DOCUMENTATIE

TEMA 1

CALCULATOR DE POLINOAME

NUME STUDENT: Fazakas Edina
GRUPA: 30223

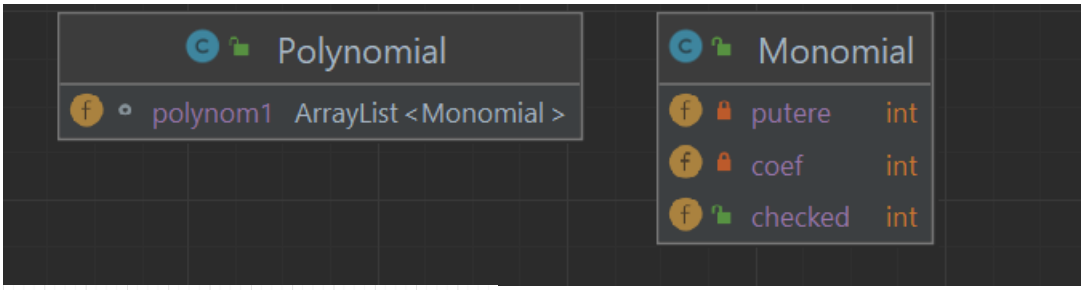
CUPRINS

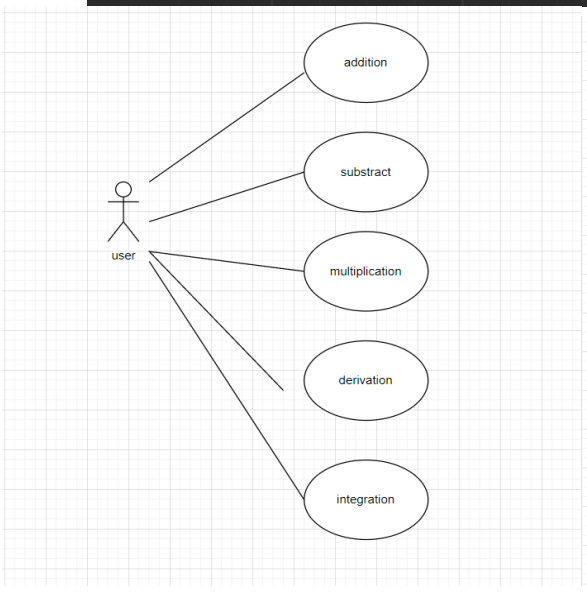
1. Obiectivul temei

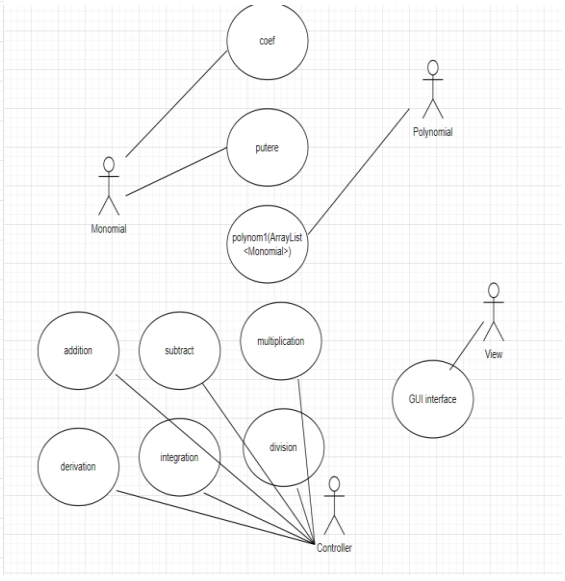
Tema aceasta constituie implementarea unui calculator de polinoame creată într-o interfață grafică în Java cu ajutorul căruia utilizatorul poate să insereze polinoame și selectând o operație matematică se va efectua operația și se va afișa rezultatul adecvat.

