

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE



UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

Disciplina: Tehnici de programare
fundamentale

Tema 1 : Calculator de polinoame

Documentație

Nume: Buleu Alexandru

Grupa: 30223

An: 2

Contents

1. Obiectivul temei	3
2. Analiza problemei, scenarii, cazuri de utilizare	4
Scenariul principala:	4
Scenariu alternativ:.....	4
Cerințele funcționale:	5
Cerințe non-funcționale:	5
Cerințele față de utilizator:.....	5
3.Proiectare.....	5
UML.....	5
Tipul de arhitectura	7
Diagrama de clase	7
4.Implementare	8
Clasa „PolinomModel”:	8
Clasa „PolinomView”	9
Clasa „PolinomController”	10
5.Rezultate	12
Adunare.....	12
Scadere.....	12
Derivare	13
6.Concluzii	13
7.Bibliografie	14

1. Obiectivul temei

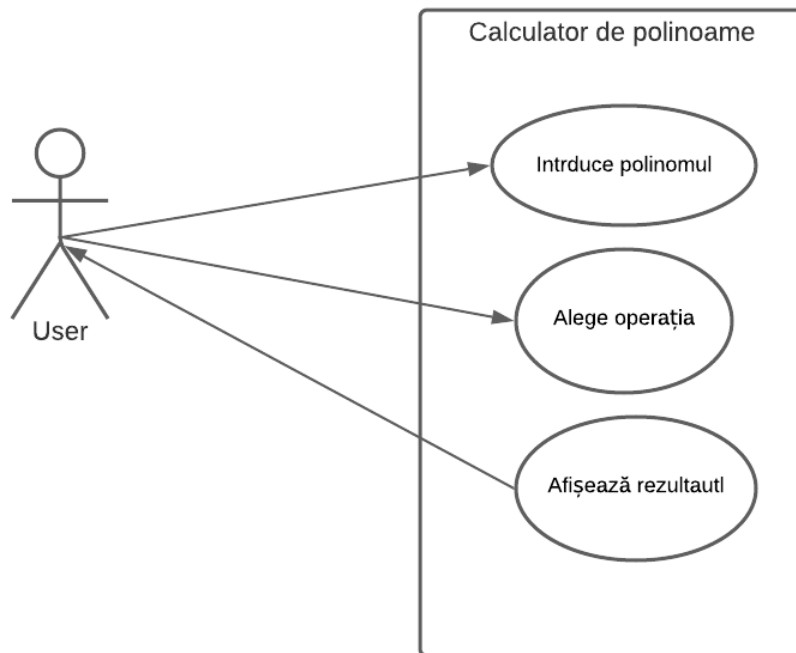
Principalul obiectiv al temei este de a implementa un calculator de polinoame cu interfață pentru utilizator. Interfața trebuie să fie simplă și intuitivă, utilizatorul trebuie să fie capabil să introducă polinoamele și să aleagă cu ușurință operația matematică care să fie efectuată iar la final să poată vedea rezultatul calculelor.

Obiectivele secundare ale temei sunt următoarele:

- Analizarea problemei și determinarea scenariilor de utilizare, acest obiectiv este analizat mai în detaliu în capitolul 2
- Proiectarea diagramei UML, acest obiectiv este analizat în detaliu în cadrul capitolului 3
- Implementarea codului, este prezentată în capitolul 4
- Descoperirea posibilităților de dezvoltare ulterioară, acest obiectiv este tratat în cadrul ultimului capitol

2. Analiza problemei, scenariii, cazuri de utilizare

Polinoamele sunt expresii aritmetice ce conțin o variabilă necunoscută "x". Fiecare polinom este format din mai multe monoame, un monom este alcătuit din coeficient și exponent. Pentru a putea să lucrez mai ușor pe un polinom, am împărțit de la citire polinomul în monoame. Operațiile sunt efectuate pe două liste de monoame iar la final lista de monoame rezultată este transformată în polinom.



