

Tehnici de Programare

**CTI romana, seria B**

**An II**

**Drăgan Lorena Diana**

**Cuprins**

* **Obiectivul Temei**
* **Analiza problemei, modelare, scenarii, cazuri de utilizare**
* **Proiectare(diagrame UML, decizii de proiectare, relații, pachete, algoritmi, interfața utilizator)**
* **Implementare**
* **Rezultate**
* **Concluzii**
* **Bibliografie**
* **Obiectivul temei**

*Specificație:* Propuneți, proiectați și implementați un sistem de procesare a polinoamelor de o singură variabilă cu coeficienți întregi.

Obiectivul acestei teme este acela de a crea un calculator cu ajutorul căruia utilizatorul să poată efectua operații asupra a două polinoame sau asupra unui singur polinom. La fel ca și la calculatorul de buzunar sau calculatorul sistemului de operare, acest calculator trebuie să aibă în dotare următoarele operații: adunarea a două polinoame, scăderea a două polinoame, înmulțirea a două polinoame, împărțirea a două polinoame și în plus față de calculatoarele obișnuite, acest calculator beneficiază și de următoarele operații: calcularea derivatei de ordinul întâi al unui polinom, calcularea integralei unui polinom. La fel ca și un calculator obișnuit și acest calculator are date de intrare și date de ieșire, acestea din urmă rezultând în urma operațiilor efectuate cu ajutorul calculatorului și fiind afișate cu ajutorul interfeței grafice.

Îmi propun să realizez un calculator ca cel descris mai sus astfel încât să fie ușor de înțeles și util pentru orice tip de utilizator. Un prim și important aspect ar fi acela că, deoarece acest calculator implementează și operația de integrare, am ales sa utilizez coeficienți și exponenți întregi, astfel la anumite date introduse de utilizator, acesta va putea vedea rezultatul în virgulă flotantă, acolo unde coeficienții vor fi numere fracționare.

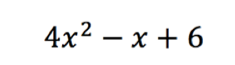
* **Analiza problemei, modelare, scenarii, cazuri de utilizare**

**Informatii despre polinoame**

In matematica, un polinom este o expresie contruita dintr-una sau mai multe variabile si constate, folosind doar operatii de adunare, scadere, inmultire, si ridicare la putere constata pozitiva intreaga.

Polinoamele sunt construite din termeni numiti monoame, care constau dintr-o constanta(numita coeficient) inmultita cu una sau mai multe variabile( de regula reprezentate prin litera “x”). Fiecare variabila poate avea un exponent constant intreg(flotant) pozitiv. Exponentul unei variabile dintr-un monom este egal cu gradul acelei variabile in acel monom. Pentru ca orice numar ridicat la puterea 0 este egal cu unu, gradul unei variabile fara exponent este 0. Un monom fara variabila se numeste monom constant. Coeficientul unui monom poate fi orice numar, inclusiv fractii, numere irationale sau negative.

Un polinom este o suma de unul sau mai multe monoame:

. El constă din trei monoame, primul are gradul doi, al doilea gradul unu iar al treilea 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 4, variabila este x iar exponentul este 2. Gradul unui polinom este cel mai mare grad al unui termen al său. În aceste exemplu, cel mai mare grad al unui termen este doi, prin urmare gradul polinomului este doi. Un polinom de gradul unu este numit liniar, unui de gradul doi este pătratic, iar unul de gradul trei este un polinom cubic. Un polinom cu un singur termen este numit monom, unul cu doi termeni binom, iar unui cu trei termeni este trinom. Un polinom care are coeficientul 1 pentru termenul de grad maxim se numește monic.

Proprietăți elementare ale polinoamelor:

* Suma a două polinoame este un polinom;
* Produsul a doua polinoame este un polinom;
* Derivata unui polinom este un polinom;
* Primitiva unui polinom este un polinom;

Toate polinoamele de o variabilă sunt echivalente cu un polinom de forma,

Această formă este considerată forma generală a polinoamelor de o singură variabilă.

