DOCUMENTATIE

TEMA *NUMARUL\_1*

NUME STUDENT: Poponet Tiberiu Sergiu

GRUPA: 30221

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)

[3. Proiectare 5](#_Toc95297887)

[4. Implementare 8](#_Toc95297888)

[5. Rezultate 9](#_Toc95297889)

[6. Concluzii 9](#_Toc95297890)

[7. Bibliografie 10](#_Toc95297891)

# Obiectivul temei

Obiectiv principal:

Proiectarea și implementarea unui calculator polinomial cu o interfață grafică dedicată prin care

utilizatorul poate insera polinoame, poate selecta operația matematică care trebuie efectuată și poate vizualiza rezultatul.

Subobiective

• Analiza problemei și identificarea cerințelor

• Proiectarea calculatorului polinomial

• Implementarea calculatorul polinomial

• Testarea calculatorului polinomial

# Analiza problemei, modelare, scenarii, cazuri de utilizare

Calculatorul polinomial permite userului sa adauge un polinom/doua polinoame.

Calculatorul polinomial permite userului sa aleaga operatia dorita. Cazuri de utilizare:

1. Cu un polinom: derivare, integrare.
2. Cu doua polinoame: adunare, scadere, inmultire, impartire.

Exemplu de utilizare (scenariu 1):

* Userul insereaza 2 polinoame in interfata grafica
* Userul alege operatia dorita, de exemplu adunare, apasand butonul specific operatiei alese
* Calculatorul polinomial afiseaza rezultatul (respectiv suma, in cazul adunarii)
* Selectand alt buton, calculatorul polinomial afiseaza urmatoarea operatie aleasa

Exemplu de utilizare (scenariu 2):

* Userul insereaza un singur polinom
* Userul alege operatia Dorita dintre derivare si integrare, apasand butonul specific operatiei alese
* Calculatorul polinomial afiseaza rezultatul derivarii/integrarii
* In cazul inserarii a doua polinoame in locul unuia singur, rezultatul va fi afisat pentru polinomul introdus in textbox-ul de sus (polinom 1)

Atentie!

Polinoamele introduse trebuie sa fie sub forma “3x^4+2x^2-1x^1-5x^0”.

Adunare

Scadere

Inmultire

Impartire

Derivare

Integrare

User

# Proiectare

X^3+2x^2

Calculator

Polinomial

**Rezultat**

X^4-2x^0

Operatie

* formarePolinom:

Codul definit în metoda formarePolinom primește un String input care reprezintă un polinom sub forma unei expresii algebrice. Metoda utilizează un obiect Map<Integer, Integer> pentru a stoca gradele și coeficienții polinomului și apoi returnează un obiect de tip Polinom care reprezintă polinomul dat.

Mai întâi, metoda inițializează un obiect de tip Map pentru a stoca coeficienții și gradele polinomului. Această mapă va fi utilizată pentru a construi obiectul Polinom la sfârșit.

În continuare, metoda utilizează expresii regulate pentru a identifica coeficienții și gradele fiecărui termen din polinom. Expresia regulată definită în variabila pattern este următoarea:

([-+]?\\d\*)?x(\\^(\\d+))?")

În continuare, metoda utilizează un obiect Matcher pentru a potrivi expresia regulată cu șirul de caractere input. În cadrul unui ciclu while, metoda parcurge toti termenii identificați de expresia regulată și extrage coeficientul și exponentul fiecărui termen utilizând metoda group a obiectului Matcher.

Dacă coeficientul lipsește sau este +, acesta este considerat a fi 1. Dacă coeficientul este -, acesta este considerat a fi -1. Altfel, coeficientul este convertit într-un întreg utilizând metoda parseInt. Se recomanda folosirea explicita a coeficientului, chiar daca este 1, atat si a exponentului 0 pentru termenii liberi.

Exponentul este convertit într-un întreg utilizând metoda parseInt. Dacă exponentul lipsește, acesta este considerat a fi 1. Totusi, se recomanda explicitarea sa „1x^1”.

În final, metoda adaugă coeficientul și exponentul la obiectul Map pentru a construi polinomul. Dacă un termen cu același exponent a fost identificat anterior, coeficientul este adunat la coeficientul existent în mapă. În cele din urmă, metoda construiește și returnează un obiect Polinom pe baza mapei de coeficienți și grade.

