

# **DOCUMENTATIE**

## **TEMA 1**

NUME STUDENT: Nemes Teodora  
GRUPA: 30226

# CUPRINS

1.	Obiectivul temei.....	3
2.	Analiza problemei, modelare, scenarii, cazuri de utilizare .....	4
3.	Proiectare .....	6
4.	Implementare .....	8
5.	Rezultate .....	10
6.	Concluzii.....	12
7.	Bibliografie .....	12

# 1. Obiectivul temei

**Obiectiv principal:** Proiectarea și implementați un 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:

Nr.crt	Obiectiv secundar	Detalii	Capitol documentatie
1	Analiza problemei si identificare cerintelor	Identificarea și definirea cerințelor funcționale și non-funcționale ale calculatorului de polinoame. Analiza utilizatorilor și a nevoilor lor în ceea ce privește utilizarea sistemului.	Cap 2. Analiza problemei, modelare, scenarii, cazuri de utilizare
2	Identificarea cazurilor de utilizare	Identificarea și descrierea cazurilor de utilizare principale ale calculatorului de polinoame, cum ar fi adunarea, scăderea, înmulțirea, împărțirea, derivarea și integrarea polinoamelor. Modelarea și documentarea scenariilor de utilizare, inclusiv fluxurile de lucru și interacțiunile utilizatorului cu sistemul.	Cap 2. Analiza problemei, modelare, scenarii, cazuri de utilizare
3	Proiectarea calculatorului	Proiectarea modelului de date pentru reprezentarea polinoamelor și a rezultatelor operațiilor polinomiale. Proiectarea interfeței utilizatorului pentru a permite introducerea polinoamelor, selectarea operației dorite și afișarea rezultatelor.	Cap 3. Proiectare
4	Implementarea calculatorul	Implementarea claselor și a metodelor asociate modelului de date pentru manipularea polinoamelor. Implementarea controlerului pentru gestionarea interacțiunii dintre modelul de date și interfața utilizatorului. Implementarea interfeței utilizatorului folosind biblioteca Java Swing.	Cap 4. Implementare
5	Testare calculator	Definirea și implementarea testelor unitare pentru metodele cheie ale sistemului. Executarea testelor și identificarea și remedierea erorilor și a comportamentului incorect.	Cap 5. Rezultate

# 1. Analiza problemei, modelare, scenarii, cazuri de utilizare

## Cerinte functionale:

- Permitearea utilizatorilor să introducă polinoame
- Permitearea utilizatorilor să selecteze operația matematică.
- Adunarea a doua polinoame
- Scaderea a doua polinoame
- Inmultirea a doua polinoame
- Impartirea a doua polinoame
- Derivarea unui polinom
- Integrarea unui polinom

## Cerinte non functionale:

- Calculatorul polinomial ar trebui să fie intuitiv și ușor de folosit de către utilizator.

## Cazuri de utilizare:

### 1. Adunare polinoame

**Actor principal:** utilizator

**Scenariu principal de succes:**

- Utilizatorul introduce cele 2 polinoame în interfața grafică.
- Utilizatorul selectează operația de "adunare" prin apăsarea butonului +
- Calculatorul polinomial efectuează adunarea celor două polinoame și afișează rezultatul în zona de text.

**Secvență alternativă:** Polinoame incorecte

- Utilizatorul introduce polinoame incorecte (de exemplu: care conțin alte caractere, cu 2 sau mai multe variabile)
- Scenariul revine la introducerea polinoamelor.
- Se așteaptă introducerea unei noi valori care să respecte formatul și se calculează valoarea sumei doar pentru aceasta

### 2. Scădere polinoame

**Actor principal:** utilizator

**Scenariu principal de succes:**

- Utilizatorul introduce cele 2 polinoame în interfața grafică.
- Utilizatorul selectează operația de "scădere" prin apăsarea butonului -.
- Calculatorul polinomial efectuează scăderea celor două polinoame și afișează rezultatul în zona de text.

