**DOCUMENTAȚIE**

TEMA 1

NUME STUDENT: BAN ALEX-FLORIN

GRUPA: 30227

CUPRINS

[1. Obiectivul temei 3](#_Toc130993170)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc130993171)

[3. Proiectare 4](#_Toc130993172)

[4. Implementare 5](#_Toc130993173)

[5. Rezultate 5](#_Toc130993174)

[6. Concluzii 6](#_Toc130993175)

[7. Bibliografie 6](#_Toc130993176)

1. **Obiectivul temei**

Obiectivul principal al temei este construirea unui calculator de polinoame care să fie capabil de a realiza operații aritmetice (adunare, scădere, înmulțire, împărțite), de asemenea derivări și integrări.

Obiective secundare:

1. Definirea unei clase care să stocheze informațiile privind polinoamle (coeficienții + puterile monoamelor)
2. Crearea unui GUI (graphical user interface) care să permită utilizatorilor să introducă două polinoame și să selecteze operația dorită
3. Implementarea operațiilor aritmetice
4. Implementarea operațiilor de derivare și integrare
5. Implementarea unui algoritm de extragere a coeficienților și puterilor polinomului
6. **Analiza problemei, modelare, scenarii, cazuri de utilizare**

Cerințe funcționale:

1. Introducerea unuia sau a două polinoame după caz
2. Selectarea unei operații aritmetice sau unei operații de derivare/integrare
3. Afișarea rezultatul corespunzător operației selectate

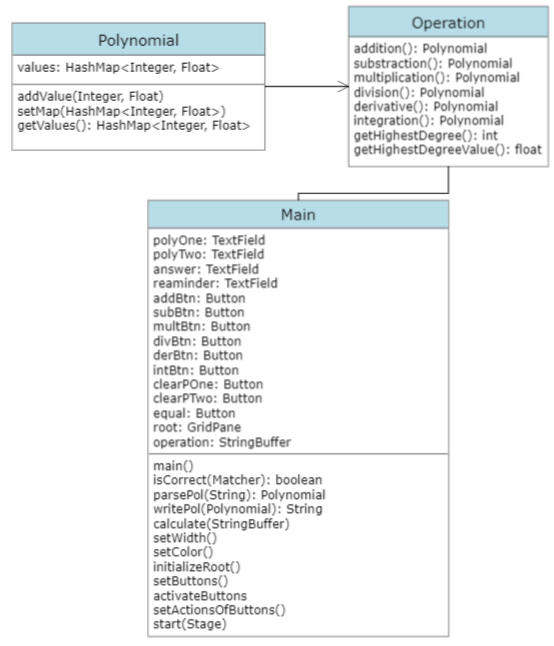
Cerințe non funcționale:

1. GUI-ul trebuie să fie unul plăcut la vedere și care să ofere utilizatorului un sentiment de familiaritate privind funcționalitatea aplicației

Cazurile de utilizare:

1. Introducerea unui polinom
2. Se dechide aplicația
3. Se introduc unul sau două polinoame
4. Executarea unei operații aritmetice:
5. Se dechide aplicația
6. Se introduc două polinoame
7. Se selectează una din primele patru operații
8. Aplicația afișează rezultatul
9. (Doar în cazul împărțirii)Aplicația afișează restul împărțirii
10. Executarea unei operații de derivare/integrare:
11. Se dechide aplicația
12. Se introduce un polinom
13. Se selectează una din ultimele două operații
14. Aplicația afișează rezultatul
15. Eliberarea unei căsuțe de text:
16. Se deschide aplicația
17. Se introduc unul sau două polinoame
18. Se apasă pe butoanele “C1” și/sau “C2” pentru ca textul să fie șters
19. **Proiectare**

Din punct de vedere al claselor, aplicația are la bază următoarea diagramă UML, formată din clasele *Polynomial*, *Operation*, *Main*:



GUI-ul a fost implementat folosind JavaFX, având un total de 9 butoane și 4 câmpuri de text. Toate acestea au fost adăugate în interiorul unui *GridPane*, iar acțiunile butoanelor au fost setate folosind funcția *setOnAction()*, fiind utilizate funcții lambda pentru a reduce complexitatea codului și a-l face mai lizibil.

