MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Facultatea de Automatica si Calculatoare

-Calculatoare si tehnologia informatiei-

CALCULATOR DE POLINOAME

Documentatie

TEMA1

Nistor Dalia

Grupa 30224

Cuprins

1.OBIECTIVUL TEMEI

Obiectivul acestei teme de laborator a fost sa propunem o solutie pentru un calculator de polinoame. Aceasta tema, de asemenea, trebuia sa contina si o interfata grafica dedicata prin care utilizatorul sa poata introduce diferite operatii (adunare, scadere, inmultire, impartire, derivare, integrare),sa poata introduce polinoamele si sa vizualizeze rezultatul. Polinoamele s-au considerat ca fiind de o singura variabila, iar coeficientii reali.

Astfel. tema a avut urmatorii pasi pentru indeplinirea obiectivului principal:

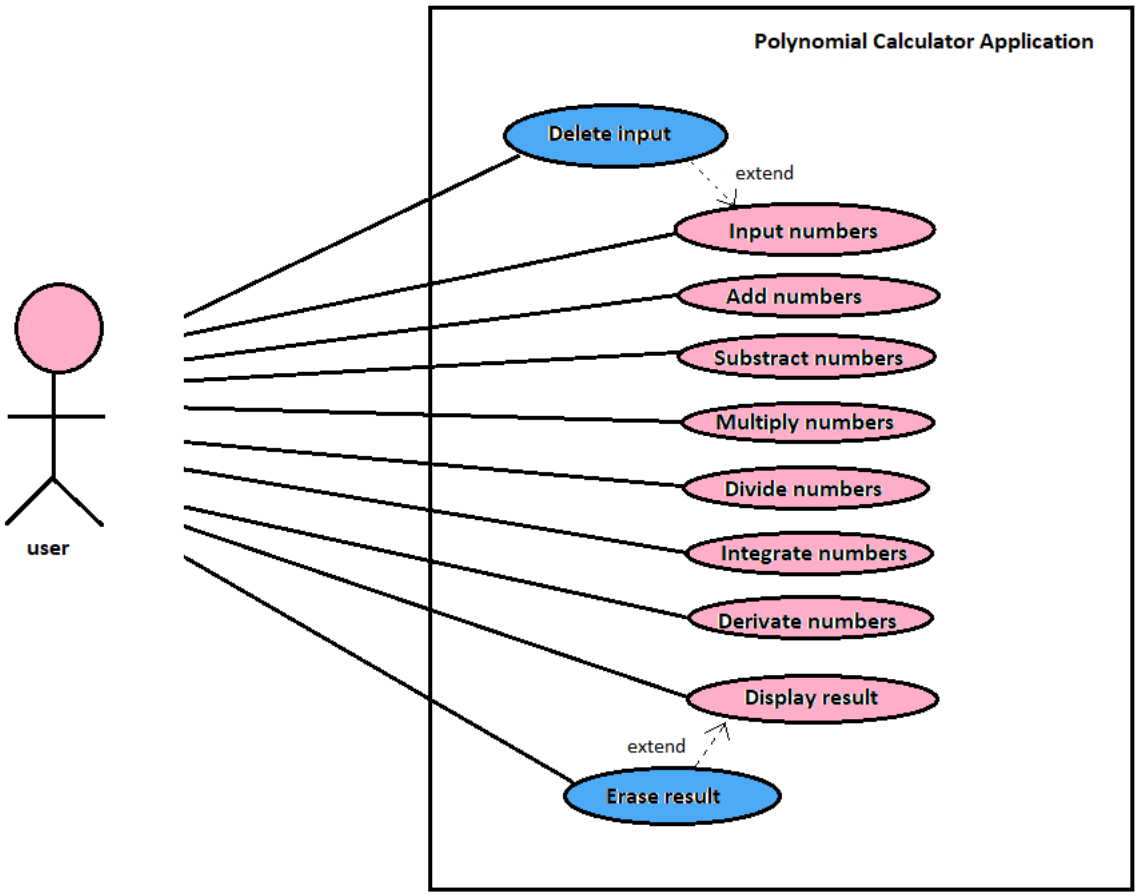
1. Polinomul trebuie sa fie stocat intr-un hashmap
2. Clase cu maxim 300 de linii (cu excepția claselor UI) și metode.
   cu maximum 30 de linii.
3. Utilizarea convențiilor de denumire Java
4. Implementarea operatiilor de adunare,scadere,inmultire,impartire,derivare,integrare
5. Folosirea Junit5 pentru testare
6. Folosirea expresiilor regulate si pattern matching pentru extragerea coeficiientilor polinomului

