**Assignment1**

**POLYNOMIAL CALCULATOR**

**NUME : OSTAFIE STANCA**

**GRUPA: 30227**

**PROFESOR LABORATOR : DAN MITREA**

CUPRINS

[1. OBIECTIVUL TEMEI 3](#_Toc66881492)

[2. ANALIZA PROBLEMEI, MODELARE, SCENARII, CAZURI DE UTILIZARE 3](#_Toc66881493)

[3. PROIECTARE 4](#_Toc66881494)

[4. IMPLEMENTARE 6](#_Toc66881495)

[5. REZULTATE 7](#_Toc66881496)

[6. CONCLUZII 7](#_Toc66881497)

[7. BIBLIOGRAFIE 8](#_Toc66881498)

# 1. **OBIECTIVUL TEMEI**

Obiectivul principal al acestei teme este proiectarea si implementarea corecta a unui calculator destinat strict operatiilor pe polinoame. Acest calculator va permite utilizatorului sa insereze polinoame, sa aleaga o operatie si sa vizualizeze rezultatul dorit prin intermediul unei interfete grafice adecvate cerintei. Operatiile pe care le va putea efectua acest calculator sunt : adunare/scaderea/inmultirea/impartire a doua polinoame , derivare/integrare unui polinom la alegere.

Vom amintii si urmatoarele obiective secundare ale temei:

* Implementarea clasei “Monomial” ne permite sa definim caracteristicile specifice fiecarui termen din polinom.
* Implementarea clasei “Polynomial” ne permite sa stocam o lista de Monoame. Aceasta lista va constitui polinomul in sine.
* Implementarea clasei “OperationsMonomial” va facilita lucrul cu Monoame deoarece descrie principalele operatii ce se pot efectua cu acestea.
* Implementarea clasei “Operations” are ca scop efectuarea operatiilor cu polinoame mentionate mai sus.
* Implementarea clasei “View” va face posibila interactiunea cu utilizatorul. Aceasta clasa foloseste componente specific pentru a descrie caracteristicile programului.
* Implementarea clasei “Controller” ne permite sa atribuim componentelor specificate in clasa “View” functiolanitati. 1 2 3 4

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

* **Analiza problemei**

Scopul acestei teme este de a realiza o aplicatie capabila sa efectueze operatii matematice cu polinoame utilizand paradigmele programarii pe obiecte. Programul va implemente 6 operatii in total. Pentru a putea efectua operatiile matematice avem nevoie sa stocam datele corespunzator astfel ca vom folosi o lista de monoame, echivalentul unui polinom. Calculatorul nostru trebuie sa fie capabil sa preia datele necesare din interfata , sa faca conversia din String in Polynomial, apoi sa efectueze operatiile astfel inca sa tina cont de cerintele minime ale unei programari eficiente.

* **Modelare**

Este important de retinut ca utilizatorul va introduce intotdeauna informatia pe care doreste sa o proceseze sub forma unui text (String). Pentru o functionare corecta trebuie sa modelam aceasta informatie si sa o aducem cat mai aproape de forma dorita de noi. Mai precis, avem nevoie sa transformam informatia preluata de la utlilizator intr-un polinom.

Aceasta transformare se va contura pe baza faptului ca deja cunoastem structura specifica unui polinom. Astfel ca datele preluate vor fi transpuse intr-un polinom, iar la randul sau polinomul va fi divizat in mai multe monoame pentru care cunoastem cele doua atribute caracteristice : gradul , precum si coeficientul. Avand forma specifica unui polinom vom putea sa implementam operatiile matematice.

* **Scenariul**

Pentru ca aceasta aplicatie sa furnizeze intotdeauna un rezultat corect utilizatorul trebuie sa urmeze un set de reguli precise, menite sa faca munca cu aceasta aplicatie mai rapida si fara problem :

Inainte de a introduce un polinom de la tastatura trebuie mentionat faptul ca programul va identifica doar termenii ce contin ca necunoscuta litera “X” si va ignora orice alta litera introdusa in locul acesteia. Termenii polinomului trebuie sa respecte forma “±coeficientX^grad”, cu anumite exceptii :

* Pentru primul termen al polinomului se poate omite semnul +.
* Daca dorim sa introducem un termen al carui coefcient sau grad este 1 putem sa folosim urmatoarele variante : ±X, ±aX, ±X^b. Aceasta exceptie nu se aplica totusi si in cazul unui monom cu gradul 0, astfel ca in acest caz suntem nevoiti sa tastam : ±aX^0.

