polinom();
                pol1 = f1.formaPolinom(polinom1.getText());
                rezultat.setText(pol2.derivare(pol1).afisare());
            }
        });

        butonIntegrare.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Polinom f1 = new Polinom();
                Polinom pol1 = new Polinom();
                Polinom pol2 = new Polinom();
                pol1 = f1.formaPolinom(polinom1.getText());
                rezultat.setText(pol2.integrare(pol1).afisare());
            }
        });

```

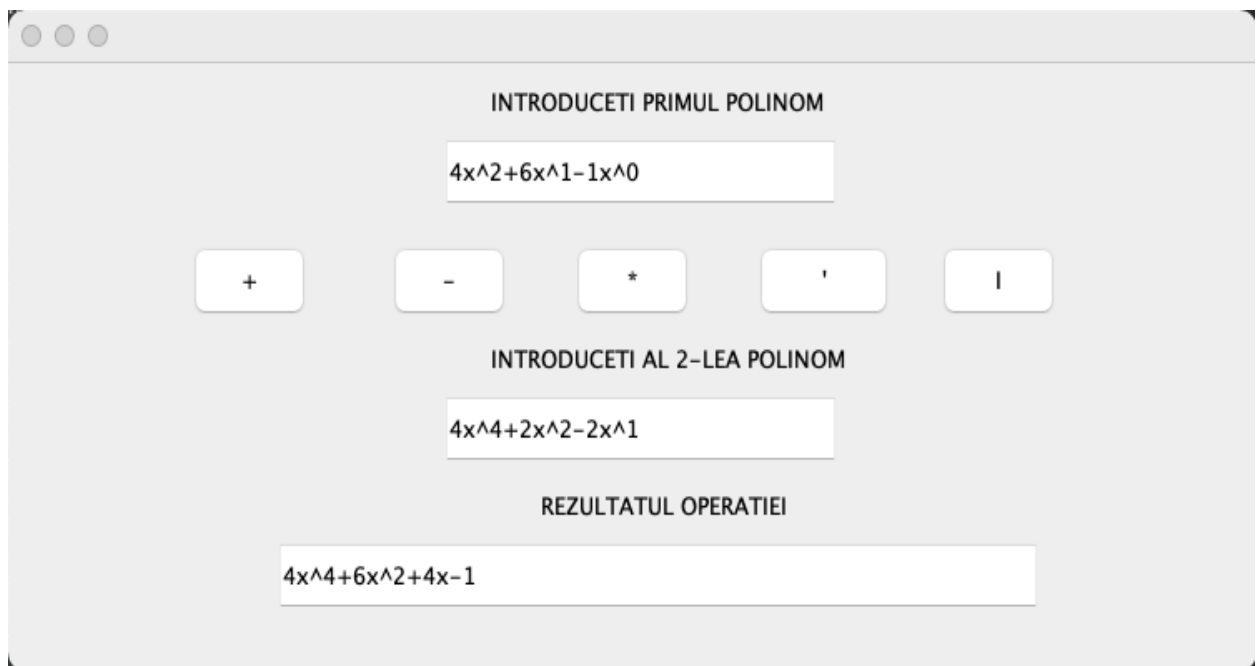
Aceste metode sunt realizate tot pentru ActionListener, dar pentru butoanele de inmultire, derivare si integrare. La fel cum am procedat si la adunare si scadere, asa vom proceda si aici. Vom avea 2 polinoame, iar, in ceea ce priveste operatia de inmultire, in pol1 vom stoca polinomul din prima caseta de text, iar in pol2 vom stoca polinomul din a doua caseta de text, iar apoi vom pune in caseta rezultat rezultatul inmultirii dintre cele doua polinoame.

In ceea ce priveste butonul de derivare, pentru a-l putea folosi, am declarant f1, pol1 si pol2, iar in pol1 vom scrie forma polinomului din prima caseta de text, iar apoi, cu ajutorul polinomului pol2 vom apela functia de derivare din clasa Polinom, si vom scrie rezultatul derivarii in caseta text pentru rezultat.

Referitor la butonul de integrare, pentru a-l putea folosi am apelat metode pentru ActionListener, in care avem 3 polinoame, iar in pol1 vom stoca polinomul din casuta text polinom1, si vom scrie rezultatul acestui polinom in casuta text rezultta.

5. REZULTATE

La finalizarea proiectului am obtinut interfata urmatoare :



The screenshot shows a graphical user interface for a polynomial calculator. It features three main input fields for polynomials and a set of operation buttons. The first input field, labeled "INTRODUCETI PRIMUL POLINOM", contains the polynomial $4x^2+6x^1-1x^0$. The second input field, labeled "INTRODUCETI AL 2-LEA POLINOM", contains the polynomial $4x^4+2x^2-2x^1$. Between these fields are five buttons for arithmetic operations: addition (+), subtraction (-), multiplication (*), division (/), and modulus (%). Below the second input field is a third input field labeled "REZULTATUL OPERATIEI" which displays the result of the addition: $4x^4+6x^2+4x-1$.

cu ajutorul careia pot sa introduc doua polinoame, si sa fac operatiile de adunare, scadere, inmultire, derivare si integrare. Pentru a nu aparea erori atunci cand dorim sa efectuam o operatie, este recomandat sa punem coeficientul in fata oricarui x (chiar si 1), sa punem exponentul oricarui x (chiar daca dorim sa reprezentam numere, fara x, va trebui sa scriem, de exemplu $5x^0$). Pentru a valida aplicatia, am testat pe cat mai multe cazuri diferite, atat inainte de a implementa interfata, cat si dupa

6. CONCLUZIE

În concluzie, acest proiect a fost unul benefic, deoarece m-a ajutat să stăpânesc mai mult limbajul JAVA, să învăț cât mai multe lucruri din erorile care mi-au apărut pe parcurs. De asemenea, mi-am aprofundat cunoștințele în ceea ce privește JAVA swing, am descoperit un mod bun de a realiza o interfață grafică, dar și cum să o fac să funcționeze util. Mi-aș fi dorit să reușesc să implementez și cealaltă operație, dar datorită faptului că nu mi-am gestionat ok timpul nu am reușit. În viitor voi îmbunătăți acest proiect, pentru a adăuga mai multe operații, dar și pentru a adăuga alte elemente.

7. BIBLIOGRAFIE

Pentru realizarea acestui proiect, m-am documentat din următoarele surse :

<https://www.javatpoint.com/java-swing>

<https://www.geeksforgeeks.org/>

<https://stackoverflow.com/>