2.ANALIZA PROBLEMEI, MODELARE, SCENARII, CAZURI DE UTILIZARE

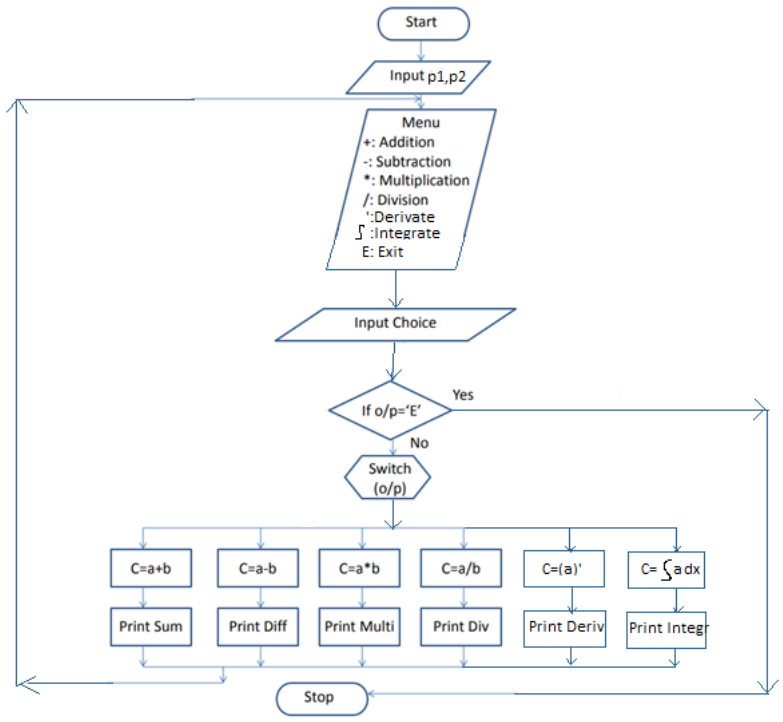
Polinoamele sunt construite din termeni numiți monoame, care sunt alcătuite dintr-o constantă (numită coeficient) înmulțită cu una sau mai multe variabile. Fiecare variabilă poate avea un exponent constant întreg pozitiv.. Exista o multitudine de variante pentru implementarea unui calculator de polinoame, insa ramane la latitudinea programatorului sa aleaga modul de lucru.

Eu am ales sa folosesc un regex pentru despartirea polinomului in monoame. Am gasit unul potrivit (cu ajutorul https://regexr.com/) astfel incat sa nu despart polinomul in functie de variabila„x”.Daca utilizatorul va introduce orice alta variabila, programul va functiona dar va afisa cu ‚x’. In cazul exponentilor, orice putere trebuie specificata, inclusiv pentru ridicarea la puterea întâi. Astfel, in cazul ridicării lui x la puterea intai, polinomul va fi completat cu valoarea: “x^1”, altfel programul nu va functiona conform așteptărilor.In cazul in care monomul nu contine o variabila, nu este necesar sa se scrie ca „x^0” . Am folosit pattern si matcher deoarece clasele Matcher și Pattern oferă facilitatea de expresie regulată Java. Am introdus intr-un hashmap valoriile astfel incat key contine exponentul iar value contine coeficientul.