**Modelare**

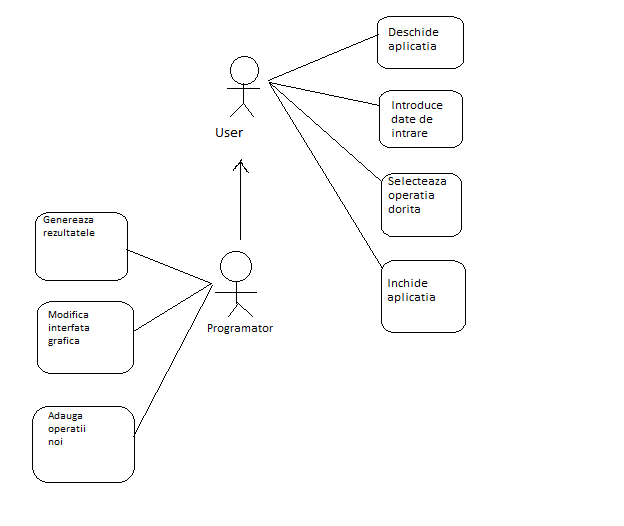
Calculatorul care trebuie implementat trebuie să aibă coeficienți întregi. Această constrângere este una importantă, deoarece se pot ivi probleme la operațiile de împărțire și integrare a două polinoame. La împărțire poate apărea această problemă, deoarece la calcularea câtului trebuie să împărțim doi cate doi întregi, care de cele mai multe ori va fi un număr real, tocai din acest motiv vom implementa un mod prin care utilizatorul va introduce de la tastatură două polinoame cu coeficienți întregi, iar calculatorul va returna inclusiv valori reale. În ceea ce privește integrarea, coeficientul va fi dat de vechiul coeficient și vechea putere plus o unitate, care este tot un număr real în unele cazuri, la fel calculatorul va returna un polinom cu coeficienți reali. În ceea ce privește celelalte operații de procesare a polinoamelor, constrângerea nu pune nici un fel de probleme.

Pentru a efectua oricare dintre operațiile propuse avem nevoie de date

precum coeficienții fiecărui polinom și exponenții acestuia. Datele acestea se vor introduce de la tastatură : “+1x^2+2x^1+2”, care înseamnă de fapt “x^2+2x+1”. Se remarcă faptul că pentru fiecare monom al polinomului trebuie introdus coeficientul cu semn, inclusiv pentru primul. De asemenea termenul care corespunde monomului de grad 0 se va pune fără variabila x sau exponent.

Polinomul se va introduce de la tastatura în ordinea descrescătoare a gradelor, monomul cu gradul cel mai mare fiind primul introdus.

**Diagrama Use - Case**



**Operații asupra polinoamelor**

*Adunarea*  a două polinoame: adunarea are ca rezultat un nou polinom. Coeficienții rezultatului sunt egali cu suma coeficienților corespunzători aceleiași puteri ai celor două polinoame de adunat. Rezultatul are lungimea polinomului cel mai lung.

*Scăderea* a două polinoame: scăderea are ca rezultat un polinom. Coeficienții rezultatului sunt egali cu diferența coeficienților corespunzători aceleiași puteri ai celor două polinoame de scăzut. Rezultatul are lungimea polinomului celui mai lung.

*Înmulțirea* a două polinoame: înmulțirea are ca rezultat un polinom. Gradul noului polinom va fi suma gradelor celor doua polinoame de înmulțit.

Fiecare coeficient al primului polinom este înmulțit cu toți coeficienții celui de-al doilea, după care se aduna toate rezultatele conform operației de adunare a polinoamelor.

*Împărțirea* cu rest a două polinoame: împărțirea cu rest are ca rezultat două polinoame. Unul reprezintă câtul și celălalt reprezintă restul. Polinomul care se împarte se numește deîmpărțit, iar cel la care se împarte se numește împărțitor.