**Secvență alternativă:** Polinoame incorecte

- Utilizatorul introduce polinoame incorecte (de exemplu: care conțin alte caractere, cu 2 sau mai multe variabile)
- Scenariul revine la introducerea polinoamelor.
- Se așteaptă introducerea unei noi valori care să respecte formatul și se calculează valoarea diferenței doar pentru aceasta.

### 3. Înmulțire polinoame

**Actor principal:** utilizator

**Scenariu principal de succes:**

- Utilizatorul introduce cele 2 polinoame în interfața grafică.
- Utilizatorul selectează operația de "înmulțire" prin apăsarea butonului \*.
- Calculatorul polinomial efectuează înmulțirea celor două polinoame și afișează rezultatul în zona de text.

**Secvență alternativă:** Polinoame incorecte

- Utilizatorul introduce polinoame incorecte (de exemplu: care conțin alte caractere, cu 2 sau mai multe variabile)
- Scenariul revine la introducerea polinoamelor.
- Se așteaptă introducerea unei noi valori care să respecte formatul și se calculează valoarea produsului doar pentru aceasta.

### 4. Împărțire polinoame

**Actor principal:** utilizator

**Scenariu principal de succes:**

- Utilizatorul introduce cele 2 polinoame în interfața grafică.
- Utilizatorul selectează operația de "împărțire" prin apăsarea butonului /.
- Calculatorul polinomial efectuează împărțirea celor două polinoame și afișează rezultatul în zona de text.

**Secvență alternativă:** Polinoame incorecte

- Utilizatorul introduce polinoame incorecte (de exemplu: care conțin alte caractere, cu 2 sau mai multe variabile)
- Scenariul revine la introducerea polinoamelor.
- Se așteaptă introducerea unei noi valori care să respecte formatul și se calculează valoarea rezultatului împărțirii doar pentru aceasta.

### 5. Derivare polinom

**Actor principal:** utilizator

**Scenariu principal de succes:**

- Utilizatorul introduce polinomul în primul text field din interfața grafică.
- Utilizatorul selectează operația de "derivare" prin apăsarea butonului dx.
- Calculatorul polinomial efectuează derivarea polinomului și afișează rezultatul în zona de text.

**Secvență alternativă:** Polinom incorect

- Utilizatorul introduce un polinom incorect (de exemplu: care conține alte caractere, cu 2 sau mai multe variabile)
- Scenariul revine la introducerea polinomului.
- Se așteaptă introducerea unei noi valori care să respecte formatul și se calculează valoarea derivatei doar pentru aceasta.

### 6. Integrare polinom

**Actor principal:** utilizator

**Scenariu principal de succes:**

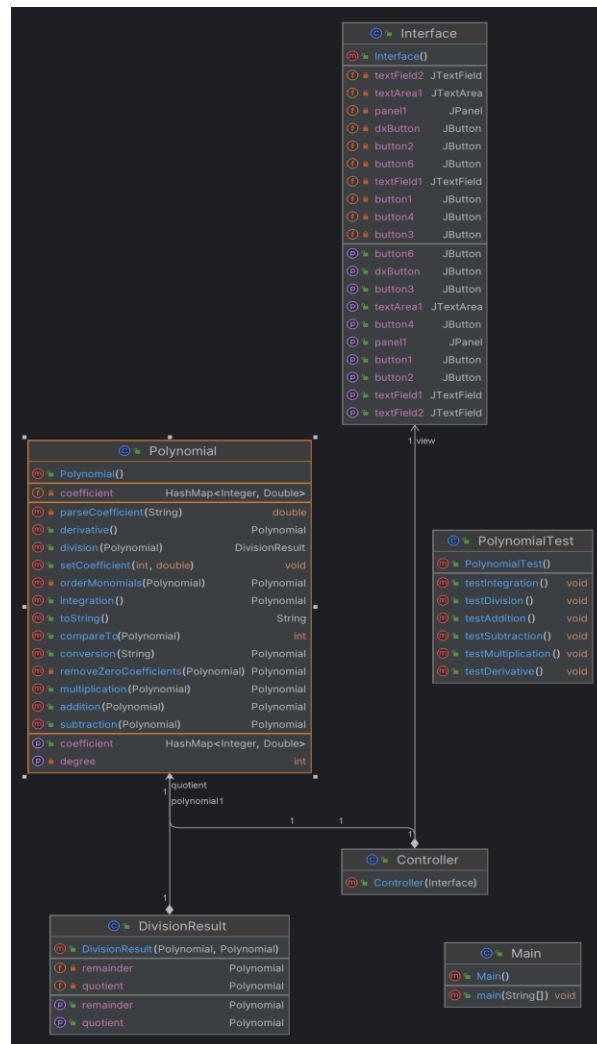
- Utilizatorul introduce polinomul în primul text field interfața grafică.
- Utilizatorul selectează operația de "integrare" prin apăsarea butonului ∫.
- Calculatorul polinomial efectuează integrarea polinomului și afișează rezultatul în zona de text.

