

Polynomial Calculator

Soponari Beatrice-Valentina

Grupa 30221

Anul 2

**Cuprins:**

1.Obiectivul temei......................................................................

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

3. Proiectare………………………………………………………………………….5

4. Implementare……………………………………………………………………6

5.Rezultate

6.Concluzii

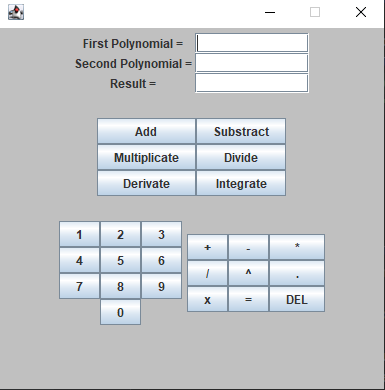
7.Bibliografie

**1.Obiectivul temei**

Scopul acestei teme este de a implementa operatiile de adunare, scadere, inmultire, impartire, derivare si integrare specifice calculului cu polinoame, utilizand un limbaj de programare orientat pe obiect, de a folosi modelarea datelor de intrare, prelucrarea si transformarea lor in obiectele necesare dezvoltarii ulterioare a aplicatiei si de a crea o interfata grafica astfel incat utilizatorul va putea introduce polinoamele si va putea alege operatiile care trebuie sa se efectueze. Obiectivele sunt obisnuirea cu anumite modele de date, prelucrarea datelor si structurarea cat mai eficienta si mai logica a rezolvarii, mai exact dezvoltarea unei aplicatii care prezinta o problema din lumea reala. Operatiile care se pot efectua sunt:

* Adunarea
* Scaderea
* Inmultirea
* Integrarea
* Derivarea

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

****

Problema proiectului presupune lucrul cu polinoame, realizarea unor operatii matematice, modelarea datelor de intrare si crearea unei interfete grafice pentru utilizator.

Aceasta este interfata grafica realizata, care permite inserarea a doua polinoame in campurile FirstPolynomial si SecondPolynomial, si afiseaza in campul Result rezultatul operatiei aplicate de utilizator prin apasarea unuia dintre butoanele asignate operatiilor.

Polinoamele se pot introduce prin tastatura sau prin butoanele create in partea de jos a imaginii. Polinoamele se vor introduce sub forma generala

P(x) = ax^b+cx^d+...+yx^z , unde a,c,y reprezinta coeficientii polinoamelor si b,d,z reprezinta puterile. Pentru inserarea polinoamelor se poate folosi tastatura sau se pot folosi butoanele din partea de jos care insereaza in textfieldurile FirstPolynomial si SecondPolynomial simbolurile afisate pe acestea (de ex. ‘1’, ’^’, ‘+’). Mai exista de asemenea si un buton ‘DEL’ desemnat sa stearga daca utilizatorul greseste la introducerea polinomului dorit. Polinoamele pot fi inserate in orice ordine a gradelor, atata timp cat sunt scrise inclusiv gradul si coeficientul 1 si 0, iar dupa realizarea unei anumite operatii alese de utilizator, acestea vor fi afisate in ordine descrescatoare, dupa gradele monoamelor.

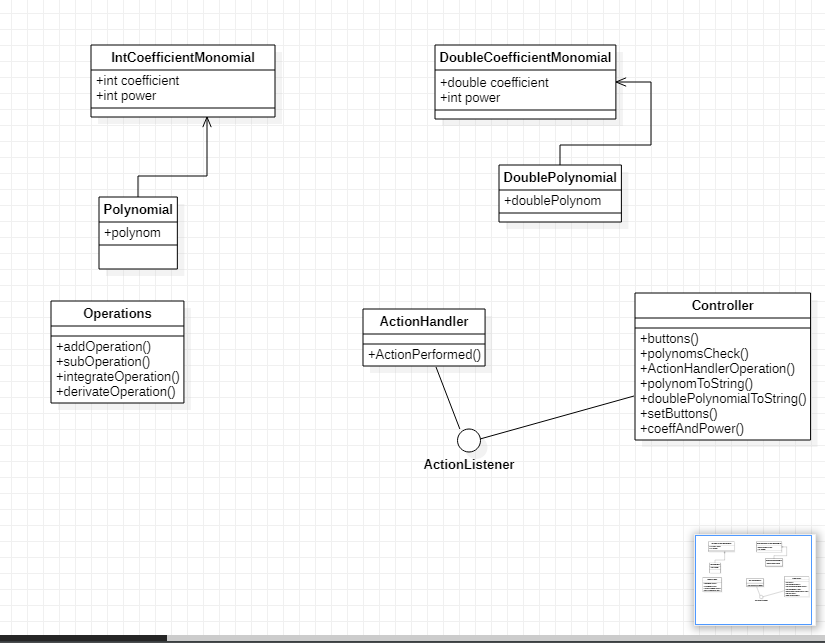
Dupa inserarea polinoamelor se poate alege operatia dorita si rezultatul se poate observa in campul Result, sub forma unui polinom, ordonat in ordinea descrescatoare a puterilor.Pentru oeratiile de adunare, scadere, inmultire si derivare rezultatele vor fi sub forma de polinoame cu coeficienti intregi. Pentru celelalte operatii reprezentarea polinoamelor va fi cu coeficienti de tip double pentru o reprezentare exacta a rezultatului. Datele au fost preluate din campurile FirstPolynomial si SecondPolynomial ca stringuri si mai apoi transformate in polinoame.