USE CASE DIAGRAM



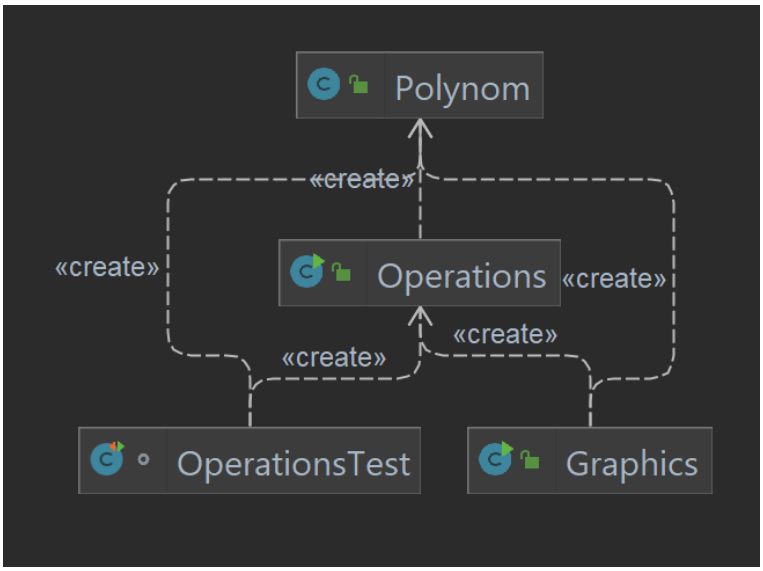
FLOWCHART

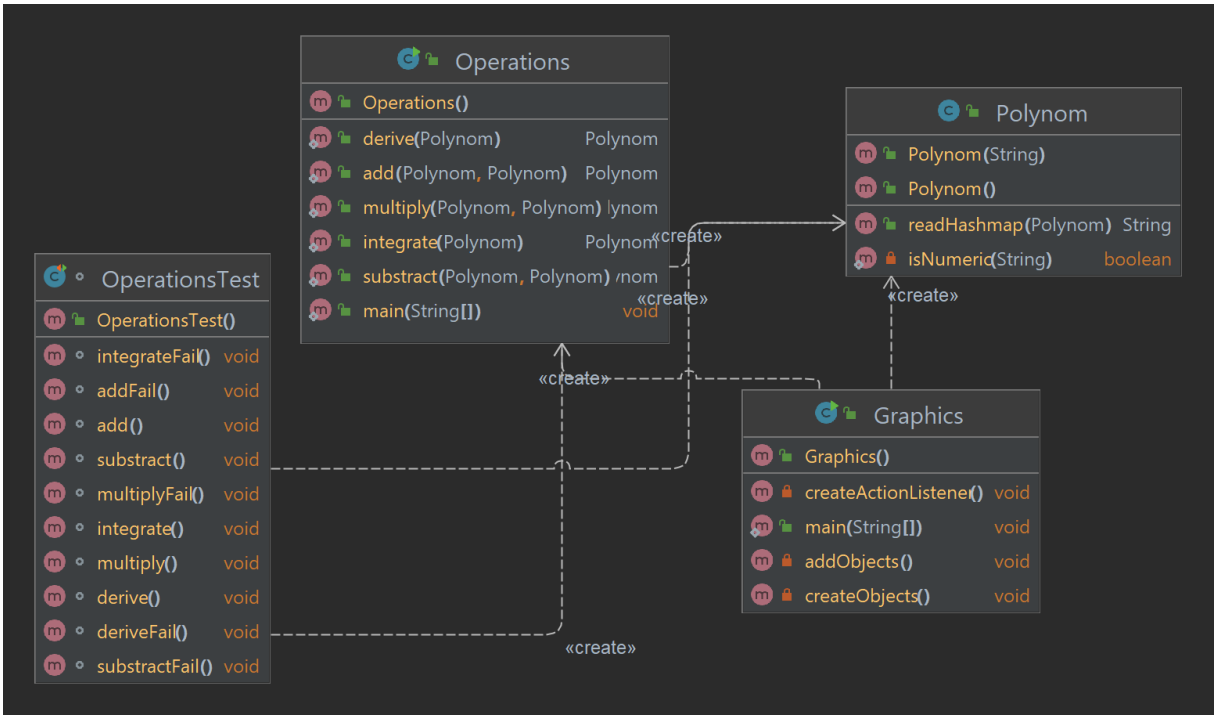


3.PROIECTARE

Diagrama UML

Diagrama de clase UML este o notație grafică utilizată pentru a construi și vizualiza sisteme orientate pe obiecte.





Stocarea datelor

Cand vine vorba despre mulțimi de obiecte(elemente) în java, în majoritatea cazurilor apare și nevoia de a le stoca într-un container pentru manipularea lor ulterioară (sortare, afișare, ștergere etc.)În Java avem *The Core Collection Interfaces*, care reprezintă o ierarhie de interfețe din care avem o serie de clase care implementează diferite tipuri de clase pentru stocarea datelor. Pentru stocarea datelor din aceasta tema eu am folosit un hashmap (HashMap <Integer,Float>).

4.IMPLEMENTARE

Clasa Polynom

In clasa Polynom am realizat un constructor ce primeste ca parametru un string, in care se efectueaza separarea polinomului, retinand coeficientii intr-un hashmap si un contructor fara parametrii. De asemenea in clasa Polynom se afla o metoda foarte importanta, formarea unui polinom din hashmap-ul rezultat in urma operatiilor.

```
public String readHashmap(Polynom p1)
{
    String polynomialString = "";
    boolean firstTerm = true;
    for (Map.Entry<Integer, Float> entry : p1.monom.entrySet())
    {
        Integer exponent = entry.getKey();
        Float coeff = entry.getValue();
        if (coeff == 0)
            continue;
        if (coeff > 0 && !firstTerm)
            polynomialString += "+";
        firstTerm = false;
        if (coeff != 1 || exponent == 0)
            polynomialString += Float.toString(coeff);