### Secvență alternativă: Polinom incorect

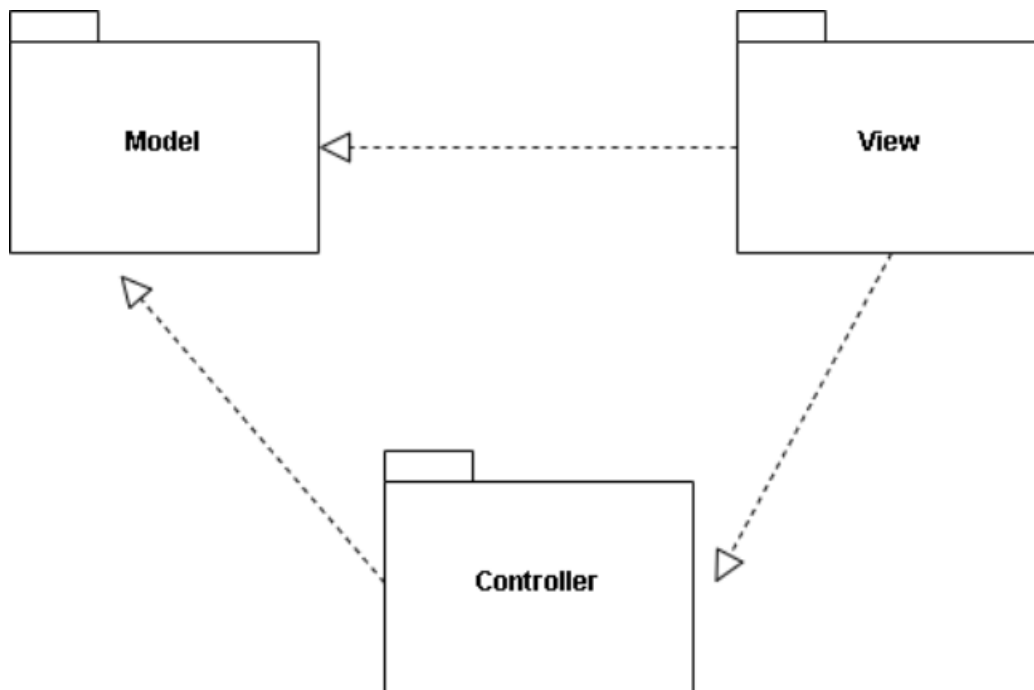
- Utilizatorul introduce un polinom incorect (de exemplu: care conține alte caractere, cu 2 sau mai multe variabile)
- Scenariul revine la introducerea polinomului.
- Se așteaptă introducerea unei noi valori care să respecte formatul și se calculează valoarea integralei doar pentru aceasta.

