**Obiective**

**Obiective principale**

Propuneți, proiectați și implementați un sistem de procesare a polinoamelor de o singură variabilă cu coeficienți întregi.

**Obiective secundare**

|  |  |  |
| --- | --- | --- |
| Obiective secundare | Descriere | Capitol |
| Dezvoltarea de use case-uri și scenarii | Urmărirea funcționării operațiilor | 3 |
| Alegerea structurilor de date | Enumerarea structurilor de date utilizate pentru realizarea acestui proiect | 4 |
| Implementarea de clase | Explicarea rolului fiecărei clase | 4 |
| Dezvoltarea algoritmilor | Procedeul de dezvoltare a algoritmului utilizat pentru acest proiect | 4 |
| Implementarea soluțiilor | Ajungerea la o soluție și modul de implementare | 4 |
| Testarea | Exemplu de funcționare și utilizare JUnit pentru testare | 4 |

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

Use case

Nume use case: adunare polinoame

Actori:

Polinom1

Polinom2

addButtonListener

Trigger:

Apăsarea butonului ‘+’

Precondiții:

Deschiderea interfeței

Introducerea celor două polinoame

Postcondiții:

Se citesc cele două polinoame

Se aduc la formă utilizabilă pentru calcul

Se parcurg ambele polinoame adunându-se coeficienții elementelor care au puterile egale pentru a crea câte un nou element al rezultatului

Se adaugă la rezultat elementele polinoamelor care nu au corespondent în celălalt adică nu există în ambele polinoame un monom de acel grad

Se copiază elementele rămase neadunate într-un polinom din cauza diferenței numărului de elemente

Se afișează rezultatul în interfață

Nume use case: scadere polinoame

Actori:

Polinom1

Polinom2

subButtonListtener

Trigger:

Apăsarea butonului ‘-‘

Precondiții:

Deschiderea interfeței

Introducerea celor două polinoame

Postcondiții:

Se citesc cele două polinoame

Se aduc la formă utilizabilă pentru calcul

Se parcurge Polinomul2 și se schimbă semnul fiecărui monom component

Se apelează metoda addPolinoame(Polinom p1, Polinom p2),unde p2 este Polinomul2 după inversarea semnelor, pentru a calcula rezultatul dorit

Se afișează rezultatul în interfață

Nume use case: înmulțire polinoame

Actori:

Polinom1

Polinom2

inmButtonListener

Trigger:

Apăsarea butonului ‘\*‘

Precondiții:

Deschiderea interfeței

Introducerea celor două polinoame

Postcondiții:

Se citesc cele două polinoame

Se aduc la formă utilizabilă pentru calcul

Se parcurge cele două polinoame termen cu termen și se formează noi monoame care se adaugă în polinomul rezultat

Se afișează rezultatul în interfață

Nume use case: derivare polinom

Actori:

Polinom

derivButtonListener

Trigger:

Apăsarea butonului ‘’‘

Precondiții:

Deschiderea interfeței

Introducerea polinoamului

Postcondiții:

Se citește polinmul

Se aduce la formă utilizabilă pentru calcul

Pentru fiecare monom component al polinomului se apelează metoda derivMonom(Monom m) pentru a obține derivata fiecăruia

Derivata fiecărui monom se adaugă în polinomul rezultat

Se afișează rezultatul în interfață

Nume use case: integrare polinoame

Actori:

Polinom

integrareButtonListener

Trigger:

Apăsarea butonului ‘I‘

Precondiții:

Deschiderea interfeței

Introducerea celor două polinoame

Postcondiții:

Se citește polinomul

Se aduce la formă utilizabilă pentru calcul

Pentru fiecare element, Monom, al polinomului se invocă metoda integrareMonom(Monom m) care va realiza integrarea fiecărui monom în parte, iar rezultatul integrări va fi trecut în polinomul rezultat ca un monom al său

Se afișează rezultatul în interfață

Nume use case: împărțire polinoame

Actori:

Polinom1

Polinom2

impButtonListener

Trigger:

Apăsarea butonului ‘/‘

Precondiții:

Deschiderea interfeței

Introducerea celor două polinoame

Postcondiții:

Se citesc cele două polinoame

Se aduc la formă utilizabilă pentru calcul

Se parcurg cele două polinoame realizând pas cu pas împărțirea. La fiecare pas al împărțirii se adaugă câte un element la cât care va fi parte din rezultat, iar în final se obține restul