Delimitarea termenilor se va face folosind caracterul + sau – . Trebuie specificat ca acest calculator nu va identifica corect coeficienti sau grade ce nu se regasesc in intervalul [0,999]. In cazul operatiilor de derivare si integrare utilizatorul poate sa decida asupra carui polinom se efectueaza operatia, cu precizarea ca in cazul in care acesta omite sa selecteze un polinom programul va genera un MessageBox cu textul “Selectati un polinom!!”.

* **Cazuri de utilizare**

La rulare acestei aplicatii se va deschide o interfata grafica compusa din doua JTextField , sapte butoane si doua JRadioButton. Design-ul pentru aceasta interfata este unul simplu, precis si accesibil pentru toti utilizatorii. Folosim cele doua JTextField pentru a introduce polinoamele asupra carora se vor efectua operatiile, informatia din acestea va fi preluata si modelata in conformitate cu cerinta problemei. Sase butoane sunt folosite pentru a implementa fiecare operatie matematica. Fiecare buton alocat operatiilor are un nume sugestiv ceea ce va facilita munca cu aceasta interfata.

Dupa ce introduce date in cele doua casute prezente, utilizatorul trebuie sa acceseze un buton , echivalent cu operatia pe care doreste sa o realizeze. Rezultatul va fi afisat intr-un JTextField setat ca “Read Only”. Pentru operatia de impartire programul va afisa in primul camp Catul operatiei, iar imediat sub acesta, intr-un alt JTextField setat ca “Read Only”, Restul operatiei. Pe langa cele sase butoane menite sa efectueze operatii, aplicatia mai dispune de un al 7-lea buton numit “Clear”. Acest buton va reseta continutul casetelor text prin adaugarea sirlui vid.

Pentru a selecta asupra carui polinom dorim sa efectuam operatia de integrare sau derivare interfata contine doua JRadioButton ce ne permit sa selectam doar o singura casuta. Rezultatul pentru aceste doua operatii va fi afisat doar dupa ce se va selecta una din cele doua casute.

# 3. PROIECTARE

Acest proiect respecta arhitecura clasica Model View Controller (MVC), fiind impartiti in mai multe pachete denumite adecvat. In pachetul “businessLogic” avem clasele “Operations” si “OperationsMonomial”, in pachetul “dataModels” se afla clasele “Monomial” si “Polynomial”, iar in pachetul “graphicalUserInterface” avem clasele “View” si “Controller”.

Principal clasa a acestui proiect este “Monomial” care contine 2 atribute : grad definit de tip int si coeficient definit de tip double; setters, getteres pentru cele doua campuri; constructorul clasei si metoda toString(). De asemenea, aceasta clasa suprascrie metoda “int compareTo(Monomial o)”.

Pornind de la clasa Monomial ne-am definit clasa Polynomial ce contine ca atribut o lista de monoame. Aceasta clasa contine si urmatoarele metode : parsePolynomial(String dataIn); addList(Monomial termen); sortPolynom(); getTheFirstMonom();isNotEmpty();clearPolynomial().

Clasa OperationsMonomial contine metode pentru operatiile de baza ce se pot efectua cu monoame: add(Monomial monom1, Monomial monom2), multiply(Monomial monom1, Monomial monom2), derivare(Monomial monom), integrare(Monomial monom), divide(Monomial monom1, Monomial monom2). Pe langa aceste metode mai exista si metodele : getNewCoeficient(Polynomial polynom, int grad), aceasta metoda determina care este coeficientul pentru un monom cu grad dat; existGrad(Polynomial polynom, int grad), aceasta metoda verifica daca in polinom exista sau nu un monom cu un grad specificat.

