

**CALCULATOR DE POLINOAME**

**DOCUMENTAȚIE**

**Facultatea:** Automatică și Calculatoare

**Specializarea:** Calculatoare și Tehnologia Informației

**Materia:** Tehnici de programare fundamentale

**Nume student:** Petricele Mihaela **Profesor coordonator**

**Grupa:** 30228 Dr. Pop Cristina Bianca

**Cuprins**

[**I.** **Obiectivul temei** 3](#_Toc66880315)

[**II.** **Analiza problemei, modelare, scenarii, cazuri de utilizare** 3](#_Toc66880316)

[***A.*** ***Diagrama Use-Case a aplicației*** 3](#_Toc66880317)

[***B.*** ***Descrierea cazurilor de utilizare:*** 4](#_Toc66880318)

[**III.** **Proiectare** 5](#_Toc66880319)

[*A)* *Proiectarea OOP a aplicației* 5](#_Toc66880320)

[*B)* *Diagrama UML* 5](#_Toc66880321)

[**IV.** **Implementare** 7](#_Toc66880322)

[**A)** **Pachetul Model** 7](#_Toc66880323)

[**B)** **Pachetul Logica** 8](#_Toc66880324)

[**C)** **Pachetul GUI** 10](#_Toc66880325)

[**V.** **Rezultate** 11](#_Toc66880326)

[**VI.** **Concluzii** 12](#_Toc66880327)

[**VII.** **Bibliografie** 13](#_Toc66880328)

# **Obiectivul temei**

Obiectivul principal al temei este de a implementa un Calculator care descrie operații cu polinoame. Obiectivele secundare pentru implementarea aplicației sunt:

* Crearea de clase, obiecte și metode pentru lucrul cu polinoame
* Descrierea operației de adunare a două polinoame
* Descrierea operației de scădere a două polinoame
* Descrierea operației de înmultire a două polinoame
* Descrierea operației de împărțire a două polinoame
* Descrierea operației de derivarea unui polinom
* Descrierea operației de integrarea unui polinom
* Afișarea rezultatului operațiilor
* Testarea aplicației

Realizarea operațiilor, crearea de clase, obiecte, metode și testarea se vor descrie în detaliu în capitolele de

Implementare, Proiectare și Testare.

# **Analiza problemei, modelare, scenarii, cazuri de utilizare**

Aplicația trebuie să permită utilizatorului să introducă polinoamele și să realizeze operații cu acestea prin

apăsarea anumitor butoane, specifice fiecărei operații.

## ***Diagrama Use-Case a aplicației***

Actor

Utilizator

Operații cu un Polinom

Operații cu două Polinoame

CALCULATOR DE POLINOAME

Sistem

Use-Case

## ***Descrierea cazurilor de utilizare:***

***Use Case***: Operații cu două Polinoame

***Actor***: Utilizator

***Scenariul de succes***:

1. Utilizatorul introduce corect de la tastatură cele două Polinoame
2. Programul verifică dacă Polinoamele introduse sunt corecte
3. Utilizatorul alege o operație apăsând pe butonul corespunzător (ex: Înmulțire)
4. Sistemul realizează corect operația de Înmulțire
5. Sistemul afișează rezultatul în field-ul corespunzător

***Scenariu alternativ:***

1. Utilizatorul nu a introdus Polinoamele conform formatului

* Sistemul atenționează utilizatorul printr-o casetă cu mesajul “Polinomul este invalid”
* Scenariul se intoarce din nou la pasul 1

1. Utilizatorul nu a introdus nici un Polinom in casetele corespunzătoare

* Calculatorul afișează același mesaj de eroare
* Scenariul revine la pasul 1

***Use Case***: Operații cu un Polinom

***Actor***: Utilizator

***Scenariul de succes***:

1. Utilizatorul introduce corect de la tastatură Polinomul în caseta corespunzătoare
2. Programul verifică dacă Polinomul introdus este corect
3. Utilizatorul alege o operație apăsând pe butonul corespunzător (ex: Derivare)
4. Sistemul realizează corect operația de Derivare
5. Sistemul afișează rezultatul în field-ul corespunzător

***Scenariu alternativ:***

1. Utilizatorul nu a introdus Polinoamele conform formatului

* Sistemul atenționează utilizatorul printr-o casetă cu mesajul “Polinomul este invalid”
* Scenariul se intoarce din nou la pasul 1

1. Utilizatorul nu a introdus nici un Polinom in casetele corespunzătoare

* Calculatorul afișează același mesaj de eroare
* Scenariul revine la pasul 1

. . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . .. . . . . . . .. . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . .

. .. . …….. . . .

# **Proiectare**

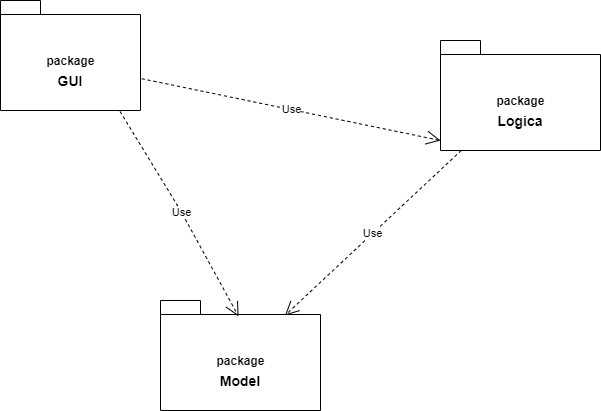
## *Proiectarea OOP a aplicației*

Aplicația este împărțită în trei pachete: GUI, Logica,Model. Pachetul GUI conține clasele: View și

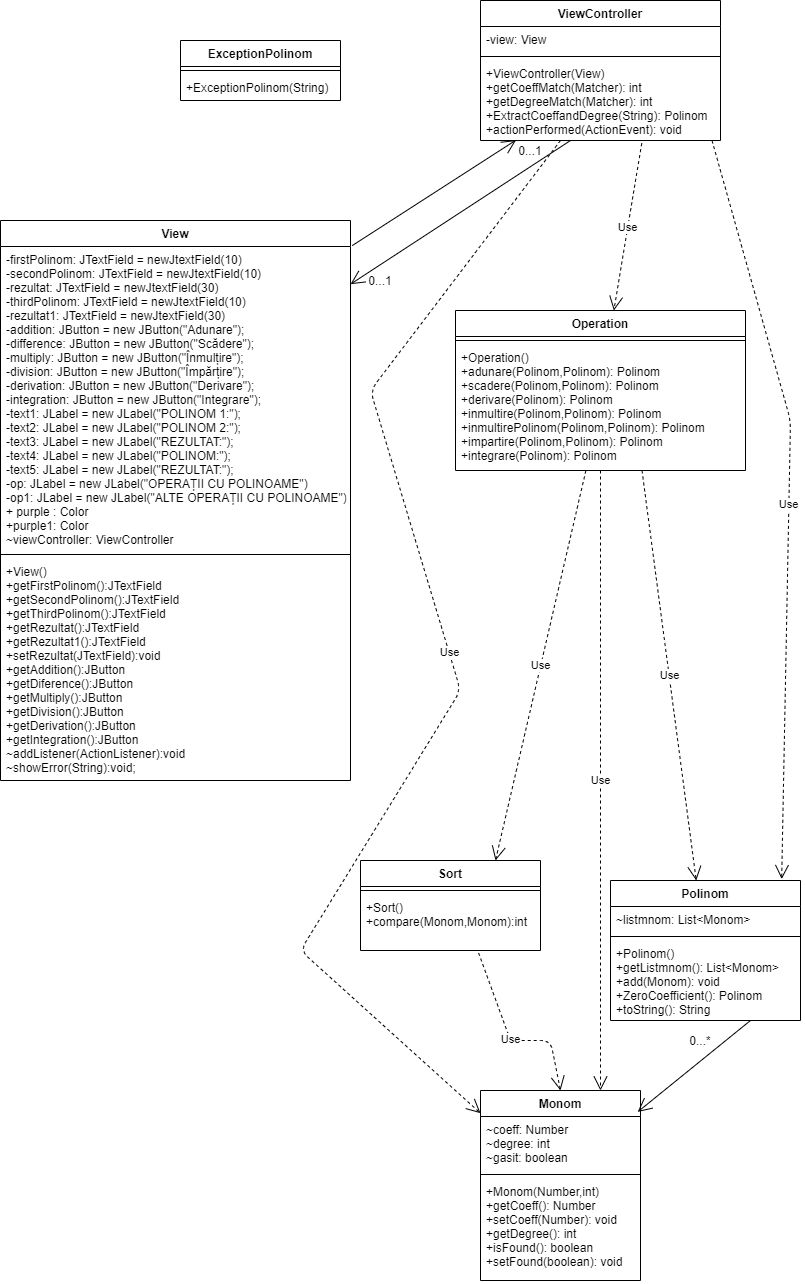
ViewController și ExceptionPolinom cu metodele aferente lor, în care se descrie codul pentru interfața cu utilizatorul. În pachetul Logica se află clasa Operation și clasa Sort cu metodele aferente în care am implementat operațiile cu polinoame și sortarea monoamelor descrescător dupa grad. Totodata în pachetul Model se află clasele Monom și Polinom pentru lucrul cu polinoame.

