DOCUMENTATIE TEMA 1

CALCULATOR POLINOAME

Student: Moldovan Teodora
Grupa: 30225
Facultatea de Automatica si Calculatoare
Profesor Laborator : Antal Marcel

Cuprins

1. Obiective

1.1. Obiectiv Principal

Aceasta tema presupune propunerea, proiectarea si implementarea unui sistem de procesare a polinoamelor de o singura variabila cu coeficienti intregi. Calculatorul de polinoame trebuie sa efectueze atat operatii cu doi operanzi(adunare, scadere, inmultire, impartire), cat si operatii cu un singur operand (derivare, integrare). Pentru a facilita lucrul cu polinoamele, programul trebuie sa dispuna de o interfata grafica.

1.2. Obective Secundare

Obiectiv Secundar	Descriere	Capitol
Analiza problemei si presupuneri	Se analizeaza diferite moduri de abordare a problemei propuse.	2
Alegerea structurilor de date	Prezentarea structurii de date alese si a argumentelor	3
Impartirea pe clase	Diagrama de clase si descriere a functionalitatii claselor	3,4
Dezvoltarea algoritmilor	Propunerea unor metode de implementare a a algoritmilor pentru operatiile propuse	3
Implementarea solutiei	Detalii referitoare la implementare in urma analizei realizate	4
Testare	Rezultatele catorva teste realizate direct sau cu JUnit	5

2. Analiza Problemei

Utilizatorul trebuie sa poata efectua doua tipuri de operatii: cele cu un singur operand si cele cu doi operanzi. In cazul in care operatia are nevoie de doi operanzi, solutia problemei ar trebui sa afiseze un mesaj in cazul in care nu se introduce decat un singur operand. Pe de alta parte, pentru operatiile unare se poate folosi doar unul dintre cele doua polinoame introduse si anume primul. In ambele cazuri, polinomul ar trebui sa fie introdus intr-un format correct, pe care aplicatia sa il poata recunoaste si procesa ulterior, iar in caz contrar sa afiseze un mesaj pentru ca utilizatorul sa poata reintroduce polinomul in mod corect. Pentru simplificarea implementarii operanzii introdusi si rezultatele nu vor fi memorate. La fiecare comanda introdusa se vor citi operanzi corespunzatori, se vor procesa si se va executa operatia dorita, iar apoi rezultatul va fi afisat.

