DOCUMENTATIE

TEMA 1

NUME STUDENT: Simina Dan-Marius
GRUPA: 30222

CUPRINS

1. Obiectivul temei

Obiectivul principal al temei este proiectarea și realizarea unui calculator pentru polinoame, cu interfata grafica dedicată prin care utilizatorul poate să interacționeze cu aplicația.
Obiective secundare:

- Analiza problemei și identificarea cerințelor (secțiunea 2)
- Proiectarea calculatorului (secțiunea 3)
- Implementarea calculatorului (sectiunea 4)
- Testarea (secțiunea 5)

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

Cerințele funcționale sunt reprezentate faptul că utilizoatorul trebuie sa poată să introducă două polinoame, iar apoi sa poata să selecteze operația pe care dorește să o facă adunare/scadere/inmulțire/împarțire/derivare/integrare, iar aplicația să îi furnizeze rezultatul dorit.
Cerințele non funcționale sunt reprezentate de faptul că aplicația ar trebui să fie intuitiva și ușor de folosit, iar funcționarea aplicației nu ar trebui să fie pusă în pericol dacă utilizatorul introduce un input neadecvat.

Use-case-uri:

1) Adunarea polinoamelor:
   Cazul 1: utilizatorul introduce un input valid
   1. Utilizatorul introduce doua polinoame valide prin intermediul interfeței grafice.
   2. Utilizatorul apasă butonul dedicat operației de adunare.
   3. Calculatorul face operația de adunare a celor 2 polinoame și afișează rezultatul.

   Cazul 2: utilizatorul introduce un input invalid
   1. Utilizatorul introduce doua polinoame, dintre care unul sau ambele nu sunt valide(nu respecta forma ...+/-cx^p+/-...).
   2. Utilizatorul apasă butonul dedicat operației de adunare.
   3. Calculatorul afișează un mesaj de eroare.

2) Scaderea polinoamelor
   Cazul 1: utilizatorul introduce un input valid
   1. Utilizatorul introduce doua polinoame valide prin intermediul interfeței grafice.
   2. Utilizatorul apasă butonul dedicat operației de scadere.
   3. Calculatorul face operația de scadere a celor 2 polinoame și afișează rezultatul.

   Cazul 2: utilizatorul introduce un input invalid
   1. Utilizatorul introduce doua polinoame, dintre care unul sau ambele nu sunt valide(nu respecta forma ...+/-cx^p+/-...).
   2. Utilizatorul apasă butonul dedicat operației de scadere.
   3. Calculatorul afișează un mesaj de eroare.

3) Înmulțirea polinoamelor
   Cazul 1: utilizatorul introduce un input valid
   1. Utilizatorul introduce doua polinoame valide prin intermediul interfeței grafice.
   2. Utilizatorul apasă butonul dedicat operației de înmulțire.
   3. Calculatorul face operația de înmulțire a celor 2 polinoame și afișează rezultatul.

   Cazul 2: utilizatorul introduce un input invalid
   1. Utilizatorul introduce doua polinoame, dintre care unul sau ambele nu sunt valide(nu respecta forma ...+/-cx^p+/-...).
   2. Utilizatorul apasă butonul dedicat operației de înmulțire.

3. Calculatorul afișează un mesaj de eroare.

4) Împărțirea polinoamelor

Cazul 1: utilizatorul introduce un input valid

1. Utilizatorul introduce doua polinoame valide prin intermediul interfeței grafice, iar al doilea nu e nul.
2. Utilizatorul apasă butonul dedicat operației de împărțire.
3. Calculatorul face operația de împărțire a celor 2 polinoame și afișează rezultatul.

Cazul 2: utilizatorul introduce un input invalid

1. Utilizatorul introduce doua polinoame, dintre care unul sau ambele nu sunt valide(nu respecta forma ...+/-cx^p+/-...).
2. Utilizatorul apasă butonul dedicat operației de împărțire.
3. Calculatorul afișează un mesaj de eroare.

Cazul 3:

1. Utilizatorul introduce doua polinoame valide prin intermediul interfeței grafice, dar al doilea e 0.
2. Utilizatorul apasă butonul dedicat operației de împărțire.
3. Calculatorul face verificarea ca polinomul cu care se face împărțirea să fie diferit de 0, iar în acest caz afișsează un mesaj de eroare.