Scenariul principala:

1. Utilizatorul introduce polinoamele sau polinomul (pentru derivare) în funcție de operația dorită.
2. Utilizatorul alege unul dintre butoanele: "adunare", "scadere" sau "derivare", și apasă pe unul dintre acestea
3. Calculatorul de polinoame efectuează operația solicitată de către utilizator pe cele două polinoame sau pe polinom dacă se alege derivarea.
4. Rezultatul este afișat de către calculator
5. Utilizatorul apasă pe butonul "resetare" dacă dorește să opereze un nou calcul astfel se întoarce înapoi la pasul 2

Scenariu alternativ:

1. În cazul în care utilizatorul introduce un polinom incorrect calculator o să afișeze un rezultat greșit

2. Se revine la pasul 1 de la scenariul principal introduându-se polinoamele corect.

Cerințele funcționale:

1. Calculatorul de polinoame trebuie să îi permită utilizatorului să introducă polinoamele
2. Calculatorul de polinoame trebuie să îi permită utilizatorului să aleagă cu ușurință operația matematică ce trebuie efectuată
3. Calculatorul de polinoame trebuie să implementeze operația matematică (adunare, scădere sau derivare) aleasă de către utilizator
4. Calculatorul de polinoame trebuie să îi permită utilizatorului să apese pe butonul de reset
5. Calculatorul de polinoame trebuie să șteargă ceea ce se află în câmpul "Rezultat"

Cerințe non-funcționale:

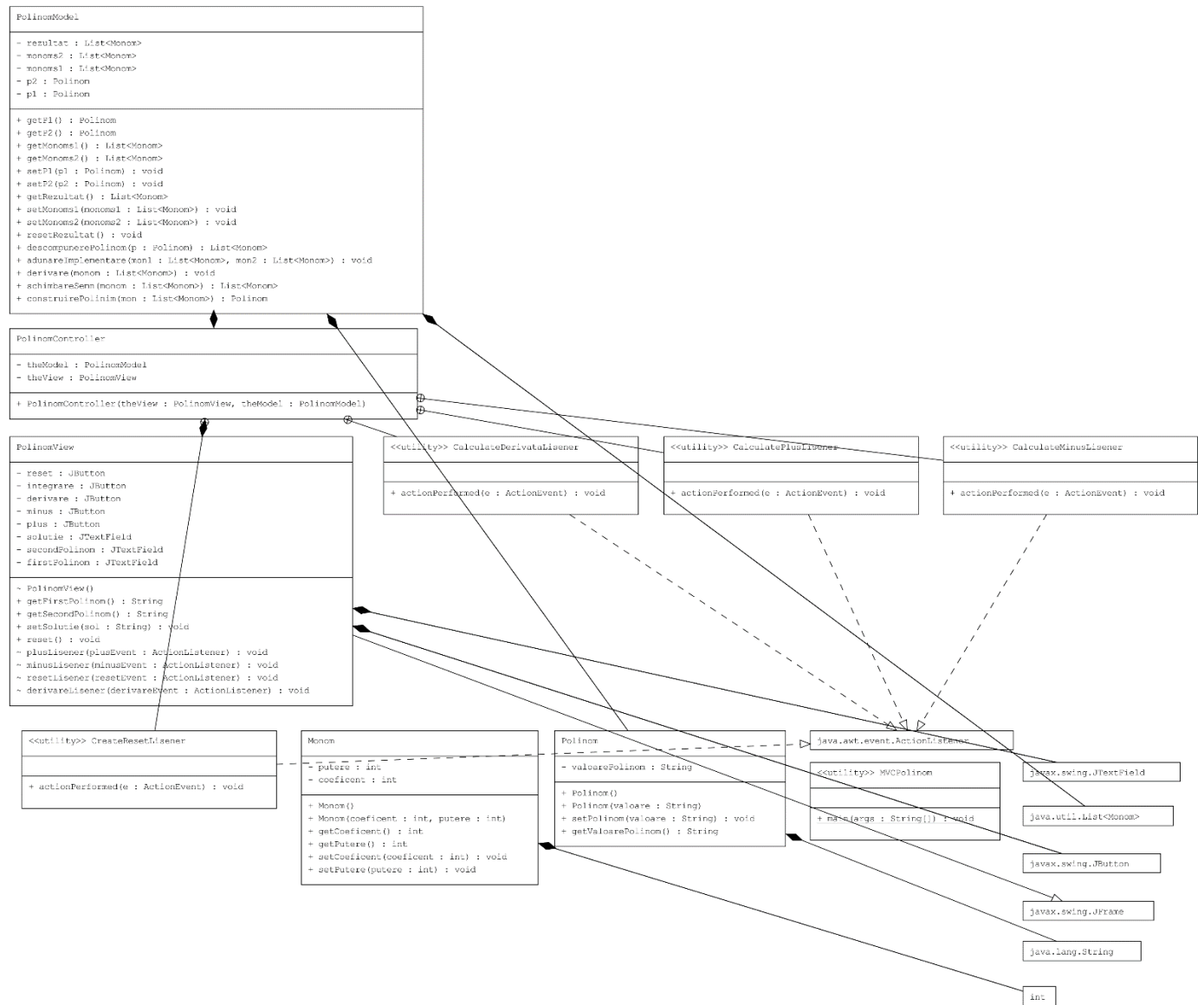
1. Calculatorul de polinoame trebuie să aibă o interfață intuitivă și ușor de utilizat