Scenariile de utilizare in care doar un polinom este introdus de catre utilizator in campurile destinate realizeaza o adunare sau scadere cu 0, inmultirea nu se realizeaza, iar in cazul integrarii si al derivarii se realizeaza pentru primul polinom din campul FirstPolynomial.Altfel, cand ambele polinoame sunt introduse se poate folosi atat adunarea cat si scaderea si inmultirea, operatii care se realizeaza corect in concordanta cu testele realizate folosind Junit.

In cazul inserarii unor polinoame care nu respecta formatul prezentat mai sus, in campul Result se va afisa mesajul “Polinoame introduse gresit”, pentru a ii semnala utilizatorului ca nu a respectat formatul impus si nu se va mai efectua operatia dorita.

**3.Proiectare**

Am realizat calculatorul de polinoame utilizand abstractizarea datelor din care au rezultat clasele IntCoefficientMonomial, DoubleCoefficientMonomial, Polynomial, DoublePolynomial, Controller, Operations, ActionHandler, Main si CalculatorTest pentru testarea corectitudinii operatiilor implementate. In implementarea lor am utilizat structuri de date de tip ArrayList, extensiile JFrame si TestCase, interfata ActionListener, din pachetele java.util.\*,java.swing.\*, java.awt.\*, junit.framework.TestCase, folosite atat pentru a lucra cu structurile de date cat si pentru a oferi o interfata potrivita pentru a putea fi folosita de catre utilizator.Am ales sa folosesc doua clase de la care sa pornesc, IntCoefficientMonomial si DoubleCoefficientMonomial pentru a putea realiza atat operatiile de adunare, scadere, inmultire si derivare care necesita monoame cu coeficienti intregi, dar si operatia de integrare si impartire, care vor avea ca rezultate monoame cu coeficienti reali. Clasale Polynomial si DoublePolynomial extind clasele IntCoefficientMonomial, respectiv DoubleCoefficientMonomial, sunt folosite pentru a implementa polinoamele, iar pentru operatii am ales sa folosesc alta clasa, si anume clasa Operations, pentru a putea lucra mai departe cu apelarea operatiilor pe polinoame.La inceput am creat o schema (diagrama UML) pentru a vedea concret, cel putin la nivel schematic, cum va arata proiectul finalizat. Am structurat clasele folosite in pachete dupa modelul View Model Controller astfel: clasa CalculatorGui apartine pachetul View, clasele IntCoefficientMonomial, DoubleCoefficentMonomial, Polynomial, DoublePolynomial si Operations apartin pachetului Model si clasa Controller apartine pachetului Controller.



**4.Implementare**

Implementarea incepe de la modelul de baza, monomul, cu coeficienti intregi sau reali, care realizeaza de fapt un polinom, asa ca primele clase definite au fost acestea.

Clasa IntCoefficientMonomial este definita de atributele coefficient si power, unde coefficient reprezinta coeficientul intreg al fiecarui monom si power reprezinta puterea fiecarui monom. La fel este si clasa DoubleCoefficientMonomial, diferenta esta ca tipul coeficientului din aceasta clasa este double.Le-am creat astfel deoarece desi coeficientii polinoamelor de intrare sunt numere intregi, la operatia de integrare si de impartire, coeficientii polinoamelor pot deveni de tip double si pentru o precizie mai buna a rezultatului final, impartirea si integrarea vor returna polinoame cu coeficienti de tip double. Pentru fiecare clasa avem constructotii, getteri si setteri pentru a putea utiliza si accesa valorile acestora.

Clasa Polynomial extinde clasa IntCoefficientMonomial si clasa DoublePolynomial extinde DoubleCoefficientMonomial.

