**Polynomial-Equation-Calculator**

**Savin Ioan-Catalin**

Contents

[1. Obiectivul temei 3](#_Toc509944993)

[2. Analiza problemei, asumptii, modelare, scenarii, cazuri de utilizare, erori 3](#_Toc509944994)

[2.1. Analiza problemei 3](#_Toc509944995)

[2.2. Asumpții 3](#_Toc509944996)

[2.3. Cazuri de utilizare 3](#_Toc509944997)

[3. Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, relații, packages, algoritmi, interfața utilizator, modul de tratare a erorilor) 4](#_Toc509944998)

[3.1. Decizii de proiectare 4](#_Toc509944999)

[3.2. Diagrame UML 4](#_Toc509945000)

[3.3. Proiectare clase 5](#_Toc509945001)

[4. Algoritmi 5](#_Toc509945002)

[5. Testare 6](#_Toc509945003)

[6. Concluzii 6](#_Toc509945004)

[7. Bibliografie 6](#_Toc509945005)

# Obiectivul temei

Propuneti, proiectati si implementati un sistem de procesare a polinoamelor de o singura variabila cu coeficienti intregi

Dezvoltarea unui calculator de polinoame care sa perimita:

* Citirea unui polinom de la tastatura sub forma 3x^3+2x^2-3x^1+2x^0
* Adunare, Scadere, Inmultire, Impartire, Derivare, Integrare
* Crearea claselor Polinom si Monom
* Testarea cu Junit Tests

# Analiza problemei, asumptii, modelare, scenarii, cazuri de utilizare, erori

## Analiza problemei

Problema presupune adunarea, scaderea, inmultirea, impartirea, derivarea si integrarea unor polinoame cu grad pozitiv si coeficienti intregi. .

## Asumpții

O asumpție o costituie faptul că polinomul este introdus corect, sub forma “ax^n+bx^n-1… zx^0” cu specificația că gradul polinomul nu poate fi negativ.

Asumptii

## Cazuri de utilizare

1. Operatia de adunare

* Se introduc cele doua polinoame
* La apasarea butonului “+”, se incarca citirea polinomului si afisarea rezulatului intr-un “JTextField”

1. Operatia de scadere

* Se introduc cele doua polinoame
* La apasarea butonului “-”, se incarca citirea polinomului si afisarea rezulatului intr-un “JTextField”

1. Operatia de inmultire

* Se introduc cele doua polinoame
* La apasarea butonului “\*”, se incarca citirea polinomului si afisarea rezulatului intr-un “JTextField”

1. Operatia de derivare

* Necesita doar un polinom
* La apasarea butonului “Derivata”, se incarca citirea polinomului si afisarea rezulatului intr-un “JTextField”

1. Operatia de integrare

* Necesita doar un polinom
* La apasarea butonului “Integrala”, se incarca citirea polinomului si afisarea rezulatului intr-un “JTextField”

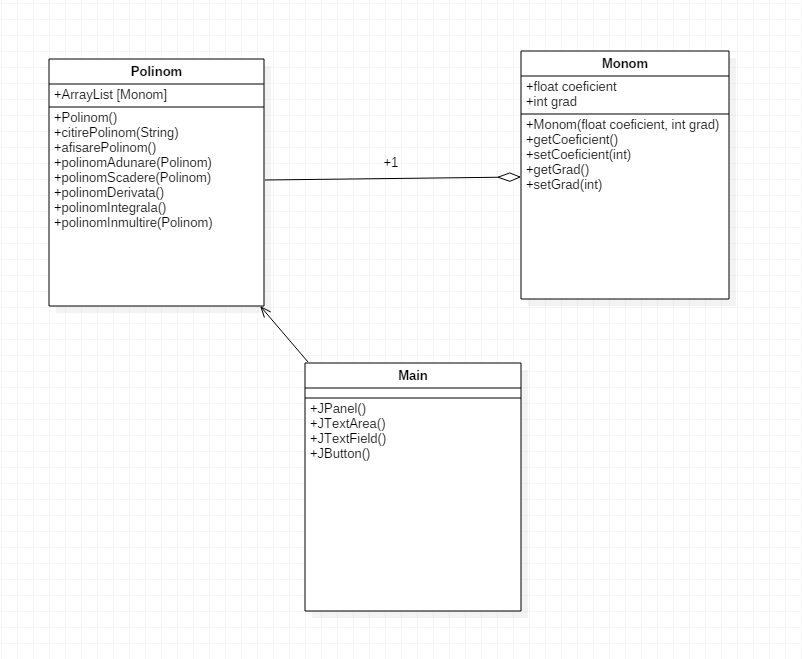
# Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, relații, packages, algoritmi, interfața utilizator, modul de tratare a erorilor)

## Decizii de proiectare

Proiectul este structurat in trei clase: Monom, Polinom si Main. Clasa “Main” este partea de interfata si clasa care face legatura cu restul claselor.