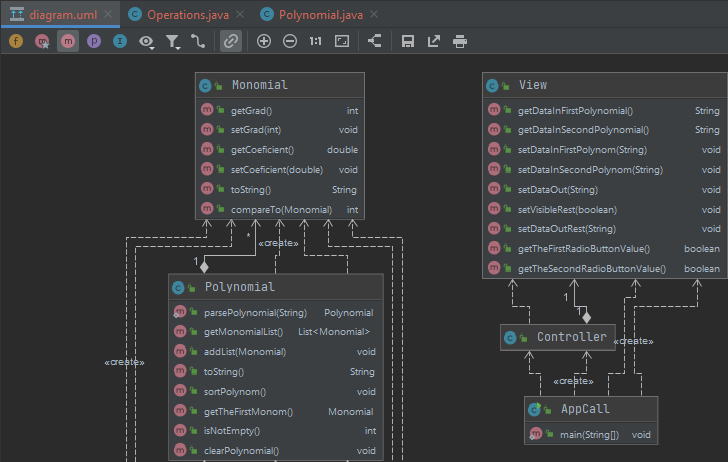
Clasa Operations implementeaza cele sase operatii matematice pentru calcul cu polinoame : add(Polynomial polynom1, Polynomial polynom2), decrease(Polynomial polynom1, Polynomial polynom2), integral(Polynomial polynom), derivative(Polynomial polynom), multiply(Polynomial polynom1, Polynomial polynom2), divide(Polynomial polynom1, Polynomial polynom2, Polynomial q). De mentionat ca toate aceste metode sunt definite static . De asemenea, trebuie tinut cont de faptul ca toate aceste metode returneaza polinomul obtinut in urma efectuarii operatiilor.

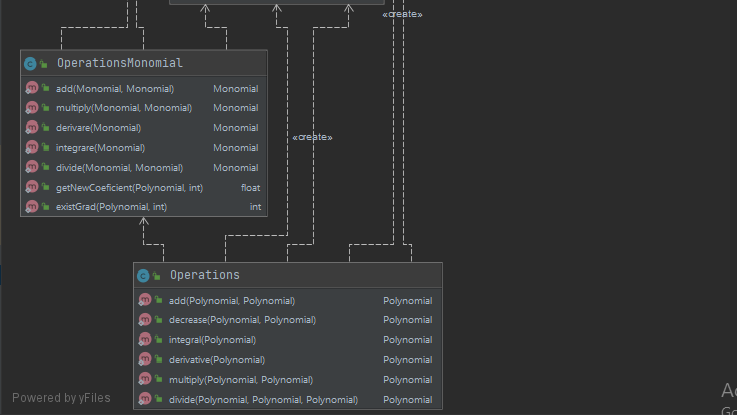
Clasa View contine elemente de structura necesare pentru interfata grafica a proiectului. Aceasta face interactiunea cu aplicatia mai practica si mai rentabila pentru utilizator. In cadrul acestei clase am utilizat elemente precum : JTextField( utilizat pentru prelucrarea textului), JButton, JRadioButton, JPanel( intermediar). Tot in aceasta clasa decidem modul in care aranjam aceste structuri in interfata finala.

Clasa Controller este clasa in care se traduce interactiunea utlizatorului cu interfata grafica in actiuni ce vor si executate de catre partea de Model.

Pentru a testa functionalitatea operatiilor matematice cu polinoame am creat o clasa de test numita “TestOperations” ce contine teste JUnit.

Clasa AppCall este cea in care se leaga toate componentele mentionate mai sus, permitandu-ne sa rulam aplicatia.