3. Proiectare

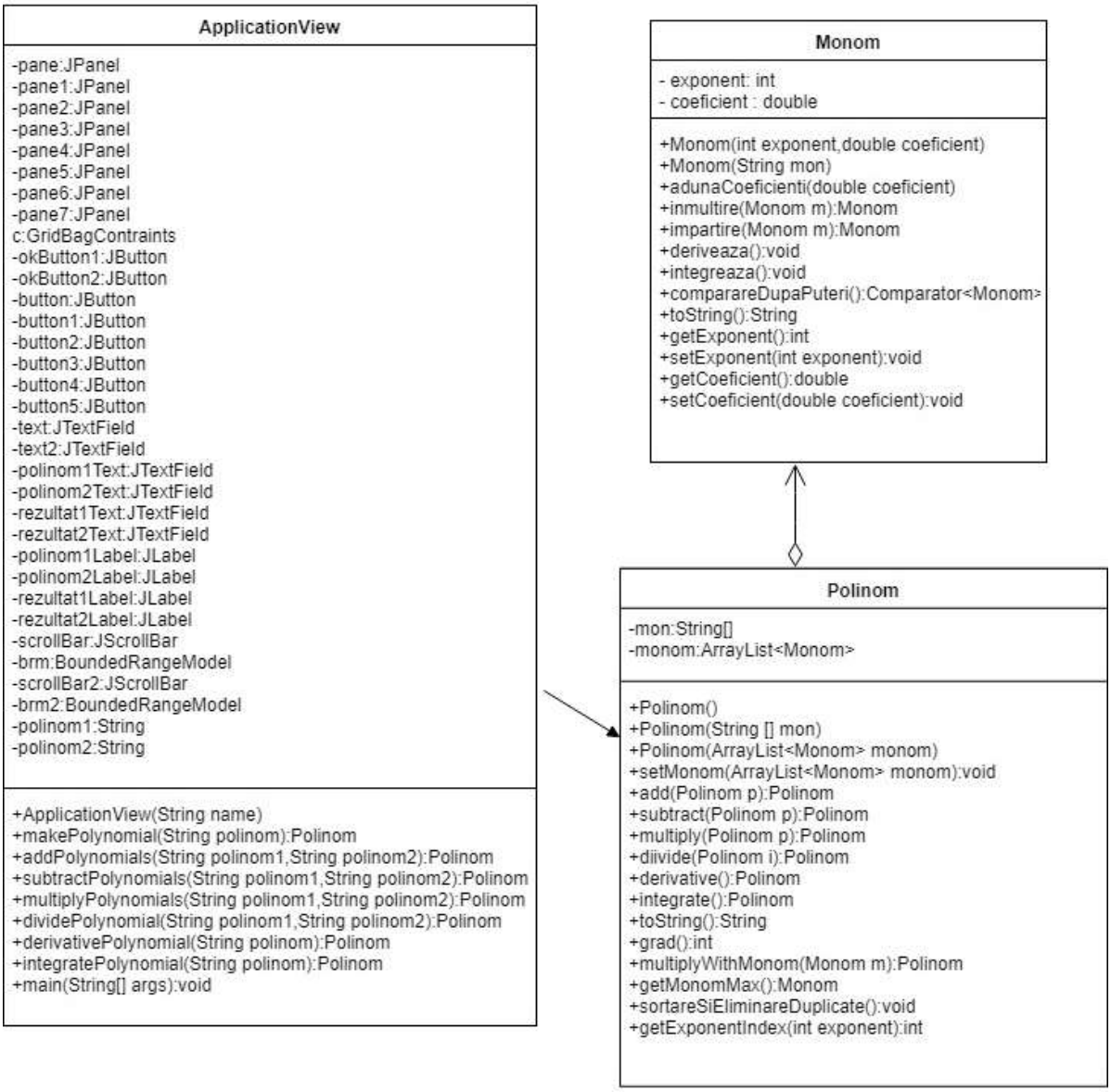
3.1. Structuri de date

Una dintre primele probleme aparute in faza de proiectare presupune alegerea unor structuri de date corespunzatoare modului de lucru cu clasele si stocarii anumitor valori pe perioada efectuarii operatiilor dorite.

O abordare posibila ar fi construirea unui vector in care sa se stocheze pe pozitia i coeficientul corespunzator fiecarui exponent. Pentru realizarea acestui proiect s-a ales in schimb folosirea colectiilor de care dispune Java si mai exact a unui ArrayList.

Un ArrayList este o colectie simpla care poate stoca orice tip de obiect, fapt ce permite utilizarea ArrayList-ului pentru definirea unui polinom. Asadar, un polinom este defapt un ArrayList de monoame. Aceasta clasa permite dimesiuni variabile ale sirului, crescand ca marime la adaugarea fiecarui element nou. Parcurgerea unui ArrayList este mai usoara, deoarece se pot folosi bucle for-each. De asemenea, clasa ArrayList dispune de anumite metode care faciliteaza anumite operatii necesare pentru realizarea proiectului cum ar fi: add (Element e), contains (Element e), indexOf (Element e), remove (int index), remove (Element e) , get(int index), size(), isEmpty().

3.2. Diagrama de clase



3.3. Algoritmi

Pentru efectuarea operatiilor necesare (adunare, scadere, inmultire, impartire, derivare si integrare) este necesara integerea notiunii de polinom si a operatiilor care se pot efectua asupra acestuia.

In practica, un polinom este o expresie construita dintr-una sau mai multe variabile si constante, folosind doar operatii de adunare, scadere, inmultire si ridicare la putere. Polinoamele sunt construite din termeni numiti monoame, care sunt alcatuite dintr-o constanta (numita coeficient) inmultita cu una sau mai multe variabile. Fiecare variabila poate avea un exponent constant întreg pozitiv.

Proprietatiile elementare ale polinoamelor sunt:

1. Suma a două polinoame este un polinom
2. Produsul a două polinoame este un polinom
3. Derivata unui polinom este un polinom
4. Primitiva unui polinom este un polinom

Avand in vedere aceste aspecte se poate incepe definirea modului de lucru astfel incat operatiile sa aiba rezultate corecte.

