MINISTERUL EDUCAȚIEI NAȚIONALE
UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

# Tema 1

## Calculator de polinoame

Student: Spinean Sebastian

Grupa: 302210

# 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

## Obiectivul principal:

Obiectivul principal al temei este proiectarea si implementarea unui calculator de polinoame de o singura variabila si cu coeficienti intregi.

## Obiective secundare:

| Obiectiv secundar: | Descriere | Capitol |
|---|---|---|
| Determinarea cazurilor de utilizare | Se identifica cazuile de utilizare si se descriu prin intermediul diagramelor use case | 2 |
| Alegerea structurilor de date | Se aleg structurile de date potrivite | 3 |
| Impartirea pe clase | Impartirea codului in pachete si clase | 3 |
| Dezvoltarea algoritmilor | Implementarea algoritmilor pentru fiecare operatie aritmetica | 4 |
| Realizarea unei interfete grafice | Proiectul trebuie sa dispuna de o interfata grafica | 4 |
| Testare | Se testeaza corectitudinea algoritmilor pentru operatii | 5 |

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

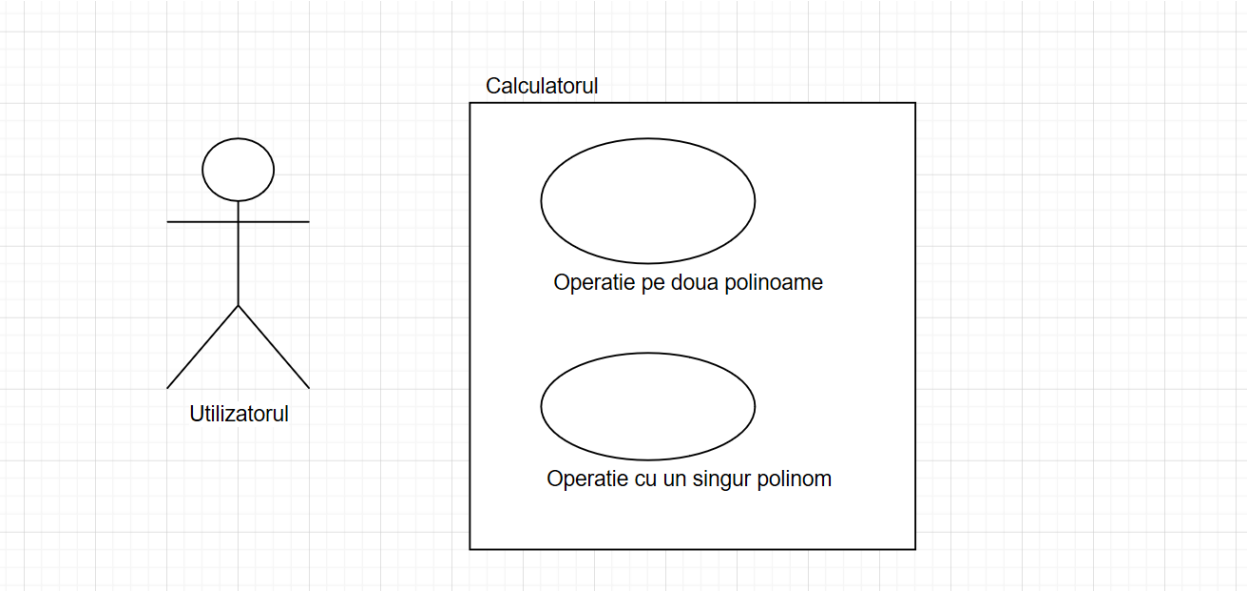
## Cerintele functionale:

In cadrul proiectului se disting urmatoarele cerinte functionale:

- Citirea unui polinom de forma +4X^6+7X^3-4X^1+5
- Validarea polinomului utilizand expresii regulate
- Efectuarea operatiei de adunare a doua polinoame
- Efectuarea operatiei de scadere a doua polinoame
- Efectuarea operatiei de inmultire a doua polinoame
- Efectuarea operatiei de impartire a doua polinoame
- Efectuarea operatiei de derivare a unui polinom
- Efectuarea operatiei de integrare a unui polinom

## Diagrame use case:

Pentru analiza cazurilor de functionare se folosesc diagramele use case. Astfel, actorul este utilizatorul care foloseste calculatorul. Cu aceasta aplicatie se pot face doua tipuri de operatii: operatii cu doua polinoame si operatii cu un singur polinom.




