

**TEHNICI DE PROGRAMARE FUNDAMENTALE
ASSIGNMENT 1**

**POLYNOMIAL CALCULATOR
DOCUMENTATIE**

Bakk Cosmin-Robert
Grupa 30227

Cuprins

1. Obiectivul temei
2. Analiza problemei, modelare, scenarii, cazuri de utilizare
3. Proiectare
 - 3.1. Diagrama UML
 - 3.2. Proiectare clase
 - 3.4. Interfata grafica
4. Implementare si testare
 - 4.1. Metode
 - 4.2. Metode interfata grafica
 - 4.3. Testare
5. Rezultate
6. Concluzii
7. Bibliografie

1. Obiectivul temei

Obiectivul acestei teme de laborator este de a implementa un calculator de polinoame. Un utilizator poate sa introduca doua polinoame si sa selecteze una dintre cele 6 operatii: adunare, scadere, impartire, inmultire, derivare, integrare. In urma operatiei alese, se va afisa rezultatul acesteia.

Obiectivele secundare sunt: respectarea paradigmei programarii orientate pe obiect, creare unei interfete grafice si utilizarea acesteia folosind modelul architectural Model-View-Controller – detaliat in capitolul 4, testarea operatiilor cu JUnit Test – detaliat in capitolul 4.

2. Analiza problemei, modelare, scenarii, cazuri de utilizare

Pentru a transforma un string introdus de un utilizator, vom filtra sirul printr-un sablon de cautare, pentru a gasi expresii regulate (Regex). String-ul va fi despartit in monoame, fiecare polinom fiind format dintr-o lista de monoame. Monoamele au cate un coeficient, real, si cate un exponent, intreg. Cerintele sunt ca utilizatorul sa introduca doar expresii care contin monoame de tipul " ax^b ", " ax ", " x ", " x^b ", " a ", unde a este de tipul double si b este de tipul int. Aceste monoame vor fi despartite de operatorii "+" sau "-".

Descriere use-case: se citeste care dintre butoane a fost apasat de catre utilizator:

-Utilizatorul apasa "Adunare". Se citesc cele doua textfield-uri de input si se apeleaza functia `adunare(Polinom x, Polinom y)`, iar rezultatul acestei functii este pus in primul textfield de output.

-Utilizatorul apasa "Scadere". Se citesc cele doua textfield-uri de input si se apeleaza functia `diferenta(Polinom x, Polinom y)`, iar rezultatul acestei functii este pus in primul textfield de output.

-Utilizatorul apasa "Inmultire". Se citesc cele doua textfield-uri de input si se apeleaza functia `inmultire(Polinom x, Polinom y)`, iar rezultatul acestei functii este pus in primul textfield de output.

-Utilizatorul apasa "Impartire". Se citesc cele doua textfield-uri de input si se apeleaza functia `impartire(Polinom x, Polinom y)`, iar rezultatele acestei functii, catul si restul, sunt puse in primul, respectiv al doilea textfield de output.