1) Analiza problemei și identificarea cerinței. – Capitolul 2
2) Proiectarea unui design pentru calculator – Capitolul 3
3) Implementarea calculatorului polinomial – Capitolul 4
4) Testarea calculatorului polinomial – Capitolul 5

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







Problema este percepută de operațiile polinoamelor într-o interfață grafică. Utilizatorul are voie să utilizeze operațiile din calculatorul de polinoame. Aici se vede diagrama use-case a proiectului în care se vede viziunea utilizatorului către fiecare clasă. Utilizatorul va introduce un Srting pe care programul va converta într-un polinom alcătuit din monoame. Astfel se va putea rezolva problema pe care utilizatorul îl va introduce și selecta. Aceste clase se vor lega între ele. Funcționalitatea calculatorului este dată dacă:

1) Să se poată introduce polinoame în calculator
2) Să se poată selecta o operație matematică
3) Să se poată efectua operația dintre cele 2 polinoame

   Use case-uri : operații – adunare, scădere, înmulțire, împărțire, integrare, derivare

   Use-case 1: adunare

   Actorul este user-ul

   Funcționalitatea:

   1) User-ul introduce 2 polinoame în interfața grafică
   2) User-ul selectează operația de adunare - addition
   3) User-ul dă click pe butonul de operație
   4) Calculatorul efectuează operația de adunare dintre cele două polinoame

   Use-case 2: scădere

   Actorul este user-ul

   Funcționalitatea:

   1) User-ul introduce 2 polinoame în interfața grafică
   2) User-ul selectează operația de scădere - subtract
   3) User-ul dă click pe butonul de operație
   4) Calculatorul efectuează operația de scădere dintre cele două polinoame

   Use-case 2: înmulțire

   Actorul este user-ul

   Funcționalitatea:

   1) User-ul introduce 2 polinoame în interfața grafică
   2) User-ul selectează operația de înmulțire- multiplicate
   3) User-ul dă click pe butonul de operație
   4) Calculatorul efectuează operația de înmulțire dintre cele două polinoame

Use-case 2: integrare

Actorul este user-ul

Funcționalitatea:

1) User-ul introduce 1 polinom în interfața grafică
2) User-ul selectează operația de integrare - integrate
3) User-ul dă click pe butonul de operație
4) Calculatorul efectuează operația de integrare pe polinom

Use-case 2: derivare

Actorul este user-ul

Funcționalitatea:

1) User-ul introduce 1 polinom în interfața grafică
2) User-ul selectează operația de derivare - derivate
3) User-ul dă click pe butonul de operație
4) Calculatorul efectuează operația de derivare pe polinom

Use-case 2: împărțire

Actorul este user-ul

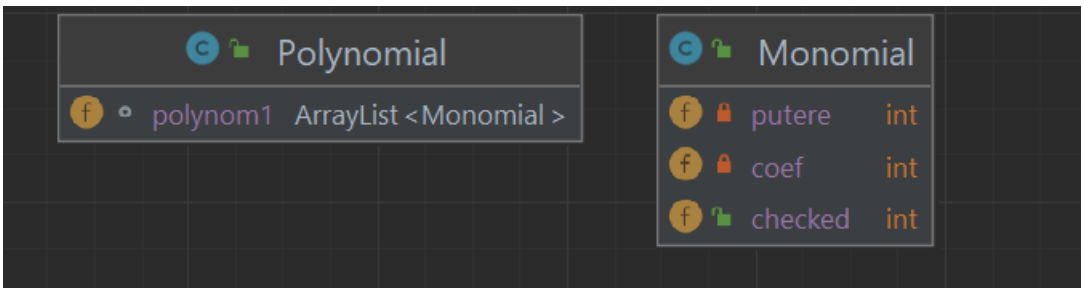
Funcționalitatea:

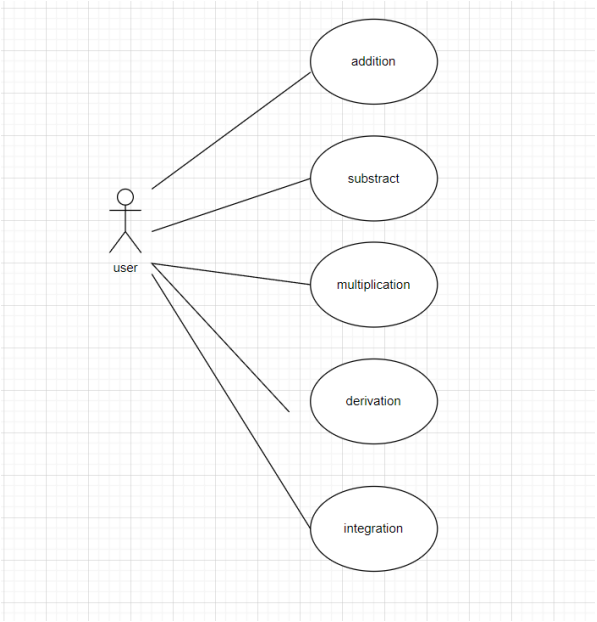
1) User-ul introduce 2 polinoame în interfața grafică
2) User-ul selectează operația de împărțire - divide
3) User-ul dă click pe butonul de operație
4) Calculatorul efectuează operația de împărțire dintre cele două polinoame

