DOCUMENTATIE

TEMA 1

NUME STUDENT: SAMOILA DENIS-VALENTIN

GRUPA: 30223

Cuprins

1. Obiectivul temei
2. Analiza problemei, modelare, scenarii, cazuri de utilizare
3. Proiectare
4. Implementare
5. Rezultate
6. Concluzii
7. Bibliografie

1.Obiectivul temei

1.1 Obiectivul principal al temei este de a crea un calculator de polinoame menit sa usureze munca persoanelor care doresc sa faca diverse operatii cu acestea. Prin definitie, un polinom este o suma de unul sau mai multe monoame, iar un monom este un produs de puteri ale variabilelor cu exponenti intregi nenegativi.

1.2 Obiective secundare

- alegerea structurilor de date folosite pentru a problema (capitol 3)

- impartirea pe clase (capitol 3)

- dezvoltarea algoritmilor (capitol 3)

- implementarea solutiei (capitol 4)

- testare (capitol 5)

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

Cerinte functionale: - proiectarea si implementarea unui calculator pentru operatii cu polinoame care sa permita utilizatorului sa introduca unul sau doua polinoame si sa efectueze operatia/ile dorita/e.

Polinoamele introduce de catre utilizator trebuie sa fie cu o singura variabila, iar coeficientii sa fie intregi.

Operatii:

- introducere de polinom de la tastatura sub forma de String cu format de tipul: anx^n+a(n-1)x^(n-1)...+a1x + a0, unde $n \in \mathbb{N}$ si $a0, a1, a2 .... an \in Z$

- adunare de doua polinoame

- scadere de doua polinoame

- inmultire de doua polinoame

- impartire de doua polinoame

- derivarea unui polinom

- integrarea unui polinom

Cerinte non-functionale: - asigurarea unei interfete responsive si a unor operatii matematice eficiente

- claritatea interfetei, asigurarea unor butoane intuitive care permite utilizatorului sa inteleaga usor modul de utilizare al aplicatiei

- gestionarea corecta a erorilor

- realizarea de teste pentru a verifica corectitudinea

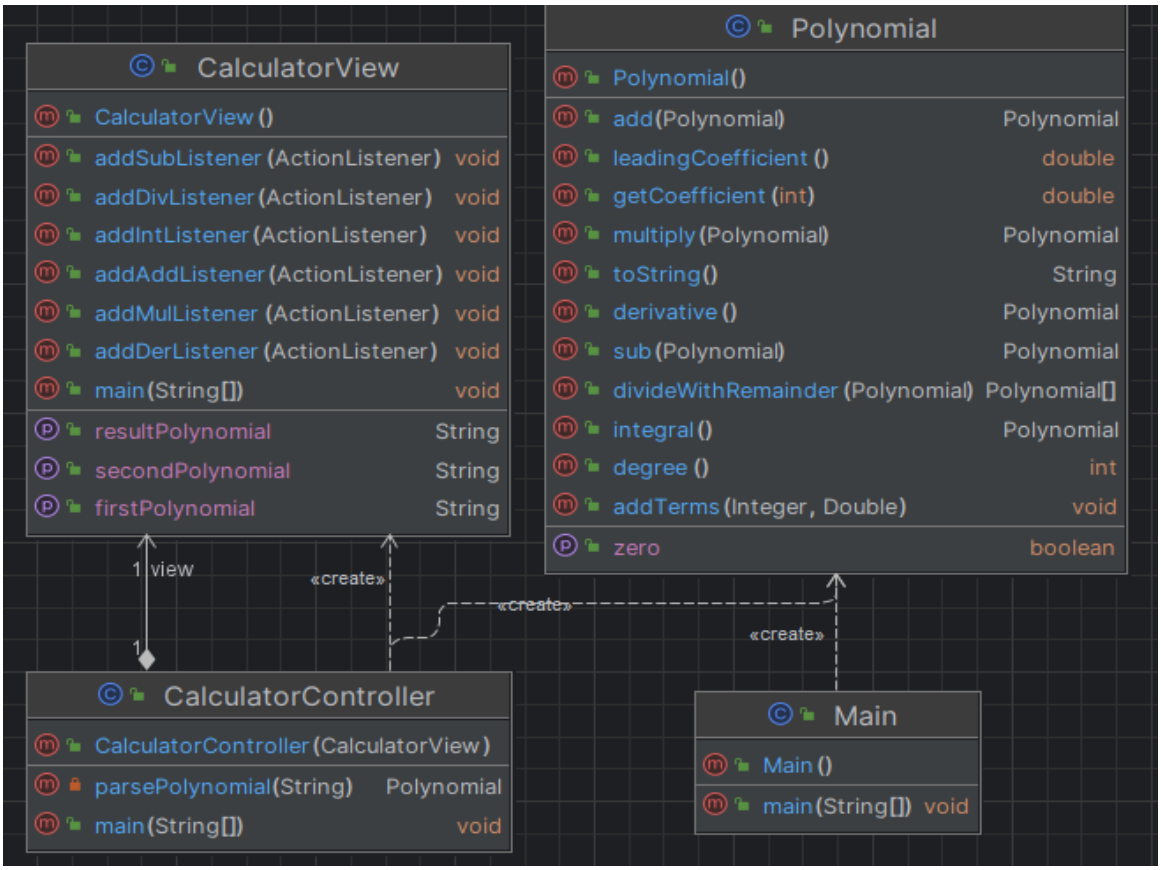
Use-case:

