

DOCUMENTATIE

TEMA 1

NUME STUDENT : COJOCARU ANA CATALINA

GRUPA: 30226

CUPRINS

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

1. Obiectivul temei

Obiectivul temei a fost crearea unui calculator de polinoame ușor de folosit de către utilizator care poate efectua următoarele operații: *Adunare, Scadere, Inmultire, Impartire, Derivare, Integrare.*

• Crearea interfeței grafice	<i>Cap. 4</i>
• Crearea clasei polinom	<i>Cap. 4</i>
• Stocarea valorilor date în interfața grafică în HashMap	<i>Cap. 4</i>
• Efectuarea operațiilor	<i>Cap. 5</i>
• Verificarea corectitudinii	<i>Cap. 5</i>
• Utilizarea de către utilizator	<i>Cap. 2</i>

-Crearea interfeței grafice: Am folosit Java Swing într-o clasa care implementeaza Action Listener;

-Crearea clasei polinom: Clasa a fost creată cu un singur parametru, HashMap-ul care are două valori întregi respective puterii și coeficientului;

-Stocarea valorilor date în interfața grafică în HashMap: se vor stoca cu ajutorul unui regex ce desparte string-ul în monoame;

-Efectuarea operațiilor: Într-o clasă separată numită *Operații*, am implementat static cele 6 operații

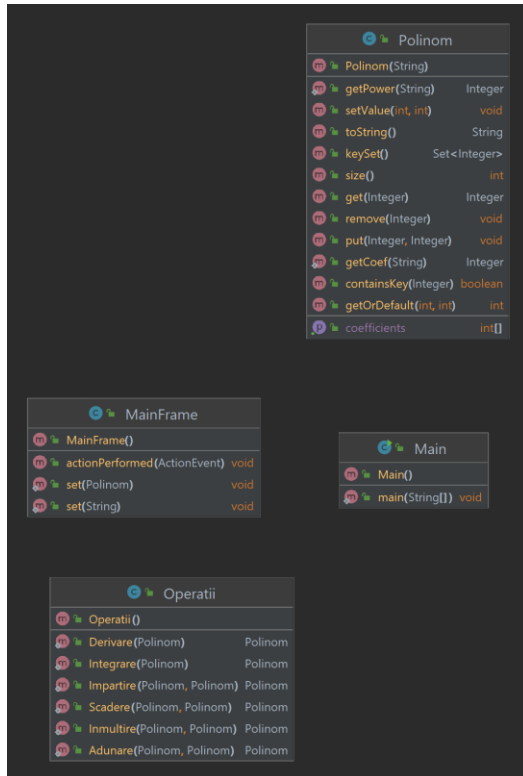
-Verificarea corectitudinii: Folosind testarea unitară, JUnit, am testat cele șase operații create

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

Calculatorul e functional in toate cazurile.

<i>Brief Description</i>	Calculatorul primeste două polinoame si se doreste calculul unor operatii de baza.
<i>Actors(s)</i>	The user
<i>Trigger</i>	Apasând butonoanele cu titlul operațiilor dorite, se va efectua operatia.
<i>Pre-conditions</i>	Calculatorul nu afiseaza nimic, toate campurile fiind goale.
<i>Post-Conditions</i>	In aria alocata rezultatului se va afisa polinomul rezultat in urma efectuării operatiilor.
<i>Basic Flow</i>	<i>Step 1:</i> Introduceți polinomul 1; <i>Step 2:</i> Introduceți polinomul 2; <i>Step 3:</i> Apasati butonul cu operatia dorita; <i>Step 4:</i> Apasati butonul ‘Clear’ pentru a introduce alte date.
<i>Exception Flow</i>	Impărțirea cu 0 nu e posibila, aceasta va rezulta intr-un mesaj în casuta de Rezultat.