Acestea contin o lista de monoame intregi si respectiv, clasa DoublePolynomial contine o lista de monoame de tip double care apartin unui polinom, pentru care sunt create metodele din clasa Operations. Lista se utilizeaza inserand pe rand cate un monom din string-urile preluate din campurile destinate scrierii de catre utilizator, in momentul in care acesta scrie polinoamele in ambele campuri(in cazul adunarii, scaderii, inmultirii) sau in cel putin unul(in cazul integrarii si derivarii), si dupa alegerea operatiei care se doreste a fi efectuata.

Clasa Operations implementeaza operatiile de adunare, scadere, inmultire, derivare si integrare a polinoamelor, fiecare metoda primind ca parametri unul sau doua polinoame, dupa caz, cu coeficienti intregi sau reali, si returneaza rezultatul mai departe ca polinom cu coeficienti intregi sau polinom cu coeficienti double in cazul impartirii si integrarii.

Metoda addOperation() implementeaza adunarea polinoamelor intregi primite ca parametrii dupa prelucrarea acestora din campurile FirstPolynomial si SeconPolynomial, parcurgand primul polinom si adunand coeficientii celui de al doilea daca gradele monoamelor sunt aceleasi, apoi monoamele care nu au corespondent ca si grad,in ordinea descrescatoare a puterilor sau daca unul dintre polinoame este null realizeaza practic doar adunarea cu 0.

Metoda subOperation() implementeaza scaderea, metoda fiind asemanatoarea cu adunarea, diferenta fiind ca coeficientii monoamelor cu acelasi grad se scad si semnul monoamelor din al doilea polinom care nu gasesc un monom cu acelasi grad in primul polinom devine minus, iar semnul coeficientilor monoamelor din primul polinom ramane acelasi.

Metoda multiplicationOperation() realizeaza inmultirea, parcurgand la fel primul polinom, si inmultind pe rand coeficientii primului cu fiecare coeficient din al doilea polinom si adunand gradele. In aceasta metoda se apeleaza si metoda addOperation() prin care avem grija ca monoamele rezultate in urma inmultirii care au acelasi grad sa fie adunate pentru a nu afisa de doua ori monoame cu acelasi grad.

Metoda derivateOperation() realizeaza derivarea, parcurgand polinomul dat ca intrare si inmultind coeficientul acestuia cu gradul, scazandu-i gradul cu o unitate, dupa regulile de derivare ale polinoamelor.Aceasta metoda primeste ca parametru un polinom cu coeficienti intregi si returneaza tot un polinom cu coeficienti intregi deoarece inmultirea nu modifica tipul coeficientului.

Metoda integrateOperation() parcurge polinomul si imparte coeficientul acestuia la grad + 1, si creste gradul fiecarui monom cu 1. Deoarece se realizeaza o impartire, aceasta metoda returneaza un polinom de tip double, pentru cazul in care rezultatul impartirii nu este un numar intreg, dar parametrul functiei este tot un coeficient intreg deoarece datele de intrare sunt polinoame cu coeficienti intregi.

Clasa Controller realizeaza apelarea operatiilor in momentul in care utilizatorul apasa unul din butoanele “Add”, “Substract”, “Multiply”, “Divide”, ”Integrate”, ”Derivate”, daca datele de intrare (polinoamele) sunt inserate corect de catre utilizator, folosindu-se de clasa ActionHandler care suprascrie metoda actionPerformed din ActionListener. De asemenea, in aceasta clasa se implementeaza metoda coeffAndPower care prelucreaza string-ul dat ca intrare de utilizator, folosind regular expressions, si returneaza un polinom cu care se poate lucra mai departe pentru apelarea operatiilor. Se prelucreaza pe rand fiecare monom, fiind mai apoi adaugate intr-un obiect din clasa Polynomial, care este returnat mai departe de aceasta metoda. Mai avem si metodele polynomToString() si doublePolynomToString () care transforma polinoamele primite ca date de intrare in string-uri pentru a putea fi afisate mai apoi ca rezultat in campul Result dupa efectuarea unei operatii. Clasa ActionHandlerOperations folosita in clasa Controller este folosita pentru a apela operatia pe care a ales-o utilizatorul.

Metoda buttons() din clasa Controller se foloseste de clasa ActionHandler care implementeaza ActionListener, unde, daca unul din butoanele din partea de jos este apasat se introduce in textfield-ul corespunzator textul de pe butoan, iar in cazul butonului DEL se sterge ultimul caracter introdus, pentru a crea o interfata user-friendly utilizatorului.