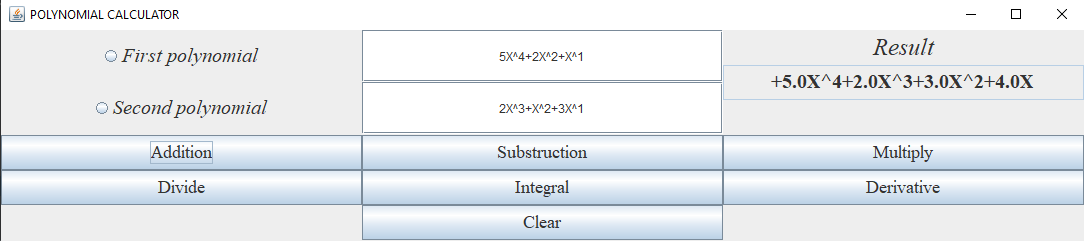


# 4. IMPLEMENTARE

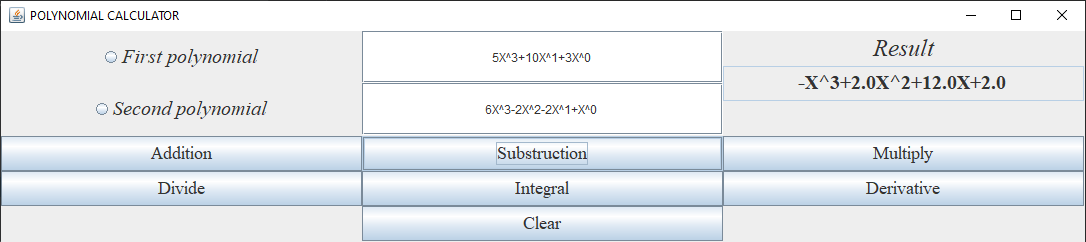
* Clasa “Monomial” are doua atibute : atributul grad (int) si atributul coefficient (double). Pentru aceste doua atribute avem implementate metode de set si get cu ajutorul carora setam valoarea unui atribut sau obtinem valoarea stocata in acesta. Tot in aceasta clasa suprascriem metoda compareTo() pentru a putea ordona o lista de monoame descrescator in functie de atributul grad.
* Clasa “Polynomial” are ca atribut o lista de monoame numita “monomialList”. In aceasta clasa sunt implementate mai multe metoda pentru a facilita manipularea polinoamelor.
* Metoda “parsePolynomial(String dataIn)” este metoda cea mai importanta implementata in aceasta clasa. Aceasta metoda este definite static astfel ca va fi apelata in raport cu numele clasei dupa cum urmeaza: Polynomial.parsePolynomia(String dataIn). Aceasta metoda primeste ca parametru un String pe care il va convertii la structura de polinom. Pentru a efectua aceasta actiune programul va folosi un Pattern cu ajutorul caruia va extrage din string-ul dataIn fiecare monom. De exemplu: pentru dataIn = “3X^2+2X-4X^0” matcher-ul va obtine pe baza pattern-ului definit urmatoarele grupuri: “3X^2”, “+2X”, “-4X^0”. La pasul urmator metoda va manipula fiecare grup in parte si va extrage valorile numerice in functie de pozitia acestora fata de necunoscuta “X”. Dupa ce seteaza gradul si coeficientul corespunzator pentru fiecare monom aplicatia introduce acest monom intr-un polinom pe care il va returna ca rezultat.
* Metoda “addList(Monomial termen)” primeste ca parametru un monom pe care il va adauga la lista noastra de monoame.
* Metoda “sortPolynom()” ordoneaza lista noastra de monoame descrescator in functie de gradul fiecarui monom.
* Metoda “getTheFirstMonom()” returneaza primul monom ce se gaseste in lista noastra de monoame.
* Metoda “isNotEmpty()” verifica daca un polinom este sau nu gol.
* Metoda “clearPolynomial()” elimina toti termenii din lista de monoame.
* Clasa “OperationsMonomial” implementeaza operatii matemmatice de baza cu monoame. Pe langa aceste operatii clasa mai are definite inca doua metode necesare efectuarii operatiilor matematice cu polinoame.
* Metoda “getNewCoeficient(Polynomial polynom, int grad)” va parcurge polinomul si va calcula coeficientul termeniilor cu acelasi grad.
* Metoda “existGrad(Polynomial polynom, int grad)” ne va semnaliza daca exista sau nu un monom cu gradul dat ca parametru in polinom.
* Clasa “Operations” este clasa in care se vor efectua operatiile matematice cu polinoame. Pentru fiecare operatie matematica este definita o metoda statica ce va returna rezultatul sub forma unui nou polinom.