5) Derivarea polinoamelor

Cazul 1: utilizatorul introduce un input valid

1. Utilizatorul introduce un polinom valid prin intermediul interfeței grafice.
2. Utilizatorul apasă butonul dedicat operației de derivare.
3. Calculatorul face operația de derivare a celor 2 polinoame și afișează rezultatul.

Cazul 2: utilizatorul introduce un input invalid

1. Utilizatorul introduce un polinom care nu este valid(nu respecta forma ...+/-cx^p+/-...).
2. Utilizatorul apasă butonul dedicat operației de derivare.
3. Calculatorul afișează un mesaj de eroare.

6) Integrarea polinoamelor

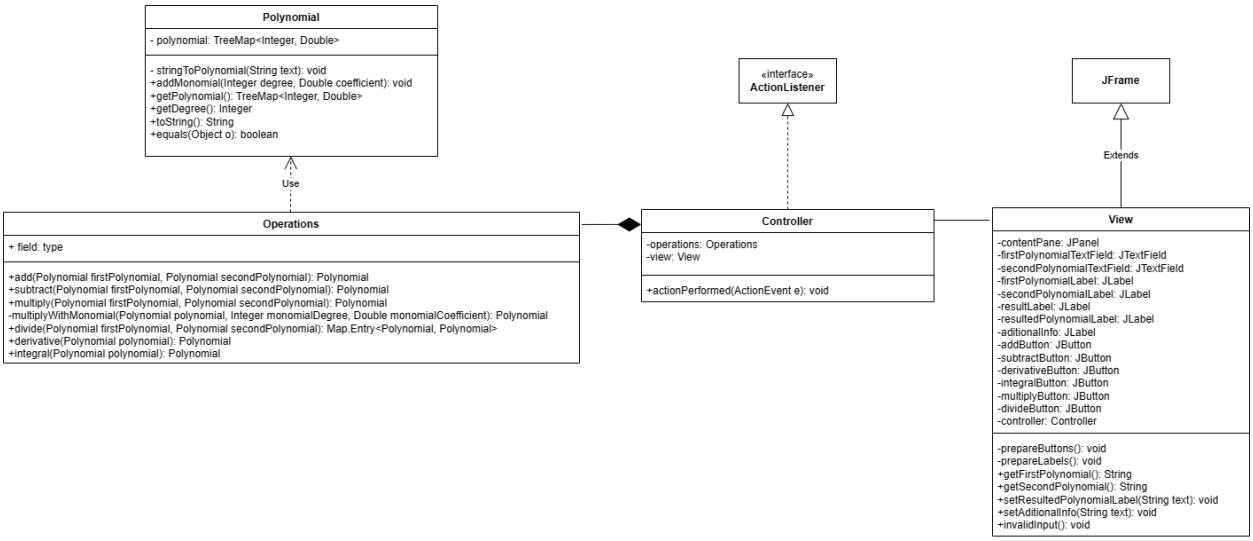
Cazul 1: utilizatorul introduce un input valid

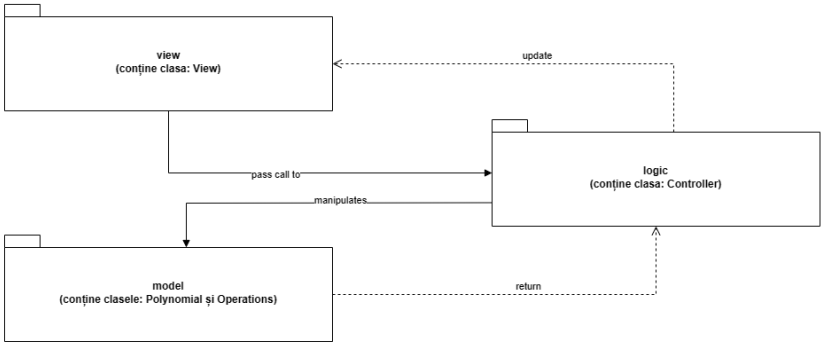
1. Utilizatorul introduce un polinom valid prin intermediul interfeței grafice.
2. Utilizatorul apasă butonul dedicat operației de integrare.
3. Calculatorul face operația de integrare a celor 2 polinoame și afișează rezultatul.