Primul caz: Operatie cu doua polinoame

Actorul principal: Utilizatorul

Scenariul cu succes:

1. Utilizatorul introduce primul polinom
2. Utilizatorul introduce si cel de-al doilea polinom
3. Se selecteaza operatia dorita: adunare, scadere, inmultire, impartire
4. Polinoamele introduse sunt valide
5. Se afiseaza rezultatul operatiei

Scenariu alternativ:

a) Unul dintre polinoamele introduse sau ambele sunt invalide
    1. Se afiseaza un mesaj de eroare in care este specificat care dintre polinoame este invalid
    2. Nu se afiseaza niciun rezultat
    3. Utilizatorul se reintoarce la pasul 1 sau 2
b) Al doilea polinom este 0 si operatia selectata a fost impartirea
    1. Se afiseaza mesaj de eroare care spune ca impartirea la 0 nu este permisa
    2. Niciun rezultat nu este afisat
    3. Utilizatorul se reintoarce la pasul 2

Al doilea caz: Operatie cu un singur polinom

Actorul principal: Utilizatorul

Scenariul cu succes:

1. Utilizatorul introduce un polinom in prima casuta text
2. Se selecteaza operatia dorita: derivare sau integrare
3. Polinomul introdus este valid
4. Se afiseaza rezultatul

Scenariu alternativ:

a) Polinomul introdus nu este valid
   1. Se afiseaza un mesaj de eroare care precizeaza ca polinomul nu este valid
   2. Nu este afisat niciun raspuns
   3. Utilizatorul revine la pasul 1

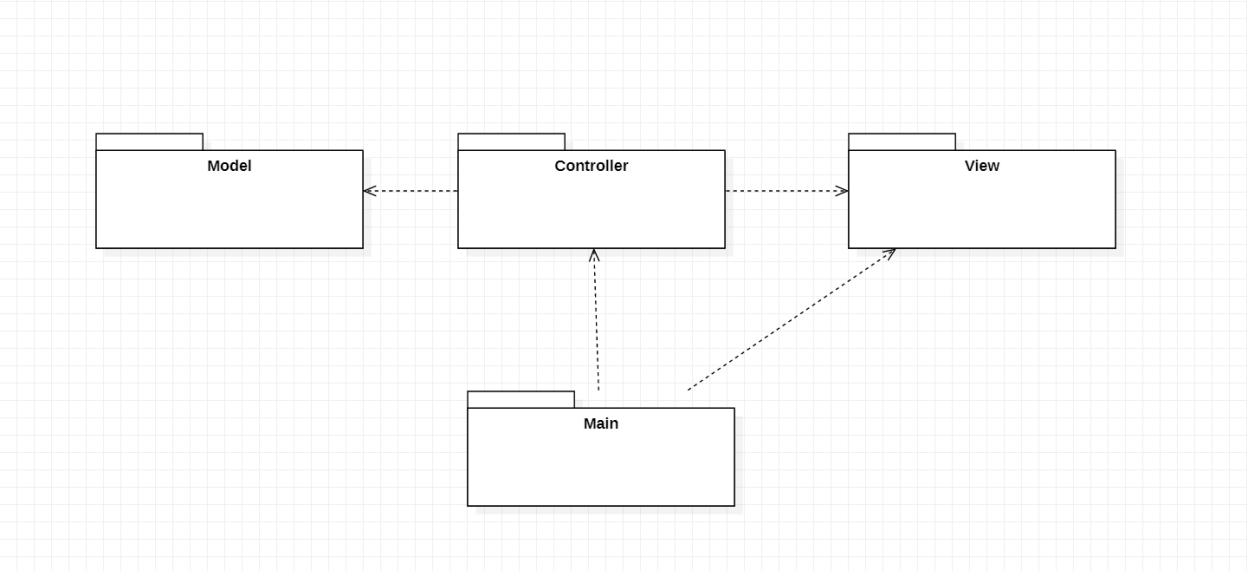
# 3 Proiectare

Codul este impartit in patru pachete:

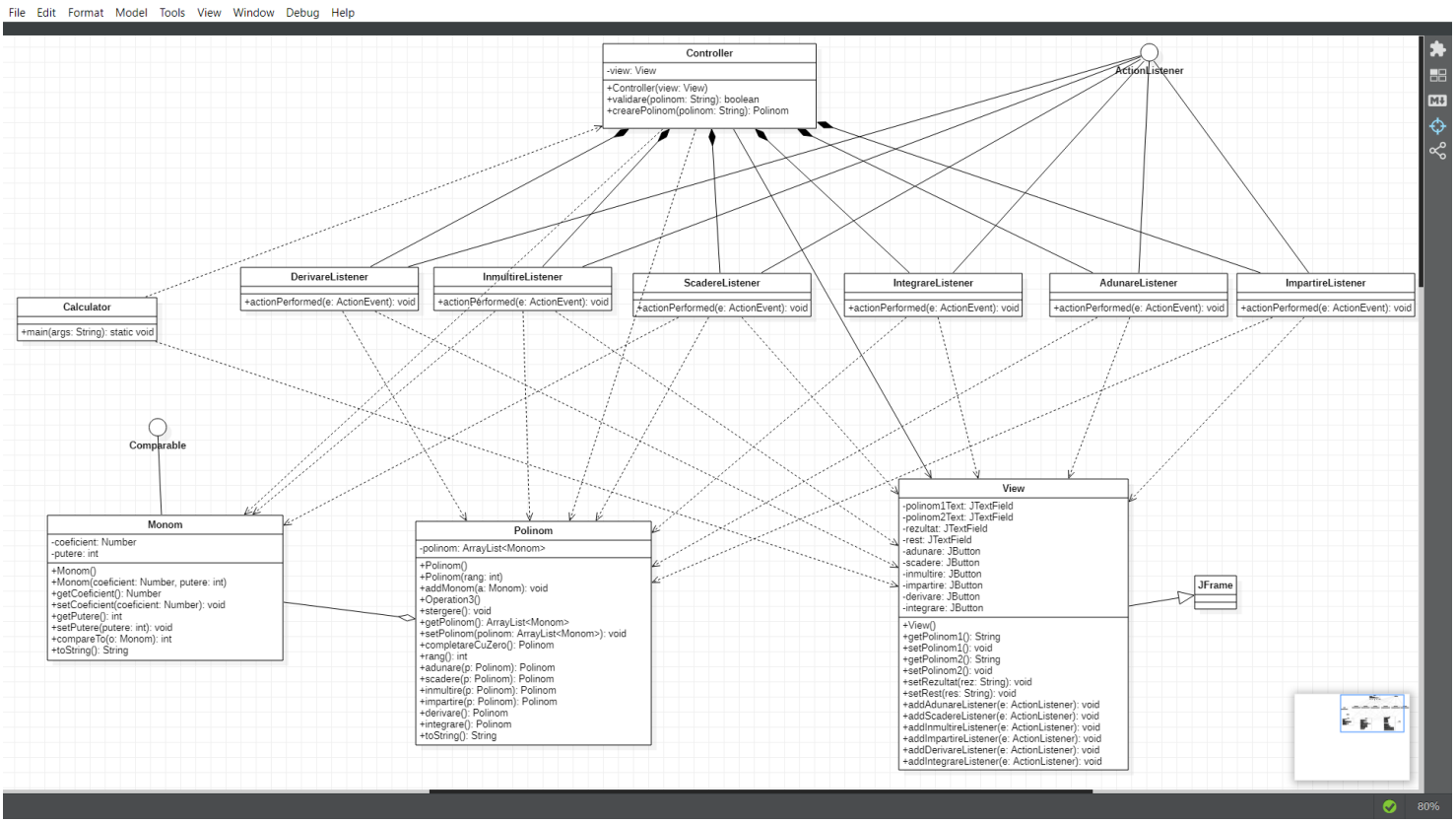
- Model
- View
- Controller
- Main

Astfel, codul este structurat pe modelul arhitecturii Model View Controller. In pachetul Model se regasesc clasele pentru monom si polinom si sunt implementate operatiile aritmetice. In pachetul View se afla codul pentru implementarea interfetei grafice. Pachetul Controller este cel in care se realizeaza legatura dintre model si vedere preluandu-se datele de intrare furnizate de utilizator prin intermediul interfetei grafice si executandu-se operatiile asupra lor. In pachetul main se gaseste clasa Calculator ce contine metoda main.

