***TEMA 1***

***CALCULATOR PENTRU POLINOAME***

***Materie: Tehinici de Programare***

***Elev: Pavel Mădălina- Denisa***

***Grupa: 30223***

***Specializare: CTI RO, an 2***

*Cuprins*

[*Obiectivul temei* 2](#_Toc131095220)

[*Analiza problemei, modelare, scenarii, cazuri de utilizare* 3](#_Toc131095221)

[*Proiectare* 4](#_Toc131095222)

[*Implementare* 5](#_Toc131095223)

[*Rezultate* 6](#_Toc131095224)

[Concluzii 9](#_Toc131095225)

[Bibliografie 10](#_Toc131095226)

# *Obiectivul temei*

Principalul obiectiv al temei este implementarea unui calculator de polinoame, functional.

Pentru a putea indeplini acest obiectiv am avut de efectuat următoarele sarcini:

* Crearea unei clase Polynomial, care salveaza polinomul sub forma unui Map de monoame;
* Folosirea expresiilor regulate pentru a converti polinomul dintr-un String într-un obiect de tip polinom;
* Implementarea operatiei de adunare;
* Implementarea operatiei de scadere;
* Implementarea operatiei de inmultire;
* Implemenatarea operatiei de impartire;
* Implementarea operatiei de derivare;
* Implementarea operatiei de integrare;
* Crearea unei interfete grafice, user-friendly.

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

Calculatorul este capabil să implementeze operațiile de adunare, scădere, înmulțire, împărțire, derivare și integrare asupra polinoamelor. Acestea sunt reprezentate printr-o listă de monoame, fiecare monom având un coeficient și un grad. Astfel, forma generală a unui monom va fi A\*X^B .

Interfața grafică a calculatorului este una foarte ușor de folosit, butoanele având denumiri suggestive(Add, Substract, Multiplicate, Derivate, Integrate, Divide). Polinoamele vor fi scrise în cele două câmpuri de text din partea superioară a paginii iar rezultatul va fi afișat în al treilea.

În cazul în care utilizatorul dorește să șteargă datele introduse, poate apăsa pe butonul CLEAR.

Operația dorită se realizează prin simpla apăsare a butonului corespunzător ei (butoanele au denumiri sugestive).

Pentru modelarea aplicației, am ales interpretare polinoamelor ca fiind o listă de monoame, mai exact un Map de tipul <cheie, valoare>, pentru a putea obține mult mai ușor informațiile despre polinom. Puterea fiecărui monom, fiind unică, va fi defapt cheia, iar valoarea este coeficientul monomului.

Exemplu de use-case:

1. Utilizatorul introduce primul polinom

2. Utilizatorul introduce cel de-al doilea polinom

3. Utilizatorul apasă butonul specific operației dorite

4. Utilizatorul obține rezultatul operației.

5. Pentru a introduce alte date, poate apăsa butonul CLEAR.

Pentru ca polinomul introdus să fie valid, el trebuie să respecte următorul tipar:

**+/-Coeficient x^putere+/-coeficientx^putere+/-…**

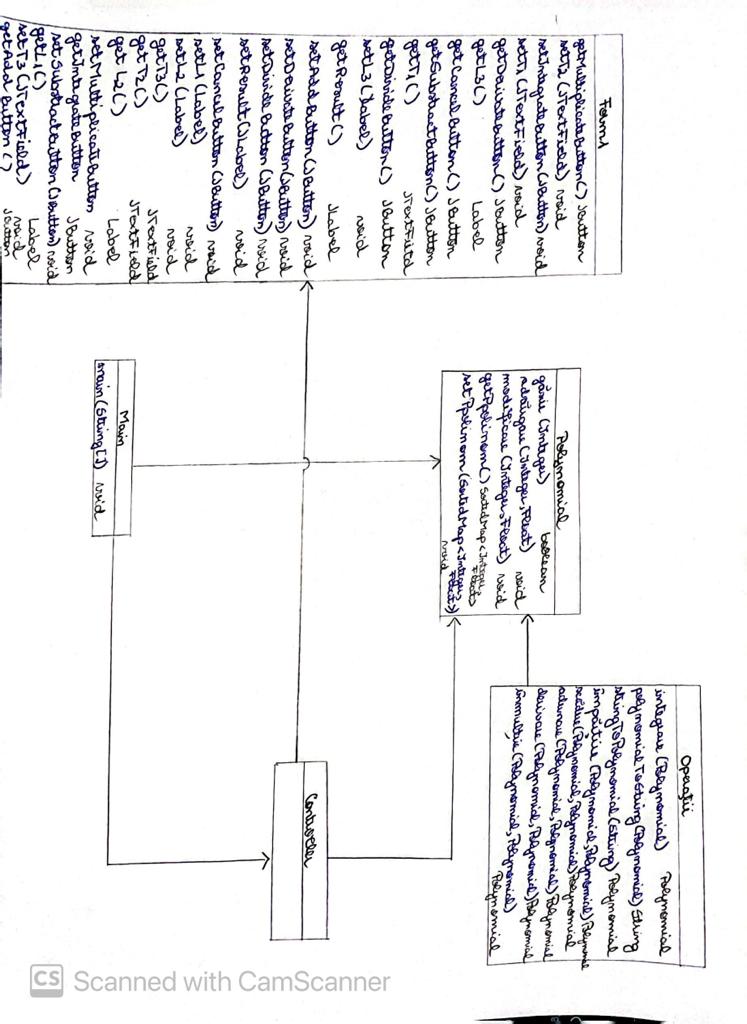
# *Proiectare*

Pentru proiectarea acestei aplicații am folosit design-pattern-ul Model-View-Controller, structurându-mi astfel proiectul în 3 pachete:

* *Model:* acest pachet conține clasele **Polynomial** și **Operații**, care constituie practic baza proiectului, unde sunt implementate toate metodele care realizează operații pe polinoame;
* *View*: în acest pachet apar doar elementele necasare construirii interfeței grafice, crearea frame-ului principal și poziționarea fiecarui element în interfața utilizatorului;
* *Controller*(aici: org.example): clasele din acest pachet conectează interfața grafică(GUI-ul) de operațiile în sine pe polinoame, pentru a putea avea efectul dorit:Apăsarea unui buton și obținerea unui rezultat.

Aceste clase sunt însoțite și de o clasă de test, unde am testat corectitudinea metodelor din clasa **Operații** folosind testarea unitară(Junit), utilă în detectarea sursei eventualelor erori.

Diagrama UML



# *Implementare*

***Clasa Polynomial***

In aceasta clasa am salvat practiv informatii despre un polinom ca si o lista de monoame. De asemenea, clasa mai contine 2 metode destul de importante:

* ***public boolean gasire(Integer p)***
* **public void modificare(Integer p,Float nVal)**

Aceste metode sunt folosite la operatia de inmultire, pentru cazul in care mai multe monoame vor avea aceeasi putere dupa efectuarea operatiei.

***Clasa Operatii***

* ***public static Polynomial adunare(Polynomial polinom1, Polynomial polinom2)***
* ***public static Polynomial scadere(Polynomial polinom1, Polynomial polinom2)***
* ***public static Polynomial derivare(Polynomial polinom1)***
* ***public static Polynomial integrare(Polynomial polinom1)***
* ***public static Polynomial inmultire(Polynomial polinom1,Polynomial polinom2)***

Aceste metode au denumiri suggestive pentru efectul lor, fiecare dintre ele fiind responsabila de o operatie pe polinoame.

* ***public static String polynomialToString(Polynomial p)***

Aceasta metoda realizeaza conversia dintre un obiect de tip polinom, intr-un String, necasara transcrierii polinomului rezultat in urma operatiilor in caseta de text corespunzatoare.

* ***public static Polynomial stringToPolynomial(String s)***

Aceasta metoda este una extrem de importanta in proiect, fiind responsabila de conversia din String intr-un obiect de tip polinom a datelor introduse in casetele de text corespunzatoare polinoamelor pe care se fac operatiile. Prin utilizarea expresiilor regulate am construit un tipar care sa ma ajute sa identific un monom correct, pe care sa il pot prelua din caseta de text si transforma intr-un polinom pe care sa pot efectua operatii.

***Clasa Controller***

In aceasta clasa nu exista nicio metoda, toate operatiile fiind introduse in constructorul clasei. Aici am implementat cateva metode de tip Action Listener, care sa declanseze efectuarea unei anumite operatiuni la apasarea butonului corespunzator.

# *Rezultate*

Pentru testarea metodelor corespunzatoare fiecarei operatii am folosit Junit.

Am incercat sa acopar majoritatea cazurilor posibile, printer care si: scaderea unui numar negative, adunarea cu opusul, inmultirea a 2 numere intregi, adunarea a doua polinoame de grade diferite, derivarea unui intreg, integrarea unui intreg, etc.

Cu ajutorul testului parametrizat, am trimis spre testare cate 3 perechi de String-uri diferite pentru verificarea corectitudinii operatiilor pe polinoame.

Acestea sunt prezentate mai jos:

* adunarea a 2 polinoame:



* scaderea a 2 polinoame:



* inmultirea a 2 polinoame:



* derivarea a doua polinoame:



* integrarea a 2 polinoame:



# Concluzii

În concluzie, implementarea unui calculator de polinoame în Java mi-a reamintit multe concepte poate uitate de la OOP, dar am utilizat si unele noi, cum ar fi testarea cu Junit si utilizarea expresiilor regulate .

Un astfel de proiect poate fi extins cu funcționalități suplimentare, cum ar fi afisarea graficului acestora sau posibilitatea de a alege gradul polinomului pe care vrem sa il furnizam. De asemenea, ar fi posibila conectarea cu o baza de date pentru a retine operatiile effectuate anterior.

# Bibliografie

<https://dsrl.eu/courses/pt/>

<https://www.baeldung.com/parameterized-tests-junit-5>

<https://www.asciitable.com/>

<https://stackoverflow.com/questions/32844417/java-parsing-a-polynomial-with-complex-coefficients-with-regex>

<https://www.jetbrains.com/help/idea/delegate-build-and-run-actions-to-maven.html>