* adunare:

Codul definit în metoda adunare primește două șiruri de caractere poly1 și poly2, care reprezintă două polinoame sub forma unei expresii algebrice (String-uri). Metoda folosește metoda formarePolinom pentru a converti cele două șiruri de caractere în două obiecte de tip Polinom. Apoi, metoda adună cele două polinoame și returnează un obiect de tip Polinom care reprezintă suma.

Metoda parcurge fiecare termen din primul polinom și adaugă coeficientul la mapă dacă același grad există și în al doilea polinom. Pentru fiecare grad care există în primul polinom, coeficientul corespunzător din al doilea polinom este obținut prin apelul metodei getOrDefault a obiectului Map din al doilea polinom. Coeficienții sunt adunați și dacă suma este diferită de 0, coeficientul este adăugat la mapă cu gradul corespunzător.

După ce sunt adăugati toti termenii din primul polinom, metoda parcurge termenii din al doilea polinom și adaugă doar termenii care nu există deja în primul polinom.

În final, metoda construiește și returnează un obiect Polinom pe baza mapei de coeficienți și grade, care reprezintă suma polinoamelor.

* scadere:

La fel ca la adunare, cu exceptia ca se scade valoarea coeficientului termenului al doilea din primul. După ce sunt adăugati toți termenii din primul polinom, metoda parcurge termenii din al doilea polinom și adaugă doar termenii care nu există deja în primul polinom, cu coeficientul negativ.

* multiplication:

Metoda parcurge fiecare exponent din primul polinom folosind un for loop, și pentru fiecare exponent, parcurge toți ceilalți exponent din al doilea polinom folosind un alt for loop. Pentru fiecare combinație de exponente, metoda înmulțește coeficienții corespunzători din cele două polinoame și stochează rezultatul în variabila "coeffMul". De asemenea, se calculează exponentul rezultat prin adunarea celor două exponente și se stochează în variabila "expMul".

În cele din urmă, se verifică dacă "mul" conține deja un termen cu exponentul "expMul". Dacă da, se adaugă noul coeficient la coeficientul existent cu acel exponent. Dacă nu, se adaugă un nou element în map cu exponentul și coeficientul rezultat.

* derivare:

Pentru fiecare exponent din polinomul de intrare se verifică dacă exponentul este diferit de zero. Dacă da, se extrage coeficientul corespunzător acelui exponent și se calculează noul exponent prin scăderea exponentului curent cu 1, iar noul coeficient este calculat prin înmulțirea coeficientului curent cu exponentul. Noul exponent și noul coeficient sunt apoi stocați în obiectul derivative prin utilizarea metodei put().

* integrare:

Integrala unui termen al polinomului se calculează astfel: se ia coeficientul termenului și exponentul acestuia și se adaugă 1 la exponent. Se împarte coeficientul la exponentul nou obținut și se adaugă termenul integral format în șirul de caractere final. Dacă coeficientul este negativ, se afișează minus înainte de acesta. Pentru fiecare termen se afișează și simbolul "x" și se afișează exponentul corespunzător. Metoda returneaza un String, deoarece polinomul opereaza cu Map-uri de tip Integer, Integer.

* impartire

Functia "grad" calculeaza gradul polinomului curent, adica valoarea maxima a exponentilor termenilor din polinom. Daca polinomul este gol, functia returneaza 0.

Functia "coeficientMaxim" calculeaza coeficientul termenului de gradul maxim din polinomul curent. Daca polinomul este gol, functia returneaza 0.0.