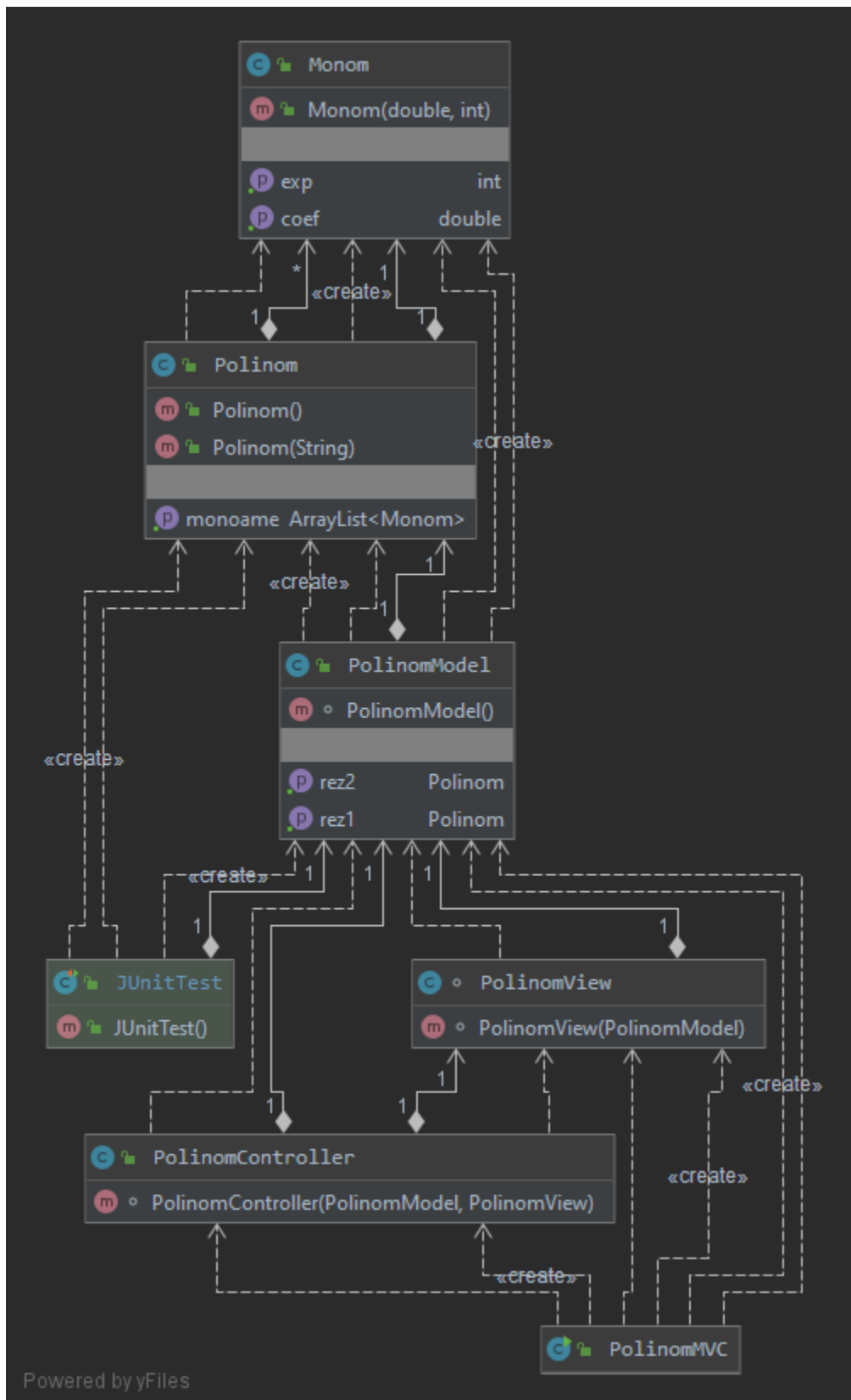
-Utilizatorul apasa "Derivare". Se citesc cele doua textfield-uri de input si se apeleaza de doua ori functia `derivare(Polinom x)`, iar rezultatele acestei functii sunt puse in textfield-urile de de output.

-Utilizatorul apasa "Integrare". Se citesc cele doua textfield-uri de input si se apeleaza de doua ori functia `integrare(Polinom x)`, iar rezultatele acestei functii sunt puse in textfield-urile de de output.

-Utilizatorul apasa "Clear". Continuturile celor doua textfield-uri de input si a celor doua textfield-uri de output sunt golite.

3. Proiectare

3.1. Diagrama UML



3.2. Proiectare clase

Clasa Monom: implementeaza Comparable. Aceasta clasa defineste conceptul de monom, acesta fiind caracterizat printr-un coeficient si printr-un exponent. Constructorul clasei seteaza coeficientul si exponentul, primite ca parametrii.

Clasa Polinom: aceasta clasa defineste conceptul de polinom, acesta fiind caracterizat printr-un ArrayList de monoame. Constructorul fara parametrii adauga listei de monoame un monom cu coeficient si exponent 0. Constructorul cu parametru de tip String apeleaza metoda crearePolinom, avand parametru string-ul trimis in constructor.

