DOCUMENTATIE

TEMA *1*

NUME STUDENT: Bondre Aurora Malina

GRUPA: 30227

# 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

**Obiectivul principal** consta in implementarea unui calculator de polinoame cu o interfață grafică dedicată prin intermediul căreia utilizatorul poate introduce polinoame, selecta operația matematică de efectuat și vizualiza rezultatul.

**Obiective secundare:**

* Analiza problemei
* Proiectarea calculatorului de polinoame
* Implementarea calculatorul de polinoame
* Testarea calculatorul de polinoame

# Analiza problemei, modelare, scenarii, cazuri de utilizare

**Cerințe Funcționale:**

1. **Introducerea Polinoamelor:**
   * Utilizatorul poate introduce polinoamele de la tastatura într-un format corespunzător (de exemplu, "3\*x^2 + 2\*x - 5").
2. **Selectarea Operației Matematice:**
   * Utilizatorul poate selecta operația matematică pe care dorește să o efectueze prin apasarea butoanelor(adunare, scădere, înmulțire, derivare, integrare).
3. **Vizualizarea Rezultatului:**
   * Utilizatorul poate vizualiza rezultatul operației matematice efectuate asupra polinoamelor introduse.

**Cerinte Non Functionale:**

1. **Fiabilitate**

* Codul gestioneaza corect situațiile neașteptate sau erori care ar putea apărea în timpul operațiilor pe polinoame( de exemplu, introducerea unor caractere invalide de catre utilizator)

1. **Flexibilitate și Extensibilitate:**

* Codul permite extinderea și adăugarea de noi funcționalități în viitor.

1. **Eficiența Performanței:**

* Codul este eficient din punct de vedere al performanței, astfel încât se pot manipula polinoamele într-un mod rapid și fără întârzieri semnificative.

**Use-Case**

**Scenariu de Utilizare**: Adăugare Polinoame

**Actor Principal:** Utilizator

**Scenariu de Succes Principal**:

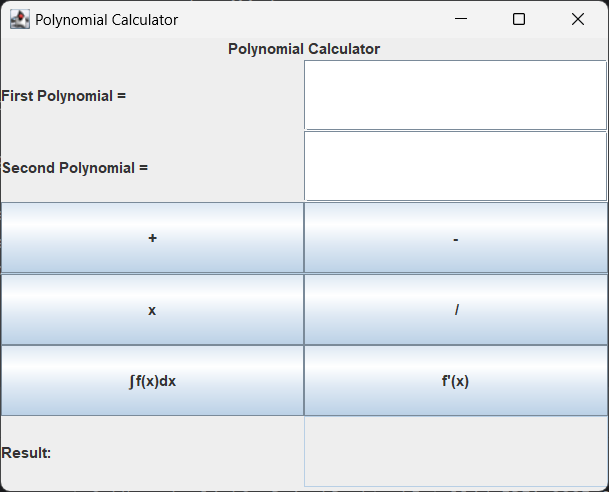
1. Utilizatorul introduce 2 polinoame în interfața grafică utilizator.
2. Utilizatorul selectează operația de "adunare".
3. Calculatorul de polinoame efectuează adunarea celor două polinoame și afișează rezultatul.

**Secvență Alternativă: Polinoame Incorecte**

* Utilizatorul introduce polinoame incorecte (de exemplu, cu 2 sau mai multe variabile).
* Scenariul revine la pasul 1.

# Proiectare

Interfata Calculator:



**Diagrama UML de clase:**

**A screenshot of a computer program

Description automatically generated**

# Implementare

***Clasa Polinom***

**1.Metoda add**

* adaug in polinomul rezultat valorile din primul polinom,
* adaug elementele din polinomul2 si verific daca in polinomul rezultat exista deja un monom cu acelasi grad, daca exista adun coeficienti.
* ***M*etoda sub** are aceasi logica dar coeficientii se scad

***2.Metoda mul***

* parcurg ambele polinoame, adun gradele si inmultesc coeficientii
* daca exista deja in polinom un monom cu acelasi grad atunci se aduna coeficienti

**3.Derivare**

* parcurg polinomul, scad din gradul monomului -1 si la coeficient ii atribui produsul dintre coeficientul initial si gradul initial
* pentru **Integrare** e aceasi abordare dar gradul creste cu 1 iar coeficientul ia valoarea raportului dintre coeficientul initial si gradul +1

**2. CalculatorInterface**

* in constructorul acestei clase implementez interfata grafica a calculatorului, folosind Java Swing
* in metoda polinomInvalid verific daca inputul se potriveste cu pattern ul regexului
* in actionPerformed :
  + folosesc getText pentru a extrage string ul polinoamelor
  + aplic metoda parsePolynomial pe cele doua string uri
  + cu un switch apelez operatiile iin functie de ce buton este apasat
  + pentru polinomul result apelez metoda polynomialTotring si il afisez cu setText
* in metoda parsePolynomial
  + creez un regez pentru polinom
  + se utilizeaza clasa Pattern pentru a compila expresia regex
  + gasesc aparitile expresiei in sir cu matcher.find()
  + se obtine textul potrivit cu metoda group() si se extrage coeficietul si gradul
  + se creaza un obiect Monom si se adauga la Polinom
* in metoda polynomialToString
  + - instantiez un obiect StringBuilder pentru a creea sirul
    - parcurg monomii si verific semnul coeficientilor pentru a adauga semnele
    - adaug coeficientul daca exista la fel si gradul
    - adaug sirul in rezultat

# 

# Rezultate

# Concluzii

**Abilitati dobandite:**

* utilizarea interfețelor grafice utilizând Java Swing
* utilizarea expresiilor regex pentru a transforma și valida polinoamele sub formă de șiruri de caractere.
* Utilizarea interfetei Java Map

**Posibile viitoare extinderi ale proiectului:**

* + adaugarea de noi functionalitati
  + optimizarea implementarilor

# Bibliografie