Scenariul în care utilizatorul introduce polinoame incorecte aduce situația în care le reintroduce

(Se va prezenta cadrul de cerinte functionale formalizat si cazurile de utilizare ca si diagrame si descrieri de use-case. Descrierile use-case-urilor se vor face sub forma unui flow-chart ori sub forma unei liste continand pasii executiei fiecarui use-case.)

3. Proiectare





Am utilizat modelul Model View Controller care se împarte aplicarea în trei domenii: procesare, ieșire și intrare. Componentele modelului încapsulează datele de bază și funcționalitate. View: afișează informații despre componente pentru utilizator, obține datele pe care le afișează din model. Controler: fiecare vizualizare are asociat o componentă controler. Controlerele primesc intrarea, de obicei ca evenimente care denotă mișcarea mouse-ului, activarea butoanele mouse-ului sau intrarea de la tastatură. Evenimentele sunt traduse în cereri de service, care sunt trimise fie la model sau la vedere.

În proiect avem 3 package-uri. Package-urile sunt : Model, View, Controller. În fiecare package sunt clase. Clasa Monomial are trei variabile, coeficientul de tip int, puterea unui monom de tip int, și variabila checked care este folosită în metodele de calculare. Clasa Polynomial are ca variabilă un Arraylist de tip Monomial polynom1. În această clasă se implementează toate metodele conversie și reprezentare. Folosesc 3 pachete pentru a avea organizare de Model View Controller. În pachetul Model am două clase, adică Monomial și Polynomial. În pachetul View am clasa Calculator în care am implementat interfața grafică. Pachetul Controller conține operațiile matematice pentru polinoame pentru a crea calculatorul, adică adunarea, scăderea, înmulțirea, împărțirea, derivarea, integrarea. Acetse operații vor fi făcute de utilizator. Interfața va fi creată astfel încât să se poată efectua aceste operații apăsând pe butonul potrivit. Utilizatorul va intorduce 2 polinoame în interfața grafică, apoi va apăsa pe butonul de operație, se va efectua operația, respectiv se va afișa polinomul rezultat pe ecran.

*(Se va prezenta proiectarea OOP a aplicatiei, diagramele UML de clase si de pachete, structurile de date folosite, interfetele definite si algoritmii folositi (*daca e cazul)