Cerințele față de utilizator:

1. Utilizatorul trebuie să introducă corect fiecare polinom
2. Fiecare monom introdus de către utilizator trebuie să conțină atât coeficientul cât și puterea, iar în cazul monoamelor care au coeficientul 1 acesta trebuie adăugat, precum următorul exemplu: " x^3 "
3. După fiecare operație matematică efectuată utilizatorul trebuie să apese pe butonul "Reset" înainte de a încerca să selecteze altă operație.
4. În cazul în care utilizatorul dorește să execute operația de derivare acesta trebuie să introducă polinomul ce urmează să fie derivat în prima casuță de text și nu în a doua, în caz contrar nu o să se întâmple nimic la apăsarea butonului de derivare

3.Proiectare

UML acronim pentru Unified Modeling Language, este un limbaj vizual de modelare utilizat pentru specificarea, construirea și documentarea sistemelor de aplicații orientate obiect și nu numai. Diagramele de clase sunt folosite în modelarea orientată obiect pentru a descrie structura statică sistemului, modulul în care este el structurat. Diagrama de clase pentru calculatorul de polinoame este următoarea:



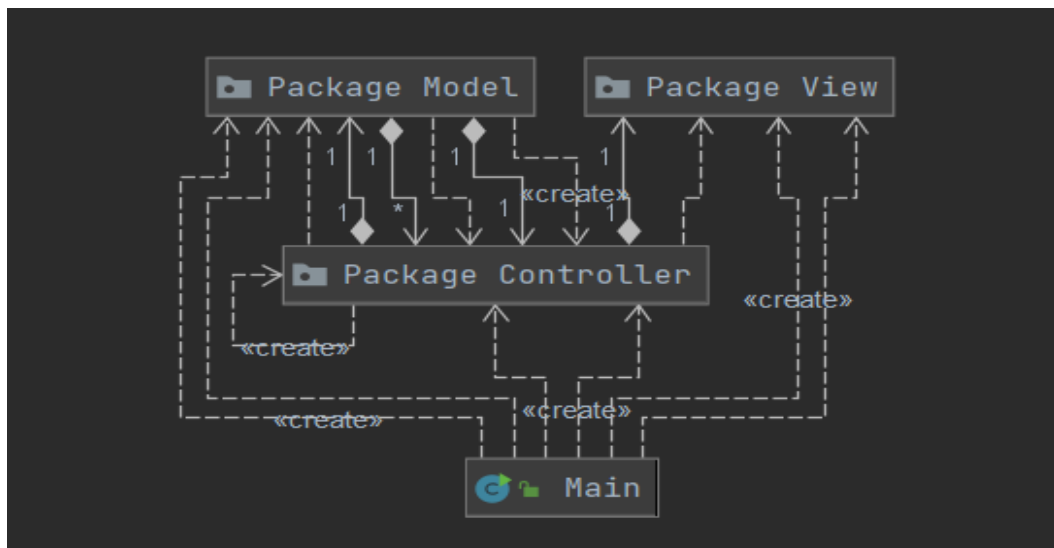
Tipul de arhitectura folosit este MVC (Model View Controller) aceasta este un model de arhitectură software care separă reprezentarea informațiilor din interacțiunea cu utilizatorul cu informațiile în sine.