## Diagrama de pachete:



# Diagrama de clase:




Dupa cum se poate observa din diagrama UML de clase, proiectul contine 11 clase impartite in cele patru pachete.

## Pachetul View:

Acest pachet contine doar o singura clasa, si anume clasa View. Aceasta clasa realizeaza interfata grafica.

## Pachetul Model:

Acest pachet este format din doua clase: Monom si Polinom. Aceste clase incapsuleaza structurile de date folosite.

Clasa Monom defineste structura unui monom acesta fiind format dintr-un coeficient si un exponent.

Clasa Polinom este principala clasa a modelului. Aceasta are ca variabila instanta un sir de monoame care definesc polinomul. Tot in aceasta clasa sunt implementate si metodele pentru operatiile de adunare, scadere, inmultire, impartire, derivare si integrare.

## Pachetul Controller

Acest pachet este format de asemena dintr-o clasa numita Controller. Aceasta clasa mai contine sase clase interne. In clasa Controller se realizeaza legatura dintre vedere si model. Astfel, in aceasta clasa sunt preluate datele de intrare, efectuate operatiile asupra lor si afisate la final.

## Pachetul Main

Acest pachet contine clasa Calculator, clasa care are doar metoda main. In aceasta metoda se instantiaza un obiect al clasei View si un obiect al calsei Controler.

# 4 Implementare

## Clasa Monom

Variabile instanta:

Aceasta clasa are doua variabile instanta. Prima variabila este de tip int si semnifica puterea unui monom. Coeficientii monoamelor ar trebui sa fie intregi dar deoarece aplicatia trebuie sa implementeze si operatiile de impartire si de integrare variabila corespunzatoare coeficientului este de tip Number.

Constructori:

Aceasat clasa are implementati doi constructori. Primul este constructorul fara parametri iar cel de-al doilea este un constructor ce primeste un parametru de tip Number pentru coeficient si inca un parametru de tip int corespunzator puteri.

Metode:

Clasa are metode de getters si setters pentru cele doua variabile instanta. De asemenea, clasa mai dispune de o metoda toString pentru a converti un monom sub forma unui string. Aceasta clasa implementeaza interfata Comparable astfel incat trebuie sa ofere o implementare pentru metoda compareTo, metoda utilizata pentru a sorta monoamele in ordine descrescatoare conform puterilor.

## Clasa Polinom:

Variabile instanta:

Un polinom este format dintr-un sir de monoame astfel incat avem ca variabila un ArrayList de monoame.

Constructori:

Clasa polinom are implementati doi constructori. Primul constructor este cel fara parametri iar celalalt este un constructor cu un parametru de tip int. Cel de-al doilea constructor initializeaza un polinom de rang corespunzator cu valoarea parametrului primit avand toti coeficientii 0.

Metode:

In aceasta clasa avem metodele getPolinom si setPolinom pentru a obtine respectiv pentru a seta sirul de monoame.

Metoda addMonom care primeste ca parametru un obiect de tip Monom are rolul de a adauga un nou monom la sirul de monoame. Metoda stergere sterge primul monom din lista

Metoda rang returneaza rangul unui polinom.

Metoda completareCuZero completeaza termenii lipsa a unui polinom setandu-le coeficientii la 0.

De asemenea, exista si metoda toString pentru a afisa un polinom sub forma unui string.

In clasa Polinom sunt implementate si metodele pentru operatiile aritmetice de adunare, scadere, inmultire, impartire, derivare si integrare.

Metoda de adunare:

Metoda primeste ca argument un obiect de tip Polinom si returneaza tot un obiect de tip Polinom. La inceput se aduc ambele polinoame la acelasi grad determinandu-se care dintre ele are gradul cel mai mare si adaugandu-se la celalalt polinom un termen de acest grad. Dupa aceea se face completarea cu zero pentru amandoua polinoamele astfel incat sa nu lipseasca niciun termen. Avand polinoamele completate, se parcurg iar atunci cand doua monoame au puterea egala se creaza un nou monom cu aceeasi putere si cu coeficientul suma dintre coeficientii celor doua monoame. Apoi se sorteaza polinomul obtinut descrescator dupa puteri. Dupa se declara un nou polinom in care se adauga doar monoamele care au coeficientul diferit de 0. In cazul in care toate monoamele au coeficientul 0 se adauga un termen care sa aiba coeficientul 0 si puterea 0. La final se returneaza polinomul ce contine rezultatul adunarii.

