Documentatie

Polynomial calculator



Tehnici de programare

Tema1

Realizat de Balc Horia-Ovidiu

Grupa 30227

# Cerinte Functionale

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

Operatiile pe care le suporta programul sunt:

* Citirea unui polinom de la tastatura sub forma de String cu formatul :

a0x^n+a1x^(n-1)+a2x^(n-2)+...+a(n-1)^x^1+anx^0, unde si

* Adunarea a doua polinoame P1(x) + P2(x)
* Scaderea a doua polinoame P1(x) - P2(x)
* Inmultirea a doua plinoame P1(x) \* P2(x)
* Impartirea a doua polinoame P1(x) / P2(x)
* Derivarea unui polinom P1(x)’
* Integrarea unui polinom ∫ P1(x
* Testarea unitara

2.Obiectivul temei

2.1 Obiectivul principal

Obiectivul temei este crearea ,in limbajul JAVA, a unui calculator pentru polinoame cu ajutorul caruia sa se poata face operatiile :adunare, scadere, inmultire, impartire, integrare si derivare.

2.2 Obiectivul secundare

Obiectivele secundare sunt: - folosirea programarii obiectuale prin utilizarea incapsularii

- scrierea variabilelor in stilul Camel Case

-utilizarea metodelor(maxim 30 de linii) si claselor( maxim 300 de linii)

- unei interfete bazata pe o arhitectura Model View Controller

- folosirea testarii unitare (junit),

- citirea polinomului ca string si convertirea acestuia din urma

- creearea polinomului ca fiind o lista de monoame

-monom-o clasa care contine un coeficient (zecimal) si o putere (intreg)

-utilizarea foreach-ului in loc de for-ul simplu folosit frecvent in limbajul C

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

3.1Analiza problemei

Activitatile pe care poate sa le faca utilizatorul sunt:

- sa scrie cele doua polinoame in TextField-urile “Input1” si respectiv “Input2” forma unui monom fiind aceasta: “coeficientx^exponent”), unde coefficient este salvat ca tip double, iar exponentul ca intreg

-sa apese click stanga pe butonul “Adunare”, in urma a acestiu eveniment in TextField-ul “Rezultat1” se va afisa rezultatul adunarii celor doua polinoame date in “Input1” si “Input2”

-sa apese click stanga pe butonul “Scadere”, in urma a acestiu eveniment in TextField-ul “Rezultat1” se va afisa rezultatul scaderii a celor doua polinoame date in “Input1” si “Input2”

-sa apese click stanga pe butonul “Inmultire”, in urma a acestiu eveniment in TextField-ul “Rezultat1” se va afisa rezultatul inmultirii celor doua polinoame date in “Input1” si “Input2”

-sa apese click stanga pe butonul “Integrala”, in urma a acestiu eveniment in TextField-ul “Rezultat1” se va afisa rezultatul integralei a primului polinoam dat in “Input1”, considerand constanta de integrare 0

-sa apese click stanga pe butonul “Derivare”, in urma a acestiu eveniment in TextField-ul “Rezultat1” se va afisa rezultatul derivarii a primului polinoam dat in “Input1”

-sa apese click stanga pe butonul “Impartire”, in urma a acestiu eveniment in TextField-ul “Rezultat1” se va afisa catul impartirii a celor doua polinoame date in “Input1” si “Input2”, iar restul impartirii se va afisa in TextField-ul “Rezultat2”

-sa apese click stanga pe butonul “Clear”, in urma a acestiu eveniment continutul tuturor TextField-urilor dispar, adica sunt inlocuite cu stringul null

3.2Scenarii si cazuri de utilizare

Deoarece e posibil ca unii utilizatori sa introduca polinoame gresite, de fiecare data cand un polinom introdus in TextField are un format gresit fata de cel normal al unui polinom, sau sunt introduse diferite Stringuri care nu formeza un polinom ( ex. Ana are mere ), la apasarea unui buton ce semnifica operatia va aparea o fereastra ce va avertiza utilizatorul ca polinomul introdus de el nu este valid cu mesajul “bad input” insa, daca utilizatorul introduce un polinom dezordonat ( gradele variabilelor nu sunt in ordine ), programul il va lua ca valid, ordonandu-l in TextField-ul in care se afiseaza rezultatul.