## 7. Proiectare

Diagrama UML de clase:



### Diagrama UML de pachete:



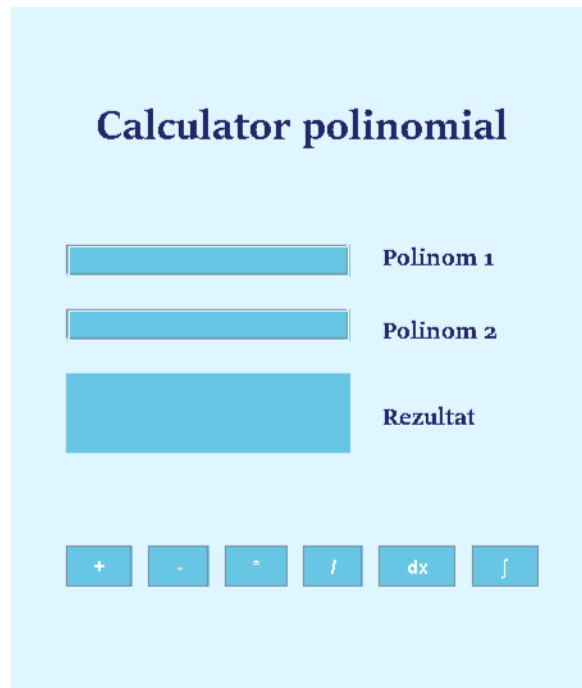
### Interfata:

Interfața utilizatorului este compusă din următoarele elemente:

**Două câmpuri de text (TextField):** Aceste câmpuri permit utilizatorului să introducă coeficienții și exponenții polinoamelor pe care dorește să le folosească pentru operațiile ulterioare. Aplicația va interpreta aceste date pentru a efectua operațiile corect. Pentru operațiile cu un singur operand, se va lua în calcul doar valoarea introdusă în primul TextField.

**Un câmp de text mare (TextArea):** Acest câmp de text este folosit pentru afișarea rezultatelor operațiilor efectuate cu polinoamele introduse de utilizator. Aici vor fi afișate polinoamele rezultate în urma operațiilor de adunare, scădere, înmulțire, împărțire, derivare și integrare.

**Șase butoane (Button):** Fiecare buton reprezintă una dintre operațiile pe care utilizatorul le poate efectua cu polinoamele introduse. Aceste operații sunt adunarea, scăderea, înmulțirea, împărțirea, derivarea și integrarea. Atunci când utilizatorul apasă un buton specific, aplicația va efectua operația corespunzătoare pe polinoamele introduse și va afișa rezultatul în câmpul de text.



## 8. Implementare

### 1. Clasa Polynomial

Reprezintă un polinom și oferă metode pentru operații precum adunare, scădere, înmulțire, împărțire, derivare și integrare.

#### Campuri:

`private HashMap<Integer, Double> coefficient:` Acest câmp reține coeficienții polinomului, fiecare asociat cu gradul corespunzător.

#### Metode Importante:

`conversion(String polynomialString):` Metodă care primește un șir de caractere ce reprezintă un polinom și returnează un obiect de tip `Polynomial` corespunzător acestuia.

`addition(Polynomial p):` Metodă pentru adunarea polinomului curent cu un alt polinom dat ca parametru.

`subtraction(Polynomial p):` Metodă pentru scăderea polinomului curent cu un alt polinom dat ca parametru.

`derivative():` Metodă pentru calculul derivatei polinomului curent.



integration(): Metodă pentru calculul integralei polinomului curent.

multiplication(Polynomial p): Metodă pentru înmulțirea polinomului curent cu un alt polinom dat ca parametru.

division(Polynomial divisor): Metodă pentru împărțirea polinomului curent la un alt polinom dat ca parametru.

toString(): Metodă care convertește polinomul într-un șir de caractere pentru afișare.

setCoefficient(int exponent, double coefficient): Metodă pentru setarea coeficientului asociat unui anumit grad al polinomului.

getCoefficient(): Metodă pentru obținerea mapării între grade și coeficienți a polinomului.

compareTo(Polynomial other): Metodă pentru compararea gradului polinomului curent cu gradul unui alt polinom dat.

getDegree(): Metodă pentru obținerea gradului maxim al polinomului.