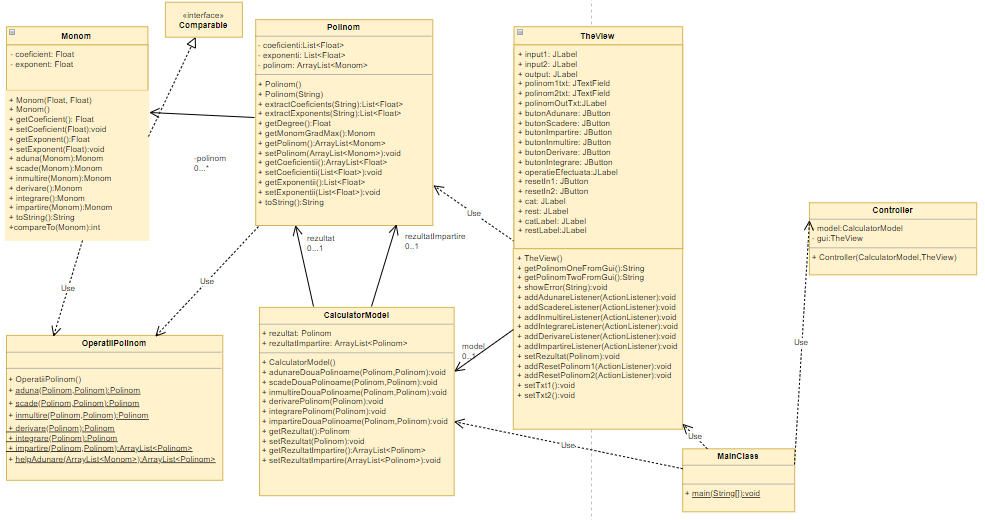
*Derivarea* unui polinom: derivarea unui polinom are ca rezultat tot un polinom. Gradul noului polinom va fi egal cu gradul vechiului polinom minus 1. Fiecare coeficient va fi înmulțit cu puterea corespunzătoare, iar exponentul va scădea cu o unitate.

*Integrarea* unui polinom: integrarea unui polinom are ca rezultat tot un polinom. Gradul noului polinom va fi egal cu gradul vechiului polinom plus 1. Fiecare monom va fi format prin înmulțirea coeficientului său cu variabila la vechea putere plus 1, totul fiind împărțit la vechea putere plus unu.

**Utilizarea aplicației**

* În domenii unde sunt necesare operații cu polinoame;
* În domeniul educațional folosit ca unealtă de verificare a unor calcule;
* **Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfețe, relații, pachete, algoritmi, interfața utilizator)**

**UML Diagram Class**



**Algoritmi folosiți**

La operația de împărțire, este folosit și implementat “Polynomial Long Division Algorithm”.

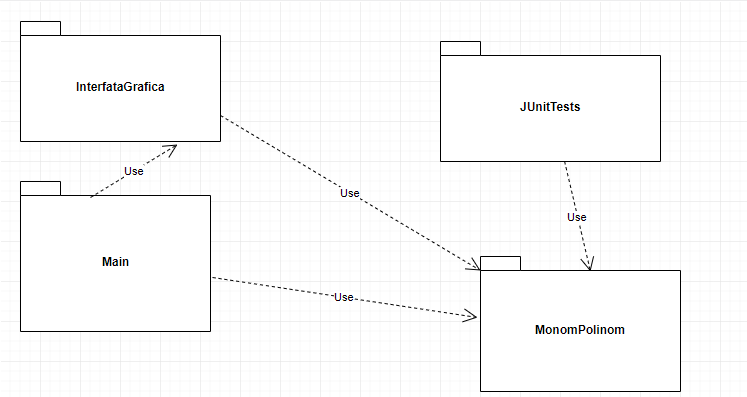
Algoritmul de împărțire decurge astfel:

* Împărțim primul termen al deîmpărțitului la primul termen al împărțitorului și astfel se va obține primul termen al catului.
* Monomul obținut în cât se va înmulți acum cu împărțitorul, iar rezultatul obținut se va scădea din deîmpărțit. Acest calcul ne dă primul rest al împărțirii.
* Se repetă procedeul luând primul rest ca deîmpărțit, iar la catul final se adună mereu monoamele rezultate în urma împărțirii primului termen al deîmpărțitului la primul termen al împărțitorului.
* Algoritmul se termină când gradul restului este strict mai mic decât gradul împărțitorului.