## *Diagrama UML*

* **Diagrama UML pentru pachete:**

****

**. . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . .. . .**

* **Diagrama UML pentru clase:**

# **Implementare**

. . . . . . . . . .. . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . .. . . . . . . .. . . . . .

În acest capitol se prezintă clasele și metode corespunzătoare din aplicația Calculator de Polinoame.

. . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . .. . . . . . . . . . . . .

## **Pachetul Model**

1. **Clasa Monom**

Această conține metodele pentru extragerea monomilor din polinoame. Clasa are doua atribute *coeff* de tipul Number care reprezinta coeficientul monoamelor și atributul *degree* de tipul int care reprezintă gradul fiecărui monom. Clasa mai conține și atributul *gasit* de tip boolean care determină dacă s-a gasit sau nu, un monom cu o anumita specificație. . . . . . . . . . . . . . . . . . . . .. . . . . . . . . .

Metodele din această clasă sunt reprezentate de getter și setter.

1. **Clasa Polinom**

Această clasă are ca attribute Lista de monoame care formează polinomul. Ca metode pe lângă gettere si settere, există metodele:

1. *public Polinom ZeroCoefficient()*

Această metodă parcurge fiecare monom din lista de monoame și șterge acele monoame care sunt 0.

Metoda este utilizată în clasa Operation în momentul în care se realizează împarțirea polinoamelor.

1. *public String toString()*

În cadrul acestei metode se realizează afișarea unui polinom. Codul metodei este următorul:

*public String toString() {*

*StringBuilder s = new StringBuilder();*

*for (Monom mon : listmnom) {*

*if (mon.getCoeff().intValue() == 0 && mon.getDegree() == 0 || (mon.getCoeff().intValue() == 0 && mon.getDegree() > 0) || (mon.getCoeff().intValue() == 0 && mon.getDegree() < 0)) {*

*s.append("");*

*}*

*if (mon.getCoeff().intValue() > 0 && mon.getDegree() == 1) {*

*s.append("+").append(mon.getCoeff()).append("x");*

*} else {*

*if (mon.getCoeff().intValue() < 0 && mon.getDegree() == 1) {*

*s.append(mon.getCoeff()).append("x");*

*}*

*}*

*if (mon.getCoeff().intValue() > 0 && mon.getDegree() == 0) {*

*s.append("+").append(mon.getCoeff());*

*} else {*

*if (mon.getCoeff().intValue() < 0 && mon.getDegree() == 0) {*

*s.append(mon.getCoeff());*

*}*

*}*

*if (mon.getCoeff().intValue() > 0 && mon.getDegree() > 0 && mon.getDegree() != 1) {*

*s.append("+").append(mon.getCoeff()).append("x^").append(mon.getDegree());*

*} else {*

*if (mon.getCoeff().intValue() < 0 && mon.getDegree() > 0 && mon.getDegree() != 1) {*

*s.append(mon.getCoeff()).append("x^").append(mon.getDegree());*

*}*

*}*

*}*

*return s.toString();*

*}*

*}*

## **Pachetul Logica**

1. **Clasa Operation**

În această clasă se implementează operațiile care se realizează cu polinoame. Câteva metode cu anumite

operații prezente în această clasă:

1. *public static Polinom inmultire(Polinom polinom1, Polinom polinom2)*

În cadrul acestei metode se realizează operația de înmultire a două Polinoame primite ca și parametru:

polinom1 și polinom2. Rezultatul reprezintă un alt polinom cu același format.

**. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . .. . . . . . . . . . . . .**

*public String toString() {*

*StringBuilder s = new StringBuilder();*

*for (Monom mon : listmnom) {*

*if (mon.getCoeff().doubleValue() == 0 && mon.getDegree() == 0 || (mon.getCoeff().doubleValue() == 0 && mon.getDegree() > 0) || (mon.getCoeff().doubleValue() == 0 && mon.getDegree() < 0)) {*