3. Proiectare



În proiect există mai multe pachete: pachetul *GUI* în care găsim funcția *Main* care creează un obiect din clasa *MainFrame* (interfata cu utilizatorul), și pachetul *Polinom* care are clasa *Operatii* unde se implementează *static* operațiile și clasa *Polinom*. De asemenea, testările se fac într-o clasă *Test* numită *OperatiiTest*.

Pentru stocarea polinoamelor se folosește *HashMap<Integer, Integer>* (coeficienții se vor lua întregi în cazul tuturor operațiilor).

4. Implementare

Clasa *Main* instanțiază un obiect din clasa *MainFrame* care va crea interfața grafică pentru utilizator. Aceasta are două câmpuri unde se introduc cele două polinoame ce vor participa la operațiile de adunare, scădere, înmulțire și împărțire. Primul polinom va putea fi derivat, respectiv integrat. Interfața este simplă, are 6 butoane specifice fiecărei operații și un buton de ștergere. Câmpul de rezultat nu poate fi editat de către utilizator. Cele două șiruri de caractere care se vor introduce de utilizator în câmpurile pentru polinoame vor fi despartite în monoame printr-un *regex* care se găsește în constructorul clasei *Polinom* care primește ca parametru string-ul. Prin metodele implementate în clasa *Polinom*, fiecare monom e stocat în *HashMap* printr-o pereche de valori respective gradului polinomului (cheia) și coeficientul acestuia (valoarea). Coeficientul este găsit printr-un *regex* care extrage primul număr/ prima cifră dinaintea literei 'x'. Găsirea puterii se bazează pe găsirea ultimelor cifre după semnul '^'. Se adaugă valorile în *HashMap*-ul

polinomului. In clasa *Operatii* se implementeaza static toate operatiile. *Adunarea* parcurge elementele primului polinom, iar daca al doilea polinom continue aceeasi putere aduna coeficientii adaugandu-i la rezultat, stergand in cel de-al doilea polinom cheile gasite. La sfarsit se parcurg elementele ramase din al doilea polinom. Similar la *Scadere*. *Inmultirea* parcurge cu doua *foreach*-uri cele doua polinoame, inmultindu-le si adaugand la rezultat numerele obtinute, iar *Impartirea* imparte cele doua polinoame, restul fiind aflat prin scaderea din primul polinom a produsului catului cu al doilea polinom. Derivarea si integrarea au ca parametru doar un polinom.

5. Rezultate

✓ OperatiiTest (org.polinom)	80 ms
✓ adunare	74 ms
✓ inmultire	1 ms
✓ impartire	1 ms
✓ integrare	2 ms
✓ derivare	1 ms
✓ scadere	1 ms
✓ Tests passed: 6 of 6 tests – 80 ms	

Testarea s-a facut cu *Junit* pe fiecare operatie in parte cu acelasi set de polinoame, toate operatiile fiind corecte, fara erori. Fiecare operatie a fost testata intr-o metoda separata in clasa de test. Rezultatele asteptate au coincis cu rezultatele calculate de metodele implementate in *clasa Operatii*.

POLINOMUL 1:
POLINOMUL 2:

Adunare

Scadere

Inmultire

Impartire

Derivare

Integrare

REZULTAT

Clear

6. Concluzii

Am invatat sa structurez un proiect mai mare decat cele precedente. De asemenea, am invatat sa , sa lucrez cu *HashMaps*, sa imi organizez clasele si metodele pentru o intelegere usoara a codului. O posibila dezvoltare a proiectului este definirea coeficientilor de tip *Double* si extinderea posibilitatilor cu alte operatii cum ar fi Laplace, inlocuirea variabilei x cu un numar anume etc. .

7. Bibliografie

1. *Bruce Eckel, Thinking in Java (4th Edition), Publisher: Prentice Hall PTR Upper Saddle River, NJ United States, ISBN:978-0-13-187248-6 Published:01 December 2005.*
2. *StackOverflow*
3. https://users.utcluj.ro/~igiosan/teaching_poo.html
4. [Fundamental Programming Techniques \(dsrl.eu\)](https://www.dsrl.eu/fundamental-programming-techniques)