**Interfețe definite**

Programul nu dispune de interfețe definite, însă implementează interfața Comparable, în clasa Monom, pentru a sorta monoamele dintr-un polinom în functie de gradul acestora, descrescător.

**UML package diagram**



* **Implementare**

Aplicația este structurata în 8 clase : TheView, AssignmentTests, Controller, MainClass, CalculatorModel, Monom, OperatiiPolinom, Polinom.

Clasa TheView creează o fereastră oferită utilizatorului și controlează în funcție de acțiunile acestuia aplicația. Interfața grafică este alcătuită din câmpuri de introducere a datelor și de afișare a rezultatelor și butoane de comandă. Aceasta extinde clasa JFrame.

De asemenea în această clasă sunt implementate metode de acces la stringurile primite de la intrare, o metoda de afisare a unui mesaj de eroare pentru date invalid introduse, si metode de adăugare a ascultătorilor pentru fiecare dintre butoanele interfeței.

Clasa Monom folosește coeficienți și exponenți de tip Float și oferă metode pentru toate cele 6 operații, precum și metode de acces pentru exponenți și coeficienți. Aceasta are și doi constructori, unul fără parametrii, care creează un monom cu coeficient și exponent 0, și una cu parametrii de initializare a coeficientului și exponentului. Metoda toString este suprascrisă în această clasă, pentru a putea afișa un monom, iar metoda compareTo compară doua monoame în functie de exponent.

Clasa Polinom folosește un ArrayList de monoame, care este polinomul propriu zis, aceasta are de asemenea și o listă de coeficienți și una de exponenți. Două metode importante din această clasă sunt metodele de extragere a coeficienților și a exponenților din string-ul primit de la utilizator prin interfața grafică, metode apelate în constructorul Polinomului deoarece atunci când un obiect din clasa Polinom este creat, constructorul primește ca și parametru un string pe care apoi îl transpune în arrayList-ul de monoame. Pe lângă aceste metode, mai sunt și metode de extragere a gradului unui polinom și metode de acces la un polinom. Metoda toString este de asemenea suprascrisă în această clasă pentru a afișa polinomul.

Clasa OperațiiPolinom este o clasă ce conține numai metode statice. Metodele sunt cele 6. Ele primesc ca și argumente două polinoame și returnează un alt polinom, cu excepția metodei de împărțire care primeste ca și argumente două Polinoame și returnează doua Polinoame, de cât și de rest, sub forma unui ArrayList.

Clasa CalculatorModel are un Polinom rezultat, si un ArrayList de polinoame rezultatImpartire. Metodele acesteia primesc ca si parametrii doua polinoame si cu ajutorul helper class-ului OperațiiPolinom, returnează cu ajutorul rezultatului răspunsul la operația efectuată.

Clasa Controller este cea care face legătura între interfața cu utilizatorul și modelul unde se efectuează operațiile cu datele de intrare primite de la utilizator. Cele două câmpuri ale acesteia sunt modelul si view-ul, pe care aceasta le foloseste, pentru a primi datele introduse de user, a le prelucra cu ajutorul metodelor din clasa CalculatorModel și a returna rezultatul și afișa cu ajutorul interfeței grafice.

Clasa AssignmentTests implementează 14 teste, unul pozitiv și unul negativ pentru fiecare dintre cele 6 operații, plus incă două teste pentru parse-ingul stringului primit ca si input de la utilizator, aceasta eșuând când datele nu sunt introduse corect.