Cazul 2: utilizatorul introduce un input invalid

1. Utilizatorul introduce un polinom care nu este valid(nu respecta forma ...+/-cx^p+/-...).
2. Utilizatorul apasă butonul dedicat operației de integrare.
3. Calculatorul afișează un mesaj de eroare.

3. Proiectare





Aplicația e împarțită în 3 pachete model, view și logic:

- Pachetul view conține clasa **View** care implementează interfața grafică.
- Pachetul model conține clasa **Polynomial** care implementeaza tipul de dată pentru polinom și clasa **Operations** care implementeaza operatiile pe polinoame.
- Pachetul logic conține clasa **Controller** care face legătura intre partea de model si partea de view.

Algoritm umtilizat pentru a împărți două polinoame P și Q:

1. Ordonează monomii celor două polinoame P și Q în ordine descrescătoare în funcție de gradul lor.
2. Împarte polinomul cu grad mai mare la celălalt polinom care are un grad mai mic (să considerăm că P are cel mai mare grad).
3. Împarte primul monom al lui P la primul monom al lui Q și obține primul termen al câtului.
4. Înmulțește câtul cu Q și scade rezultatul înmulțirii din P, obținând restul împărțirii.
5. Repetă procedura de la pasul 2, considerând restul ca noul deîmpărțit al împărțirii, până când gradul restului este mai mic decât Q.

4. Implementare

Clasa Polynomial – implementează tipul de dată polinom.
Câmpul **polynomial** – polinomul este stocat într-un TreeMap unde unde în key este stocat gradul, iar în value coeficientul monomului cu gradul respectiv.
Metode importante:

- Metoda **addMonomial** inserează în **polynomial** un monom dat prin grad și coeficient, dar înainte de a face inserarea verifică daca există deja un monom cu gradul respectiv, dacă nu exista face inserarea, iar dacă există aduna coeficientul monomul pe care dorim să-l inserăm, dacă valoarea rezultată e nenulă inserează, și dacă nu șterge monomul.
- Metoda **stringToPolynomial** este folosită de catre constructor atunci când vrem să construim un polinom dintr-un sir de caractere introdus de la tastatură, se folosește de un regex care identifica subsirurile de forma +/- c*x^n și le sparge în semn, coeficient, x, și putere.

Clasa Operations – implementează operațiile pe polinoame.
Metode importante:

- Metoda **add** realizează operația de adunare. Crează un polinom nou și adaugă în el toate monoamele din cele 2 polinoame folosind **addMonomial**. Returnează polinomul obținut.
- Metoda **subtract** realizează operația de scadere. Funcționează la fel ca **add** doar ca toate monoamele din al doilea polinom sunt înmulțite cu -1.
- Metoda **multiply** realizează operația de înmulțire. Crează un polinom nou, apoi inmultește fiecare monom din primul polinom cu fiecare monom din al doilea polinom și le adaugă in noul polinom folosind **addMonomial**. Returnează polinomul obținut.
- Metoda **multiplyWithMonomial** realizează operația de înmulțire a unui polinom cu un monom. Funcționează pe același principiu ca și metoda de înmulțire a două polinoame, este folosită in cadrul metodei **divide**. Returnează polinomul obținut.
- Meoda **divide** realizează operația de împărțire a doua polinoame. Crează două polinoame, iar într-unul pune câtul(monom cu monom pe masură ce se calculează), iar în al doilea pune restul obținut. Metoda se folosește de algoritmul prezentat la punctul 3. Returnează polinoamele obținute.
- Metoda **derivative** realizează operația de derivare. Crează un polinom nou, iar apoi parcurge polinomul primit și pentru fiecare monom aplică operația de derivare și îl adaugă în noul polinom. Returnează polinomul obținut.
- Metoda **integral** realizează operația de integrare. Funcționează pe același principiu ca și metoda de derivare.

Clasa Controller – realizează legătura intre partea de **view** și partea de **model**.
Metode importante:

- Metoda **actionPerformed** în momentul apăsării unu buton de către utilizator metoda apelează funcția respectivă din **model**, iar apoi face update în interfața grafică cu rezultatul obținut.

