DOCUMENTATIE

TEMA 1

NUME STUDENT: Frim Eleazar

GRUPA: 7

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

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

[3. Proiectare 3](#_Toc95297887)

[4. Implementare 3](#_Toc95297888)

[5. Rezultate 3](#_Toc95297889)

[6. Concluzii 3](#_Toc95297890)

[7. Bibliografie 3](#_Toc95297891)

# Obiectivul temei

Scopul principal al proiectului este să implementăm un calculator care poate efectua operații pe polinoame. Pentru a atinge acest obiectiv, trebuie să parcurgem câțiva pași secundari:

* ***Construcția unei interfețe grafice***

Pentru a facilita utilizarea aplicației, este necesară crearea unei interfețe grafice intuitive. Aceasta va include câmpuri de text în care utilizatorul poate introduce polinoame și în care va fi afișat rezultatul, precum și butoane pentru selectarea operației dorite. Detaliile implementării vor fi discutate în secțiunea 4.

* ***Parsarea polinomului introdus de utilizator***

Pentru a interpreta corect datele, este crucial să transpunem polinomul introdus de utilizator într-un obiect care să permită efectuarea ușoară a operațiilor pe polinoame. Această implementare va fi descrisă în detaliu în secțiunea 4.

* ***Operațiile asupra polinoamelor***

Pentru a obține rezultatele dorite, programul va prelua polinoamele introduse și va efectua operația selectată. Procesul de implementare va fi explicat în secțiunea 4.

* ***Testarea***

Pentru a asigura corectitudinea operațiilor efectuate de program, va trebui să testăm aplicația folosind calcule predefinite și să verificăm că rezultatele sunt conforme. Aceasta va fi abordată în secțiunea 5.

# Analiza problemei, modelare, scenarii, cazuri de utilizare

*Cerințele funcționale conturează acțiunile și comportamentele specifice așteptate de la calculator. Aceste cerințe detaliază diferitele operații matematice pe care trebuie să le susțină calculatorul.*

***Cerințe Funcționale:***

* Calculator-ul trebuie să permită utilizatorilor să introducă polinoame.
* Utilizatorul trebuie să poată selecta operația matematică dorită.
* Calculator-ul trebuie să permită adunarea, scăderea, înmulțirea și împărțirea a două polinoame.
* Calculator-ul trebuie să permită derivarea sau integrarea unui polinom.
* Calculator-ul trebuie să afișeze rezultatul corect în toate cazurile.

***Cerințe Non-Funcționale:***

* Calculator-ul trebuie să fie intuitiv și ușor de utilizat.
* Calculator-ul trebuie să permită introducerea unui polinom în mai multe tipuri de formate.

*Aceste cerințe funcționale și non-funcționale definesc cadrul general al funcționării și experienței utilizatorului pentru calculatorul de polinoame.*

Calculator de polinoame

# 3. Proiectare

*Structura aplicației este organizată în trei părți importante, fiecare definind o caracteristică esențială a aplicației, fiind împărțită astfel în:*

* ***UI*** (Interfață Utilizator): Acest pachet se ocupă de dezvoltarea și gestionarea interfeței grafice, controlul butoanelor și a intrărilor utilizatorului.
* ***logic*** : Pachetul logic gestionează partea logică a aplicației, adică operațiile efectuate între polinoame.
* ***dataModels*** (Modele de Date): În cadrul pachetului de date, este definită clasa Polynomial, care menține datele despre un anumit polinom într-o structură de tip TreeMap.

Această structură modulară facilitează organizarea și dezvoltarea aplicației, asigurând o separare clară între aspectele de interfață grafică, logica aplicației și gestionarea datelor.**A screenshot of a computer

Description automatically generated**

# Implementare

La baza aplicației stă clasa Polynomial, care reprezintă un polinom. Acesta este menținut printr-o structură de tip TreeMap, în care cheia reprezintă puterea unui termen din polinom, iar valoarea este coeficientul termenului. Utilizarea unei structuri de tip TreeMap oferă avantajul de a menține întotdeauna ordinea în intrările sale, ceea ce ajută la afișarea rezultatului într-un mod ordonat.

Clasa Polynomial dispune de trei constructori: unul care creează un polinom gol, un alt constructor care face o copie a unui obiect de același tip primit ca parametru, iar ultimul și cel mai util constructor creează un polinom bazat pe un șir de caractere primit ca parametru. Parsarea șirului de caractere se bazează pe utilizarea expresiilor regulate, astfel încât termenii polinomului sunt identificați și adăugați în TreeMap. O funcție utilă este funcția toString(), care transformă polinomul din structura TreeMap într-un șir de caractere pentru a putea fi afișat.

O altă clasă care implementează logica aplicației este clasa PolynomialOperations. Aceasta implementează toate metodele în care se efectuează operațiile cu polinoame. Funcțiile sunt statice, astfel încât pot fi apelate fără a fi nevoie să se creeze un obiect. Acest lucru este util deoarece acestea sunt apelate din metodele care definesc acțiunile pentru butoanele din interfață, reducând astfel gradul de complexitate.

Interfața grafică este implementată cu JavaFX și constă din două fișiere importante: view.fxml și clasa Controller. Fișierul view.fxml este specific acestei platforme și conține date despre elementele grafice. Pe lângă acest fișier, sunt prezente și fișierele forms.css și style.css, care stilizează aplicația. Interfata este implementata dinamic astfel incat in cazul in care dorim sa selectam operatia de impartire, va aparea un field nou care va afisa restul impartirii, iar in cazul derivarii sau a integrarii va disparea field-ul pentru cel de al doilea polinom nefiind nevoie de el. Clasa Controller face legătura între partea grafică și partea logică a aplicației. În această clasă sunt definite metodele care sunt executate la apăsarea unui buton. Metodele au rolul de a apela alte metode fără a efectua procesare în interiorul lor, ceea ce ajută la separareainterfeței de utilizator de partea logică a aplicației, facilitând localizarea eventualelor erori care pot apărea.

A screenshot of a math application

Description automatically generated**A screenshot of a calculator

Description automatically generated**

**A screenshot of a calculator

Description automatically generated**

# Rezultate

A screenshot of a computer

Description automatically generatedPentru a fi siguri ca aplicatia ofera rezultatele corecte s-au implementat patru teste pentru fiecare operatie incercand sa simuleze diferite scenarii de uz real de ex: folosirea coeficientilor reali sau lipsa lor, folosirea puterilor(puterea 1 nu se scrie) sau cazuri speciale. Testele sau implementat folosind Junit, unde s-a creat o clasa noua pentru testare si pentru fiecare operatie s-a scris o functie speciala de testare

1. **Concluzii**

In concluzie, tema curenta a ajutat la acumularea cunostintelor legate de folosirea librariei JavaFX si a testarii unitare folosind JUnit. Aplicatia poate fi dezvoltata in continuare prin a face o interfata mai interactive si prin adaugare altor operatii de ex: evaluarea intr-un punct, aflarea radacinilor, trasarea graficului etc.

# Bibliografie

1. Java Map Interface - https://www.javatpoint.com/java-map
2. Junit 5 User Guide - https://junit.org/junit5/docs/current/user-guide/