**Interfața Utilizator**

Pentru interfața utilizator am folosit un JPanel principal, pe care am adăugat 5 alte JPanel-uri, cu diferite layout-uri. In primele două JPanel am adaugat cate un JTextField unde sunt introduse datele utilizatorului, un JTextLabel care specifică cum vor fi evaluate cele două polinoame, care este primul și care este cel de-al doilea. A se remarca faptul că pentru operația de integrare și derivare datele trebuie introduse în primul polinom. De asemenea cele doua JPanel-uri au cate un buton de reset, cu care utilizatorul poate șterge tot conținutul JTextField-urilor.

În cel de-al treilea JPanel sunt puse două JTextLabel, unul care afișează polinomul rezultat și unul care specifică faptul că acel polinom este rezultatul. JPanel-urile 4 si 5 sunt rezervate operației de împărțire, afișând câtul și restul.

Înainte de cele 6 butoane care se afla la baza JPanel-ului principal, se găsește un JTextLabel care specifică ce operație s-a efectuat. Am ales să implementez astfel interfața deoarece panoul principal are ca și layout stabilit BoxLayout iar cele 5 panouri au layout-ul implicit. Panoul de butoane are ca și layout grid layout.

* **Rezultate**

Pentru acest pas, am folosit framework-ul JUnit.

Metoda assertEquals(Object expected, Object actual) afirmă că cele două obiecte sunt egale, iar în caz de neegalitate se afișează message.

Metoda assertNotEquals(Object expected, Object actual) afirmă că cele doua sunt diferite, iar în cazul în care sunt egale, afișează mesaj de eroare.

* Date de test pentru operația de **adunare**:

Polinom p1 = new Polinom("+1x^2+1x^1+2");

Polinom p2 = new Polinom("+1x^1+1");

Polinom rezultatAsteptat = new Polinom("+1x^2+2x^1+3");

Polinom rezultatObtinut = new Polinom();

rezultatObtinut = OperatiiPolinom.aduna(p1, p2);

Cele două polinoame care conțineau rezultatul așteptat și rezultatul obtinut au fost comparate element cu element. Pentru testul pozitiv am folosit metoda assertEquals iar pentru cel negativ, am eronat intenționat rezultatul așteptat și am folosit metoda assertNotEquals, pentru a demonstra că cele două nu sunt egale.

* Date de test pentru operația de **scădere:**

Polinom p1 = new Polinom("+1x^2+1x^1+2");

Polinom p2 = new Polinom("+1x^1+1");

Polinom rezultatAsteptat = new Polinom("+1x^2");

Polinom rezultatObtinut = new Polinom();

rezultatObtinut = OperatiiPolinom.scade(p1, p2);

La fel ca și la operația de adunare, este un test pozitiv și unul negativ.

* Date de test pentru operația de **înmulțire:**

Polinom p1 = new Polinom("+1x^2+1x^1+2");

Polinom p2 = new Polinom("+1x^1+1");

Polinom rezultatAsteptat = new Polinom("+1x^3+2x^2+3x^1+2");

Polinom rezultatObtinut = new Polinom();

rezultatObtinut = OperatiiPolinom.inmultire(p1, p2);

* Date de test pentru operația de **derivare:**

Polinom p1 = new Polinom("+1x^2+1x^1+2");

Polinom rezultatAsteptat = new Polinom();

Monom m1 = new Monom(2f,1f);

Monom m2 = new Monom(1f,1f);

rezultatAsteptat.getPolinom().add(m1);

rezultatAsteptat.getPolinom().add(m2);

Polinom rezultatObtinut = new Polinom();

* Date de test pentru operația de **integrare:**

Polinom p1 = new Polinom("+3x^2+1");

Polinom rezultatAsteptat = new Polinom();

Monom m1 = new Monom(1f,3f);