Metoda de scadere este implementata asemanator cu cea de adunare.

Metoda de inmultire:

Metoda primeste ca argument un obiect de tip Polinom si returneaza tot un Polinom. La inceput se parcurg polinoamele si pentru fiecare doua monoame se creeaza un nou monom avand puterea ca suma dintre cele doua puteri si coeficientul ca inmultirea dintre cele doi coeficienti ce se adauga la un nou polinom rez. La pasul urmator se declara un nou polinom, pol_rez, avand gradul egal cu suma gradelor celor doua polinoame si toti coeficientii egali cu 0. Se parcurge polinomul rez si se aduna la coeficientii lui pol_rez coeficientii acestuia atunci cand puteriile sunt egale. Urmeaza sortarea descrescztoare a listei de monoame din polinomul pol_rez. Dupa aceea in polinomul rez2 se adauga doar monoamele a caror coeficienti sunt diferiti de 0. Daca polinomul rezultat este gol atunci se mai adauga un termen care are coeficientul 0 si puterea tot 0.

Metoda de impartire:

Aceasta metoda primeste ca argument un obiect de tip Polinom si returneaza un ArrayList de polinoame deoarece in urma impartirii se afiseaza atat catul cat si restul. La inceput se declara un ArrayList de polinoame numit poilnome, un polinom a, un polinom cat si doua polinoame deimp si imp care se egaleaza cu polinoamele care trebuiesc sa fie impartite. Dupa aceea cat timp deimpartitorul are gradul mai mare sau egal cu impartitorul si impartitorul este diferit de 0 se repeat urmatorii pasi: se extrage primul element din deimpartit, deimpMonom, si din impartitor, impMonom, se creeaza un nou monom r a carui coeficient este egal cu coeficientul lui deimpMonom impartit la coeficientul lui impMonom iar puterea este diferenta puterilor lor. Monomul r se adauga la polinomul cat si la polinomul a. In polinomul rest este salvata inmultirea impartitorului cu polinomul a. Dupa aceea stergem monomul din a iar deimpartitul devine egal cu diferenta dintre el insusi si polinomul rest. La iesirea din bucla while se verifica daca catul este gol, caz in care se adauga un monom cu puterea 0 si coeficientul nul. Se sorteaza catul si deimpartitul,acum devenit rest, si se adauga in ArrayList-ul polinoame care urmeaza a fi returnat.

Metoda de derivare:

Aceasta metoda nu primeste niciun parametru dar returneaza un obiect de tip Polinom. Se parcurge tot polinomul iar pentru monoamele a caror putere este diferita de 0 coeficientul se inmulteste cu puterea iar puterea scade cu 1. La final se verifica daca polinomul rezultat este nul, caz in care se mai adauga un monom cu coeficientul nul si puterea 0.

Metoda de integrare:

Metoda nu primeste niciun parametru dar returneaza un obiect de tip Polinom. Se parcurge polinomul iar pentru fiecare monom coeficientul se imparte cu puterea plus unu iar puterea se incrementeaza cu 1. Inainte de returnarea noului polinom, acesta se sorteaza.

# Clasa View:

Variabile instanta:

Cate un text field pentru introducerea celor doua polinoame si pentru afisarea rezultatului si al restului pentru impartire

Sase butoane, fieacre corespunzand uneia dintre operatiile aritmetice pe care le poate realiza aceasta aplicatie.

Constructor:

Clasa are un singur constructor fara parametrii. In acest constructor sunt instantiate cele doua panel-uri pe care sunt adaugate butoanele si casutele text. Aceste panel-uri sunt adaugate la randul lor pe view care este setat sa devina vizibil.

Metode:

Metoda getPolinom1 care returneaza textul din casuta corespunzatoare primului polinom. Metoda setPolinom1 goleste casuta primului polinom.

Asemanator functioneaza si metodele getPolinom2 si setPolinom2.

Metododele setRezultat si setRest primesc ca argument un string pe care il afiseaza in casuta text pentru rezultat respectiv pentru rest.

