DOCUMENTATIE

TEMA *1*

NUME STUDENT:Deac Cristian Dumitru

GRUPA: 30229

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)-5

[3. Proiectare 5](#_Toc95297887)-6

[4. Implementare 6-9](#_Toc95297888)

[5. Rezultate 1](#_Toc95297889)0

[6. Concluzii 1](#_Toc95297890)0

[7. Bibliografie 1](#_Toc95297891)1

# Obiectivul temei

**Main Objective**

Obiectivul temei este acela de a proiecta si implementa un calculator de polinoame cu o interfata grafica dedicata prin care utilizatorul sa poata introduce polinoame, iar mai apoi sa poata selecta o operatie care doreste sa fie efectuata pe acestea, rezultatul urmand sa fie afisat pe ecran.

**Sub-objectives**

Obiectivele secundare ale temei sunt:

* Sa se analizeze problema si sa se identifice cerintele (acest sub obiectiv va fi prezentat mai in amanunt in capitolul 2).
* Proiectarea calculatorului de polinoame (acest sub obiectiv va fi prezentat mai in amanunt in capitolul 3)
* Implementarea calculatorului de polinoame (acest sub obiectiv va fi prezentat mai in amanunt in capitolul 4)
* Testarea calculatorului de polinoame (acest sub obiectiv va fi prezentat mai in amanunt in capitolul 5)

# Analiza problemei, modelare, scenarii, cazuri de utilizare

**Cerinte functionale:**

* Calculatorul de polinoame ar trebui sa permita userului sa introduca polinoame .
* Calculatorul de polinoame ar trebui sa permita userului sa selecteze operatia matematica pe care acesta doreste sa o execute.
* Calculatorul de polinoame ar trebui sa poata aduna doua polinoame.
* Calculatorul de polinoame ar trebui sa poata scadea doua polinoame.
* Calculatorul de polinoame ar trebui sa poata inmulti doua polinoame.
* Calculatorul de polinoame ar trebui sa poata imparti doua polinoame.
* Calculatorul de polinoame ar trebui sa poata deriva un polinom.
* Calculatorul de polinoame ar trebui sa poata integra un polinom.
* Calculatorul de polinoame ar trebui sa afiseze rezultatul operatiei matematice in interfata.

**Cerinte non-functionale:**

* Calculatorul de polinoame ar trebui sa fie intuitiv si usor de folosit de orice user.

**Use Case:** Adunarea polinoamelor

**Primary Actor :** user

**Main Success Scenario:**

1. Userul introduce 2 polinoame in interfata grafica.
2. Userul selecteaza operatia de adunare .
3. Calculatorul de polinoame va executa operatia de adunare a celor 2 polinoame si va afisa rezultatul in interfata grafica.

**Use Case:** Scaderea polinoamelor

**Primary Actor :** user

**Main Success Scenario:**

1. Userul introduce 2 polinoame in interfata grafica.
2. Userul selecteaza operatia de scadere .
3. Calculatorul de polinoame va executa operatia de scadere a celor 2 polinoame si va afisa rezultatul in interfata grafica.

**Use Case:** Inmultirea polinoamelor

**Primary Actor :** user

**Main Success Scenario:**

1. Userul introduce 2 polinoame in interfata grafica.
2. Userul selecteaza operatia de inmultire .
3. Calculatorul de polinoame va executa operatia de inmultire a celor 2 polinoame si va afisa rezultatul in interfata grafica.

**Use Case:** Impartirea polinoamelor

**Primary Actor :** user

**Main Success Scenario:**

1. Userul introduce 2 polinoame in interfata grafica.
2. Userul selecteaza operatia de impartire .
3. Calculatorul de polinoame va executa operatia de impartire a celor 2 polinoame si va afisa rezultatul in interfata grafica.

**Use Case:** Derivarea polinomului

**Primary Actor :** user

**Main Success Scenario:**

1. Userul introduce 1 polinom in prima casuta din interfata grafica in textFieldul din dreptul labelului cu textul “First Polynomial”.
2. Userul selecteaza operatia de derivare .
3. Calculatorul de polinoame va executa operatia de derivare a polinomului si va afisa rezultatul in interfata grafica.