Suma si diferenta a doua polinoame se realizeaza prin adunarea sau scaderea coeficientiilor a doua monoame daca exponentul este egal. Monoamele care apar in unul singur dintre cele doua polinoame se adauga la rezultat astfel: pentru adunare se aduna la rezultatul final, iar pentru scadere , daca monomul apartine primului polinom, se procedeaza ca in cazul adunarii, in caz contrar adaugandu-se monomul la rezultatul final cu semn schimbat.

Operatia de inmultire se realizeaza prin parcurgerea unuia dintre polinoame si, pe rand, inmultirea fiecarui monom din acesta, cu fiecare monom al celuilalt polinom. La final se poate obtine un polinom in care există mai multe monoame cu acelasi exponent, caz in care trebuie sa eliminam duplicatele si sa adunam coeficientii respectivi.

Cea mai complexa operatie care trebuie efectuata este impartirea, aceasta bazandu-se pe teorema impartirii cu rest. Modul in care va fi implementata impartirea necesita functionarea corecta a celorlalte trei operatii cu doi operanzi(adunare, scadere, inmultire). Relatia pe care se bazeaza operatia de impartire a polinoamelor este: D=I*C+R, unde D este deimpartitul, I este impartitorul, C este catul si R este restul.

Primul termen al deimpartitului se imparte la primul termen al impartitorului, astfel obtinandu-se primul monom din cat. Acest monom se inmulteste cu fiecare monom din impartitor, iar rezultatul astfel obtinut se scade din deimpartit si se obtine primul rest. Algoritmul se repeta pana cand restul obtinut are gradul mai mic decat gradul impartitorului.

Operatiile de derivare si de integrare se efectueaza iterand prin monoamele polinomului si pentru fiecare monom se aplica operatia de derivare sau de integrare. In cazul derivarii se inmulteste coeficientul cu puterea initiala si apoi, exponentul se decrementeaza cu o unitate. Pentru integrare, care este operatia opusa derivarii, se incrementeaza cu o unitate exponentul, iar valoare coeficientului nou se calculeaza prin impartirea coeficientului initial cu valoarea exponentului in urma incrementarii sale.

4. Implementare

Metoda de implementare aleasa contine 4 pachete:

- Unul in care se gaseste clasa Main(contine metoda main), reponsabil pentru crearea unei instante a clasei ApplicationView, mai exact pentru generarea frame-ului care va contine panel-urile din care este alcatuita interfata grafica cu utilizatorul.
- Al doilea pachet contine clasa ApplicationView care contine elementele interfetei grafice si controlul aplicatiei.
- Al treilea pachet este cel care contine clasele necesare pentru a implementa operatiile cerute si anume clasa Polinom si clasa Monom.
- Ultimul pachet este folosit pentru testarea metodelor cu JUnit

Clasa ApplicationView este responsabila pentru modul in care sunt aranjate componentele folosite in interfata grafica cu utilizatorul. Pentru introducerea polinoamelor se folosesc doua campuri JTextField in cazul in care operatia are doi operanzi (in cazul operatiilor unare, derivare si integrare, se va lua in considerare doar polinomul introdus in primul camp destinat introducerii polinoamelor). Dupa fiecare astfel de camp urmeaza un buton “OK”, a carui apasare duce la afisarea polinomului introdus in campul respectiv pentru verificarea corectitudinii acestuia. Polinoamele introduse, la apasarea butoanelor “OK” corespunzatoare campurilor, vor fi afisate astfel: cel din primul camp in campul JTextField corespunzator etichetei “First Polynomial”, iar urmatorul in dreptul etichetei “Second Polynomial”. Intre campurile JTextField de afisare a polinoamelor si cele de introducere se regasesc sase butoane, fiecare corespunzand uneia dintre cele 6 operatii care se pot efectua. Pentru afisarea rezultatului operatiilor există niste campuri speciale, care in faza initiala nu sunt vizibile, dar devin vizibile la apasarea butonului corespunzator operatiei dorite. Pentru operatia de impartire sunt

necesare doua campuri unul pentru cat si unul pentru rest, iar pentru celelalte operatii este suficient un singur camp. Daca s-au introdus corect polinoamele, operatia se va efectua, va aparea campul/campurile necesare afisarii rezultatului si o eticheta care marcheaza ce operatie s-a efectuat in dreptul campului.

