**DOCUMENTATIE**

**TEMA 1**

**CALCULATOR POLINOMIAL**

NUME STUDENT: HIREAN ROXANA-MARIA

GRUPA: 30224

**OBIECTIVE:**

Aceasta tema are ca scop principal proiectarea si implementarea unui calculator polinomial care sa fie capabil sa efectueze urmatoarele operatii cu polinoame: adunarea, scaderea, inmultirea, impartirea, derivarea si integrarea. Ultilizatorul poate introduce polinoamele prin interfata grafica, de unde poate selecta si operatia matematica pe care doreste sa o efectueze, iar apoi va putea vedea si rezultatul operatiei.

Obiectivul principal are in vedere indeplinirea urmatoarelor:

1. Realizarea unei interfete grafice – care realizeaza interactiunea dintre utilizator si program.
2. Procesarea datelor de intrare – inputul(textul) pe care utilizatorul il introduce trebuie transformat in obiecte de tip polinom pentru a putea fi procesate.
3. Efectuarea operatiei de adunare – suma dintre cele doua polinoame introduse de la tastatura de catre utilizator.
4. Efectuarea operatiei de scadere – din primul polinom introdus de utilizator se scade cel de-al doilea polinom.
5. Efectuarea operatiei de inmultire – fiecare termen al primului polinom este inmultit cu fiecare termen al celui de-al doilea polinom.
6. Efectuarea operatiei de impartire – se efectueaza impartirea cu rest a primului polinom la cel de-al doilea polinom si se returneaza catul.
7. Efectuarea operatiei de derivare – este derivat primul polinom pe care utilizatorul il introduce. Faptul ca utilizatorul mai introduce de la tastatura si cel de-al doilea polinom, nu influenteaza rezultatul operatiei.
8. Efectuarea operatiei de integrare – se integreaza primul polinom introdus.
9. Rezultatul adus la cea mai simpla forma – polinomul rezultat, precum si polinoamele introduse de la tastatura de catre utilizator pot contine ori coeficienti nuli, ori termeni asemenea, iar noi trebuie sa fim siguri ca am adus polinomul la cea mai simpla forma a alui inainte de a-l utiliza.
10. Afisarea rezultatului final – rezultatul obtinut dupa fiecare operatie in parte trebuie sa fie facut vizibil pentru utilizator.
11. Testarea unitara – programul rezultat trebuie testat pentru a ne asigura ca acesta functioneaza cum trebuie.

**ANALIZAREA PROBLEMEI: SCENARII SI CAZURI DE UTILIZARE:**

Cerinte functionale:

* Calculatorul polinomial trebuie sa permita utilizatorilor sa introduca doua polinoame care au coeficientii numere intreci, iar puterile sunt numere naturale.
* Calculatorul trebuie sa permita utilizatorilor sa selecteze operatia care doresc sa fie efectuata.
* Calculatorul ar trebui sa poata sa efectueze cu succes oricare din operatiile implementate, pentru orice date valide de intrare.
* Calculatorul ar trebui sa faca posibila vizualizarea rezultatului fiecarei operatii, intr-o forma simplificata.
* Calculatorul trebuie sa fie usor de utilizat de catre orice fel de utilizatori.

Cazuri de utilizare:

* **Inmultirea polinoamelor:**

Scenariul principal de succes:

1. Utilizatorul introduce cele 2 polinoame.  
2. Utilizatorul apasa butonul cu textul ,,Inmultire”.

3. Calculatorul efectueaza operatia selectata prin inmultirea fiecarui termen din primul polinom cu fiecare termen din cel de al doilea polinom.

4. Calculatorul afișează rezultatul inmultirii celor doua polinoamepe.

Secventa alternativa: a) Rezultatul inmultirii este prea mare pentru a putea sa fie afisat in text field-ul corespunzator.

1. Utilizatorul introduce date de intrare prea lungi sau care au coeficienti cu valori mari.

2. Se afiseaza partial rezultatul.

3. Se revine pasul de introducere a polinoamelor.

* **Impartirea polinoamelor:**

Scenariul principal de succes:

1. Utilizatorul introduce polinoamele si selecteaza operatia de impartire.

2. Calculatorul efectueaza operatia si afiseaza rezultatul.

Secvență alternativă: a) Date de intrare incorecte: se afiseaza un mesaj de eroare.

b) Gradul primului polinom este mai mic decat gradul celui de al doilea: nu se afiseaza niciun rezultat.

c) Coeficientul primului termen al celui de al doilea polinom este prea mare pentru a fi scazut din unul din coeficientii termenilor cu rang mai mare sau egal cu rangul acestuia din primul polinom: nu se afiseaza niciun rezultat.

d) Cel de al doilea polinom este nul: se afiseaza un mesaj de eroare.