        if (exponent > 0)
        {
            polynomialString += "x";
            if (exponent > 1)
                polynomialString += "^" + exponent;
        }
    }
    return polynomialString;
}
```

Clasa Operations

In aceasta clasa am implementat 5 metode care returneaza adunarea, inmultirea, scaderea, derivarea sau integrarea. Toate metodele primesc ca parametrii 2 variabile de tip Polynom.

1. Pentru operatiile de adunare,scadere si inmultire rationamentul a fost asemanator. Am retinut intr-o alta variabila, Polynom result, unul dintre cei doi parametrii primiti, iar apoi am parcurs hashmap-ul celuilalt cu un foreach.Am verificat daca result contine key-ul polinomului parcurs. In caz afirmativ am facut adunarea, scaderea respectiv inmultirea, iar in caz negativ am introdus in result coeficientii si cheile respective. Nu am implementat operatia de impartire.

2. Pentru operatia de derivare am parcurs polinomul primit ca parametru si m-am folosit de formula: (x^n)'=n*x^(n-1). Am implementat operatia astfel incat sa se afiseze doar doua variabile dupa virgula.

3. Pentru operatia de integrare am parcurs polinomul primit ca parametru si m am folosit de formula integralei lui x^n: x^(n+1)/(n+1). De asemenea, si aceasta operatie a fost implementata astfel incat sa afiseze doua variabile dupa virgula.\

Clasa Graphics

In aceasta clasa am implementat interfata cu care interactioneaza utilizatorul. Am folosit pentru aceasta Java Swing. Swing în Java este un set de instrumente de interfață grafică cu utilizatorul (GUI) care include componentele GUI.