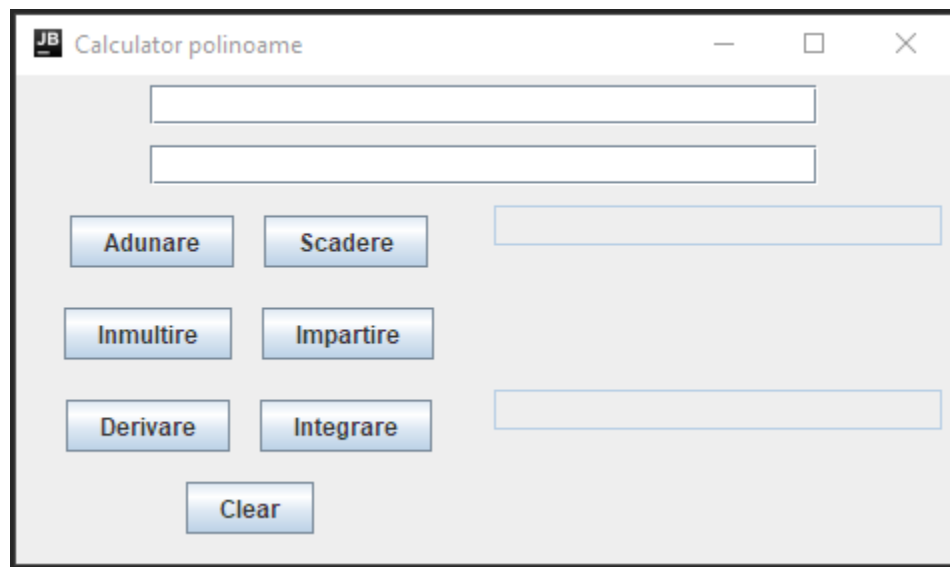
Clasa PolinomModel: aceasta clasa contine toate operatiile care pot fi aplicate pe polinoame: adunare, scadere, inmultire, inmultire, derivare si integrare si are doua polinoame care vor lua valorile rezultatelor operatiilor. Constructorul apeleaza metoda reset, care reseteaza cele doua rezultate.

Clasa PolinomView: extinde JFrame. Aceasta clasa este formata din componentele interfetei grafice si asezarea lor in JPanel-uri. Metodele acestei clase sunt cele pentru preluare de polinoame, pentru setare a rezultatelor si pentru adaugare de ascultatori pe butoane. Constructorul aranjeaza toate componentele in JPanel-uri.

Clasa PolinomController: aceasta clasa contine clasele de ascultatori pe butoane. Constructorul primeste un view si un model ca parametrii si adauga ascultatorii in view.

Clasa PolinomMVC: aceasta clasa este clasa main a proiectului, aici se realizeaza aparitia interfetei grafice

3.3. Interfata grafica



Interfata grafica este o interfata User-Friendly. Aceasta este formata dintr-un Frame contine doua text field-uri de input, 7 butoane si doua text field-uri de output. Dupa introducerea a doua polinoame si apasarea unui buton, rezultatul sau rezultatele operatiilor vor fi afisate in text field-urile de output.



Butoanele Adunare, Scadere, Inmultire, Impartire, Derivare si Integrare sunt incluse fiecare in cate un panel de tip `FlowLayout`: 1, 2, 3, 4, 5, 6.

(1,2),(3,4),(5,6)–aceste perechi intra fiecare in componenta unui panel de tip `BorderLayout`, cu o linie si doua coloane: 7, 8, 9.

Butonul Clear e inclus intr-un panel de tip `FlowLayout`: 10.

Cele doua text field-uri pentru output sunt incluse fiecare in cate un panel de tip `FlowLayout`: 11, 12.

(7,8,9,10)–aceste panel-uri intra in componenta unui panel de tip `BorderLayout`, cu 4 linii si o coloana: 13.

(11,12)–aceste panel-uri intra in componenta unui panel de tip `BorderLayout` cu 2 linii si o coloana: 14.

Cele doua text field-uri pentru input sunt incluse fiecare in cate un panel de tip `FlowLayout`: 15, 16.

(13,14)–aceste panel-uri intra in componenta unui panel de tip `BorderLayout` cu o linie si doua coloane: 17.

Cele 3 panel-uri, 15, 16 si 17 sunt apoi adaugate in frame pe pozitiile NORTH, CENTER si SOUTH.