## 2. Clasa DivisionResult

Reprezintă rezultatul unei operații de împărțire a două polinoame și conține un polinom cât și restul rezultatului împărțirii.

### Campuri:

private Polynomial quotient: Acest câmp reprezintă polinomul cât obținut în urma împărțirii.

private Polynomial remainder: Acest câmp reprezintă polinomul rest obținut în urma împărțirii.

### Metode Importante:

DivisionResult(Polynomial quotient, Polynomial remainder): Constructorul clasei, care primește ca parametri polinomul cât și polinomul rest și inițializează câmpurile corespunzătoare.

getQuotient(): Metodă pentru obținerea polinomului cât.

getRemainder(): Metodă pentru obținerea polinomului rest.

## 3. Clasa Controller

Este responsabilă pentru gestionarea interacțiunii dintre modelul de date (Polynomial și DivisionResult) și interfața utilizatorului (Interface). Aceasta conține ascultători pentru acțiunile utilizatorului și implementează logica asociată fiecărei acțiuni.

### **Campuri:**

private Interface view: Acest câmp reprezintă interfața utilizatorului cu care este asociată clasa Controller.

private Polynomial polynomial1: Acest câmp reține primul polinom introdus de utilizator.

private Polynomial polynomial2: Acest câmp reține al doilea polinom introdus de utilizator.

### **Metode Importante:**

Controller(Interface view): Constructorul clasei, care primește interfața utilizatorului și inițializează câmpurile asociate.

Button1Listener, Button2Listener, Button3Listener, Button4Listener, DxButtonListener, Button6Listener: Aceste clase interne implementează interfețele ActionListener și sunt responsabile pentru gestionarea acțiunilor utilizatorului în funcție de butoanele apăsate. Fiecare ascultător preia polinoamele introduse de utilizator din câmpurile de text, realizează operația corespunzătoare și afișează rezultatul în câmpul de text asociat.

## **4. Clasa PolynomialTest**

Este responsabilă pentru testarea unitară a metodelor din clasa Polynomial. Aceasta conține mai multe metode de test, fiecare testând comportamentul unei anumite metode sau operații asociate cu polinoamele. Pentru fiecare test, se construiesc polinoame specifice pentru a apela metoda corespunzătoare, iar apoi se verifică dacă rezultatul obținut este cel așteptat.

### **Metode Importante:**

testAddition(): Această metodă testează corectitudinea operației de adunare a două polinoame.

testSubtraction(): Această metodă testează corectitudinea operației de scădere a două polinoame.

testMultiplication(): Această metodă testează corectitudinea operației de înmulțire a două polinoame.

testDerivative(): Această metodă testează corectitudinea operației de derivare a unui polinom.

testIntegration(): Această metodă testează corectitudinea operației de integrare a unui polinom.

testDivision(): Această metodă testează corectitudinea operației de împărțire a două polinoame și corectitudinea rezultatului cât și a restului.

## **9. Rezultate**

Pentru a prezenta scenariile de testare, s-a utilizat un exemplu pentru fiecare operație de bază a calculatorului polinomial: adunare, scădere, înmulțire, împărțire, derivare și integrare.

Cazurile de testare care au fost luate în calcul la testarea aplicației au fost:

### **Testare pentru Adunare Polinoame:**

Scenariu 1: Adunare corectă a două polinoame cu coeficienți și grade corecte.

Scenariu 2: Adunare a două polinoame cu coeficienți negativi.

Scenariu 3: Adunare a două polinoame cu grade diferite.

### Testare pentru Scădere Polinoame:

Scenariu 1: Scădere corectă a două polinoame cu coeficienți și grade corecte.

Scenariu 2: Scădere a două polinoame cu grade egale.

Scenariu 3: Scădere a două polinoame cu grade diferite.

### Testare pentru Înmulțire Polinoame:

Scenariu 1: Înmulțire corectă a două polinoame cu număr diferit de termeni.

Scenariu 2: Înmulțire a două polinoame cu același număr de termeni.

Scenariu 3: Înmulțire a două polinoame cu grade diferite.