Biblioteca Java Swing este construită pe baza Java Abstract Widget Toolkit (AWT), un set de instrumente GUI mai vechi, dependent de platformă. Se pot utiliza componente simple de programare Java GUI, cum ar fi butonul, caseta de text etc., din bibliotecă și nu trebuie să se creeze componente de la zero.

Aplicatia mea contine 25 de butoane:

1. 6 butoane pentru cele 6 operatii (adunare, inmultire, impartire, derivare, integrare, scadere) pentru ca la apasarea acestora sa se execute operatia dorita. In cazul in care utilizatorul nu introduce ambele polinoame pentru operatiile de adunare,scadere,inmultire se va afisa un mesaj : „Te rog sa introduci polinoame in ambele campuri", iar in cazul in care nu introduce polinomul in primul camp pentru operatiile de derivare si integrare se va afisa mesajul: „Te rog sa introduci polinom in primul camp pentru integrare/derivare".
2. 17 butoane implementate pentru a putea fi introdus polinomul si cu ajutorul acestora (0,1,2,3,4,5,6,7,8,9,*,/,x,^,+,-,.)
3. un buton numit del, care sterge rezultatul pentru a putea fi executata o noua operatie, pentru acelasi polinom
4. un alt buton numit clear all care sterge atat cele 2 polinoame cat si rezultatul

De asemena contine 3 TextField-uri:

TextField= spatii dreptunghiulare in care se pot introduce date de la tastatura. Pot fi folosite si pentru a afisa rezultate.

1. 2 pentru introducerea polinoamelor
2. unul pentru afisarea rezultatului

In plus, mai are si 4 label-uri:

Label = este o etichetă ce poate fi folosita drept titlu sau pentru a ajuta utilizatorul

1. 1 Label ca titlu „Polynomial Calculator”
2. 3 pentru a indica unde se intorduc polinoamele si unde va aparea rezultatul („First Polynomial”, „Second Polynomial”, „Result”).

Pentru fiecare button, s-a făcut o metodă nouă care implică ActionEvent. Astfel, de fiecare dată când are loc o acțiune la un buton, de exemplu a fost apăsat, ActionEvent-ul denumit e, transmite informația la ActionListener care așteaptă astfel informații. Apoi, au loc evenimentele ce se afla în metodă respectiva

Pentru a putea introduce de la cele 17 butoane in textField-ul selectat, am implementat 2 metode care implica FocusEvent, numite focusLost si focusGained.

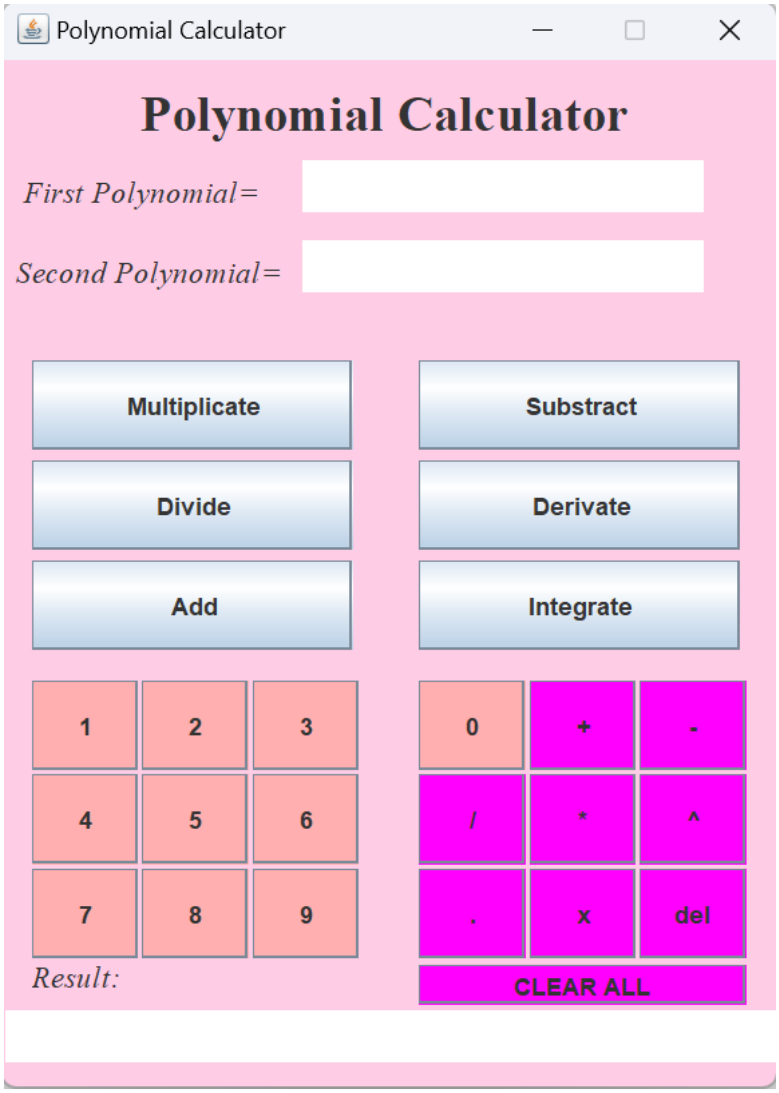
Interfata grafica propriu zisa este formata in constructor unde se initializeaza un frame cu proprietatile dorite.