Modelul este responsabil cu gestionarea datelor din aplicație. Răspunde la cereri care vin din View, deasemenea și instrucțiunilor din Controller. Este cel mai jos nivel care se ocupă cu menținerea datelor.

View-ul este partea în care se afișează datele pentru utilizator.

Controller-ul este partea aplicației care se ocupă de interacțiunea cu utilizatorul. În controller se citesc datele introduse de utilizator, se trimit către model, se execută operațiile, după care se trimite răspunsul către view.

Diagrama de clase generată de către IntelliJ este următoarea:



Pachetul Controller apelează atât pachetul View cât și Model în timp ce pachetele View și Model nu știu unul de existența celuilalt.

Structurile folosite în cadrul acestui proiect sunt listele, în principal listele de monoame iar un algoritm folosit este cel de tipul merge sort pentru implementarea operației de adunare.

4.Implementare

În cadrul implementării a folosit tipul de arhitectură MVC(Model View Controller). Pentru a stoca polinoamele am construit clasa Polinom, este o clasă simplă care conține doar un String cu valoarea polinomului. După care am împărțit polinomul în monoame astfel am realizat clasa Monom care conține puterea și coeficientul fiecărui monom.

Clasa „PolinomModel”:

Are drept attribute două Polinoame și trei liste de Monoame, câte o listă de Monoame pentru fiecare Polinom și încă o listă pentru rezultatele operațiilor care sunt realizate pe monoame. În cadrul acestei clase se găsesc settere și gettere ce sunt folosite pentru a obține valorile atributelor sau pentru a schimba valorile acestora. Se folosesc settere și gettere pentru a ajuta la încapsularea datelor, astfel datele din cadrul clasei sunt declarate private iar singura metodă prin care se poate ajunge la ele este prin intermediul metodelor `get` și `set`.

Prima metodă importantă ce se găsește în cadrul acestei clase este `”descompunerePolinom”`, această metodă returnează o listă de monoame și primește ca parametru un polinom. Prima dată se verifică dacă polinomul trimis ca parametru este introdus corespunzător cu ajutorul funcțiilor `Regex(Pattern și Matcher)` și a unui șir `Regex` ce compune un template pentru polinoame. După aceea fiecare monom din polinom este luat în mod separat și este adăugat în lista de monoame care urmează să fie returnată. Am folosit funcția `”split”` pentru a împărți monoamele de tipul `”3x^2”` în coeficient și putere care sunt trimise ca attribute pentru monomul nou creat.

Următoarea metodă din cadrul acestei clase este `”adunareImplementare”`. Această funcție primește ca parametru două liste de Monoame și nu returnează nimic, rezultatul adunării este stocat în lista de Monoame `”rezultat”` de la începutul clasei. Algoritmul folosit pentru adunarea celor două liste este de tipul `Merge`. Se parcurg cele două liste primite ca parametrii în același timp, iar în cazul în care puterile celor două obiecte din listă sunt egale atunci se adună coeficienții și se adaugă în lista `„rezultat”` un nou Monom ce are drept coeficient suma obținută și drept putere are puterea elementului din prima listă. În cazul în care puterile celor două monoame nu sunt egale atunci se adaugă în lista `„rezultat”` Monomul cu puterea mai mare. La final dacă mai există elemente în una din cele două liste acestea sunt adăugate și ele în lista `„rezultat”`.