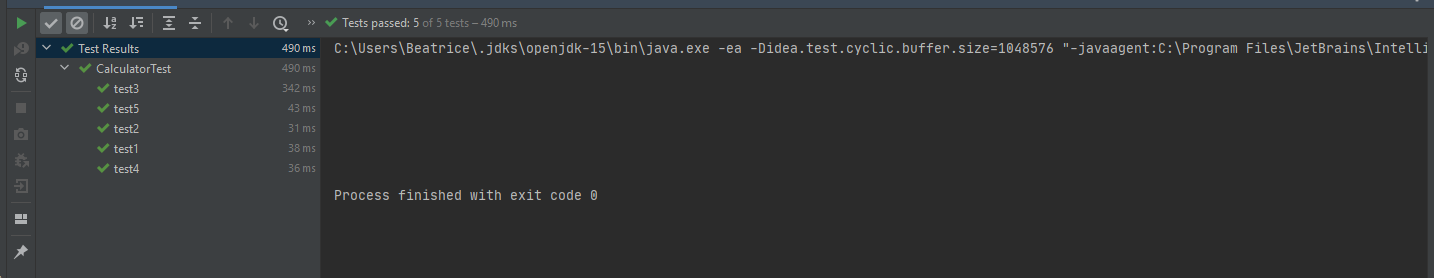
Metoda setButtons() este folosita pentru adaugarea de ActionListeners pentru butoanele prin care efectueaza operatiile de adunare, scadere, inmultire, integrare si derivare implementate in clasa Operations.

Clasa CalculatorGui implementeaza interfata prezentata mai sus, adauga toate butoanele, toate label-urile si toate textField-urile corespunzatoare, asezate pe un panel la diferite coordonate,avand aceleasi dimensiuni pentru fiecare grupare asemanatoare si mesaje care reprezinta scopul fiecaruia,fiind o interfata simpla, usor de utilizat. Butoanele din partea stanga scriu ciferele in campurile FirstPolynomial si SecondPolynomial, butonul DEL sterge ultimul caracter scris in cazul eventualelor greseli ale utilizatorului, iar butoanele din partea dreapta reprezinta operatia care se executa la apasarea fiecaruia. Toate acestea sunt apelate mai apoi in clasa Controller, unde am hotarat mai exact ce trebuie sa se intample dupa fiecare click al unui anumit buton. Rezultatul care apare in campul Result se genereaza automat dupa apasarea unuia din butoanele “Add”, “Substract”, “Integrate”, “Derivate”, “Multiply”.

Clasa CalculatorTest are rolul de a testa functionalitatile aplicatiei si de a demonstra corectitudinea fiecarei metode implementate asupra polinoamelor. Testele introduse au scopul de a prezenta anumite cazuri deosebite si de a verifica corectitudinea in cazurile acestea, dar si in cazuri generale, pentru a demonstra per total corectitudinea implementarii operatiilor.

**5.Rezultate**

Pentru testarea operatiilor am folosit framework-ul Junit cu metoda de testare fixa.Am reusit sa implementez operatiile de adunare, scadere, inmultire, derivare si integrare.Datele de testare au fost introduse pentru a acoperi cat mai bine cazurile generale si in urma acestor teste se poate constata ca toate operatiile au fost realizate corect.Toate testele realizate au fost corecte, deci operatiile sunt bine implementate.Rezultatul fiecarei operatii este afisat sub forma unui polinom ordonat in ordinea descrescatoare a gradelor, iar polinoamele care pot avea nevoie de o reprezentare de coeficienti reali sunt reprezentate cu cifre zecimale.



**6.Concluzii**

Aceasta tema a fost folositoare pentru invatarea folosirii de regex in java, pentru a invata sa testam folosind JUnit si pentru a delimita clar niste clase, relatiile dintre ele si pentru a le structura cat mai bine.De asemenea am invatat sa folosim modelul Controller View Model pentru a structura eficient proiectul si am exersat lucrul cu ArrayList si de a implementa operatii matematice.De asemenea am exesat lucrul cu GUI in java si consider ca am realizat o interfata user-friendly pentru utilizatorii calculatorului.

**7.Bibliografie**

1. Cursurile si notitele de la laboratoarele de la “Tehnici de programare fundamentale”
2. <https://stackoverflow.com/questions/28859919/java-regex-separate-degree-coeff-of-polynomial/44188848>
3. <https://www.vogella.com/tutorials/JUnit/article.html>
4. <https://docs.oracle.com/javase/tutorial/essential/regex/>