In main-ul programului este instantiata clasa Graphics, astfel de fiecare data cand se initialieaza clasa main se va instantia o noua interfata grafica.

Clasa OperationsTest

In clas OperationsTest am implementat 10 teste cu ajutorul JUnit5, 5 failed si 5 passed.

Interfata arata in felul urmator:



5.REZULTATE

Pentru a testa aplicatia trebuie parcursi urmatorii pasi:

1. Introducerea celor 2 polinoame tinand cont de specificarea puterii, inclusiv pentru ridicarea la puterea intai
2. Apasarea butonului cu operatia dorita
3. Vizualizarea rezultatului

Testare cu Junit5

Platforma JUnit este responsabilă pentru lansarea cadrelor de testare pe JVM. Aceasta definește o interfață stabilă și puternică între JUnit și clienții săi, cum ar fi instrumentele de construcție.Platforma integrează cu ușurință clienții cu JUnit pentru a descoperi și executa teste.De asemenea, definește API-ul TestEngine pentru dezvoltarea unui cadru de testare care rulează pe platforma JUnit. Prin implementarea unui TestEngine personalizat, putem conecta bibliotecile de testare ale părților terțe direct în JUnit.

Am implementat 5 passed test, cate unul pentru fiecare operatie:

Pentru operatiile de adunare,scadere,inmultire am initializat 2 polinoame, o operatie si un string care reprezinta rezultatul asteptat. Intr-o variabila de tipul polinom am apelat metoda ce trebuie testata si am convertit-o ,cu ajutorul metodei din clasa Polynom, intr-un string.

```java
@Test
void add() {
  Polynom p1=new Polynom( s: "4+5x^2+3.6x^7+2.5x^10");
  Polynom p2=new Polynom( s: "7+2.5x^1-8x^7+2x^8-2.8x^10");
  Operations op=new Operations();
  String expectedresult="11.0+2.5x+5.0x^2-4.4x^7+2.0x^8-0.29999995x^10";
  Polynom actualResult=op.add(p1,p2);
  String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

```java
@Test
void substract() {
    Polynom p1=new Polynom( s: "4+3x^1-12x^6+7x^8");
    Polynom p2=new Polynom( s: "12-3.5x^2+4x^3-5.5x^6");
    Operations op=new Operations();
    String expectedresult="-8.0+3.0x+3.5x^2-4.0x^3-6.5x^6+7.0x^8";
    Polynom actualResult=op.substract(p1,p2);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
@Test
```