*s.append("");*

*}*

*if (mon.getCoeff().doubleValue() > 0 && mon.getDegree() == 1) {*

*s.append("+").append(mon.getCoeff()).append("x");*

*} else {*

*if (mon.getCoeff().doubleValue() < 0 && mon.getDegree() == 1) {*

*s.append(mon.getCoeff()).append("x");*

*}*

*}*

*if (mon.getCoeff().doubleValue() > 0 && mon.getDegree() == 0) {*

*s.append("+").append(mon.getCoeff());*

*} else {*

*if (mon.getCoeff().doubleValue() < 0 && mon.getDegree() == 0) {*

*s.append(mon.getCoeff());*

*}*

*}*

*if (mon.getCoeff().doubleValue() > 0 && mon.getDegree() > 0 && mon.getDegree() != 1) {*

*s.append("+").append(mon.getCoeff()).append("x^").append(mon.getDegree());*

*} else {*

*if (mon.getCoeff().doubleValue() < 0 && mon.getDegree() > 0 && mon.getDegree() != 1) {*

*s.append(mon.getCoeff()).append("x^").append(mon.getDegree());*

*}*

*}*

*}*

*return s.toString();*

*}*

*}*

1. *public static Polinom integrare (Polinom polinom)*

În această metodă se realizează operația de integrare a unui polinom.

Rezultatul este sortat înainte să fie afișat în caz că polinomul a fost introdus cu gradele monoamelor în ordine aleatorie.

*public static Polinom integrare(Polinom polinom) {*

*List<Monom> listaPol;*

*listaPol = polinom.getListmnom();*

*Monom monres;*

*Polinom result = new Polinom();*

*for (Monom mon : listaPol) {*

*monres = new Monom(mon.getCoeff().doubleValue() / (mon.getDegree() + 1), mon.getDegree() + 1);*

*result.add(monres);*

*}*

*Collections.sort(result.getListmnom(), new Sort()); //Sortez monoamele in ordinea descrescatoare a gradelor*

*return result;*

*}*

*}*

1. *public static Polinom impartire(Polinom polinom1, Polinom polinom2)*

Această metodă realizează împărțirea dintre cele două polinoame primite

ca și parametru. Rezultatul returnat este reprezentat de câtul împărțiri și restul acesteia. La împărțire am folosit următorul algoritm:

* Am ordonat polinoamele descrescător în funcție de gradul acestora cu ajutorul unei clase noi și cu metoda sort din clasa Collections.
* Cât timp gradul primului polinom este mai mare decat gradul celui de-al doilea polinom împart primul monom al primului Polinom, la primul monom al celui de-al doilea Polinom pentru a obtine primul termen al câtului.
* Am înmulțit câtul cu cel de-al doilea Polinom, apoi am scazut din primul Polinom rezultatul pentru a afla restul
* În caz că restul ajunge la 0 se iese din buclă

1. **Clasa Sort**

Această clasă am utilizat-o pentru a sorta lista de monoame dintr-un

Polinom. Clasa implementează interfața Comparator cu metoda aferentă acesteia pentru a realiza operația de ordonare a polinoamelor descrescător după gradele. Această clasă am folosit-o la afișarea rezultatului operațiilor.

## **Pachetul GUI**

1. **Clasa View**

În cadrul acestei clase am implementat interfața cu utlizatorul a aplicației.

Aceasta este formată din două zone:

1. Operații cu polinoame: adunarea, scăderea, înmulțirea, împărțirea

La această parte utilizatorul poate să introducă în cele două zone de text cele două polinoame și să aleagă dintr-un panel cu butoane operația pe care dorește să o realizeze.

1. Alte operații cu polinoame: derivarea și integrarea

În această zonă utilizatorul poate să introducă într-o zonă de text un polinom și să aleagă dintr-un panel cu butoane operația pe care dorește să o realizeze.

Rezultatul fiecărei operații se afișeză în zona de text corespunzătoare.

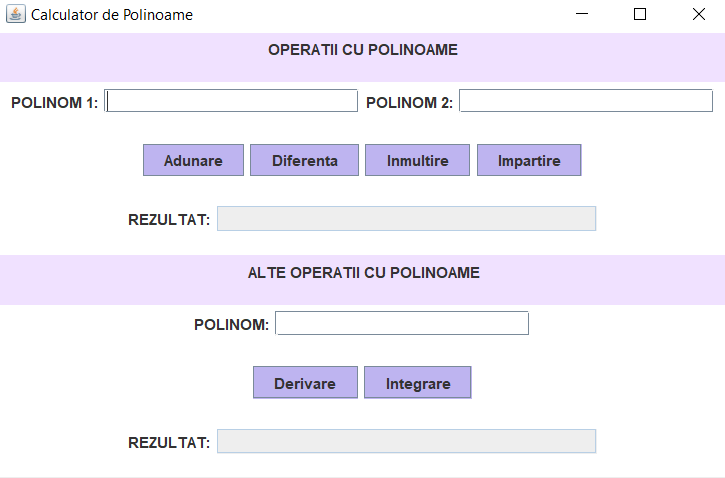


Figura 4.1 Interfața cu utilizatorul

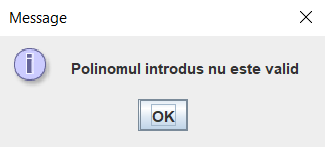
În cazul în care utilizatorul introduce un polinom cu un format greșit sau nu introduce nici un polinom aplicația aruncă o excepție sub forma unei casete cu un mesaj.