**Use Case:** Integrarea polinomului

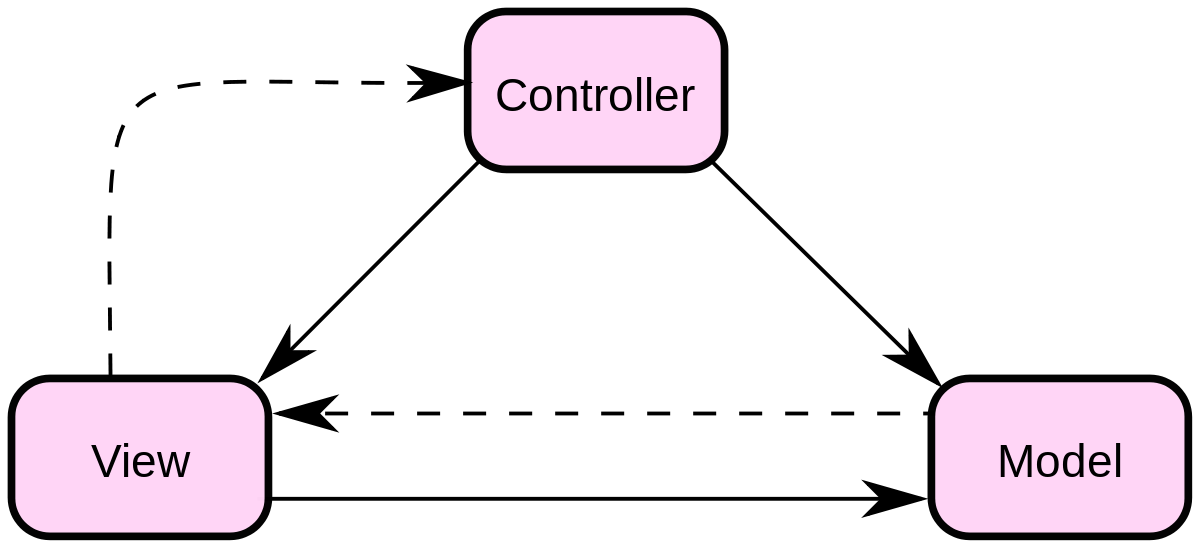
**Primary Actor :** user

**Main Success Scenario:**

1. Userul introduce 1 polinom in prima casuta din interfata grafica in textFieldul din dreptul labelului cu textul “First Polynomial”.
2. Userul selecteaza operatia de integrare .
3. Calculatorul de polinoame va executa operatia de integrare a polinomului si va afisa rezultatul in interfata grafica.

# Proiectare

Proiectare aplicatiei a fost realizata utilizand arhitectura de tip model-view-controller, astfel am impartit programul in pachete ca in diagrama de mai jos:



Fiecare pachet are un rol aparte in crearea acestui program, iar acestea sunt:

1. Pachetul Controller

Pachetul Controller actioneaza atat asupra modelului cat si asupra viewului. Acesta controleaza datele care intra in obiectele din model si updateaza viewul de fiecare daca cand datele se schimba.

1. Pachetul View

Pachetul View reprezinta vizualizarea datelor pe care modelul le contine, acesta defineste interfata grafica.