4.Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator)

4.1Decizii de proiectare si Structuri de date

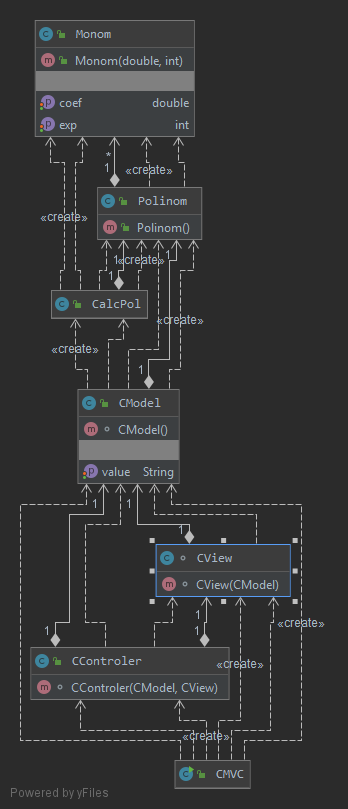
Am ales sa utilizez o clasa Polinom case contine ca element principal o lista (ArrayList) de monoame, unde Monom este o clasa ce contine doua variabile de tip private una care reptrezinta coeficientul (si este reala), iar cealalta exponentul (si este intreaga).

De asemenea am decis sa folosesc o clasa noua doar pentru operatiile cu polinoame numita CalcPol.

4.1.1Interfata utilizator

Interfata utilizator contine 4 TextField-uri fiecare fiind insotit de cate un label si 7 Butoane.Totodata interfata este sectionata in 3 parti nord(Inputuri), central(Outputuri) si sud(butoanele).

4.2Diagrame UML



5.Implementare

Clasa Monom

Aceasta clasa implementeaza interfata default Comparable <Monom> pentru a putea suprascrie metoda compareTo

Am folsit getteri (getCoef si getExp) si setteri (setCoef si setExp) pentru variabilele coef (coeficient) si exp (putere)

Clasa Polinom

Aici am folosit urmatoarele metode:

-gradMax este o metoda ce returneaza, in urma cautarii cu ajutorul unui foreach, monomul cu gradul maxim din lista de monoame a polinomului primit ca parametru

-sortare este o metoda ce sorteaza ArrayList-ul de monoame primit ca parametru descrescator in functie de grad cu ajutorul suprascrierii metodei compareTo din Clasa Monom

-adauga este o metoda ce adauga un mono m(dat ca parametru) in ArrayList-ul de monoame al polinomului (cu care il apelezi)

doar daca nu exista element in lista cu acelasi grad, altfel se va modifica doar coeficientul monomului existent cu acelasi grad. Metoda returneaza 0 daca a fost adaugat in lista de monoame si 1 in caz contrar.Verificarea daca exista un monom cu acelasi grad in polinom este facutacu ajutorul unui if, care verifica daca gradul monomului dat ca parametru coincide cu gradul monomului din lista(incepand cu primul), aflat intr-un foreach care parcurge toate elementele listei. De asemenea instructiunea: elem.removeIf(x->(x.getCoef()==0)); are rolul de a sterge toate monoamele care au coeficientul 0.

-schimbaSemn este o metoda ce cu ajutorul unui foreach, in care setam coeficientul (cu setCoef) cu negatul sau (-getCoef), schimba semnul fiecarui monom din lista de monoame a polinomului dat ca parametru.

-toString este o metoda, care suprascrie metoda default, ce returneaza stringul s care este initializat la null (la inceputul metodei), si care ,dupa ce se sorteaza polinomul (cu care am apelat functia) cu ajutorul metodei sortare descrisa mai sus, pentru fiecare monom din lista lui s i se va adauga semnul + sau - si un string de formatul acesta:“coeficientulX^putere” pentru fiecare monom din lista (cu ajutorul unui foreach)

Clasa CalcPol

Aici am folosit urmatoarele metode:

-regexx este o metoda ce primeste ca parametrii un string si un polinom care converteste stringul intr-un polinom. Folosim metodele predefinite replaceAll, pentru a inlocui - cu +-, si split pentru a imparti stringul dat ca parametru in mai multe stringuri, impartitorul fiind semnul “+”,salvate intr-un sir de stringuri.De asemenea, utilizam un foreach cu ajutorul caruia parcurgem sirul de stringuri, daca stringul nu este null atunci se aplica split unde “x^” are rol de impartitor, astfel vom obtine 2 stringuri primul fiind convertit din string in double valoarea este pastrata in variabila locala coef, iar al doilea din string in int, valoarea este pastrata in variabila locala put. Ulterior vom initializa un monom cu valoarea coef pentru coefficient si cu valoarea put pentru exponent, dupa care vom adauga monomul in lista polinomului din parametrul functiei.