Introducerea corecta a unui polinom presupune introducerea lui sub forma "ax^2+bx^1+cx^0"(de exemplu, unde a,b,c sunt coeficienti), dar se accepta si omiterea coefientiilor in cazul in care sunt 1 sau -1, a x^0 sau a omiterii exponentului daca acesta este 1.

Pe langa modul in care arata si functioneaza interfata, in clasa ApplicationView se mai regasesc niste metode auxiliare pentru transmiterea mai departe la clasa polinom a sirului introdus. Prima metoda si cea mai importanta,"Polinom makePolynomial(String polinom)" inlocuieste "- " cu "+-" si apoi imparte sirul de caractere dupa ""+". Astfel se creeaza un tablou de siruri de caractere care este transmis la clasa Polinom care are un constructor special, utilizat doar la trasformarea polinoamelor din siruri de caractere in monoame, care apoi sunt adaugate in ArrayList-ul de monoame al polinomului. Functia returneaza polinomul astfel creat pentru a-l putea utiliza in alte functii sau pentru a-l afisa. Celelalte metode sunt: addPolynomials, subtractPolynomials, multiplyPolynomials, dividePolynomials, derivativePolynomial, integratePolynomial. Toate primesc ca argument unul sau doua siruri de caractere, apeleaza metoda makePolynomial pentru a le transforma in polinoame, iar apoi apeleaza metodele corespunzatoare din clasa Polinom. Acestea returneaza un polinom.

Clasa Polinom are o singura varibila instanta, variabila monom, care este un ArrayList de Monoame. Aceasta clasa are 3 contructori, unul folosit la crearea unui polinom atunci cand se foloseste metoda makePolynomial din clasa ApplicationView, unul care creeaza o instanta goala a clasei Polinom (se creeaza astfel ArrayList-ul de Monoame la care se poate adauga) folosita in general la construirea pas cu pas a unui nou polinom si altul utilizat pentru copierea listei de monoame. Metodele continute in aceasta clasa realizeaza in mare parte operatiile cerute, iar pe langa aceste metode mai există cateva metode auxiliare pentru a nu efectua aceleasi operatii de foarte multe ori. Metodele care realizeaza operatiile sunt: add(Polinom p), subtract(Polinom p), multiply(Polinom p), divide(Polinom i), derivative() si integrate(). Aceste metode functioneaza pe baza algoritmiilor descrisi in sectiunea 3.3. Metodele auxiliare sunt: grad() – returneaza gradul polinomului, multiplyWithMonom(Monom m)- returneaza un polinom in urma inmultirii unui polinom cu un monom, getMonomMax() –returneaza monomul maxim dintr-un polinom, getExponentIndex(int index)-returneaza indexul exponentului dat ca si parametru din ArrayList. In clasa Polinom se suprascrie metoda toString folosita pentru afisarea polinomului in forma finala (cu coeficienti reali) si mai există o metoda numita sortareSiEliminareDuplicate() care sorteaza polinomul pe baza

exponentului monomului cu ajutorul metodei sort din clasa Collections. Polinomul este astfel sortat in ordine descrescatoare a exponentiilor si daca există duplicate, acestea sunt eliminate prin adunare coeficientilor si pastrarea exponentului comun. Aceasta metoda este foarte utila dupa operatia de inmultire, dar contribuie si la efectuarea corecta a celorlalte operatii in cazul in care utilizatorul nu introduce un polinom sortat si minim.