Metoda `„schimbareSemn”` primește drept parametru o listă de Monoame și returnează tot o listă de Monoame asemănătoare cu lista primită ca parametru doar că fiecare monom din cadrul listei returnate are semnul schimbat. Această metodă este folosită pentru a realiza scăderea a două polinoame.

```
public List<Monom> schimbareSemn(List<Monom> monom){  
    for(Monom m: monom){  
        m.setCoeficient(m.getCoeficient() * (-1));    }  
    return monom;  
}
```



```
}
```

O altă metodă din clasa model care realizează o operație matematică este metoda denumită „derivare”. Aceasta primește drept parametru o listă de Monoame și, asemănător cu metoda „adunareImplementare”, stochează rezultatul operației în lista „rezultat”. După cum spune și numele metodei, aceasta realizează operația de derivare.

Ultima metodă din cadrul clasei „PolinomModel” este „construirePolinim”, metoda primește drept parametru o listă de Monoame și returnează polinomul format de monoamele din lista dată. Fiecare monom este concatenat la un string care este mai apoi folosit pentru a seta valoarea polinomului returnat. În cadrul acestei metode am folosit funcția „replace()” pentru a evita cazurile în care se dublează semnul „+” sau semnul „-” sau atunci când acestea apar amândouă concomitent.

```
public void derivare (List<Monom> monom ) {  
    for(Monom m : monom) {  
        int c = m.getCoeficient();  
        int p = m.getPutere();  
        if (p > 0) {  
            m.setCoeficient(p * c);  
            m.setPutere(p - 1);  
            rezultat.add(m);  
        }  
    }  
}
```

Clasa „PolinomView” reprezintă interfața pentru utilizator, interfața implementată este foarte simplă și intuitivă, aceasta este generată cu ajutorul Winodow Builder din Eclipse. Interfața conține 5 butoane (plus, minus, derivare, integrare și reset) 3 câmpuri de text (două în care se introduc polinoamele și unul în care se afișează rezultatul operațiilor matematice) și două câmpuri tip label („Introduceți polinoamele” și „Rezultat:”). În cadrul acestei clase am declarat și acțiunile implementate în cadrul clasei „PolinomController” iar acestea reprezintă ce se întâmplă atunci când unul dintre butoane este apăsat.

```
private JTextField firstPolinom = new JTextField(20);  
private JTextField secondPolinom = new JTextField(20);  
private JTextField solutie = new JTextField(20);  
private JButton plus = new JButton("+");
```

```

private JButton minus = new JButton("-");

private JButton derivare = new JButton("Derivare");

private JButton integrare = new JButton("Integrare");

private JButton reset = new JButton("reset");

```

Clasa „PolinomController” ține legatura atât cu clasa „PolinomView” cât și cu clasa „PolinomModel”. Clasa dată conține două atribute, unul de tipul PolinomView și unul de tipul PolinomModel. În cadrul acestei clase sunt implementate subclase care definesc acțiunile ce au loc la apăsarea butoanelor.

```

private PolinomView theView;

private PolinomModel theModel;

```

Prima clasă este „CalculatePlusLisener”, acesta implementează operația de adunare pe cele două polinoame atunci când butonul „+” este apăsat. În cadrul acestei clase sunt declarate două polinoame iar apoi se apelează metoda „adunareImplementare” din clasa „PolinomModel”. Următoarea linie de cod reprezintă adunarea celor două polinoame care sunt descompuse în liste de monoame:

```

theModel.adunareImplementare(theModel.descompunerePolinom( firstPolinom )
, theModel.descompunerePolinom( secondPolinom ));

```

După ce se execută linia de cod de mai sus, rezultatul care este tot o listă de monoame, este transformat într-un polinom și este afișat în interfața grafică prin intermediul metodei „setSolutie” din clasa „PolinomView”