Monom m2 = new Monom(1f,1f);

rezultatAsteptat.getPolinom().add(m1);

rezultatAsteptat.getPolinom().add(m2);

Polinom rezultatObtinut = new Polinom();

rezultatObtinut = OperatiiPolinom.integrare(p1);

* Date de test pentru operația de **împărțire**:

Polinom p1 = new Polinom("+1x^2+2x^1+1");

Polinom p2 = new Polinom("+1x^1");

ArrayList<Polinom > rezultatAsteptat = new ArrayList<Polinom>();

ArrayList<Polinom> rezultatObtinut = new ArrayList<Polinom>();

rezultatObtinut = OperatiiPolinom.impartire(p1, p2);

* Date de test pentru operația de **prelucrare a stringurilor :**

Polinom p1 = new Polinom("+1x^2+1x^1+1");

Polinom p2 = new Polinom();

Monom monom1 = new Monom(1f,2f);

Monom monom2 = new Monom(1f,1f);

Monom monom3 = new Monom(1f,0f);

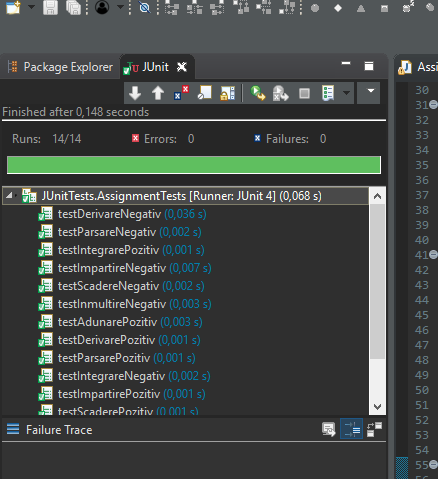
p2.getPolinom().add(monom1);

p2.getPolinom().add(monom2);

p2.getPolinom().add(monom3);

S-a reușit crearea unei aplicații care să realizeze operațiile uzuale întâlnite în lucrul cu polinoame. Aplicația oferă o interfață grafică intuitivă și tratează eventualele greșeli de introducere a datelor de intrare prin emiterea de avertismente specifice greșelii de utilizare.

Cazurile de test menționate mai sus au fost trecute cu bine, rezultatele operațiilor fiind cele așteptate:



**Mode de utilizare a aplicației**

Aplicația realizează operațiile de adunare, scădere, înmulțire, împărțire, derivare și integrare.

Polinoamele sunt introduse de către utilizator în două câmpuri în fereastra aplicației. Inclusiv primul element din polinom trebuie introdus cu semn, +/-, iar coeficientul trebuie specificat, chiar daca acesta este 1.

Un exemplu de test introdus corect ar fi: +1x^2+2x^1+1. Se poate observa că termenul liber nu are nevoie de x^0, la celalte fiind nevoie sa fie introdus. În cazul în care coeficienții sunt introduși în alt format sau se introduc alte simboluri, pe ecran va apărea un dialor prietenos cu utilizatorul care îl va avertiza ca a introdus date invalide.

Dupa ce se introduc coeficienții polinomului în mod corect se pot efectua operații asupra lor prin apăsarea butoanelor. Dacă se dorește introducerea unor polinoame noi, se folosește butonul de reset care joacă rolul de a curăța textField-urile destinate rezultatului.

* **Concluzii**

În aceasă temă am învățat să structurez o aplicație folosind întâi descrierea în limbajul UML.

**Dezvoltări ulterioare:**

* Introducerea de noi operații ce se pot efectua asupra polinoamelor.
* Oferirea functionalității de trasare grafic a unui polinom
* Oferirea posibilității rezultatului obținut în urma unei operații.
* **Bibliografie**

<http://ro.wikipedia.org/wiki/Polinom>

<http://meditatiionline.ro/44100-114-0-0-0-Formule_Matematica_Polinoame.html>

<http://www.math.wfu.edu/Math105/Operations%20on%20Polynomials(revised).pdf>