Câtul și restul se așează în polinomul rezultat sub formatul dorit

Se afișează rezultatul în interfață

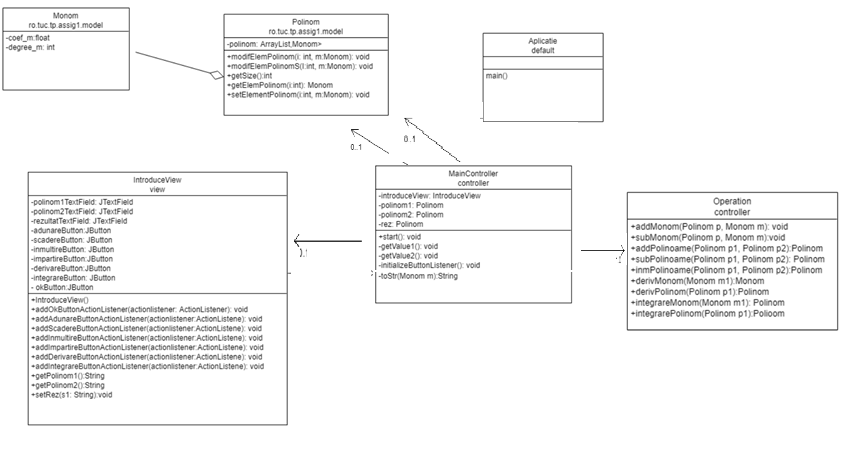
**Proiectare**

**Diagrama UML**

**Hdadkajfal;f’a;f’a l l l l l l l l l l l l l l l l l l l l l l l l l l l ll l l l l l l l l l ll l l l l l l l l l l l l l l l l l l l l l l l l l l l ll l l l l l l l l l l l l l l l l l ll l l**

Pentru a putea urmări cu ușurință conținutul aplicației proiectul se organizează în multiple pachete care vor conține diverse clase. Această organizarea simplifică găsirea anumitor porțiuni de cod în cazul în care acestea trebuie modificate și totodată fac codul mult mai lizibil încât fiecare pachet se conține clase care au rolurisimilare.

**L l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l ll l l l l l l l l l l l l l l l l l l ll ll l ll l l l l l l l l l l l ll ll l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l l**



**Structuri de date**

Acest proiect folosește ArrayList care conține elemente de tipul Monom pentru a crea un polinom. ArrayList va ține elementele unui polinom citit din interfața grafică și va fi folosit în operații pentru a obține rezultatele dorite. Deci ArrayListul va fi folosit pentru toate operatiile din cadrul programului acestia tinand elementele care trebuie adunate, scazute , inmultite , impartite , derivate sau integrate .

In final tot ArrayListul se va folosi pentru a retine rezultatul calculului efectuat in urma operatiei alese si acesta se va prelucra intr – un string si va fi afisat.

**Implementare**

Proiectul pentru crearea unui calculator pentru polinoame include patru pachete în care sunt organizate 5 clase.

Pachetul *model* inlude doua clase *Monom* și *Polinom* utilizate pentru a defini structurile necesare pentru calcule. Monomul are un coeficient real și o putere întreagă. Această clasă ține structura principală cu care se va lucra, polinoamele fiind alcătuite din mai mute monoame. Conține metode care permit citirea și scrierea coeficientului și a puterii, getere și settere. Clasa Polinom are un atribut polinom care este o listă de Monoame. Această structură va fi cea care în final va fi utilizată pentru operații. Metodele din acestă clasă permit citirea sau adăugarea uni element din/la listă, totodotă permit citirea întregii liste. Aceste metode permit efectuarea operaților care trebuie să modifice fiecare monom din cadrul polinomului și totodată adăugarea elementelor la un nou polinom pentru a crea rezultatul.

Pachetul View conține o singură clasă, IntroduceView, care crează setează dimensiunea ferestrei pentru interfața de utilizator și crează câmpurile și putoanele din cadrul acesteia. Astfel interfața va conține trei TextField-uri și șase butoane. Două textfield-uri vor fi folosite pentru a introduce polinoamele pe care va urma să fie efectuată o operație(pentru derivare și integrare se va utiliza doar unul din cele două câmpuri deoarece aceste operații nu necesită două polinoame), iar cel de-al treilea va fi folosit pentru a returna un rezultat. Fiecare polinom introdus și returnat va fi de forma “ax^y+bx^(y-1)+…+z”, excepție fiind împărțirea care trebuie să returneze și câtul și restul și pentru a evita crearea unui camp separate doarpenntru rest, rezultatul va fi de forma: “C: ax^y+bx^(y-1)+…+z R: fx^e+…+z”. Operațiile vor fi efectuate în urma apăsării unuia din cele șase butoane, fiecăruia corespunzând uneia din următoarele metode: adunare, scădere, înmulțire, împărțire, derivare, integrare. Polinoamele trebuie introduse înaintea selectării unei operații în caz contrar programul nu va putea efectua operația. Pentru o funcționare corectă(în conformitate cu dorința utilizatorului) la operațiile de scădere,împărțire, derivare și integrare contează în care câmp se introduc polinoamele. La opearția de scădere se introduce în field-ul Polinom1 descăzutul operațiiei, iar în field-ul Polinom2 se va introduce scăzatorul. La operația de împărțire field-ul Polinom1 va conține deîmpărțitul, iar Polinom2 va conține împărțitorul. La derivare și integrare se va utilize un singur câmp și acesta va fi Polinom1. În urma apăsării oricăruia dintre cele șase butoane în câmpul Rezultat v-a apărea rezultatul returnat de operația selectată. Această clasă conține metode pentru a accesa elementele din câmpurile Polinom1 și Polinom2 și metode care crează actionListeneri pentru fiecare din cele șase butoane.