A doua clasă internă este „CalculateMinusLisener” în care se declară două polinoame, după care cel de al doilea polinom este descompus în listă de monoame iar semnele fiecărui monom din listă este schimbat. După ce semnele au fost schimbate se realizează operația de adunare între lista de polinoame cu semnele schimbate și primul polinom declarat care este și el descompus în listă de monoame. La fel ca la clasa anterioară rezultatul este transformat în polinom după care este afișat în interfața grafică. Codul din clasa „CalculateMinusLisener” :

```

theModel.schimbareSemn(theModel.descompunerePolinom(secondPolinom));
theModel.adunareImplementare(theModel.descompunerePolinom(firstPolinom), theModel
.schimbareSemn( theModel.descompunerePolinom( secondPolinom )));

Polinom rez = new Polinom();

```

```
rez = theModel.construirePolinim(theModel.getRezultat());  
theView.setSolutie(rez.getValoarePolinom());
```

O altă casă internă din cadrul clasei „PolinomController” este „CalculateDerivataLisener” în care se declară un polinom ce urmează să fie descompus în listă de monoame care apoi este derivată iar în final lista rezultat este transformată în polinom și polinomul este afișat în interfața grafică.

```
Polinom firsPolinom = new Polinom(theView.getFirstPolinom());  
theModel.derivare(theModel.descompunerePolinom(firsPolinom));  
Polinom rez = new Polinom();  
rez = theModel.construirePolinim(theModel.getRezultat());  
theView.setSolutie(rez.getValoarePolinom());
```

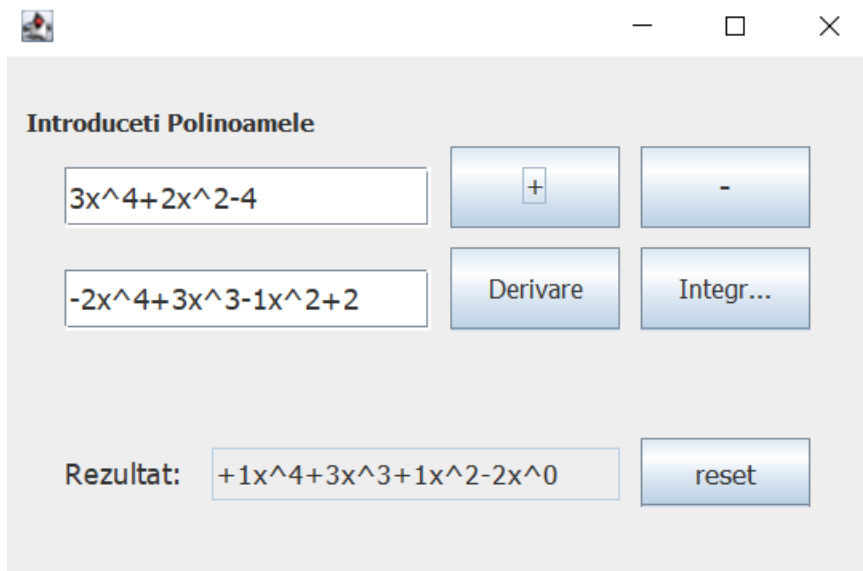
Ultima clasă internă este „CreateResetLisener” care golește lista „rezultat” de monoame și șterge din interfața utilizatorului ceea ce se află în câmpul în care se afișează rezultatul.

```
public void actionPerformed(ActionEvent e) {  
    theView.reset();  
    theModel.resetRezultat();  
}
```

5.Rezultate

În cadrul acestui proiect nu am reușit să folosesc testarea cu un anumit framework. Drept teste am introdus manual cateva operații pentru :

Adunare



A screenshot of a web-based calculator interface for adding polynomials. The window has a title bar with a small icon, a minus sign, a maximize button, and a close button. The main area is titled "Introduceti Polinoamele". It contains two input fields for polynomials. The first field contains $3x^4+2x^2-4$. The second field contains $-2x^4+3x^3-1x^2+2$. Between the fields are two buttons: "+" and "-". Below the second field are two buttons: "Derivare" and "Integr...". At the bottom, there is a "Rezultat:" label followed by a text box containing $+1x^4+3x^3+1x^2-2x^0$ and a "reset" button.