Metodele addAdunareListener, addScadereListener, addInmultireListener, addImpartireListener, addDerivareListener, addIntegrareListener, primesc ca argument un obiect de tipul ActionListener si adauga un listener pentru fiecare buton corespunzator unei operatii.

# Clasa Controller

Variabile instanta:

Clasa contine o singura variabila instanta de tipul View.

Constructor

Clasa are doar un constructor care primeste un argument de tipul View. In constructor sunt apelate metodele addAdunareListener, addScadereListener, addInmultireListener, addImpartireListener, addDerivareListener, addIntegrareListener din view.

Metode:

Metoda validare primeste ca argument un string si returneaza un Boolean. Aceasta verifica daca un polinom introdus de utilizator este valid.

Metoda creare Polinom primeste ca argument un string si returneaza un polinom. Aceasta metoda extrage coeficientii si puterile dintr-un string creand astfel un obiect de tipul polinom.

Clasa Controller are sase clase interne: AdunareListener, ScadereListener, InmultireListener, ImpartireListener, DerivareListener, IntegrareListener. Toate aceste clase au cate o singura metoda numita actionPerformed.

Clasa AdunareListener: In stringurile polinom1 si polinom2 se extrag datele introduse de utilizator. Se verifica validitatea acestora iar daca ambele sunt corecte se executa operatia de adunare si se afiseaza rezultatul. Daca cel putin unul dintre polinoame este invalid se afiseaza un mesaj de eroare.

Clasa ScadereListener si InmultireListener functioneaza asemanator cu AdunareListener.

Clasa ImpartireListener mai verifica suplimentar daca primul polinom este egal cu 0 caz in care rezultatul este 0. De asemenea se verifica daca polinomul al doilea este 0 caz in care se afiseaza mesaj de eroare care spune ca nu este posibila impartirea cu 0.

Clasa DerivareListener: Se extrag datele din prima casuta text si se verifica validitatea acestora. Daca sunt valide se executa operatia si se afiseaza rezultatul, altfel se afiseaza un mesaj de eroare.

Clasa IntegrareListener functioneaza asemanator cu DerivareListener.