4. Implementare si testare

4.1. Metode

Clasa Monom

```
public double getCoef();
```

-returneaza coeficientul monomului

```
public int getExp();
```

-returneaza exponentul monomului

```
public int compareTo(Object x);
```

-compara doua monoame in functie de exponentul acestora

Clasa Polionom

```
public void crearePolinom(String x);
```

-converteste string-ul primit in monoame, astfel: pentru inceput, am ales ca in expresia introdusa de utilizator, toate semnele “-” sa fie inlocuite cu “+”. Astfel, se va putea face split in functie de “+”, fara a pierde semnul coeficientilor. Apoi, am inlocuit expresia “x+” cu “x^1+”. Apoi, tuturor numerele de tip double carora le lipsea “x” langa, le-am adaugat “x^0”. Expresiile de tipul “+x” sau “-x” vor fi inlocuite cu “+1x”, respectiv “-1x”. In final, se face split dupa semnul “+”. Fiecarui sir format dupa split i se va face din nou split, dupa “x^”. Daca lungimea vectorului format este 2, atunci primul termen va fi coeficientul, iar cel din dreapta, exponentul. Altfel, constructorul va arunca o exceptie de tipul `NumberFormatException`

```

public ArrayList<Monom> getMonoame();
    -returneaza lista de monoame a polinomului

public void adaugaMonom(Monom x);
    -cauta in lista de monoame a polinomului un monom cu acelasi exponent ca si al monomului primit ca parametru. Daca se gaseste, atunci in locul monomului gasit se pune un alt monom cu acelasi exponent, iar coeficientul este suma coeficientilor celor doua monoame

public void restrangere();
    -restrange termenii polinomului, astfel incat sa nu existe doua monoame cu acelasi exponent in polinom

public String toString();
    -returneaza polinomul sub forma de string, pentru a putea fi afisat utilizatorului

```

Clasa MonomModel

```

public void reset();
    -cele doua rezultate sunt resetate, luand valoarea 0

public void adunare(Polinom x, Polinom y);
    -in primul rezultat se pune suma dintre cele doua polinoame primite ca parametru

public void diferenta(Polinom x, Polinom y);
    -in primul rezultat se pune diferenta dintre cele doua polinoame primite ca parametru

public void inmultire(Polinom x, Polinom y);
    -in primul rezultat se pune produsul dintre cele doua polinoame primite ca parametru

public void impartire(Polinom x, Polinom y);
    -in primul rezultat se pune catul impartirii dintre cele doua polinoame primite ca parametru, iar in al doilea se pune restul acestei impartiri

public void derivare(Polinom x);
    -in primul rezultat se pune rezultatul obtinut in urma derivarii polinomului primit ca parametru

public void integrare(Polinom x);
    -in primul rezultat se pune rezultatul obtinut in urma integrarii polinomului primit ca parametru

public Polinom getRez1();
    -returneaza primul rezultat

public Polinom getRez2();
    -returneaza cel de-al doilea rezultat

```

4.2. Metode interfata grafica

Clasa MonomView

```

void reset();
    -reseteaza cele 4 text field-uri

String getPolinom1();
    -returneaza string-ul din primul text field de input

String getPolinom2();
    -returneaza string-ul din al doilea text field de input

```

```

void setRezultat1(String rez);
    -afiseaza string-ul rez in primul text field de output

void setRezultat2(String rez);
    -afiseaza string-ul rez in al doilea text field de output

void showError(String errMessage);
    -afiseaza o eroare pe ecran, intr-un JFrame nou

void addAdunareListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Adunare"

void addDiferentaListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Scadere"

void addInmultireListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Inmultire"

void addImpartireListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Impartire"

void addDerivareListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Derivare"

void addIntegrareListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Integrare"

void addClearListener(ActionListener a);
    -se adauga ActionListener-ul a pe butonul "Clear"