Utilizatorul introduce de la tastatura in campurile destinate polinoamelor un sir de caractere care se va folosi de formatul mentionat anterior (anx^n+a(n-1)x^(n-1)...+a1x + a0), iar ai apoi va putea efectua diferite operatii (adunare, scadere, inmultire, impartire, derivare si integrare). Fiecare buton din interfata descrie una din operatii. Apasandu-l pe cel dorit, utilizatorul va putea observa rezultatul afisat pe ecran.

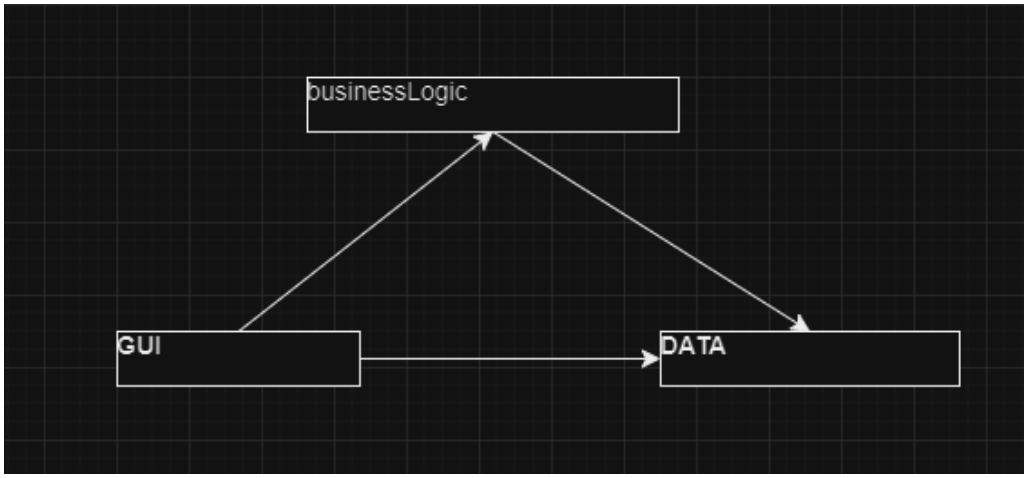
Pentru a opera corect, trebuie sa ne asiguram ca datele introduce sunt valide. Daca nu se respecta formatul polinomului, utilizatorului i se va afisa un mesaj de eroare, dar are posibilitatea de a introduce din nou date, fara a reporni aplicatia.

3.Proiectare

3.1 Diagrama UML de clase



3.2 Diagrama de pachete



3.3 Structuri de date

```
private Map<Integer,Double> terms;
```

-Este utilizata pentru a stoca termenii polinomului, unde cheia reprezinta exponentul, iar valoarea coeficientul.

3.4 Algoritmi

Algoritmii folositi sunt cei uzuali invatati din liceu, nu am folosit implementari aditionale, in mare fiecare polinom este parcurs de la inceput pana la final si se efectueaza operatia dorita.