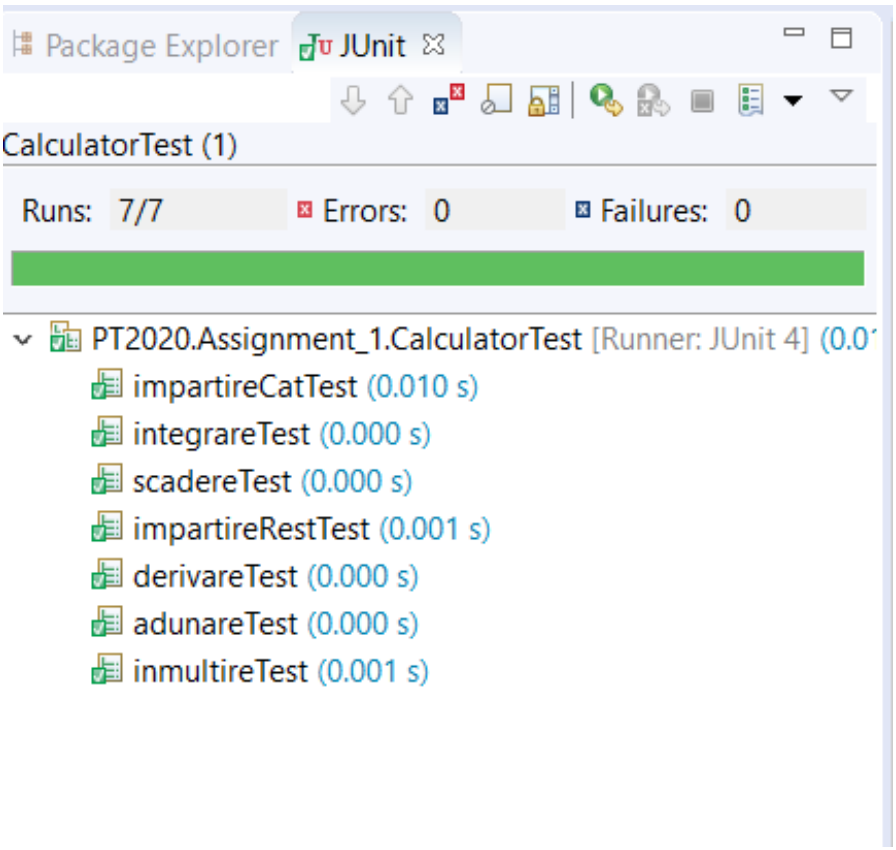
# 5 Rezultate

Pentru testarea corectitudinii operatiilor aritmetice s-a folosit Junit. Astfel, a fost scrisa o clasa de testare numita CalculatorTest. Aceasta clasa contine metodele adunareTest, scadereTest, inmultireTest, impartireCatTest, impartireRestTest, derivareTest, integrareTest care verifica corectitudinea fiecarei operatii.

```
public class CalculatorTest {

	@Test
	public void adunareTest() {
		Polinom p1 = new Polinom();
		Polinom p2 = new Polinom();
		p1.addMonom(new Monom(2,3));
		p1.addMonom(new Monom(1,1));
		p2.addMonom(new Monom(3,3));
		p2.addMonom(new Monom(-4,2));
		p2.addMonom(new Monom(5,0));
		String suma = "+5X^3-4X^2+1X^1+5";

		assertEquals(suma, p1.adunare(p2).toString());
	}

	@Test
	public void scadereTest() {
		Polinom p1 = new Polinom();
		Polinom p2 = new Polinom();
		p1.addMonom(new Monom(4,6));
		p1.addMonom(new Monom(-2,4));
		p1.addMonom(new Monom(8,0));
		p2.addMonom(new Monom(3,6));
		p2.addMonom(new Monom(-3,4));
		p2.addMonom(new Monom(5,0));
		String diferenta = "+1.0X^6+1.0X^4+3.0";
		assertEquals(diferenta, p1.scadere(p2).toString());

	}

	@Test
	public void inmultireTest() {
		Polinom p1 = new Polinom();
		Polinom p2 = new Polinom();
		p1.addMonom(new Monom(1,2));
		p1.addMonom(new Monom(1,1));
```

```
        p2.addMonom(new Monom(3,3));
        p2.addMonom(new Monom(5,2));
        String rezultat = "+3.0X^5+8.0X^4+5.0X^3";
        assertEquals(rezultat, p1.inmultire(p2).toString());
}

@Test
public void impartireCatTest() {
        Polinom p1 = new Polinom();
        Polinom p2 = new Polinom();
        p1.addMonom(new Monom(4,4));
        p1.addMonom(new Monom(2,2));
        p2.addMonom(new Monom(2,1));
        String rezultat = "+2.0X^3+1.0X^1";
        ArrayList<Polinom> polinoame;
        polinoame=p1.impartire(p2);
        assertEquals(rezultat, polinoame.get(0).toString());
}

@Test
public void impartireRestTest() {
        Polinom p1 = new Polinom();
        Polinom p2 = new Polinom();
        p1.addMonom(new Monom(4,4));
        p1.addMonom(new Monom(2,0));
        p2.addMonom(new Monom(2,5));
        String rezultat = "+4X^4+2";
        ArrayList<Polinom> polinoame;
        polinoame=p1.impartire(p2);
        assertEquals(rezultat, polinoame.get(1).toString());
}

@Test
public void derivareTest() {
        Polinom p1 = new Polinom();
        p1.addMonom(new Monom(3,5));
        p1.addMonom(new Monom(2,3));
        p1.addMonom(new Monom(1,2));
        p1.addMonom(new Monom(5,0));
        String rezultat = "+15X^4+6X^2+2X^1";
        assertEquals(rezultat, p1.derivare().toString());

}

@Test
public void integrareTest() {
        Polinom p1 = new Polinom();
        p1.addMonom(new Monom(10,4));
        p1.addMonom(new Monom(9,2));
        p1.addMonom(new Monom(5,0));
        String rezultat = "+2.0X^5+3.0X^3+5.0X^1";
```

```
            assertEquals(rezultat, p1.integrare().toString());
        }

}
```



# 6 Concluzii

In concluzie acest calculator implementeaza operatiile de adunare, scadere, inmultire, impartire, derivare si integrare.

Aceasta aplicatie poate fi imbunatatita prin adaugarea de noi operatii cum ar fi gasirea radacinilor. De asemenea s-ar putea extinde polinoamele la mai multe variabile.

# 7 Bibliografie

http://www.mkyong.com/tutorials/junit-tutorials/