Figura 4.2 Mesajul de eroare afisat

1. **Clasa ViewController**

În această clasa am implementat metoda *public Polinom ExtractCoefandDegree(String polinom)*

*throws ExceptionPolinom* pentru convertirea polinoamelor din String și crearea listei de monoame pentru fiecare Polinom. Am folosit un pattern pentru a extrage coeficienții și gradele polinoamelor.

*public Polinom ExtractCoefandDegree(String polinom) throws ExceptionPolinom {*

*Polinom resultPolinom = new Polinom();*

*Pattern pattern = Pattern.compile("((-?\\d+(?=x))?(-?[x])(\\^(-?\\d+))?)|((-?)[x])|(-?\\d+)");*

*Matcher match = pattern.matcher(polinom);*

*List<Integer> coeffpol = new ArrayList<>();*

*List<Integer> degreepol = new ArrayList<>();*

*int i = 0;*

*if (!match.find() || (!Pattern.matches("[+\\-x^\\d]+", polinom))) {// daca nu s-a gasit nici un match*

*throw new ExceptionPolinom("Polinomul introdus nu este valid");*

*}*

*match = pattern.matcher(polinom);*

*while (match.find()) {*

*coeffpol.add(getCoeffMatch(match));*

*degreepol.add(getDegreeMatch(match));*

*}*

*while (i < coeffpol.size()) {*

*Monom monin = new Monom(coeffpol.get(i), degreepol.get(i));*

*resultPolinom.add(monin);*

*i++;*

*}*

*System.out.println(resultPolinom);*

*return resultPolinom;*

*}*

*. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . .. . . . . . . . . .. . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . .. . . . .. . .. .*

# **Rezultate**

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . .. . . . . . . . . . . .

Pentru testarea aplicației am folosit Junit. Am creat patru polinoame pentru a le folosi în scenariile de test. Clasele pentru care am create scenarii de test sunt următoarele:

* Clasa Operation cu Clasa de testare OperationTest
* Clasa Polinom cu Clasa de testare PolinomTest

1. **Clasa OperationTest**

Scenariile de test pentru această clasa sunt:

* Test pentru operația de adunare a două polinoame
* Test pentru operația de scădere a două polinoame
* Test pentru operația de înmulțire
* Test pentru operația de împărțire
* Test pentru operația de derivare
* Test pentru operația de integrare

Aplicația a trecut fiecare test descris.

1. **Clasa PolinomTest**

Scenariile de test pentru această clasa sunt:

* Test pentru cazul în care un monom este 0.

Aplicația a trecut scenariul de test.

# **Concluzii**

În concluzie această temă m-a ajutat să îmi amintesc aspectele legate de limbajul Java. Totodata mi-a aprofundat și cunoștințele despre acest limbaj. Am învățat cum să structurez o aplicație în Java pe pachete, eu înainte scriind toate clasele într-un singur pachet, acest mod fiind mult mai ușor de înțeles. Am învățat să nu mai scriu coduri de sute de linii la metode sau clase, acest lucru face codul mai lizibil. Mai am mult de lucru pentru a stăpâni acest limbaj, această temă fiind un bun început în drumul meu de a fi mai bună la a coda în acest limbaj.

Ca dezvoltări ulterioare aș propune ca polinomul afișat în rezultat, în caz că este pozitiv să nu se afișeze semnul. O altă dezvoltare ar fi posibilitatea ca utilizatorul să introducă polinoame și cu coeficienți reali nu doar reali. Se pot realiza multe îmbunătățiri și se poate extinde la o aplicație mult mai complexă.

# **Bibliografie**

* <https://regex101.com/>
* <https://ro.siwib.org/960791-collections-sort-with-multiple-fields-QLDMYS>
* Stack Overflow
* Youtube
* <https://docs.oracle.com/javase/tutorial/essential/regex/quant.html>
* <https://www.baeldung.com/parameterized-tests-junit-5>