4.Implementare

```java
public class Polynomial {
    27 usages
    private Map<Integer,Double> terms;

    23 usages
    public Polynomial(){
        terms=new HashMap<>();
    }

    60 usages
    public void addTerms(Integer exponent, Double coefficient){
        terms.put(exponent,coefficient);
    }

    public Polynomial add(Polynomial polynomial){
        Polynomial result=new Polynomial();
        for(Map.Entry<Integer,Double> term: terms.entrySet()){
            Integer exponent=term.getKey();
            Double coefficient=term.getValue();
            result.addTerms(exponent,coefficient);
        }
        for(Map.Entry<Integer,Double> term: polynomial.terms.entrySet()) {
            Integer exponent=term.getKey();
            Double coefficient=term.getValue();
            if(result.terms.containsKey(exponent)){
                coefficient+=result.terms.get(exponent);
            }
            result.addTerms(exponent,coefficient);
        }
        return result;
    }
```

Clasa Polynomial contine campul de termini care definesc un polinom. Tot aici se afla si metoda de adaugare a termenilor in polinom, precum si metodele specifice tuturor operatiilor de operare cu polinoamele introduse de catre utilizator.

Clasa CalculatorController este cea care controleaza interactiunile dintre polinom/polinoame si interfata utilizator.

Clasa CalculatorView este interfata grafica a calculatorului polynomial.

Implementarea interfetei utilizator

Interfata utilizator (UI) este implementata folosind Java Swing pentru a crea o aplicatie simpla.

CalculatorView este o clasa care extinde JFrame si continetoate componentele vizuale necesare pentru interfata calculatorului. Componentele sunt organizate folosind JPanel-uri.

CalculatorController este o clasa care gestioneaza logica aplicatiei, continand clase interne care implementeaza interfete “ActionListener” pentru fiecare buton. Atunci cand un buton este apasat, datele sunt preluate si sunt apelate metodele corespunzatoare clasei “Polynomial”.

5.Testare

5.1 Adunare

5.0 + 4.0x + 3.0x^2

- x + 2.0x^3

5.0 + 3.0x + 3.0x^2 + 2.0x^3

Test add finished successfully

5.2 Scadere

5.0 + 4.0x + 3.0x^2

- x + 2.0x^3

5.0 + 5.0x + 3.0x^2 - 2.0x^3

Test sub finished successfully

5.3 Inmultire

4.0x + 3.0x^2

1.0 + 2.0x

4.0x + 11.0x^2 + 6.0x^3

Test multiply finished successfully

5.4 Impartire

4.0 + 7.0x + 6.0x^2

1.0 + 2.0x

2.0 + 3.0x

2.0

Test division finished successfully

5.5 Derivare

4.0x + 2.0x^2 + 3.0x^3

4.0 + 4.0x + 9.0x^2

Test derivative finished successfully

5.6 Integrare

4.0 + 3.0x + 2.0x^2

4.0x + 1.5x^2 + 0.6666666666666666x^3

Test integral successfully

6.Concluzii

Dupa realizarea acestui proiect mi-am dat seama ca inceputul conteaza cel mai mult, cum reusesti sa planifici prima data “schita” proiectului. Dupa ce am realizat un mic “template” dupa care sa ma ghidez, mi-a fost mult mai usor sa am implementari ulterioare sau sa modific anumite chestii, deoarece

scheletul proiectului a fost realizat bine de la bun inceput. Cazurile marunte de implementare consider ca au fost cele mai costitistoare din punct de vedere al timpului.

O posibila dezvoltare ulterioara ar putea fi implementarea generica a operatiilor pe mai multe polinoame, nu doar doua.

7.Bibliografie

https://www.youtube.com/?app=desktop&hl=ro&gl=RO

https://chat.openai.com/

https://gitlab.com/mariustotoian225/POO_Lab_Materials_30223_2023/-/blob/main/lab_12/src/test/java/CalculatorTest.java?ref_type=heads
https://www.baeldung.com/junit-5
https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-engine
https://www.jetbrains.com/help/idea/run-debug-configuration-junit.html#logs