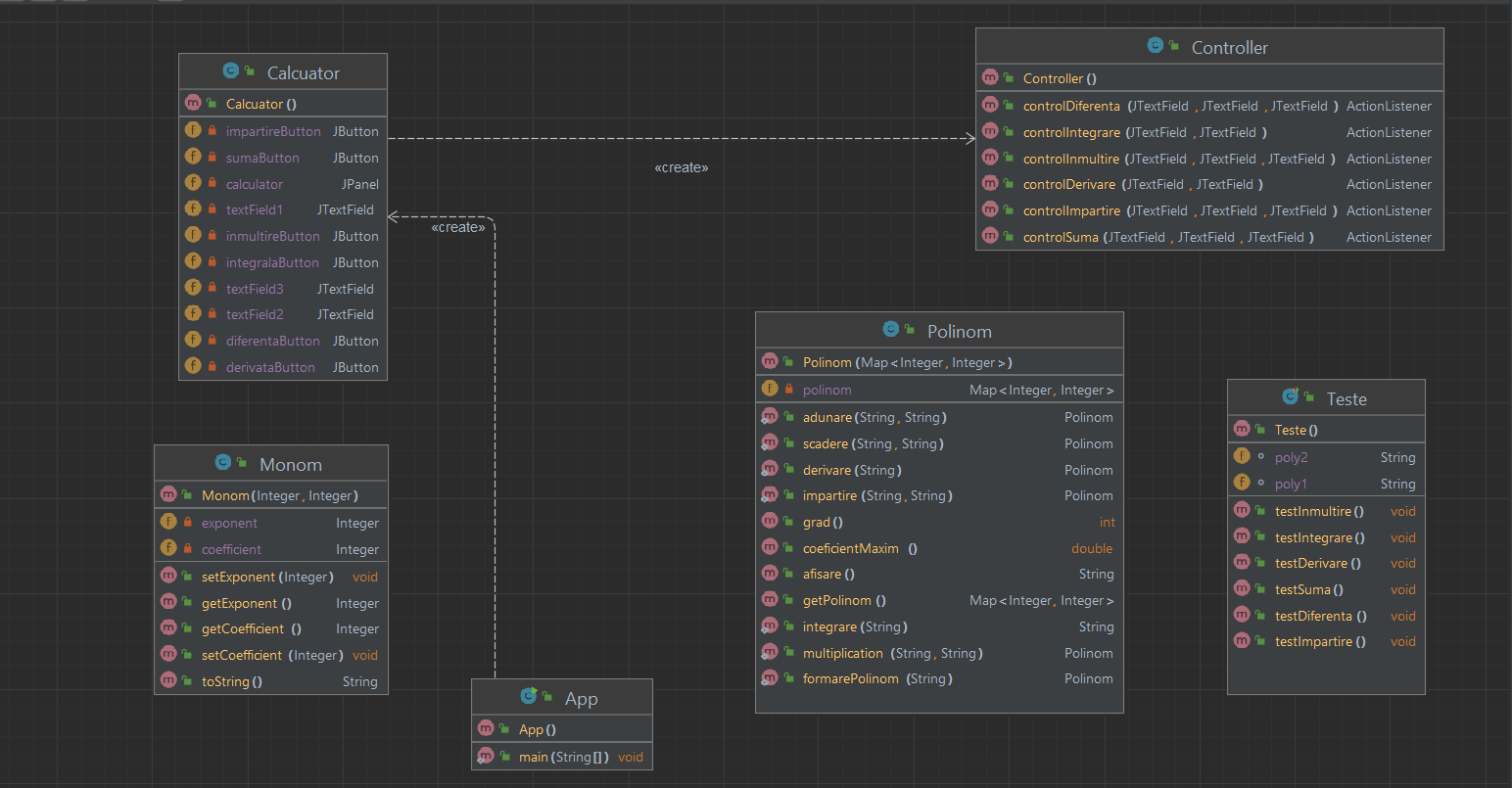
Functia "impartire" primeste doua polinoame sub forma de siruri de caractere si le transforma in obiecte Polinom. Apoi, se initializeaza un polinom cat cu sirul vid.

Intr-un while, cat timp gradul primului polinom este mai mare sau egal cu gradul celui de-al doilea polinom, se efectueaza urmatoarele operatii:

1. Se calculeaza coeficientul primului termen al catului prin impartirea coeficientilor termenilor de gradul maxim din fiecare polinom.
2. Se calculeaza exponentul termenului curent din cat, prin scaderea gradului celui de-al doilea polinom din gradul primului polinom.
3. Se initializeaza un polinom monomCat cu sirul vid si se adauga termenul curent la el.
4. Se adauga monomul curent la polinomul cat.
5. Se calculeaza produsul dintre polinomul de impartit si monomul cat.
6. Se calculeaza diferenta dintre polinomul de impartit si produsul dintre polinomul cat si cel de-al doilea polinom.
7. Daca gradul diferentei este mai mare sau egal cu gradul polinomului de impartit initial, se iese din while.
8. In caz contrar, se actualizeaza polinomul de impartit cu noua diferenta calculata.

La final, functia returneaza polinomul cat.

Functia nu functioneaza perfect, deoarece exista posibilitatea de a da un numar flotant, iar eu lucrez doar cu Integeri.