Pentru definirea clasei *Polynomial* a fost utilizată structura de date *HashMap*, o structură de date care se folosește de o cheie pentru a găsi o valoare.

1. **Implementare**

Clasa *Polynomial* va conține în *values* coeficienții și puterile monoamelor polinomului introdus. *Values* este de tip HashMap, cheia reprezentând puterea și valoare reprezentând coeficientul. Această clasă conține metode pentru calcularea sumei, scăderii, înmulțirii, împărțitii, derivării și integrării polinoamelor.

Clasa *Operation* conține în interiorul ei doar metode statice, fiecare ocupându-se cu câte un tip de operație (metodele getHighestDegree și getHighestDegreeValue fiind o excepție, ele fiind utilizate de metoda *division* pentru efectuarea unor operații din interiorul metodei).

Clasa *Main* se va ocupa de tot ceea ce ține de inițializare GUI-ului , efectuarea citirilor și extragerea coeficienților și puterilor polinoamelor. Această clasă conține mai multe elemente de tip buton, însoțite de patru câmpuri de text, două utilizate pentru introducerea polinoamelor, iar două fiind folosite de către aplicație pentru a transmite rezultatul operațiilor.

1. **Rezultate**

Testarea aplicației a fost făcută utilizând JUnit. Au fost luate în considerare următoarele scenarii: utilizatorul dorește să efectueze o operație aritmetică sau o operație de derivare/integrare. Pentru toate operațiile au fost luate în considerare aceleași polinoame:

P1: x^2+x^1

P2: x^3+3x^2+1

În cazul testelor care verifică funcționarea operaților aritmetice, toate au trecut cu succes, fiecare operație fiind testată o singură dată.

În interiorul metodelor care au testat operația de derivare și integrare, au fost făcute două verificări: una pentru polinomul *P1* și una pentru polinomul *P2*, acestea oferind un rezultat satisfăcător.

1. **Concluzii**

Această temă a ajutat la dezvoltarea cunoștințelor atât de Java cât și de JavaFX, dar pe lângă acestea ne-a ajutat să înțelegem mai bine procesul prin care se dezvoltă o aplicație care are scopul de a fi utilizată de cât mai multă lume. Ne-a obligat să ne gândim la toate situațile în care un utilizator ar putea să greșească când introduce un polinom. Din acest motiv am fost nevoiți să proiectăm aplicația din punctul de vedere al unui utilizator, contrar modului în care lucram până acum când dezvoltam coduri în C, C++ ,Java, etc.

Proiectul ne-a învățat de asemenea să lucrăm cu *containerele* deja existente în Java, în această temă utilizând *HashMap*-uri, iar în unele cazuri și *TreeMap*-uri pentru a realiza sortarea polinoamelor. Am învățat în acest mod cât de important este să utilizăm toate materialele pe care le avem deja la dispoziție în loc să creăm noi toate părțile codului de la zero.

Cu toate acestea, aplicația nu este în niciun caz perfectă, ea ar putea beneficia de multe îmbunătățiri atât din punct de vedere al codului cât și al GUI-ului. Ex: GUI-ului ar avea nevoie de un *polish*, codul ar putea fi mai eficient.

1. **Bibliografie**
2. JavaFx Calculator with Java code - <http://www.discoversdk.com/blog/javafx-calculator-with-java-code>
3. Sort a HashMap in Java - <https://www.baeldung.com/java-hashmap-sort>
4. StringBuffer class in Java - <https://www.geeksforgeeks.org/stringbuffer-class-in-java/>
5. String Buffer - <https://docs.oracle.com/javase/7/docs/api/java/lang/StringBuffer.html>
6. GridPane- https://docs.oracle.com/javase/8/javafx/api/javafx/scene/layout/GridPane.html