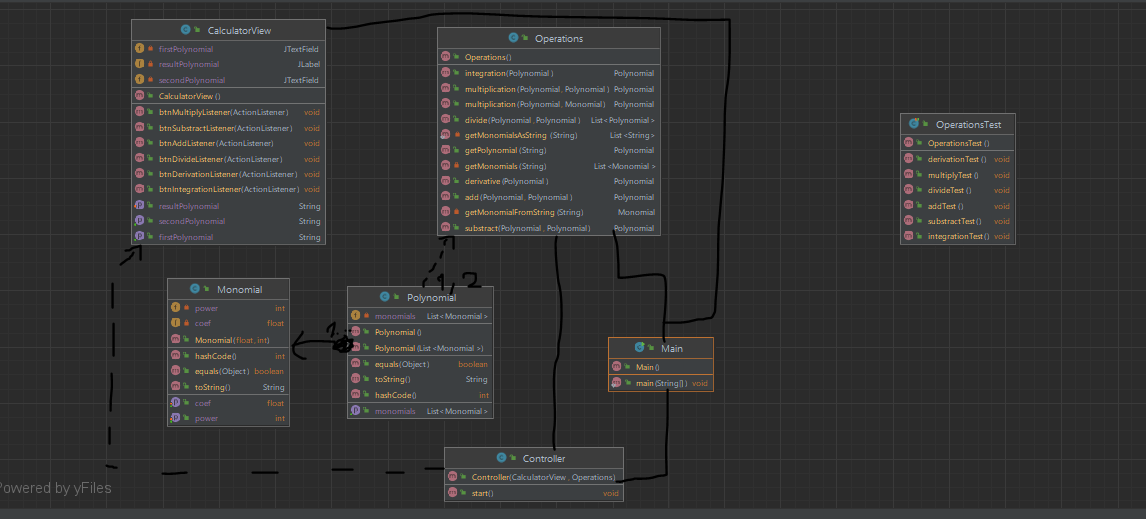
1. Pachetul Model

Pachetul Model contine clasele necesare pentru crearea si modificarea polinoamelor.

Acesta are la baza clasele Monomial si Polynomial care ajuta la stocarea datelor intr un obiect de tip Polynomial.

Un monom contine puterea si coeficientul, iar polinomul este o lista de monoame. Polinomul va retine monoamele in lista in ordine descrescatoare in functie de putere ( acest lucru este posibil deoarece metodele create pentru stocarea datelor in polinoame asigura asta), deoarece asta ne va face viata mai usoara pentru implementare operatiilor.

Diagrama UML:



# Implementare

# **Clasa Controller**

Clasa Controller face posibila interactiunea utilizatorului cu interfata grafica, astfel aceasta clasa primeste din Clasa CalculatorView butoanele si polinoamele introduse. Clasa Controller va stoca polinoamele introduse in obiecte de tip Polynomial si va da un sens fiecarui buton. In urma operatiunilor executate in aceasta clasa imediat ce utilizatorul va apasa un buton polinoamele vor fi stocate iar operatia dorita se va executa.

Singura metoda din aceasta clasa este insusi constructorul clasei care va primi ceea ce are nevoie pentru a putea implementa efectul dorit la apasarea butoanelor .

public Controller(CalculatorView calculatorView, Operations operations)

**Clasa CalculatorView**

Clasa CalculatorView (Interfata Grafica) este alcatuita din:

* 4 labeluri(Un label pentru a afisa un titlu in interfata grafica si anume ,,Polynomial Calculator,, , 2 labeluri care ne conduc spre casutele unde trebuie introduse polinoamele si 2 labeluri care se folosesc pentru afisarea rezultatului, unul va indica unde este va fi afisat rezultatul iar celalalt fiind rezultatul insusi.
* 2 textFielduri, in fiecare text field urmand sa fie introdus cate un polinom care mai apoi va fi preluat de controller utilizand metodele de get.
* 6 butoane, fiecare buton reprezinta o operatie in interfata

Metodele din interfata grafica sunt doar gettere sau setere care vor fi utilizate in Controller pentru a stoca polinoamele si a da un rol butoanelor.

**Clasa Monomial**

Clasa Monomial contine 2 variabile instanta, un int pentru puterea monomului si un float pentru coeficientul acestuia

private float coef;  
private int power;

Aceasta clasa nu contine decat metode obisnuite, conventionale, cum ar fi getterele si setrerele pentru coeficient si putere, si metodele toString() si equals().

**Clasa Polynomial**

Clasa polynomial contine o lista de monoame numita monomials

private List<Monomial> monomials = new ArrayList<>();

Aceasta clasa la fel ca si clasa Monomial nu contine decat metodele obisnuite, conventionale, un getter si un setter pentru lista de monoame si metodele toString() (folosita pentru afisarea polinomului) si equals().