-adunare este o metoda ce primeste ca parametrii doua polinoame si returneaza alt polinom care reprezinta suma primelor doua.Metoda utilizeaza doua foreach-uri unul care parcurge lista de monoame a primului polinom si in care se apeleaza metoda adauga (polinomulRezultat.adauga(monom)) si altul face acelasi lucru utilizand celalalt polinom. Polinomul este returnat correct cu ajutorul instructiunii : elem.removeIf(x->(x.getCoef()==0)); care nu sterge toate monoamele care au coeficientul 0, dupa care se returneaza polinomul rezultat.

-scadere este o metoda ce primeste ca parametrii doua polinoame si returneaza alt polinom care reprezinta diferenta primelor doua.Metoda apeleaza metoda schimbaSemn ,explicate mai sus, aplicata pe al doilea polinom pentru a se putea utilize ulterior metoda adunare (descrisa precedent) intre primul polinom si polinomul negat (cu -) , dupa care se returneaza polinomul rezultat.

- inmultireMonom este o metoda ce primeste ca parametrii un polinoam si un monom si returneaza alt polinom care reprezinta produsul polinomului cu monomul vazut ca fiind un polinom cu un singur element. Metoda foloseste un foreach in care se inmulteste coeficientul monomului cu fiecare coeficient de monom din lista de monoame al polinomului si in care se aduna exponentul monomului ci fiecare putere de monom dim lista de monoame a polinomului.dupa aceste operartii se adauga noul monom in polinomul rezultat cu ajutorul metodei adauga(tot in foreach). Dupa iesirea din structura repetitive se returneaza polinomul rezultat.

-inmultire este o metoda ce primeste ca parametrii doua polinoame si returneaza alt polinom care reprezinta produsul primelor doua.Metoda utiilizeaza 2 foreach-uri imbricate ,unul care parcurge lista de monoame unui polinom, iar altul care il traverseaza pe celalat. In aceasta instructiune este declarat un monom care are exponentul setat cu suma puterilor celor doua monoame selectate de foreach si coeficientul ca produsul coeficientilor celor doua elemente.Acest monom este imediat inserat in lista cu ajutorul metodei adauga.Dupa iesirea din cele doua foreach-uri imbricate se va returna polinomul rezultat.

-impartire este o metoda ce primeste ca parametrii 4 polinoame din care primul este deimpartitorul al doilea este impartitorul al treilea este catul si ultimul reprezinta restul. Metoda foloseste un polinom pol1 ca fiind copie a deimpartitului si altul aux initializat ca fiind null.De asemenea am utilizat o secventa while cu conditia !pol1.toString().isEmpty() && deimp!=d.adunare(d.inmultire(cat,imp),rest) care verifica daca pol1 nu este null si daca conditia teoremei impartiirii cu rest nu este indeplinita,cand una din aceste conditii au loc inseamna ca am gasit catul si restul dorit si se iese din while. Daca in structura repetitive pol1 are gradul maxim (cu ajutorul metodei gradMax) mai mix decat cel maxil al impartitorului inseama ca restul ia valoare lui pol1 si pol1 devine null(I se atribuie aux) . Altfel, utilizam o variabila reala c (tip double) care ia valoarea raportului dintre coeficientii monoamelor cu grad maxim a polinoamelor pol1 si impartitor si o variabila intreaga e (tip int) care ia valoarea diferentei dintre exponentul maxim a lui pol1 si ce al impartitorului.Se creeaza un monom m cu coeficientul c si cu exponentul e care va fi adaugat in lista de monoame a polinomului cat (cu ajutorul metodei adauga), dupa care pol1 val lua valoarea lui: pol1-imp\*m, care va fi calculate ccu ajutorul functiilor scadere si inmultireMonom.

-integrare este o metoda ce primeste ca parametru un polinom si returneaza alt polinom care reprezinta integrala acestuia cu constanta 0.Metoda foloseste un foreach in care initializam un monom m cu coef si exp 0.Exponentul acestuia il setam la puterea+1 a monomului din lista si coefficientul acestuia il setam ca fiind raportul dintre coeficientul monomului din lista si exponentul monomului m. Adaugam in lista de monoame a polinomului rezultat monomul m.Dupa iesirea din structura repetitive returnam polinomul rezultat.

-derivare este o metoda ce primeste ca parametru un polinom si returneaza alt polinom care reprezinta derivata acestuia. La inceput se initializeaza polinomul rezultat si un monom m cu coeficientul si puterea 0. Metoda utilizeaza un foreach in care se initalizeaza un monom m1 cu coeficientul si exponent 0, dupa care daca elementul curent din lista are puterea 0 se va adauga in lista de monoame a polinomului rezultat monomul m.Altfel,Monomul m1 se seteaza astfel: coeficientul ia valoarea produsului dintrecoeficientul si exponentul a monomului curent (din lista), iar exponentul va fi egal cu exponentul monomului current -1, ulterior se adauga monomul m in lista de monoame a polinomului rezultat. Dupa iesirea din structura repetitiva se sterge monomul m din lista polinomului rezultat, dupa care returneaza polinomul returnat.