Pachetul controller conține o singură clasă MainController care conține metode pentru a controla funcționarea aplicației. Aceasta va fi clasa care va conține metodele pentru fiecare din cele șase operații. În principiu pentru fiecare din aceste metode se accesează conținutul câmpurilor Polinom1 și Polinom2 din interfață. Cele două polinoame obținute astfel se evaluează element cu element, adică se evaluează fiecare Monom din cadrul lor. Totodată aceasta este clasa care setează vizibilitatea interfeței și care inițializează actionListenerii pentru fiecare buton astfel acestea pot reacționa penru a efectua o operație. Astfel când în interfață se apasă unul din cele șase butoane în interiorul metodei de inițializare a actionListenerilor,initializeButtonListeners(), se va apela metoda operației corespunzătoare butonului selectat. Spre exemplu, în cazul selectării butonului ‘+’ se va apela o metodă numită addPolinoame(Polinom p1, Polinom p2) care va efectua adunarea celor două polinoame și va afișa rezultatul. În cazul selactării butoanelor ‘-‘, ‘\*’, ‘/’, ’’’, ’I’ se vor utiliza metodele subPolinoame(Polinom p1, Polinom p2), inmPolinoame(Polinom p1, Polinom p2), impPolinoame(Polinom p1, Polinom p2), derivPolinom(Polinom p1) respectiv integrarePolinom(Polinom p1). Metodea derivPolinom(Polinom p1) se va folisi de metoda derivMonom(Monom m), iar metoda integrarePolinom(Polinom p1) se va folosi de metoda integrareMonom(Monom m).

Ultima clasă implemantaă este o clasă pentru test care va folosi pachetul Junit. Această clasă va conține main- ul programului și va permite rularea aplicației pentru a putea fi utilizată. Pentru aceasta metoda main va lucra cu un obiect de tip MainController pe care se va apela metoda start() ceea ce va porni interfața și va permite astfel introducerea polinoamelor și a efectuării operațiilor asupra acestora.

O metoda toStr transforma polinomul in String pentru a fi afisat, aceasta metoda analizeaza fiecare coeficient si fiecare putere din polinom apoi in funtie de valorile acestuia construieste un String folosind un StringBuilderp p p p p p p p p p p p p p pp p p p p p p p p p p p p p pp p p p

**Rezultate**

Următorul exemplu arată modul de funcționare a fiecărei operații:

Inputs:l l l l l l l l l l l l l l l ll l l l l l l l l ll l l l l l l l l l l l l l l l l l l lll l l l l l l l l l l l l l l l l l l l l l l l l

Polinom1: 3x^4-2x^3+3x-34p p p p p p p p p p p p p p pp pp p p p pp p p p p p p p p p p p p p

Polinom2: 2x^2-x+43p p p p p p p p p p p p p p p p p p p p p p p p p p p p p p p p p p pp p p p p p p p p

Outputs:p p p p p p p p p p p p p p p p p pp p p p p p p p p p p p p p p p p p p p p p p p p p p pp p

Adunare: 3x^4-2x^3+2x^2+2x+9

Scădere: 3x^4-2x^3-2x^2+4x-77

Înmulțire: 6x^6-7x^5+131x^4-80x^3-71x^2+172x-1462

Împărțire: C: lksod R: jsjja

Derivare ( se ia pentru Polinomul1 ): 12x^3-6x^2+3

Integrare ( se ia pentru Polinomul1 ): 0.6x^5-0.5x^4+1.5x^2-34x

Testul cu Junit pornește aplicația creân interfața unde se vor introduce polinoamele dorite și se va selecta operașia dorită. După se va afișa un rezultat. Se verifică rezultatul manual pentru asigurarea corectiduni acestuia. În cazul lipsei rezultatului se verifică eventuala apariție a unei erori. Dacă nu există erori semnalate se verifică logica utilizată și se reia testare. Astfel în final se ajunge la o aplicație funcțională.

**Concluzii**

Proiectul ușurează lucrul cu polinoamele astfel

Dezvlotări ulterioare

Proiectul i s-ar putea face o pagină web astfel aplicația nu v-a trebui să ruleze local pe computerul fiecărui user, astfel acesta nu ar mai trebui sa aibă codul sau un executabil pentru el pe propriul sau calculator/laptop ci va putea să îl acceseze simplu. Aceasătă metodă ar face mai simplă utilizarea pentru user, făcând aplicația mult mai accesibilă pentru ei. Ca altă dezvoltare calculatorul ar putea fi extins să supote operații mai complexe cu polinoame sau ar putea fi inclus în cadrul unui proiect pentru calcule mai complexe care printre altele ar necesita și operații cu polinoame.