DOCUMENTAȚIE

TEMA_1

Student: Pop Raul-George
Grupa: 30225

CUPRINS

1.Obiectivul temei 3
2.Analiza problemei, modelare, scenarii, cazuri de utilizare 4
3. Proiectare 6
4. Implementare 7
5. Rezultate 10
6. Concluzii 11
7. Bibliografie 12

1.Obiectivul temei

Obiectivul temei este reprezentat de implementarea și proiectarea unui sistem de calcul polinomial cu o interfață grafică dedicată menită să ușureze și faciliteze introducerea de date (polinoame) de către utilizator cât și alegerea de operații posibile care pot fi realizate pe datele introduse și vizualizarea rezultatului final. Operațiile implementate sunt: adunarea, scăderea, înmulțirea, împărțirea, derivarea și integrarea polinoamelor. Sistemul de calcul trebuie să fie ușor de folosit și ușor de înțeles de orice utilizator indiferent de nivelul de pregătire în domeniul matematicii cât și al tehnologiei.

Obiectivele secundare considerate în cadrul implementării obiectivului principal sunt:

1. Separarea problemei în structuri ușor de implementat: realizarea de clase și pachete care vor lua parte la formarea sistemului de calcul;
2. Realizarea interfeței grafice: crearea unei interfețe care va fi folosită de către utilizator pentru introducerea și prelucrarea datelor;
3. Implementarea unei metode de conversie și stocare a datelor introduse de utilizator;
4. Implementarea operațiilor fundamentale realizabile pe polinoame;
5. Testarea operațiilor menționate anterior.

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

În matematică, un polinom este o expresie construită din una sau mai multe variabile și constante, folosind doar operații de adunare, scădere, înmulțire și ridicare la putere constantă pozitivă întreagă.

X^2-4X+7 este un polinom

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. Exponentul unei variabile dintr-un monom este egal cu gradul acelei variabile în acel monom. Pentru că $X^1 = X$ gradul unei variabile fără exponent este unu. Un monom fără variabile se numește monom constant, sau doar constantă. Gradul unui termen constant este 0. Coeficientul unui monom poate fi orice număr, inclusiv fracții, numere iraționale sau negative.

De exemplu, $-5X^2Y$ este un monom. Are coeficientul -5, variabilele sunt x și y, gradul lui x este doi, iar gradul lui y este unu. Gradul întregului monom este suma gradelor tuturor variabilelor din el. În exemplul de mai sus, gradul este 2 + 1 = 3.

Un polinom este o sumă de unul sau mai multe monoame. De exemplu, următoarea expresie este un polinom: $3X^2-5X+4$

El constă din trei monoame: primul are gradul doi, al doilea are gradul unu, iar al treilea are gradul zero.

Când un polinom este dispus în ordinea naturală, el are termenii de grad mai mare înaintea celor de grad mai mic. În primul termen, coeficientul este 3, variabila este x, iar exponentul este doi. În al doilea termen, coeficientul este -5. Al treilea termen este o constantă. Gradul unui polinom este cel mai mare grad al unui termen al său. În acest exemplu, polinomul are gradul doi.

La baza analizei problemei stau elementele programării orientate pe obiecte, deci procesul de analizare va începe prin extragerea de verbe și substantive din enunțul problemei care să creeze o bază de pornire pentru implementarea Java. Un alt aspect important este interacțiunea utilizatorului cu aplicația. Soluția este reprezentată de o interfață grafică ușor de utilizat care va cuprinde toate funcționalitățile aplicației împreună cu o modalitate de introducere a datelor și afișare a acestora.