**Clasa Operations**

private static List<String> getMonomialsAsString(String polynomial)

Metoda getMonomialAsString primeste ca parametru polinomul sub forma de string si returneaza o lista de stringuri,in aceasta lista de stringuri fiecare element reprezentand un monom,acest lucru este posibil cu ajutorul regexului -?\\d\*(x(\\^\\d+)\*)\* care identifica toate monoamele din polinomul dat ca string, acesta avand un pattern anume pentru a putea identifica fiecare monom. Folosindu-ne de un while care ne ajuta sa ne oprim cand nu mai gasim monoame stocam in lista toate monoamele pe rand pe care le gasim in stringul de polinom.

private Monomial getMonomialFromString(String monomialString)

Metoda getMonomialFromString primeste ca parametru un String care reprezinta un monom si returneaza un obiect de tipul Monomial.

Rolul acestei metode este acela de a obtine coeficientul si puterea dintr un string de monom si a le stoca intr-un monom.

In aceasta metoda folosim metoda split din clasa String care primeste ca parametru regexul x\\^\* si returneaza intr-un array de stringuri coeficientul si puterea care este gasita datorita regexului, aceasta putere urmand sa fie stocata cu ajutorul constructorului din clasa Monomial intr-un monom.

private List<Monomial> getMonomials(String polynomial)

Metoda getMonomials primeste ca parametru stringul de polinom si returneaza lista de monoame.

Aceasta metoda utilizeaza metodele de mai sus, practic metoda getMonomialsAsString este utilizata pentru a rupe stringul de polinoame intr o lista de stringuri de monoame cum am mentionat mai sus, apoi utilizam un for each pentru a lua fiecare string din lsita de monoame si il adaugam in lista de monoame dupa ce aplicam getMonomialFromString pe acesta. Astfel am obtinut lista noastra de monoame care este defapt polinomul nostru care va fi introdusa in polinom direct utilizand metoda de mai jos:

public Polynomial getPolynomial(String polynomial) {  
 return new Polynomial(getMonomials(polynomial));  
}

public Polynomial add(Polynomial p1, Polynomial p2)

Metoda add primeste ca parametrii2 polinoame si returneaza tot un polinom care este rezultatul adunarii dintre acestea.

Pentru a realiza adunarea celor 2 polinoame facem o interclasare intre cele doua astfel daca gasim 2 monoame cu puteri egale le adunam coeficientii, iar daca monoamele nu au puteri egale adaugam in lista monomul cu puterea mai mare pentru a mentine ordinea in polinomul rezultat.

public Polynomial substract(Polynomial p1, Polynomial p2)

Metoda substract primeste ca parametrii 2 polinoame si returneaza tot un polinom care este rezultatul scaderii dintre acestea.

Pentru a realiza adunarea celor 2 polinoame facem o interclasare intre cele doua astfel daca gasim 2 monoame cu puteri egale le scadem coeficientii, iar daca monoamele nu au puteri egale adaugam in lista monomul cu puterea mai mare pentru a mentine ordinea in polinomul rezultat.

public Polynomial multiplication(Polynomial p1, Polynomial p2)

public Polynomial multiplication(Polynomial p1, Monomial m2)

Metoda multiplication primeste ca parametrii 2 polinoame sau un polinom si un monom si returneaza rezultatul inmultirii dintre acestia.

In prima metoda folosim 2 for eachuri pentru a parcurge fiecare termen din primul polinom cu tot polinomul al doilea, astfel la fiecare parcurgere inmultim coeficientii intre ei si adunam puterile,

Mai apoi pentru ca rezultatul operatiei de mai sus ne va da probabil mai multe puteri de aceleasi fel in lista noastra de monoame o sa parcurgem lista de monoame si vom verifica daca avem puteri de acelasi fel si daca acestea exista le vom aduna coeficientii.

In final vom sorta descrescator in functie de puterea monomului pentru a obtine rezultatul sub forma de polinom.