Clasa CModel

In aceasta clasa am implementat operatiile prin intermediul metodelor descrise mai sus, in clasa CalcPol. Astfel, in metode se va folosi functia regexx pentru a transforma dintr-un string intr-un polinom, dupa care se aplica metoda specifica pentru operatia aleasa.

Clasa CView

In aceasta clasa avem declarate 4 TextField-uri(doua pentru inputuri si 2 pentru outputuri) 7 butoane (un buton pentru fiecare operatie). Constructorul clasei contine un container c, acesta contine 3 jpaneluri (content (nord), butoane(sud) , rezultat(central)), fiecare avand un new FlowLayout. In rezultat adaugam cele 2 TextField-uri ce rerezinta outputurile si 2 labeluri(cu textul “Rezultat1” si “Rezultat2”), in content adaugam celelalte TextField-uri si 2 labeluri(cu textul “Input1” si “Input2”), in butoane adaugam toate cele 7 butoane.

Clasa aceasta contine metoda reset care seteaza continutul tuturor TextField-urilor cu stringul null.De asemenea tot in aceasta clasa adaugam Listenere pentru fiecare buton(operatie).

Clasa CControler

In aceasta clasa folosim un obiect de tip CModel si altul de tip CView.In Constructorul functiei adaugam cele 7 listenere pentru butoane(operatii).Am implementat 7 clase care implementeaza interfata ActionListener in care inputurile sunt preluate din TextFiel si declaram un polinom p care ia valoarea functiei din CModel care folosecte functia din clasa CalcPol, dupa care afisam rezultatele in TextFild-urile pentru output.

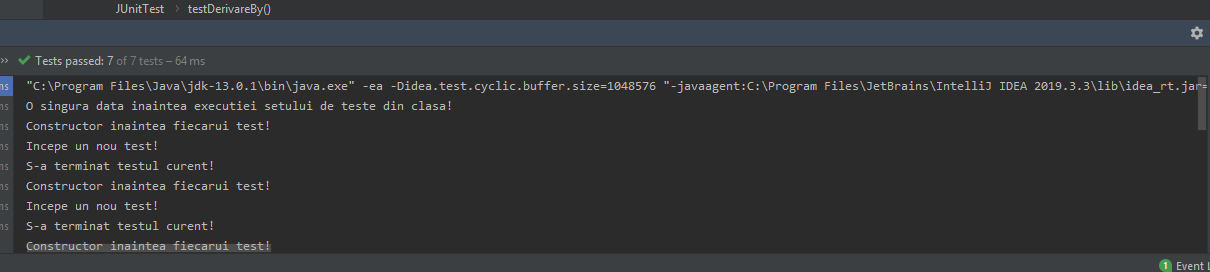
Clasa CMCV

Aceasta este clasa principala care contine o metoda static main in care instantiem un obiect de tip CModel, altul de tip CView si altul de til CControler,dupa care stetam variabila de tip CView ca fiin vizibila.

5. Rezultate

Clasa JUnitTest

In aceasta clasa am testat cele 7 operatii de pe butoane, cu un exemplu ales, toate avand mesaj de success.



6. Concluzii

6.1Ce s-a invatat

Mi-am imbunatatit cunostiintele in limbajul JAVA, mai ales in ceea ce consta crearea unei interfete si utilizarea testarii unitare.De asemenea, am invatat cum sa folossc metode precum split si replaceAll si cum sa impart un string intro lista de o structura de tip.

6.1Dezvoltari ulterioare

Imbunatatirea functiei regexx pentru a suporta mai multe variante de polinoame cum ar fi in loc de “1x^2” sa fie “x^2”.Totodata, o alta dezvoltare ar fi afisarea unui numar cu format intreg, nu real, daca numarul este intreg.

7.Bibliografie

-Am cautat pe <https://stackoverflow.com/> si pe <https://www.geeksforgeeks.org/> pentru implementrea functiei regexx, pentru interfata si pentru implementarea instructiunii elem.removeIf(x->(x.getCoef()==0));.

-De asemenea m-am inspirat din laboratoarele de POO din semestrul 1 (anul 2019-2020) pentru implementarea interfetei si a testarii unitare.