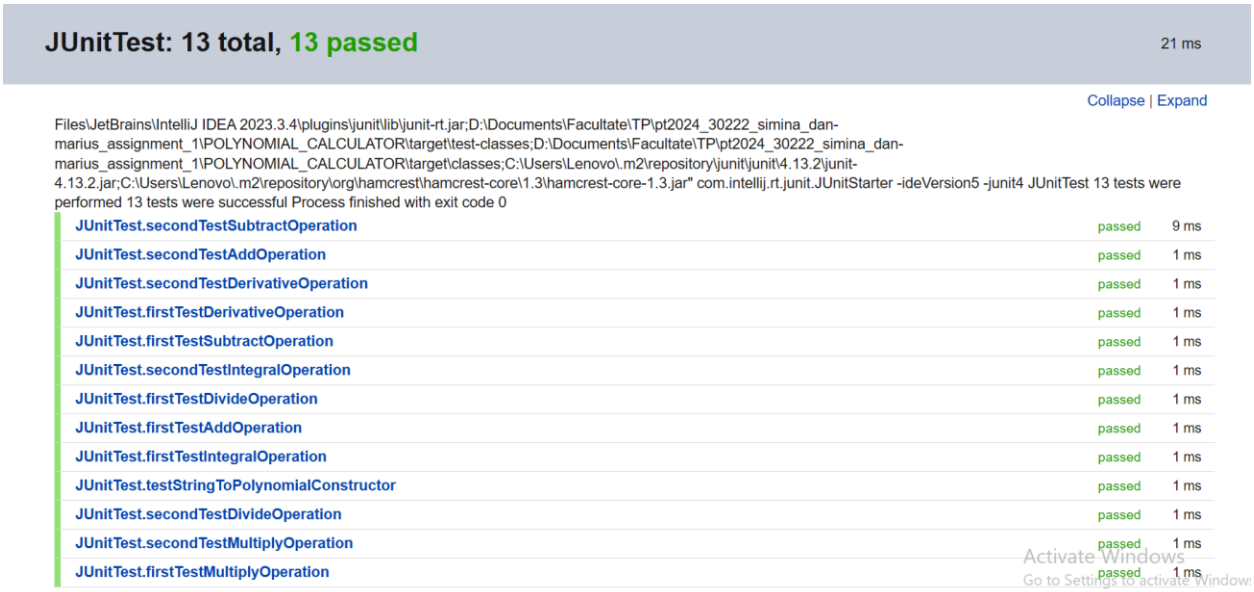
Clasa View – realizează interfața grafică.
Metode importante:

- Metoda **prepareButtons** setează textul, culoarea, comanda și action listener pentru fiecare buton.
- Metoda **prepareLabels** setează textul, culoarea pentru fiecare label.
- Metodele **getFirstPolynomial** și **getSecondPolynomial** returnează polinoamele introduse de utilizator în text box-ul repectiv.
- Metodaele **setResultedPolynomialLabel** și **setAditionalInfo** afișează rezultatele în label-urile respective.
- Metoda **invalidInput** afișează o fereastră cu un mesaj de eroare.

Pozitionarea elementelor în fereastră a fost realizată cu Swing UI Designer.

5. Rezultate

Am realizat testarea unitară folosind JUnit pentru toate operațiile pe polinoame și pentru constructorul prin care se crează un polinom dintr-un șir de caractere. Pentru constructor am realizat un singur test, iar pentru operații câte două teste pentru fiecare operație, perechile de polinoame pentru teste sunt create înaintea testării, și pentru fiecare operație se compară rezultatul obținut cu rezultatul corect.



6. Concluzii

În urma acestei teme am învățat să lucrez mai ordonat, să folosesc Swing UI Designer, iar cel mai important cred că a fost faptul că am învățat ce este un regex și am vazut cum îți poate ușura munca.
Ca dezvoltări ulterioare se poate adăuga funcția de calcul a radacinilor, interfața grafică poate să mai fie polișată, iar unele operații pot să fie făcute mai eficient.

7. Bibliografie

1. regular expressions 101 - regex101: build, test, and debug regex
2. ASSIGNMENT 1 – SUPPORT PRESENTATION (PART 1)
3. ASSIGNMENT 1 – SUPPORT PRESENTATION (PART 2)
4. ASSIGNMENT 1 – SUPPORT PRESENTATION (PART 3)
5. Interface Map.Entry<K,V> – Map.Entry (Java Platform SE 8) (oracle.com)
6. draw.io
7. Maven – Maven – Introduction (apache.org)