* Metoda “add(Polynomial polynom1, Polynomial polynom2)” primeste ca parametrii doua polinoame si are ca scop returnarea sumei dintre acestea. Initial aplicatia va parcurge primul polinom si pentru fiecare termen va cauta alti termeni in al doilea polinom care sa aiba acelasi grad. Atunci cand gaseste termeni cu acelasi grad va aduna coeficientii acestora cu scopul de a obtine un termen nou cu acelasi grad, dar coeficient diferit. La acest pas va adauga noul termen la rezultat doar daca coeficitul gasit este diferit de 0. Dupa aceasta parcurgere este nevoie de inca o parcurgere in care sa verificam daca in polinomul rezultat exista toate gradele din al doilea polinom. Daca nu, programul va adauga acesti termeni la rezultat. La final metoda returneaza polinomul ordonat.
* Metoda “decrease(Polynomial polynom1, Polynomial polynom2)” are ca scop returnarea diferentei dintre doua polinoame. Initial metoda va parcurge cel de-al doilea polinom si va schimba semnul fiecarui coeficient. Pentru a simplifica aceasta metoda, dupa pasul 1 se va apela metoda “add” prezentata mai sus.
* Metoda “multiply(Polynomial polynom1, Polynomial polynom2)” returneaza produsul a doua polinoame. Aceasta metoda va face inmultirea termen cu termen exact ca la matematica urmand sa grupam termenii cu acelsi grad pentru a obtine un polinom cu grade distincte. Pentru a face acest pas metoda va parcurge polinomul mulPolynoms obtinut in urma inmulitii termen cu termen si pentru fiecare monom va determina un posibil coefficient nou. Daca in polinomul result nu exista deja un termen cu gradul termenului curent metoda va adauga acest termen la rezultat.
* Metoda “divide(Polynomial polynom1, Polynomial polynom2, Polynomial q)” are ca parametrii 3 polinoame. Primii doi parametrii sunt polinoamele pe care dorim sa le impartim iar al treilea parametru este polinomul in care vom stoca catul acestei operatii. Initial metoda face o copie Deimpartitului in polinomul r. Cat timp polinomul r mai contine termeni si gradul polinomului r (deimpartitul) este mai mare decat gradul polinomului polynom2 (impartitorului) vom parcurge pasii pentru impartire exact ca la matematica. Aceasta metoda returneaza ca rezultatul restul obtinut in urma impartitii, polinomul r.
* Metoda “integral(Polynomial polynom)” primeste ca parametru un polinom asupra caruia dorim sa efectuam operatia de integrare. Pentru a face aceasta operatie am folosit formual matematica : .
* Metoda “derivative(Polynomial polynom)” primeste ca parametru un polinom asupra caruia dorim sa aplicam operatia de derivare dupa cum urmeaza : .
* Clasa “View” genereaza partea de interfata grafica ce se poate vizualiza la rularea aplicatiei. In aceasta clasa se descrie modul de aranjare al componentelor in interfata grafica si adauga ActioListener() pentru butoane.
* Clasa “Controller” se coupa cu implementarea “evenimentelor” in momentul in care se efectueaza oa actiune asupra unei componente din interfata ce genereaza un ActionListener().