4. Implementare

Implementarea este făcută în trei package-uri: Model, View, Controller. În package-ul Model sunt 2 clase: Monomial și Polynomial. În package-ul View se află interfața grafică, clasa Calculator. În package-ul Controller se află clasa Controller. În clasa aceasta se efectuează operațiile. Clasa Monomial are trei variabile, coeficientul de tip int, puterea unui monom de tip int, și variabila checked care este folosită în metodele de calculare. Clasa Polynomial are ca variabilă un Arraylist de tip Monomial polynom1. În această clasă se implementează toate metodele conversie și reprezentare. În clasa Monomial nu este nicio metodă în afară de setteri și getteri. În clasa Polynomial este o metodă convertion în care se convertește prin regular expressions stringuri în monoame, validare de polinoame, și convertire de monoame în stringuri. În clasa Controller se află metodele pentru operații. Operația de adunare se face prin parcurgere foreach în cele două polinoame. Atunci când cele două puteri sunt egale, atunci setăm coeficientul ca suma celor două coeficienți, puterea este egală cu puterile celor două monoame. Pentru a nu parcurge de mai multe ori monoamele deja parcurse, punem o variabilă checked pe care o modificăm în 1 după ce am adunat cele două monoame. Variabila found o incrementăm când le adunăm. Această variabilă este folosită pentru a aduna monoamele în polinomul rezultat care au putere ce nu se află în celălalt polinom. La final se adună și monoamele din polinomul al doilea care nu au avut ”pereche” în polinomul 1. Aceste operații se fac și în metoda de scădere cu diferența că coeficienții se scad. În metoda de multiplication în primul rând avem un polinom în care se înmulțesc monoamele. Fiecare cu fiecare. Acest lucru se face prin adunarea puterilor și înmulțirea coeficienților. După aceea avem un alt polinom în care o să punem rezultatul. Parcurgem polinomul în care am adăugat înmulțirea monoamelor și ne uităm dacă avem mai multe elemente cu aceași putere, să nu fie vorba de același element, și aici vom verifica dacă elemntul a fost verificat sau nu. Dacă găsim elmente care au aceași putere, îi adunăm coeficienții, și schimbăm checked. Înainte de bucla a doua ne uităm dacă contorul din bucla 1 a fost verificată pentru a nu aduna elementul pe care deja l-am pus în sumă. După ce punem sumele în polinomul rezultat, adăugăm elementele care nu au fost adăugate încă. În metoda de derivare este o singură parcurgere a polinomului în care după formula de derivare a unui monom, decrementăm puterea și înmulțim coeficientul cu putere. În metoda de integrare, conform formulei de integrare a unui monom, puterea se incrementează și coeficientul va fi coeficientul împărțit cu puterea incrementată. Împărțirea se face prin alegerea polinomului cu puterea mare, polinoamele vor fi sortate în ordinea descrescătoare a puterilor. Apoi se vor lua coeficienții în ordine, q va fi întotdeauna suma dintre el însuși și împărțirea celor două monoame. Restul va fi dat de scăderea între rest și înmulțirea dintre împărțirea monoamelor și monomul din polinomul al doilea. Interfața grafică este rezolvată prin Swing GUI. Clasa implementează clasa ActionListener pentru a putea folosi mouse-ul și alte butoane de pe tastatură. Interfața are un frame, 3 textfielduri, unul cu primul polinom, cu al doilea polinom și cu polinomul rezultat. Operațiile și numerele pentru a scrie un polinom se afișeaza și se efectuează prin JButton-uri. Avem și butoane de delete și clear pentru a șterge ce am scris în TextFielduri. Prin metoda setBounds se aranjează fiecare buton cu coordinate, lungime și lățime. Toate butoanele și labelurile se adaugă în frame și fiecare buton se adaugă prin metoda ActionListener la operația de click. Pentru a diferenția în care TextField vrem să scriem o să folosim MouseListener, adică ne uităm pe care TextField am dat click și în așa fel o să modificăm valoarea unei variabile și o să testăm unde printăm polinomul respectiv. Metoda actionPerformed are ca parametru o variabilă de tip ActionEvent prin care o să judecăm evenimentele. În primul rând luăm în considerare butoanele pentru ”tastatură”. Numerele și caracterul x, respectiv caracterele speciale ca și + - ^ / etc sunt luate în considerare prin click și atunci când apăsăm butonul respectiv se printează caracterele în care TextField dăm click. Când apăsăm butonul de clear sau delete se vor șterge ori textFieldurile ori caracter cu caracter. Unele butoane sunt pentru a alege o operație. În această situație se va converti stringul de polinom, se va efectua metoda respectivă și se afișează rezultatul. Pentru a converti stringuri în monoame folosim regular expression. Pentru asta folosim un pattern care ne va ajuta să găsim expresii în string-uri care arată ca pattern-ul. Setul de match ne va ajuta să găsim aceste expresii. Expresia din algoritmul meu este dat de un semn, apoi un număr – ori dintr-un digit, ori mai multe *cifre*. Apoi va veni opțional x -ul, opțional caracterul ”^” apoi iarăsi cifre. Astfel, funcția match *după tipul matcher va arăta dacă va găsi un grup care arată ca expresia dată. Aceste expresii se vor converti în monoame. După ce convertim expresia în monom, pentru a afișa rezultatul, trebuie să o convertim iarăși în string. (Se va descrie fiecare clasa cu campuri si metodele importante. Se va descrie implemantarea interfetei utilizator.)*