```

Clasa MonomController

Aceasta clasa nu are metode proprii, are doar clase care implementeaza ActionListeneri:

```

class AdunareListener implements ActionListener
    -contine metoda public void actionPerformed(ActionEvent e);
    -sunt citite cele 2 text field-uri de input si se apeleaza metoda adunare(A, B); primul text field este setat cu rez1, iar al doilea text field este setat cu null

class DiferentaListener implements ActionListener
    -contine metoda public void actionPerformed(ActionEvent e);
    -sunt citite cele 2 text field-uri de input si se apeleaza metoda scadere(A, B); primul text field este setat cu rez1, iar al doilea text field este setat cu null

class InmultireListener implements ActionListener
    -contine metoda public void actionPerformed(ActionEvent e);
    -sunt citite cele 2 text field-uri de input si se apeleaza metoda inmultire(A, B); primul text field este setat cu rez1, iar al doilea text field este setat cu null

class ImpartireListener implements ActionListener
    -contine metoda public void actionPerformed(ActionEvent e);
    -sunt citite cele 2 text field-uri de input si se apeleaza metoda impartire(A, B); primul text field este setat cu rez1, iar al doilea text field este setat cu rez2

class DerivareListener implements ActionListener
    -contine metoda public void actionPerformed(ActionEvent e);

```


-sunt citite cele 2 text field-uri de input si se apeleaza metoda derivare(A); primul text field este setat cu rez1, iar apoi se apeleaza derivare(B); al doilea text field este setat cu rez1

class IntegrareListener implements ActionListener

-contine metoda public void actionPerformed(ActionEvent e);

-sunt citite cele 2 text field-uri de input si se apeleaza metoda integrare(A); primul text field este setat cu rez1, iar apoi se apeleaza integrare(B); al doilea text field este setat cu rez1

class ClearListener implements ActionListener

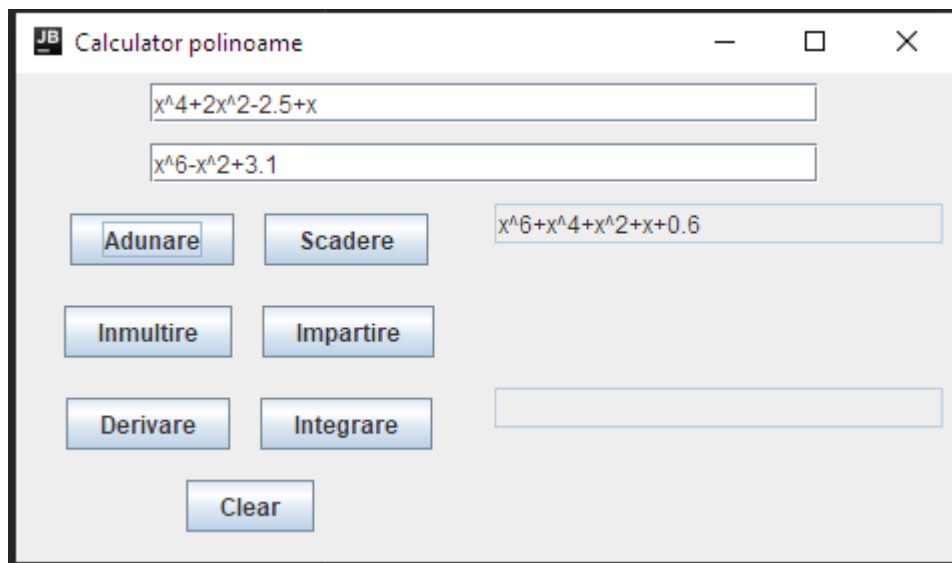
-contine metoda public void actionPerformed(ActionEvent e);