# 5. REZULTATE

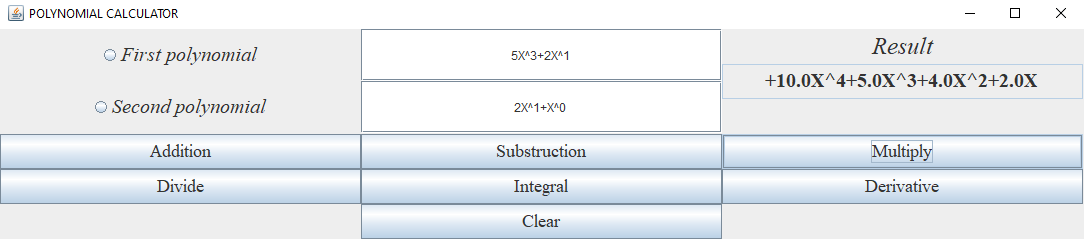
* Adunare



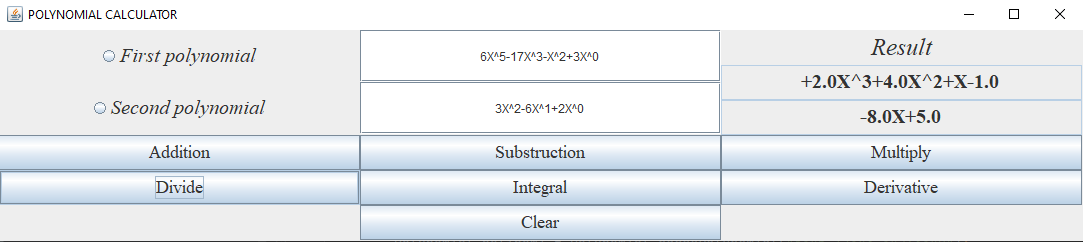
* Scadere



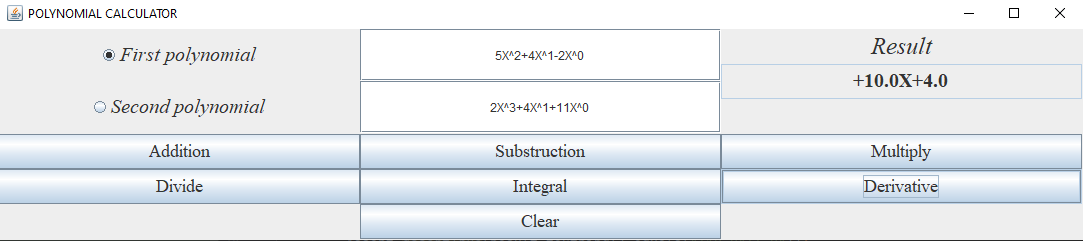
* Inmultire



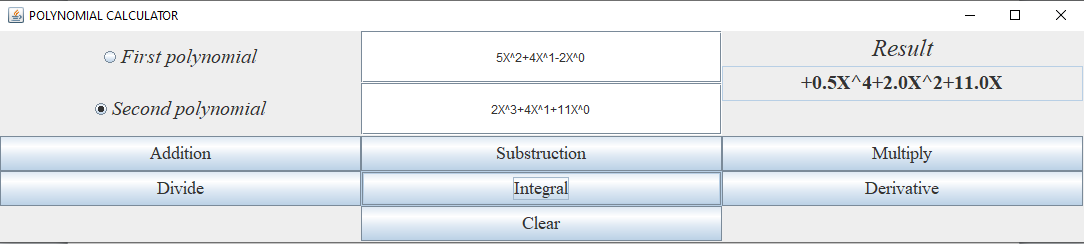
* Impartire



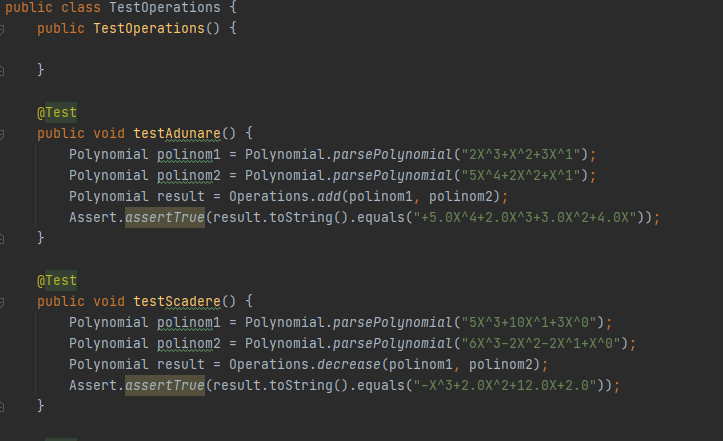
* Derivare



* Integrare



**TESTARE:**

****

# 6. CONCLUZII

Pentru mine aceasta tema nu a ridicat mari dificultati din punct de vedere al operatiilor matematice implementate pentru polinoame, gandirea design-ului sau aducerea acestei aplicatii de la 0 la punctul in care sa ne genereze rezultate conform cerintei. Am intampinat dificultati la convertirea unui String in Polynomial folosint pattern si matcher, insa dupa incercari repetate am ajuns la o forma convenabila care sa imi genereze un rezultat corect si am incercat sa implementez aceasta aplicatie respectand paradigmele programarii orientate pe obiecte .

# 7. BIBLIOGRAFIE

* <https://regexr.com/>
* <https://www.geeksforgeeks.org/>
* <https://stackoverflow.com/>