5. Rezultate

Un test este o bucată de cod a cărei funcție este de a verifica dacă o altă bucată de cod funcționează corect. Pentru a face verificarea, apelează metoda testată și compară rezultatul cu rezultatul predefinit așteptat. Un rezultat așteptat poate fi, de exemplu, o anumită valoare returnată sau o excepție. Pentru a lucra cu JUnit trebuie să configurăm Maven-ul. Creăm clasa de testare. Trebuie să implementăm metoda pentru testare și adăugăm în clasă. Apoi trebuie să dăm run la test. Aserțiunile sunt metod statice definite în clasa org.junit.jupiter.api.Assertions: assertEquals , assertAll , assertNotEquals , assertTrue. În cazul în care facilitățile de afirmare oferite de JUnit Jupiter nu sunt suficiente, terț pot fi folosite biblioteci (de exemplu, AssertJ, Hamcrest. Suitele de testare reunesc mai multe clase de testare într-o suită, astfel încât acestea să pot fi rulate. Testele parametrizate fac posibilă rularea unui test de mai multe ori cu diferite argumente. Trebuie să declare cel puțin o sursă care va furniza argumentele pentru fiecare invocare și apoi consumă argumentele din metoda de testare.

Se vor prezenta scenariile pentru testare cu Junit sau alt framework de testare.

6. Concluzii

Din această team se poate învăța multe lucruri. În primul rând operațiile cu polinoame ce se referă la partea matematică. Operațiile sunt de adunare, scădere, înmulțire, împărțire, derivare, integrare. Aceste operații fiind scrise în clase trebuie conectate de interfața grafică. Polinoamele se introduc în interfața grafică. De acolo trebuie convertite prin expresii regulate ceea ce a fost un lucru de învățat și revizuit. Interfața grafică trebuie să fie capabilă de a introduce polinoamele, acest lucru se face prin ActionListener. Folosirea butoanelor, panelului, framului, a metodei ActionPerformed. Designul Model View Controller și cum se creează este un alt lucru pe care a trebuit să îl cercetez în cursul acestei proiect. Junit este un alt domeniu pe care l-am putut folosi pentru beneficiul nostrum. Testările sunt folositoar pentru a testa dacă am lucrat correct și dacă funcționează metodele folosite correct. Folosirea expresiilor regulate este un alt aspect pe care l-am învățat. Pattern-ulile și match-ul sunt structura care ne ajută la conversia din string în monom. Aceste lucruri au fost învățate pe lângă lucrurile deja previzuite în Java.

(*Se vor prezenta concluziile, ce s-a invatat din tema, dezvoltari ulterioare.)*

7. Bibliografie

https://google.github.io/styleguide/javaguide.html

https://www.google.com/search?q=use+case+diagram&rlz=1C1BNSD_enRO980RO980&sxsrf=APq-WBsxOC_EZsCXCum1TfekwOVbddYYdg:1646594624210&source=lnms&tbm=isch&sa=X&ved=2ahUKEwj

2t6j9mrL2AhUVAhAIHdRMDrUQ_AUoAXoECAIQAw&biw=767&bih=704&dpr=1.25#imgrc=dfmnzqrEOzBCwM

https://stackoverflow.com/questions/8942751/use-intellij-to-generate-class-diagram

https://www.google.com/search?q=uml&rlz=1C1BNSD_enRO980RO980&sxsrf=APq-WBugGiQYi7p4XJ_GCMleyWCf4e56iA:1646581013150&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjPvIej6LH2AhUqMuwKHQGUDKwQ_AUoAXoECAIQAw&biw=767&bih=704&dpr=1.25#imgrc=PHR4AloTe-FkhM

https://stackoverflow.com/questions/18754490/using-compareto-and-collections-sort

baeldung.com/regular-expressions-java

https://www.vogella.com/tutorials/JavaRegularExpressions/article.html

https://www.tutorialspoint.com/java/java_regular_expressions.htm

https://stackoverflow.com/questions/17969436/java-regex-capturing-groups

https://math.stackexchange.com/questions/215734/pseudo-code-for-polynomial-long-division

https://www.jetbrains.com/help/idea/testing.html

https://www.guru99.com/java-swing-gui.html

https://docs.oracle.com/javase/tutorial/uiswing/