Diagrama UML:

*Se va prezenta proiectarea OOP a aplicatiei, diagramele UML de clase si de pachete, structurile de date folosite, interfetele definite si algoritmii folositi (*daca e cazul)

# Implementare

Pentru aplicatia mea, am creat 4 pachete: Controller, Interfata, org.example si Teste.

In org.exaple, avem:

1. O clasa Monom, cu 2 elemente de tip Integer: exponent si coefficient, de asemena functii de set si get pentru ambele elemente. Clasa Monom nu este folosita pe parcursul proiectului, idea principala, anume impartirea polinoamelor in obiecte de tip Monom a fost inlocuita cu cea a crearii polinoamelor sub forma de Hash-Map-uri.
2. Clasa Polinom, care foloseste Map-uri de tip <Integer, Integer>. Avem o functie getPolinom, cae returneaza un Map<Integer, Integer> si o metoda formarePolinom, care formeaza polinomul sub forma de HashMap-uri, dupa ce primeste un String ca input. De asemenea, in clasa Polinom am implementat metodele specifice operatiilor pe polinoame: adunare, scadere, multiplication(inmultire), derivare, integrare si impartire (pentru care mai avem nevoie de 2 metode auxiliare: grad si coeficientMaxim). Toate metodele sunt de tipul public static, inafara celor auxiliare.
3. Clasa App, care este main-ul aplicatiei, in care apelam interfata.

In Interfata, avem implementarea cu Java Swing a interfetei. Avem 3 text field-uri pentru cele 2 polinoame date ca input, respectiv al treilea, care va fi output-ul si 6 butoane pentru fiecare operatie: sumaButton, diferentaButton, inmultireButton, impartireButton, derivataButton, integralaButton.

In Controller, conectam interfata la cod, prin 6 ActionListeneri pentru fiecare operatie.

In Teste, avem teste pentru fiecare operatie, cu Junit.

# Rezultate

Avem 2 polinoame: poly1="1x^3+2x^2+3x^1+5x^0", poly2="4x^2+6x^1".

Metoda testSuma are rezultatul dorit, adica "+5x^0+9x^1+6x^2+1x^3".

Metoda testDiferenta are rezultatul dorit, adica "+5x^0-3x^1-2x^2+1x^3".

Metoda testInmultire are rezultatul dorit, adica "+30x^1+38x^2+24x^3+14x^4+4x^5”.

Metoda testDerivare are rezultatul dorit dupa derivarea poly1, adica „3x^0+4x^1+3x^2".

Metoda testIntegrare are rezultatul dorit dupa integrarea poly1, adica ”5/1x^1+3/2x^2+2/3x^3+1/4x^4".

Pentru metoda testImpartire, pentru o mai usoara urmarire, am folosit alte 2 polinoame mai simple: "1x^4+3x^2+3x^1"si "1x^2+1x^0", iar in urma impartirii am obtinut rezultatul dorit, anume "+2x^0+1x^2".

# Concluzii

Din temă, am învățat câteva concluzii importante:

* Polinoamele pot fi reprezentate în programe prin intermediul unei colecții de perechi de forma (exponent, coeficient), reprezentate de obicei printr-un map.
* Există o serie de operații matematice pe care le putem realiza asupra polinoamelor, precum adunarea, scăderea, înmulțirea, derivarea și integrarea.
* Pentru a putea realiza operațiile de mai sus într-un program, trebuie să avem o metodă pentru a forma un polinom dintr-un șir de caractere (String) și o metodă pentru a afișa un polinom sub formă de șir de caractere.
* În programare, este important să avem grijă la condițiile de limitare, cum ar fi cazurile când un polinom este gol sau când încercăm să împărțim un polinom cu gradul mai mic decât cel al divizorului.
* Algoritmul de împărțire a polinoamelor se bazează pe algoritmul de împărțire a polinoamelor din matematică, iar utilizarea lui într-un program necesită atenție și înțelegere a procesului matematic.

# Bibliografie

1. *What are Java classes? -* [*www.tutorialspoint.com*](http://www.tutorialspoint.com)
2. [*https://dsrl.eu/courses/pt/materials/PT2023\_Laboratory\_Resources.pdf*](https://dsrl.eu/courses/pt/materials/PT2023_Laboratory_Resources.pdf)
3. *code-it.ro*
4. [*www.jetbrains.com*](http://www.jetbrains.com)
5. *inf.ucv.ro*
6. [*www.scritub.com*](http://www.scritub.com)
7. *dsrl.eu*