* **Adunarea polinoamelor:**

Actor principal: utilizatorul

Scenariul principal de succes:

1. Utilizatorul introduce cele 2 polinoame în interfata grafica.  
2. Utilizatorul selectează operatia de adunare, apasand pe butonul cu textul corespunzator.

3. Calculatorul polinomial efectuează suma celor două polinoame și afișează rezultatul

Secventa alternativa: date de intrare necorespunzatoare

1. Utilizatorul introduce texte care nu corespund formatului specificat pentru polinoame, de exemplu:

- polinoame cu 2 sau mai multe variabile;

- polinoame care nu au coeficientii numere intregi;

* **Scaderea polinoamelor:**

Actor principal: utilizatorul

Scenariul principal de succes:

1. Utilizatorul introduce cele 2 polinoame.  
2. Utilizatorul selectează operatia de scadere.  
3. Calculatorul polinomial efectuează diferenta celor două polinoame, scazand din valoarea primului polinom, valoarea celui de-al doilea.

4. Calculatorul afișează rezultatul pe interfata grafica.

**- Integrarea unui polinom:**

Scenariul principal de succes**:**  
1. Utilizatorul introduce polinomul si selecteaza operatia de integrare.

2. Calculatorul realizeaza integrarea si afiseaza rezultatul.

Secvență alternativă: a) Formatul polinoamelor introduse nu corespunde celui cerut.

- **Derivarea unui polinom:**

Scenariul principal de succes:

1. Utilizatorul introduce primul polinom (valoarea celui de al doilea nu este utilizata) si selecteaza operatia de derivare.

2. Calculatorul realizeaza derivarea polinomului.

Secvență alternativă: a) Datele de intrare sunt incorecte.

b) Polinomul este nul

-  **Resetarea calculatorului:**

Scenariul principal de succes:

1. Utilizatorul apasa butonul de reset din coltul din dreapta, jos.

2. Este resetat modelul din interfata grafica.

3. Se observa ca toate campurile din interfata grafica au revenit la valoarea initiala (0).

**PROIECTARE:**

Diagrama UML:Diagram, engineering drawing

Description automatically generated

**IMPLEMENTARE:**

**Clase:**

Principalele clase create in cadrul acestui proiect, exceptandu-le pe cele utilizate la interfata grafica sunt urmatoarele: clasa Monomial, clasa DoubleMonomial, clasa Polinomial, clasa PolinomialDouble si clasa Main.

**Clasa Monomial**

Clasa Monomial cuprinde doi parametri intregi: un parametru pentru memorarea coeficientului varibilei ,,x”, iar unul pentru a salva valoarea exponentului acesteia. Clasa are un constructor cu parametrii, dar si metode care ajuta la accesarea variabilelor din afara acestei clase.

Pe langa metodele pentru accesarea variabilelor, in aceasta clasa mai sunt implementate si metode pentru efectuarea operatiilor de inmultire, impartire si derivare a monoamelor, dar si pentru transformarea unui sir de caractere din string in monom. Conversia din String in monom se realizeaza in mai multe metode: se verifica daca stringul reprezinta un termen liber si se incarca conversia acestuia intr-un intreg dupa care atribuim valoarea rezultata coeficientului. Daca exponentul este mai mare decat 0, verificam fiecare caz in parte: daca variabila x este precedata de semnul “-”, atunci coeficientul are valoarea -1, iar exponentul are valoarea 1; daca x este precedat de un numar, coeficientul ia valoarea acelui numar iar puterea ramane la valoarea 1 daca sirul de caractere nu contine caracterul “^”, iar daca il contine, puterea monomului ia valoarea numarului aflat dupa acest simbol.

In aceasta clasa se mai gasesc si doua metode suprascrise: :”compareTo”, care compara doua obiecte de tip monom, avand un rol important in ordonarea elementelor polinomului in ordine descrescatoare a puterilor, si metoda “toString” care returneaza un sir de caractere.

**Clasa DoubleMonomial**

Aceasta clasa este una asemanatoare clasei Monomial, aceasta fiind diferita prin faptul ca are coeficientul de tip double, nu de tip intreg. Aceasta clasa contine un constructor cu parametri de tip intreg pe care ii converteste prin conversie implicita in coeficienti de tip double. In aceasta clasa este implementata operatia de integrare, Pe langa metoda pentru integrare, clasa mai contine o metoda suprascrisa pentru afisarea termenilor polinoamelor care au coeficienti rationali (toString), identica celei din clasa monom.