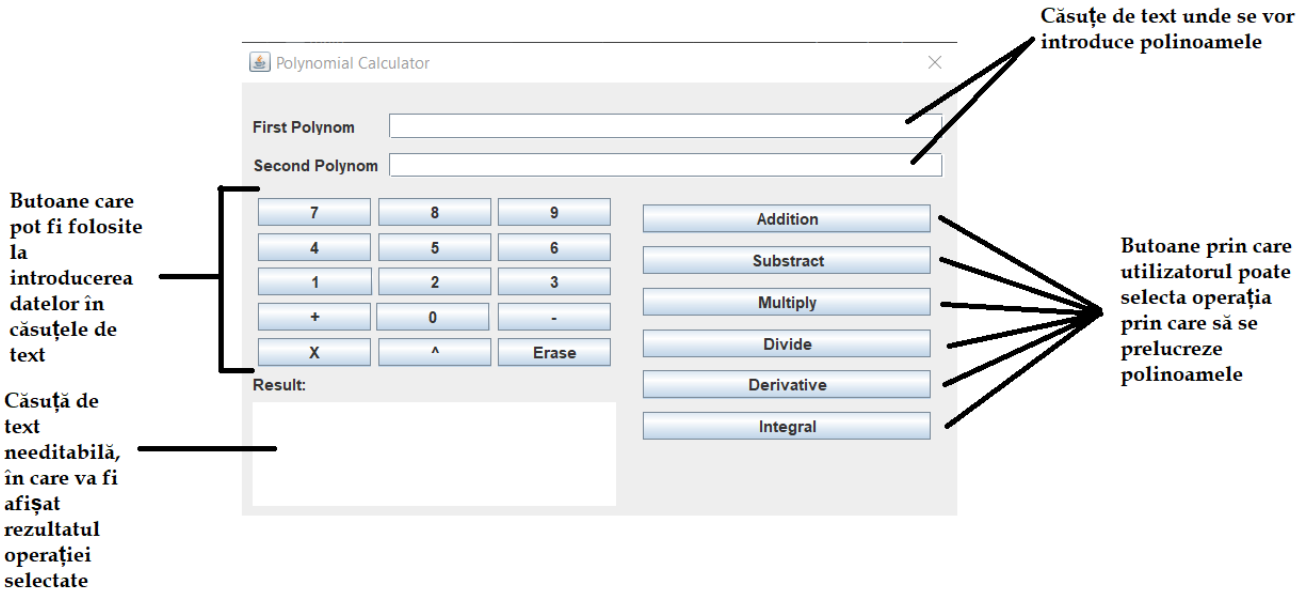
Cazurile de utilizare se împart în:

- Operații pe 2 polinoame: adunare, scădere, înmulțire, împărțire;

- Operații pe un polinom: derivare și integrare.

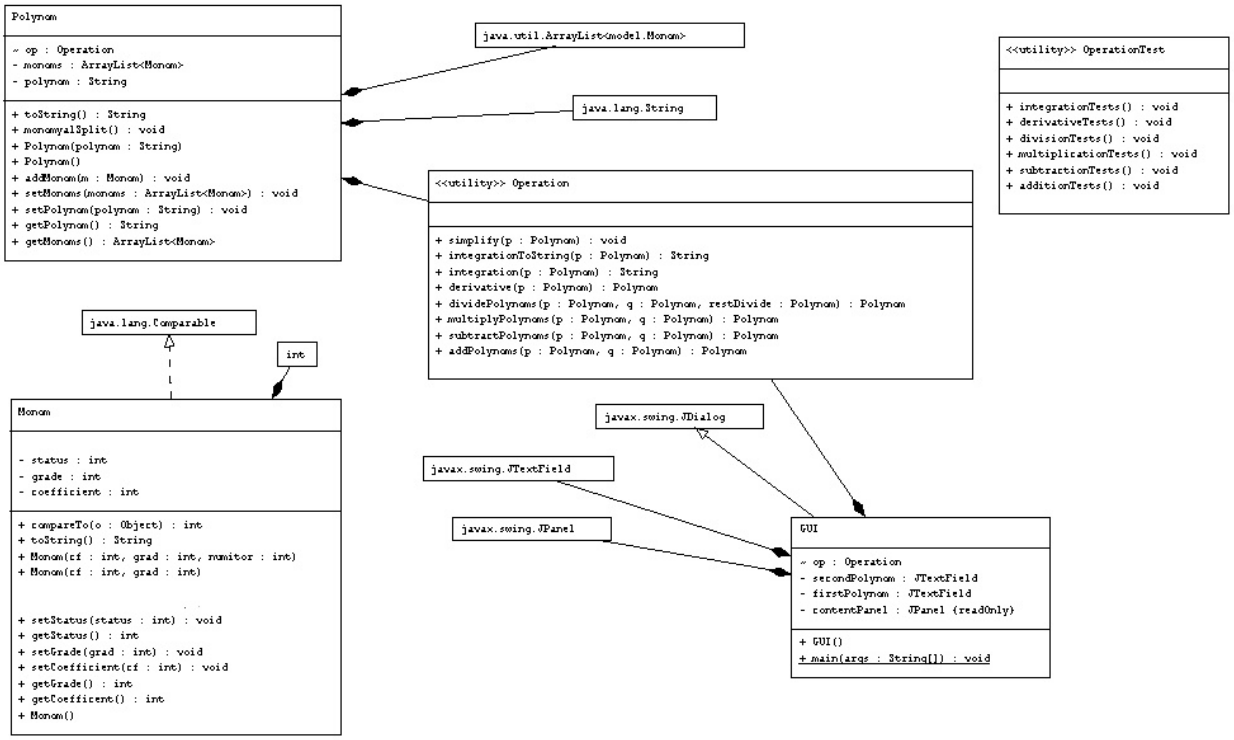
Pentru ambele cazuri, rezultatul operației va fi afișat într-un panou destinat.