### Testare pentru Împărțire Polinoame:

Scenariu 1: Împărțire corectă a două polinoame cu același număr de termeni.

Scenariu 2: Împărțire a două polinoame cu coeficienți negativi.

Scenariu 3: Împărțire a două polinoame cu număr diferit de termeni.

### Testare pentru Derivare Polinom:

Scenariu 1: Derivare corectă a unui polinom format dintr-un singur monom de gradul 0.

Scenariu 2: Derivare a unui polinom cu coeficienți negativi.

Scenariu 3: Derivare a unui polinom cu coeficienți pozitivi.

### Testare pentru Integrare Polinom:

Scenariu 1: Integrare corectă a unui polinom format dintr-un singur monom de gradul 0.

Scenariu 2: Integrare a unui polinom cu coeficienți negativi.

Scenariu 3: Integrare a unui polinom cu coeficienți pozitivi.

În cadrul testării prin intermediul utilitarului **Junit**, s-au luat în considerare următoarele scenarii:

#### 1. Adunare

Polinom 1:  $3x^2 + 4x$

Polinom 2:  $2x^2 + 5$

Rezultat obținut:  $5.0x^2 + 4.0x + 5.0$

#### 2. Scadere

Polinom 1:  $3x^2 + 4x$

Polinom 2:  $2x^2 + 5$

Rezultat obținut:  $x^2 + 4.0x - 5.0$

#### 3. Înmulțire

Polinom 1:  $3x^2 + 4x$

Polinom 2:  $2x^2 + 5$

Rezultat obținut:  $6.0x^4 + 8.0x^3 + 15.0x^2 + 20.0x$

#### 4. Împărțire

Polinom 1:  $5x^2 + x$

Polinom 2:  $x$

Rezultat obtinut:  $5x + 1$

## 5. Derivare

Polinom 1:  $3x^3 + 2x^2 + x$

Rezultat obtinut:  $9.0x^2 + 4.0x + 1.0$

## 6. Integrare

Polinom 1:  $3x^3 + 2x^2 + x$

Rezultat obtinut:  $0.75x^4 + 0.6666666666666666x^3 + 0.5x^2$

# 10. Concluzii

### Concluzii si lucruri invatate din tema:

- Realizarea acestei teme a facilitat procesul de invatare si utilizare practica a expresiilor regulate, pe care nu le-am mai folosit pana in prezent
- Am înțeles necesitatea de a crea scenarii de utilizare și de testare pentru a asigura funcționalitatea și corectitudinea aplicației.
- Am obținut experiență în utilizarea unor instrumente precum JUnit pentru testare unitară și în dezvoltarea de clase de testare pentru diverse funcționalități ale aplicației.
- M-am familiarizat cu utilizarea Java Swing pentru crearea unei interfete grafice simple si intuitive pentru utilizator.
- Am constientizat importanta impartirii proiectului in cele 3 pachete principale, View, Model si Controller, care sa faciliteze o buna organizare si structura a codului

### Posibile dezvoltarii ulterioare:

- Implementarea unor funcții avansate pentru manipularea polinoamelor, cum ar fi factorizarea, găsirea rădăcinilor etc.
- Adaugarea de butoane care sa contina cifre si termenul x in cadrul interfetei, pentru a evita cazul in care utilizatorul introduce si alte caractere.
- Extinderea interfeței utilizatorului pentru a oferi mai multe opțiuni de personalizare și vizualizare a rezultatelor.

# 11. Bibliografie

### Surse utilizare:

1. Bruce Eckel, Thinking in Java (4th Edition), Publisher: Prentice Hall PTR Upper Saddle River, NJ United States, ISBN:978-0-13-187248-6 Published:01 December 2005.
2. What is Package Diagram? <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/>
3. Curs Programare Orientata pe Obiecte: [https://users.utcluj.ro/~igiosan/teaching\\_poo.html](https://users.utcluj.ro/~igiosan/teaching_poo.html)
4. draw.io : <https://www.drawio.com/>