**Clasa PolinomialInteger**

Clasa PolinomInteger contine o lista de tipul array list in care sunt adaugate mai multe elemente de tip monom reprezentand termenii polinoamelor. Am ales implementarea folosind array list deoarece aceste structuri au anumite facilitati ca posibilitatea accesului indexat al elementelor, ceea ce a fost foarte util in implementarea tuturor metodelor. Lista este folosita pentru a memora polinoamele care au coeficienti intregi. Pe langa lista de monoame, clasa mai contine si o variabila de tip intreg care memoreaza gradul polinomului care are coeficienti de tip intreg.

In cadrul acestei clase, este implementata o functie “simplify” care reduce polinomul la forma cea mai simpla prin reducerea termenilor asemenea. Procedeul acesta se realizeaza prin: sortarea termenilor polinomului in ordine descrescatoare a gradului, adunarea tuturor termenilor consecutivi care au acelasi grad, stergerea termenilor care au corficientul egal cu zero, iar apoi se actualizeaza gradului polinomului, egalandul cu gradul primului termen din lista.

Alta metoda implementata in aceasta clasa este cea pentru afisarea polinoamelor: toString, care returneaza un sir de caractere obtinut prin concatenarea unui sir gol cu sirul returnat de metoda toString apelata pentru fiecare termen al polinomului.

O alta metoda importanța este „transformarePolinom” care transforma sirul de caractere intr-un obect de tip polinom. Conversia se realizeaza prin impartirea polinomului in mai multe obiecte de tip monom cu ajutorul metodei “transformareMonom” prezentata in clasa Monomial, dupa care, monoamele sunt adaugata in lista din polinom care este apoi redus.

Restul metodelor din clasa Polinom implementeaza operatiile cu polinoame: adunarea este realizata prin adaugarea tuturor elementelor din cele doua polinoame o lista de monoame goala a unui obiect de tip polinom, urmata de reducerea polinomului. Pentru scadere am inversat semnul celui de al doilea polinom dupa care am aplicat algoritmul de adunare. Inmultirea se realizeaza prin inmultirea tuturor termenilor ambelor monoame, adunarea termenilor, urmata de reducerea polinomului. Atat derivarea, cat si integrarea se realizeaza prin derivarea, respectiv integrarea, fiecarui termen al primului polinom.

**Clasa PolinomialDouble**

Aceasta Clasa este una asemanatoare clasei PolinomialInteger, fiind diferita prin faptul ca lista este folosita la memorarea polinoamelor care au coeficienti rationali, obtinuti in urma integrarii. Implementarea clasei este aproape acceasi cu cea a clasei PolinomialInteger. Avem de asemenea si metoda toString, implementata si in clasa PolinomialInteger, care returneaza un sir de caractere.

**Interfata grafica(GUI)**

Interfata grafica este implementata cu ajutorul a trei clase foarte imprtante: Model, View si Controller. Am ales acest model pentru a face codul mai usor de inteles si mai organizat.

**Clasa View**

Clasa View cuprinde mai multe butoane si textfield-uri care sunt adaugate in panouri si aranjate corespunzator. Prin intermediul acestei componente sunt afisate informatii utilizatorului si sunt obtinute datele transmise si procesate de model, avand cel mai important rol pentru realizarea interactiunii dintre program si utilizator. Contine metode pentru returnarea datelor introduse de utilizator, dar si pentru afisare si pentru a declara ascultatorii pentru butoane.

**Clasa Controller**

Fiecare componenta view are un controler asociat. Controlerele primesc intrări, de obicei ca evenimente care denotă mișcarea mouse-ului sau apasarea butoanelor.

**Clasa Model**

Clasa Model încapsulează datele de bază și funcționalitate avand ca variabile doua polinoame, reprezentand datele de intrare pe care le proceseaza in metode definite pentru fiecare operatie, dar si o metoda pentru reset. In clasa Model avem toate operatiile pe care urmeaza sa le efectuam cu polinoamele noastre.

**Clasa Main**

Clasa Main reprezinta clasa principala a acestui proiect. In clasa Main se gaseste metoda main care uneste principalele componente din cadrul proeictului, continand trei instante ale claselor ce implementeaza interfata grafica, unite in componenta controller.

**REZULTATE:**

In cadrul testarii unitare din clasa Tests s-au efectuat teste pentru fiecare operatie. In urma testarii s-a obtinut un rezultat favorabil, toate testele fiind efectuate cu succes.

**CONCLUZII / DEZVOLTARI ULTERIOARE:**

Acest proiect poate sa fie dezvoltat prin marirea intervalului de valori pe care il pot lua coeficientii sau exponentii termenilor, realizandu-se un calculator care sa efectueze aceleasi peratii pentru variabile cu exponent negativ sau rational sau pentru polinoame care au coeficienti rationali sau prin adaugarea mai multor operatii ca returnarea restului.

**BIBLIOGRAFIE:**

-https://www.vogella.com/tutorials/JUnit/article.html

-https://www.baeldung.com/junit-5

- https://regex101.com/