Sistemul de calcul este modelat în așa fel încât, utilizatorul va avea la dispoziție 2 câmpuri de text editabile, în cadrul cărora va introduce datele sub formă polinomială (*ex. x^2+2x+1*) de unde vor fi preluate de către program și prelucrate anterior fiind afișate rezultatele.



(*)*Pentru operațiile de derivare și integrare se va introduce în prima căsuță de text polinomul asupra căruia se vor realiza operațiile.*

3. Proiectare



Aplicația conține 4 clase principale și o clasă de test care va avea rolul de a verifica corectitudinea fiecărei operații implementate. În fiecare clasă se respectă principiul încapsulării.

4. Implementare

- Clasa *Monom*:

  Are 3 variabile instanță care descriu coeficientul și gradul fiecărui monom din polinomul din care face parte, iar variabila status va fi folosită în cadrul operațiilor. Clasa Monom implementează interfața *Comparable* pentru a fi ușor de ordonat în funcție de gradul fiecărui monom.

- Clasa *Polinom*:

  Are 3 variabile instanță. Variabila *polynom* va memora polinomul introdus de către utilizator sub formă de String; variabila *monoms:* va memora o listă de monoame; variabila *op*: va reprezenta o instanță a unui obiect de tip *Operation* de care ne vom folosi în cadrul implementării unor metode din clasa Polinom. Metoda *monomialSplit* va transforma polinomul introdus de utilizator sub forma de String într-o listă de monoame care va fi memorată în variabila *monoms* a obiectului de tip Polinom.

- Clasa *Operation*:

  Această clasă nu are variabile instanță, însă această implementează toate operațiile pe care utilizatorul le poate aplica polinoamelor introduse în căsuțele de text.

  Astfel ca,

  - Metoda *addPolynoms* va realiza adunarea a două polinoame astfel: va parcurge inițial atât primul polinom cât și al doilea și va verifica dacă gradul monoamelor curente sunt egale. În cazul în care acestea sunt egale, coeficienții se vor aduna, rezultatul fiind stocat într-o variabila *result* iar monoamelor adunate li se va seta variabila status la 2. După parcurgerea polinoamelor, vom mai parcurge o dată fiecare polinom pe rând și vom verifica valoarea variabilei status a fiecărui monom, în cazul în care aceasta are valoarea 2, nu adăugăm monomul respectiv la rezultat, în caz contrar îl adăugăm.
  - Metoda *substractPolynoms* va realiza scăderea a două polinoame urmând același algoritm prezent și la metoda anterioară.
  - Metoda *multiplyPolynoms* va realiza înmulțirea a două polinoame astfel: se vor parcurge simultan ambele polinoame și vor fi înmulțite pe rând toate

monoamele din cele două polinoame, rezultatul fiind memorat în variabila *result*.

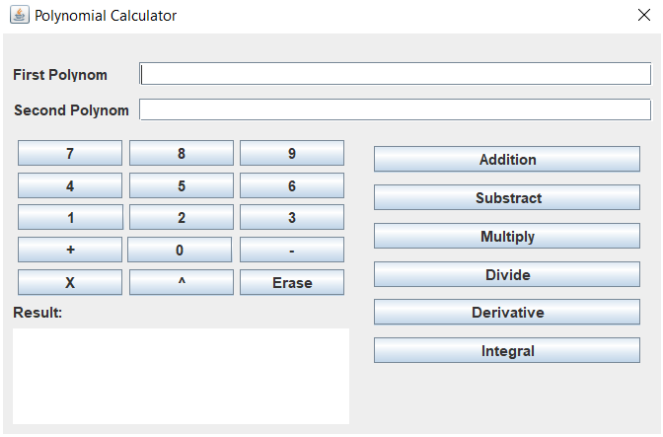
- Metoda *dividePolynoms* va realiza împărțirea a două polinoame urmărind următorul algoritm:

```
function n / d is
    require d ≠ 0
    q ← 0
    r ← n             // At each step n = d × q + r

    while r ≠ 0 and degree(r) ≥ degree(d) do
        t ← lead(r) / lead(d)       // Divide the leading terms
        q ← q + t
        r ← r − t × d

    return (q, r)
```