Introduceti Polinoamele

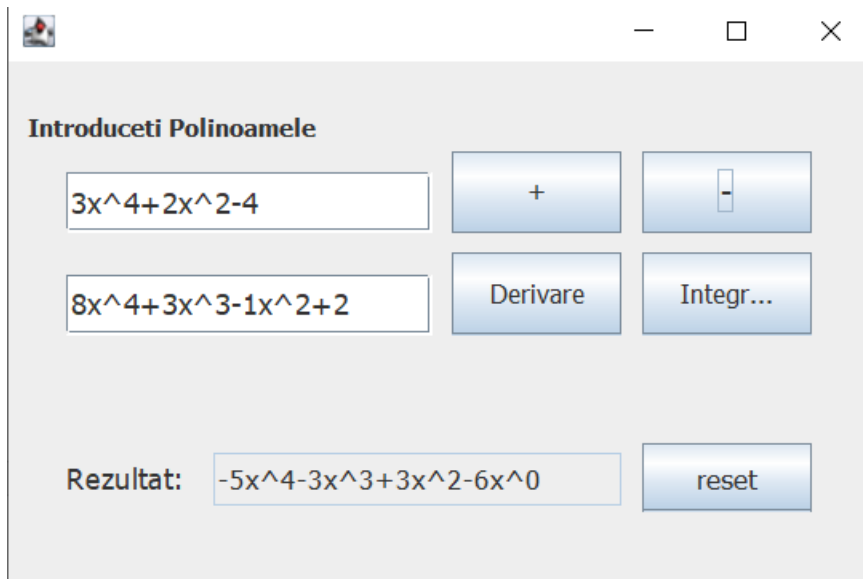
$3x^4+2x^2-4$

$-2x^4+3x^3-1x^2+2$

Derivare Integr...

Rezultat: $+1x^4+3x^3+1x^2-2x^0$ reset

Scadere



A screenshot of a web-based calculator interface for subtracting polynomials. The window has a title bar with a small icon, a minus sign, a maximize button, and a close button. The main area is titled "Introduceti Polinoamele". It contains two input fields for polynomials. The first field contains $3x^4+2x^2-4$. The second field contains $8x^4+3x^3-1x^2+2$. Between the fields are two buttons: "+" and "-". Below the second field are two buttons: "Derivare" and "Integr...". At the bottom, there is a "Rezultat:" label followed by a text box containing $-5x^4-3x^3+3x^2-6x^0$ and a "reset" button.

Introduceti Polinoamele

$3x^4+2x^2-4$

$8x^4+3x^3-1x^2+2$

Derivare Integr...

Rezultat: $-5x^4-3x^3+3x^2-6x^0$ reset

Derivare

Introduceti Polinoamele

$3x^4+5x^3-2x^2+8x-4$

+

-

Derivare

Integr...

Rezultat: $+12x^3+15x^2-4x^1+8x^0$

reset

După cum se poate observa în pozele de mai sus, rezultatele sunt corecte, în interiorul câmpului rezultat polinomul este reprezentat cu toate valorile lui x chiar și „ $8x^0$ ”

6.Concluzii

Din această temă pot să spun că am învățat cum să lucrez cu arhitectura tip MVC și să organizez mai bine codul, un nou lucru pe care am învățat să îl folosesc este încapsularea. În ceea ce privește interfața grafică am descoperit că Layoutul poate să fie setat la „null” ceea ce oferă o flexibilitate mai mare în aranjarea elementelor pe panou.

Din punct de vedere al dezvoltării ulterioare se poate implementa operațiile de înmulțire, împărțire și integrare. Pentru a putea implementa operația de integrare și de împărțire cu succes este nevoie să se schimbe tipul coeficienților monoamelor din int în double astfel în cazul în care împărțirea returnează un număr cu zecimale acesta să fie reprezentat corect.

7.Bibliografie

1. <https://en.wikipedia.org/wiki/Model–view–controller>
2. https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm
3. ASSIGNMENT_1_SUPPORT_PRESENTATION
4. <https://app.diagrams.net>