```java
@Test
void multiply() {
    Polynom p1=new Polynom( s: "4+3x^1-12x^6+7x^8");
    Polynom p2=new Polynom( s: "12-3.5x^2+4x^3-5.5x^6");
    Operations op=new Operations();
    String expectedresult="48.0+3.0x-3.5x^2+4.0x^3+66.0x^6+7.0x^8";
    Polynom actualResult=op.multiply(p1,p2);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

Pentru operatiile de integrare si derivare am folosit acelasi rationament dar am initializat un singur polinom

```
@Test
void integrate() {
    Polynom p1=new Polynom( s: "2+3.6x^2+2.5x^5-3.6x^7");
    Operations op=new Operations();
    String expectedresult="2.0x+1.2x^3+0.42x^6-0.45x^8";
    Polynom actualResult=op.integrate(p1);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

```
void derive() {
    Polynom p1=new Polynom( s: "3.3+5x^4+3x^5+2x^10");
    Operations op=new Operations();
    String expectedresult="20.0x^3+15.0x^4+20.0x^9";
    Polynom actualResult=op.derive(p1);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

In plus, am implementat si 5 failed tests:

```
    @Test
    void addFail()
    {
        Polynom p1=new Polynom( s: "2+3.6x^2+2.5x^5-3.6x^7");
        Polynom p2=new Polynom( s: "");
        Operations op=new Operations();
        String expectedresult="5+3.6x^2+5x^4";
        Polynom actualResult=op.add(p1,p2);
        String actualResultprinted=actualResult.readHashmap(actualResult);
        Assert.assertEquals(actualResultprinted,expectedresult);

    }
```

```java
@Test
void substractFail() {
    Polynom p1=new Polynom( s: "2+3.6x^2+2.5x^5-3.6x^7");
    Polynom p2=new Polynom( s: "4+3x^1-12x^6+7x^8");
    Operations op=new Operations();
    String expectedresult="5+3.6x^2+5x^4";
    Polynom actualResult=op.add(p1,p2);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

```java
@Test
void multiplyFail(){
    Polynom p1=new Polynom( s: "4+3x^1-12x^6+7x^8");
    Polynom p2=new Polynom( s: "12-3.5x^2+4x^3-5.5x^6");
    Operations op=new Operations();
    String expectedresult="40+3.0x-3.5x^2+4.0x^3+66.0x^6";
    Polynom actualResult=op.multiply(p1,p2);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
```

```java
@Test
void integrateFail() {
    Polynom p1=new Polynom( s: "2+3.6x^2+2.5x^5-3.6x^7");
    Operations op=new Operations();
    String expectedresult="2x+12x^3+0.42x^6-0.45x^8";
    Polynom actualResult=op.integrate(p1);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

```java
@Test
void deriveFail() {
    Polynom p1=new Polynom( s: "3.3+5x^4+3x^5+2x^10");
    Operations op=new Operations();
    String expectedresult="x^3+15.0x^4+20.0x^6";
    Polynom actualResult=op.derive(p1);
    String actualResultprinted=actualResult.readHashmap(actualResult);
    Assert.assertEquals(actualResultprinted,expectedresult);
}
```

6.CONCLUZII

In concluzie, consider ca aceasta tema mi-a imbunatatit remarcabil abilitatile de a scrie cod Java, mi-a aprofundat cunostiintele in ce semnifica implementarea paradigmelor OOP, mi-a imbunatatit modul in care incep implementarea unei aplicatii.

Posibile dezvoltari ulterioare: sa nu fie obligatorie puterea variabilei ridicate la puterea intai, implementarea operatiei de impartire.

7.BIBLIOGRAFIE

- https://medium.com/@lucian.ritan/colec%C8%9Bii-de-date-java-69eb6f9056e7
- https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/#:~:text=A%20class%20diagram%20in%20the,and%20the%20relationships%20among%20objects.
- https://medium.com/@lucian.ritan/colec%C8%9Bii-de-date-java-69eb6f9056e7
- https://www.baeldung.com/junit-5
- https://www.w3schools.com/java/java_hashmap.asp