- Metoda *derivative* va realiza derivarea unui polinom astfel: va parcurge întreg polinomul și va salva în variabila *result* polinomul modificat (coeficientul original al monomului se va înmulți cu gradul monomului iar gradul monomului se va scădea cu 1).
- Metoda *integration* va realiza integrarea unui polinom astfel: va parcurge întreg polinomul și va salva în variabila *result* polinomul modificat (gradul original al fiecărui monom va crește cu 1). Deoarece variabila *coefficient* din clasa monom este de tip întreg și pentru a păstra o mai bună acuratețe, am ales ca reprezentarea integrării sa se facă astfel: Coeficientul monoamelor nu se va modifica în cadrul metodei *integration* decât în funcția *integrationToString* unde coeficienții vor fi afișați sub formă de fracție. (Coeficient original al monomului / Gradul monomului după integrare).
- Metoda *simplify* va verifica dacă în polinom există polinoame cu gradul 0 și coeficientul 0 sau doar coeficientul 0, și le va elimina. De asemenea va verifica și dacă există mai multe monoame de același grad, iar în cazul în care există, le va aduna.

- Clasa *GUI*

  Implementează interfața grafică și face legătura dintre utilizator și operațiile implementate prin intermediul căsuțelor de text și a butoanelor.



- Clasa *OperationTest*

  Realizează teste de verificare unitară a tuturor operațiilor implementate prin intermediul framework-ului *JUnit*.

5. Rezultate

În cadrul alegerii testelor din clasa OperationTest au fost considerate anumite cazuri.

- Pentru operația de adunare au fost alese polinoame care au monoame de grad egal, polinoame care nu au monoame de grad egal, polinoame care au atât polinoame cu grad egal cât și diferit, polinoame cu coeficienți negativi, pozitivi sau nuli, polinoame ale căror sumă are ca rezultat un număr întreg nenul, polinoame ale căror sumă are ca rezultat 0, polinoame a căror grad maxim este 0, polinoame de grad mare.
- Pentru operația de scădere au fost alese polinoame urmând același principiu precum la operația de adunare.
- Pentru operația de înmulțire au fost alese polinoame care au monoame de grad egal, polinoame care nu au monoame de grad egal, polinoame care au atât polinoame cu grad egal cât și diferit, polinoame cu coeficienți negativi, pozitivi sau nuli, polinoame ale căror înmulțire are ca rezultat un număr întreg nenul, polinoame ale căror înmulțire are ca rezultat 0, polinoame a căror grad maxim este 0, polinoame de grad mare.
- Pentru operația de împărțire au fost alese polinoame care au monoame de grad egal, polinoame care nu au monoame de grad egal, polinoame care au atât polinoame cu grad egal cât și diferit, polinoame de grad egal și diferit, polinoame cu coeficienți negativi, pozitivi sau nuli, polinoame ale căror împărțire are ca rezultat un număr întreg nenul, polinoame ale căror împărțire are ca rezultat 0 (cazuri în care împărțirea nu poate avea loc), polinoame a căror grad maxim este 0, polinoame de grad mare.
- Pentru operația de derivare au fost alese polinoame cu coeficienți negativi, pozitivi sau nuli, polinoame ale căror derivare are ca rezultat un număr întreg nenul, polinoame ale căror derivare are ca rezultat 0, polinoame a căror grad maxim este 0, polinoame de grad mare.
- Pentru operația de integrare au fost alese polinoame urmând același principiu precum la operația de derivare.

6. Concluzii

În concluzie, obținem un sistem de calcul ușor de folosit și bine organizat. În urma implementării proiectului au fost aprofundate cunoștințele de implementare a unei interfețe grafice, de testare unitară folosind framework-ul *JUnit,* de organizare a claselor și cunoștințe de logică generală.
Ca și dezvoltări ulterioare putem aminti: evaluarea polinomului într-un punct întreg dat, lucrul cu numere flotante, operația de generare a graficului polinomului, implementarea unui istoric al operațiilor realizate.

7. Bibliografie

https://ro.wikipedia.org/wiki/Polinom

https://dsrl.eu/courses/pt/materials/A1_Support_Presentation.pdf

https://junit.org/junit5/docs/current/user-guide/