-se apeleaza functia reset() din PolinomView

4.3. Testare

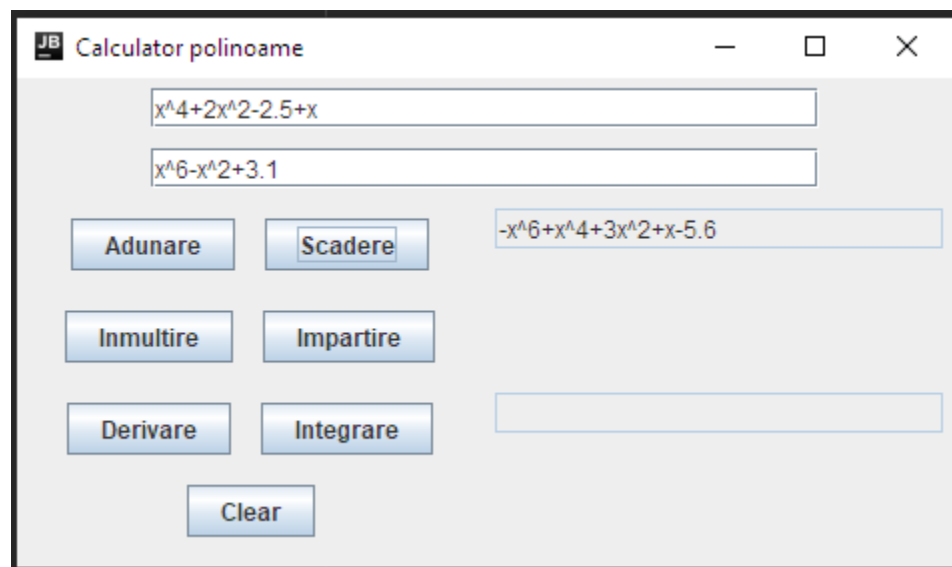
Adunarea

Se introduc cele doua polinoame si se apasa butonul “Adunare”. Rezultatul va fi afisat in primul text field de output.



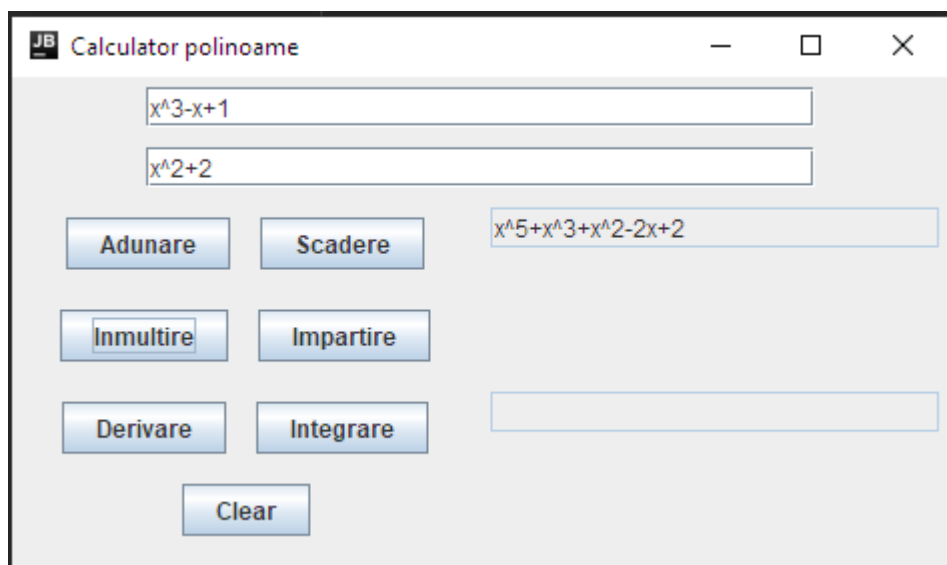
Scaderea

Se introduc cele doua polinoame si se apasa butonul “Scadere”. Rezultatul va fi afisat in primul text field de output.



Inmultirea

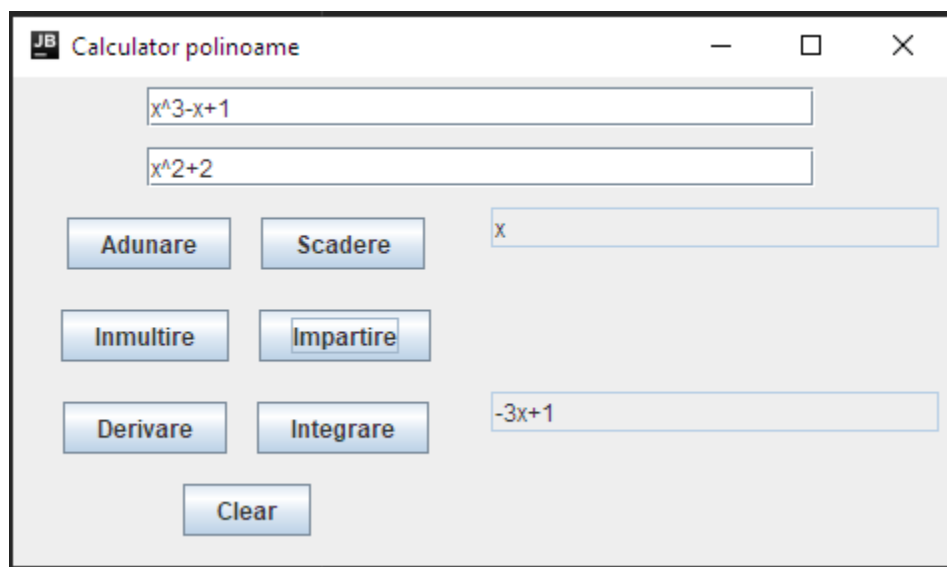
Se introduc cele doua polinoame si se apasa butonul “Inmultire”. Rezultatul va fi afisat in primul text field de output.