Clasa Monom are doua variabile instanta, exponentul care este de tipul int (intreg) si coeficientul de tipul double (real). Decizia sa existe un coeficient de tipul double a fost luata in urma analizarii modului de efectuare a operatiei de integrare, care in majoritatea cazurilor obtine coeficienti reali pentru monoame. Primul constructor este cel utlizat la inceput, la despartirea sirurilor care vin din clasa Polinom in coeficient si exponent. Pentru aceasta se foloseste parsarea bazata pe regex(Regular Expression) si o serie de teste pentru corectitudine, dar si pentru tratarea catorva cazuri in care polinomul nu a fost introdus exact in forma indicata(ex. ax^2+bx^2+cx^0). Cel de-al doilea constructor este folosit pentru formarea de monoame atunci cand se cunosc deja exponentul si coefientul (int si double). Metodele din aceasta clasa sunt: adunaCoeficienti(double), compararеDupaPuteri()-returneaza un Comparator<Monom> pentru metoda sort folosita in sortareSiEliminareDuplicate() din Polinom, toString()-metode auxiliara care transforma un monom(coeficientul si exponentul) intr-un string de forma ax^b,unde a reprezinta coeficientul si b exponentul. Tot pentru simplificarea operatiilor din clasa Polinom, clasa Monom contine metodele: inmultire(Monom m), impartire(Monom m), deriveaza(), integreaza() si gettere si settere pentru coeficienti.

5. Testare si Rezultate

O prima operatie de testare care se executa ar fi introducerea unui polinom in fiecare dintre cele doua campuri destinate acestui lucru si apasarea butoanelor “OK” corespunzatoare, iar apoi verificarea ca polinoamele s-au introdus corect si ca sunt afisate corect in JTextField-urilor corespunzatoare etichetelor “First Polynomial” si “Second Polynomial”.

Vom lua cate un exemplu si pentru testarea operatiilor, atat cele unare cat si cele binare.

Pentru operatiile unare (derivare si integrare) se va folosi doar primul camp disponibil in interfata grafica a aplicatiei pentru introducerea polinomului. Se va apasa butonul

corepunzator operatiei dorite, moment in care va devein viizibil un camp in care va fi afisat rezultatul, campul fiind precedat de o eticheta care indica operatia efectuata.

Procedeul de testare este asemanator si pentru testarea operatiilor cu doi operanzi. Singura exceptie este impartirea. Pentru aceasta, se introduce doi operanzi, iar rezultatul va fi afisat in doua campuri distincte, primul va fi catul, iar al doilea restul impartirii.

	Date de intrare		Rezultat asteptat	Rezultat obtinut	PASS/FAIL
adunare	2x^2+4x^3-1	3x^2+5x+6	4.0x^3+5.0x^2+5.0x^1+5.0	4.0x^3+5.0x^2+5.0x^1+5.0	PASS
scadere	2x^2+4x^3-1	3x^2+5x+6	4.0x^3-1.0x^2-5.0x^1-7.0	4.0x^3-1.0x^2-5.0x^1-7.0	PASS
inmultire	x+1	x+2	1.0x^2+3.0x^1+2.0	1.0x^2+3.0x^1+2.0	PASS
impartire	2x^2+4x+3	x+1	2.0x^1+2.0 si 1.0	2.0x^1+2.0 si 1.0	PASS
derivare	2x^2+4x+3		4.0x^1+4.0	4.0x^1+4.0	PASS
integrare	x+1		0.5x^2+1.0x^1	0.5x^2+1.0x^1	PASS

Toate metodele care implementeaza cele sase operatii sunt testate si prin intermediul testelor Junit. Fiecare test poate fi executat individual prin intermediul claselor de tipul JUnit Test Case sau toate testele pot fie executate concomitent prin intermediul Suitei de Test , clasa care poarta numele AllTests. Aceste teste se afla intr-un al patrulea pachet pe care doar l-am amintit anterior.

6. Concluzii si Dezvoltari Ulterioare

Acest proiect este unul care lasa programatorului o posibilitate crescuta de a interpreta problema si modalitatiile de rezolvare, de aceea din punct de vedere al dezvoltariilor ulterioare există numeroase posibilitati. Cateva dintre acestea ar fi adaugarea unei reprezentari grafice, in reper cartezian al polinomului sau o operatie pentru aflarea radaciniilor unui polinom . Pe langa posibilitatiile de dezvoltare din punct de vedere al functionalitatii aplicatiei, aceasta poate fi imbunatatita prin realizarea unei interfete mai bine organizate si mai vizibile.

7. Bibliografie

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

http://blog.zeltera.eu/?p=370

https://docs.oracle.com/javase/7/docs/api/java/util/ArrayList.html