## Diagrame UML

Clasa Main se ocupa de realizarea interacțiuni dintre utilizator și aplicație, si facand legatura cu restul claselor. Unified Modeling Language (UML) este un limbaj standard pentru descrierea de modele si specificatii pentru software. Este folosita pentru reprezentarea vizuala a claselor si a interdependentelor, taxonomiei si a relatiilor de multiplicitate dintre ele. Diagramele de clasa sunt folosite si pentru reprezentarea concreta a unor instante de clasa, asadar obiecte, si a legaturilor dintre acestea.



# Proiectare clase

Organizarea proiectului are forma Model-View-Controller.

Clasa Polinom manipulează polinoamele asupra cărora se efectuează: citirea polinoamelor de la utilizator, scrierea acestora ca liste de monoame și returnează rezultatul operațiilor dorite.

Clasa Monom permite manipularea monoamelor din listele procesate de clasa Polinom prin operațiile implementate.

# Algoritmi

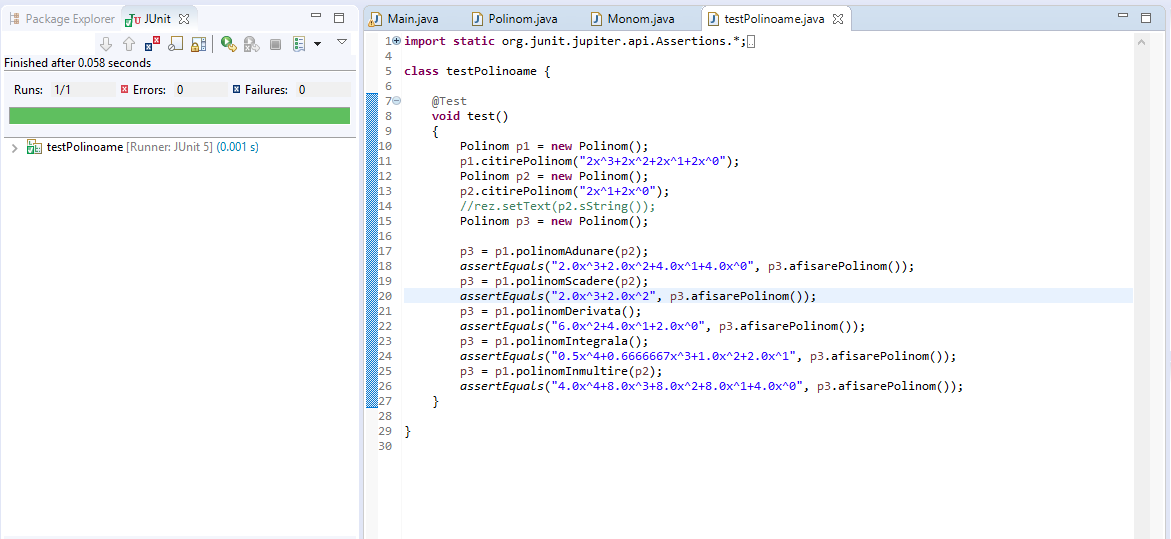
Algoritmul de adunare și scădere a polinoamelor

Algoritmul se bazează pe faptul că polinoamele conțin monoamele deja sortate după grad. Se folosesc 2 variabile care memoreaza starea in care se afla fiecare polinom, dacă mai există monoame de parcurs. Complexitatea algoritmului este liniara.

Algoritmul de înmulțire a polinoamelor

Algoritmul parcurge fiecare monom din p1 și îl înmulțește cu fiecare monom din p2, rezultatul fiind introdus în alt polinom. Daca exista in polinoml rezultat monom cu grad egal, atunci coeficinetii se vor aduna si va rezultat un alt polinom ordonat.

# Testare



Cu excepția unor mici cazuri când programul este forțat la în zona de input-uri, programul nu va rula corespunzător deoarece nu există parte de descompunere pentru eventualele string-uri aleatoare. În rest, atât rezultatele, cât și valorile se află în grila celor așteptate.

# Concluzii

Din această primă temă, mi-am dat seama, de importanța sepărării, la nivel conceptual, adescomunerii string-urilor fată de interpretarea lor de către clasele Polinom și Monom

# Bibliografie

<http://stackoverflow.com/>

<http://wikipedia.com/polynom>

[http://java2novice.com/java-collections-and-util](http://java2noveice.com/java-collections-and-util)