The screenshot shows a window titled "JB Calculator polinoame". It has two input fields at the top containing the polynomials $x^3 - x + 1$ and $x^2 + 2$. Below these are six buttons: "Adunare", "Scadere", "Inmultire", "Impartire", "Derivare", and "Integrare", along with a "Clear" button at the bottom. The "Inmultire" button is highlighted. To the right of the buttons, there are two output fields. The top output field contains the result of the multiplication: $x^5 + x^3 + x^2 - 2x + 2$. The bottom output field is empty.

Impartirea

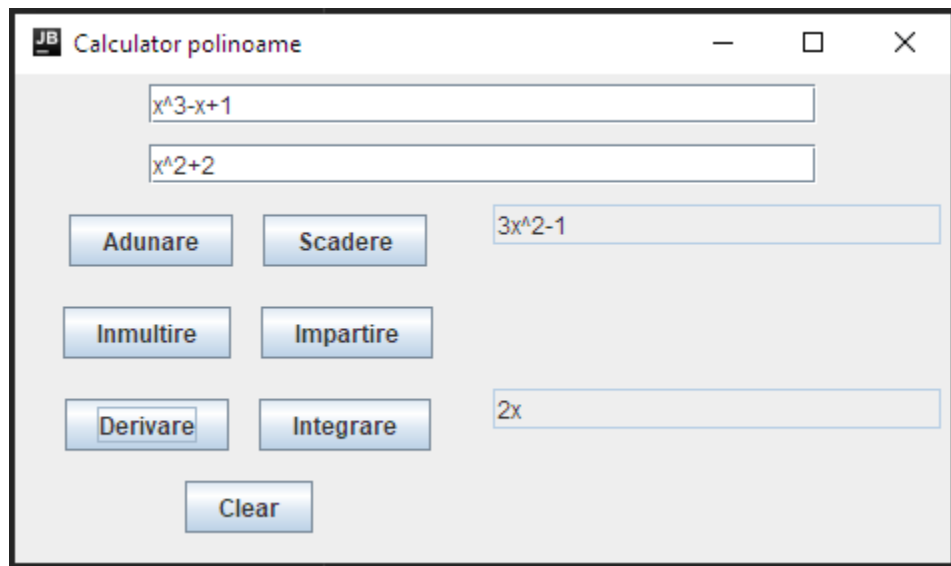
Se introduc cele doua polinoame si se apasa butonul “Impartire”. Catul va fi afisat in primul text field de output, iar restul va fi afisat in al doilea text field de output.



The screenshot shows the same "JB Calculator polinoame" window. The input fields still contain $x^3 - x + 1$ and $x^2 + 2$. The "Impartire" button is now highlighted. The output fields show the result of the division: the top field contains the quotient x and the bottom field contains the remainder $-3x + 1$.

Derivarea

Se introduc cele doua polinoame si se apasa butonul “Derivare”. Primul rezultat va fi afisat in primul text field de output, iar al doilea rezultat va fi afisat in al doilea text field de output.



The screenshot shows a window titled "JB Calculator polinoame". It has two input fields at the top containing the polynomials $x^3 - x + 1$ and $x^2 + 2$. Below these are six buttons: "Adunare", "Scadere", "Inmultire", "Impartire", "Derivare", and "Integrare". A "Clear" button is at the bottom. To the right of the buttons are two output fields. The first output field displays $3x^2 - 1$, which is the derivative of the first polynomial. The second output field displays $2x$, which is the derivative of the second polynomial. The "Derivare" button is highlighted with a blue border.