Iar in cea de a doua metoda in care este primit un monom ca parametru in loc de cel de al doilea polinom se realizeaza acelasi lucru doar ca nu a mai fost nevoie de 2 for eachuri, ajungand un for pentru parcurgerea polinomului si inmultirea acestuia cu monomul.

public Polynomial derivative(Polynomial p)

Metoda derivative primeste ca parametru un polinom si returneaza tot un polinom in urma operatiei de derivare

In aceasta metoda parcurge polinomul cu un for each iar la fiecare iteratie coeficientul va fi inmultit cu puterea ( puterea != 0) iar puterea va scadea cu 1.

public Polynomial integration(Polynomial p)

Metoda integration primeste ca parametru un polinom si returneaza tot un polinom in urma operatiei de integrare

In aceasta metoda parcurgem polinomul cu un for each iar la fiecare iteratie coeficientul se va imparti cu puterea adunata cu 1, iar puterea va creste cu 1, daca puterea este 0 atunci monomul va primi puterea 1 si isi pastreaza coeficientul.

public List<Polynomial> divide(Polynomial p1, Polynomial p2)

Metoda divide primeste ca parametru 2 polinoame si returneaza tot un polinom in urma operatiei de impartire.

Aceasta metoda verifica daca puterea primului polinom este mai mare decat puterea celui de al doilea polinom urmand sa ii imparta coeficientii si sa scada puterea intre monoamele cu gradul cel mai mare din dreptul fiecarui polinom, mai apoi se va inmulti monomul rezultatul cu monomul la care se realizeaza impartirea, rezultatul acestei operatii urmand sa fie scazut din primul monom. Acest lucru se intampla intr un while cat timp gradul primului polinom este mai mare sau egal cu gradul celui de al 2 lea polinom.

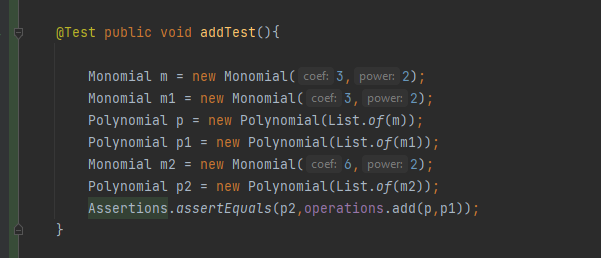
In clasa Main ne vom crea cate o instanta de view, de clasa de operatii si le vom trimite in constructorul de controller, urmand ca view sa fie facut vizibil.

# Rezultate

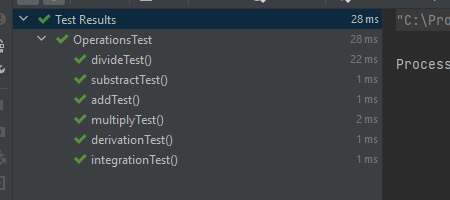
Pentru testarea unitara am utilizat JUnit versiunea 5.7.2

Am testat matematic fiecare operatie creand cate un test separat pentru fiecare

Exemplu cum arata un test:



Functionarea testelor:



# Concluzii

In concluzie, tema aceasta m-a ajutat sa ma obisnuiesc cu un nou IDE (Intellij) pe care il folosesc pentru prima data, totodata am invatat sa stapanesc mai bine lucrul cu modelul arhitectural model-view-controller.

Testarea cu JUnit tot asa a fost noua pentru mine, precum si regexul, astfel am invatat cum sa creez testari automate in java si cum sa ma folosesc de regex pentru a putea manipula stringurile si pentru a putea gasi un pattern in ele.

Proiectul poate fi dezvoltat in continuare, poate prin adaugarea unui buton de executare dupa apasarea butonului de operatie in caz ca nu se doreste executia directa a operatiei dupa apasarea butonului acesteia, sau prin implementare mai multor operatii.

# Bibliografie

<https://docs.oracle.com/javase/7/docs/api/java/lang/String.html>

<https://regex101.com/>

<https://www.w3schools.com/java/java_regex.asp>

<https://www.parasoft.com/blog/junit-tutorial-setting-up-writing-and-running-java-unit-tests/>