Integrarea

Se introduc cele doua polinoame si se apasa butonul “Integrare”. Primul rezultat va fi afisat in primul text field de output, iar al doilea rezultat va fi afisat in al doilea text field de output.



The screenshot shows the same "JB Calculator polinoame" window. The input fields still contain $x^3 - x + 1$ and $x^2 + 2$. The "Integrare" button is now highlighted with a blue border. The first output field displays $0.25x^4 - 0.5x^2 + x + C$, which is the integral of the first polynomial. The second output field displays $0.3333x^3 + 2x + C$, which is the integral of the second polynomial.

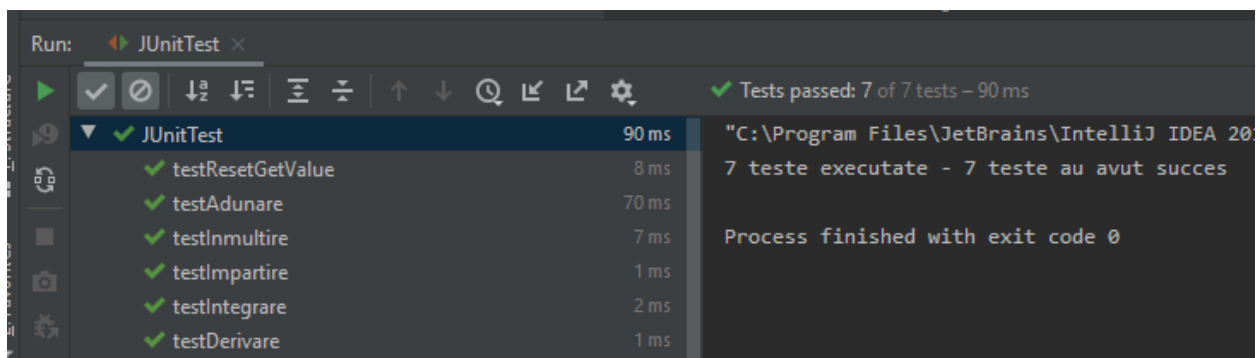
5. Rezultate

Pentru a testa rezultatele operatiilor cu niste date de intrare prestabilite, am creat o clasa de testare unitara, JUnitTest. Aceasta clasa are un PolinomModel si doi intregi de tip static, nrTesteExecutate si nrTesteCuSucces.

Metodele clasei sunt:

```
public static void setUpBeforeClass();  
    -se executa o singura data inaintea inceperii executiei testelor  
  
public static void tearDownAfterClass();  
    -se executa o singura data dupa terminarea executiei testelor  
  
public void setUp();  
    -inceperea unui nou test  
    -valoarea intregului nrTesteExecutate creste  
  
public void tearDown();  
    -se executa dupa fiecare test  
  
public void testResetGetValue();  
    -se testeaza metoda reset() din PolinomModel  
  
public void testAdunare();  
    -se testeaza metoda adunare(A,B) din PolinomModel  
  
public void testDiferenta();  
    -se testeaza metoda scadere(A,B) din PolinomModel  
  
public void testInmultire();  
    -se testeaza metoda inmultire(A,B) din PolinomModel  
  
public void testImpartire();  
    -se testeaza metoda impartire(A,B) din PolinomModel  
  
public void testDerivare();  
    -se testeaza metoda derivare(A), apoi derivare(B) din PolinomModel  
  
public void testIntegrare();  
    -se testeaza metoda integrare(A), apoi integrare(B) din PolinomModel
```

Dupa executia testelor:



6. Concluzii

Acest proiect este unul pentru uz general, se poate folosi oricand e nevoie de aplicarea unor operatii pe doua polinoame. Functioneaza fara probleme, au fost tratate diferite cazuri, astfel incat sa se obtina rezultatul dorit sau ca pe ecran sa fie afisata o fereastra de eroare in cazul introducerii unor date de intrare incorecte. Proiectul m-a ajutat sa dobandesc mai multe cunostiinte in programarea orientata pe obiect si in folosirea modelului arhitectural Model-View-Controller.

7. Bibliografie

1. <https://regexr.com/>
2. <https://google.github.io/styleguide/javaguide.html>
3. MVC, JUnit – laboratorul de POO din semestrul trecut
4. ArrayList, Collections documentation