DOCUMENTATIE

TEMA *1*

NUME STUDENT: MARGINEAN TEODOR IOAN

GRUPA: 6

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)

[3. Proiectare 4](#_Toc95297887)

[4. Implementare 5](#_Toc95297888)

[5. Rezultate 7](#_Toc95297889)

[6. Concluzii 8](#_Toc95297890)

[7. Bibliografie 8](#_Toc95297891)

# Obiectivul temei

Obiectivul principal al acestei teme este realizarea unui calculator polinomial care sa poata sa faca adunarea, scaderea, inmultirea, impartirea, derivarea si integrarea unor polinoame. Obiectivele secundare sunt:

* + Realizarea structurii claselor Monome, Polinom;
  + Realizarea unei interfete grafice;
  + Realizarea convresiei din stringurile introduce de utilizator in obiecte de tip Monome si Polinom cu ajutorul regex;
  + Implementarea operatiei de adunare;
  + Implementarea operatiei de scadere;
  + Implementarea operatiei de inmultire;
  + Implementarea operatiei de impartire;
  + Implementarea operatiei de derivare;
  + Implementarea operatiei de integrare;
  + Realizarea unui model de tip Model View Controller;
  + Realizare de teste cu ajutorul Junit.

Clasa Monome va avea trei atribute de tip integer(int): coeficient, vizita, putere. In atributul coeficient se va stoca coeficientul respectivului monome, in atributul putere se va specifica puterea acestuia, iar atributul vizitat va fi folosit in cadrul adunarii si scaderii, ca un semafor care va decide daca acesta a fost utilizat in adunare sau scadere. Aceste clase vor fi detaliate in capitolul patru.

Interfata grafica va contine un frame in care se vor afla doua TextField-uri, doua TextArea-uri, patru Label-uri si sase butoane. Aceasta va fi detaliata in cadrul capitolului patru.

Metoda de conversie cu regex se gaseste in clasa Polinom, si se foloseste de inca doua metode prin care se parseaza stringul pentru a returna valori de tip int. Aceasta va fi detaliata in cadrul capitolului patru.

Implementare operatiilor s-a facut intr-o clasa separata denumite Operatii, o clas de metode. Acestea vor fi detaliate in cadrul capitolului patru.

Modelul de tip Model View Controller va fi cosntrusit din 3 pachete denumite Model, View si Controller care cuprind mai multe clase. Se poate vedea in capitolul trei.

Realizare de teste cu ajutorul JUnit a fost realziata in Maven in folderul test, unde fiecare operatie a fost testata in parte pe unul dintre cazurile nefavorabile.

# Analiza problemei, modelare, scenarii, cazuri de utilizare

Sunt sase use-case-uri prezente in aceasta tema: adunare, scaderea, inmultirea, impartirea, derivarea, integrarea. Actorii cu care interactioneaza use-case-urile sunt in fiecare caz polinoame asupra caraora se realizeaza aceste operatii.

Primul use-case este adunarea, acesta se rezuma la doua polinoame a caror monoame o sa fie adunate intr-o alta lista de monoame petnru a crea un alt polinom care se reprezinte suma primelor doua.

Al doilea use-case este scaderea, se rezuma la doua polinoame a caror monoame o sa fie scazute si rezultatul acestei scaderi va fi stocat intr-un alt polinom.

Al treilea use-case este inmultirea, se rezuma la dooua polinoame a caror monoame o sa fie inmultite si rezultatul acesta o sa fie stocat intr-un alt polinom.

Al patrulea use-case este impartirea, se rezuma la impartirea lunga a doua polinoame, rezultatul o sa fie trimis ca string spre iesire.

Al cincealea use-case este derivarea care ia un polinom is ii deriveaza monoamele, rezultatul o sa fie stocat in alt polinom.

Al saselea use-case este integrarea care ia un polinom si ii integreaza monoamele, rezultatul o sa fie stoact in alt polinom prin intermediul unui arraylist.

# Proiectare

In cadrul acestei teme au fost folosite patru pachete denumite: model, view, controller si testing. Pachetul model are trei clase denumite Monome, Polinom si Operatii. Pachetul controller are sase clase fiecare dintre acestea corescpunzand unei operatii, aceste sunt denumite: ConttrollerAdunare, ControllerScadere, ControllerImaprtire, ControllerInmultire, ControllerDerivare si ControllerIntegrare. Pachetul view contine doar o clasa View care contine interfata grafica a temei. Ca si algoritmi cunoscui folositi, s-au folosit metoda bulelor pentru metoad de sortare: sortare care se regaseste in clasa polinom din pachetul model.

Diagrama pachet controller: Diagrama pachet model:

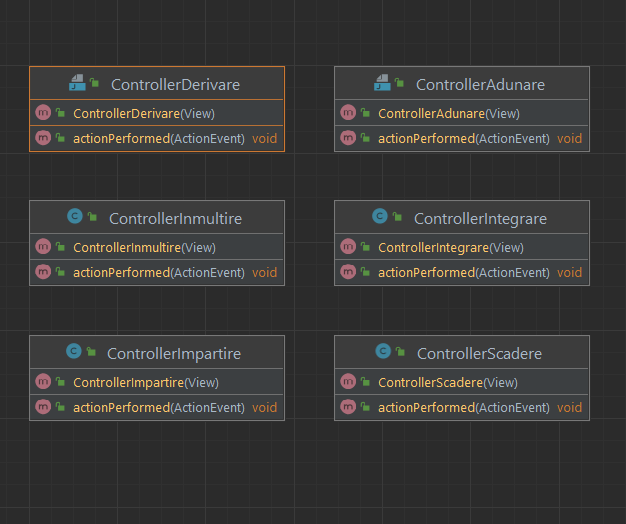
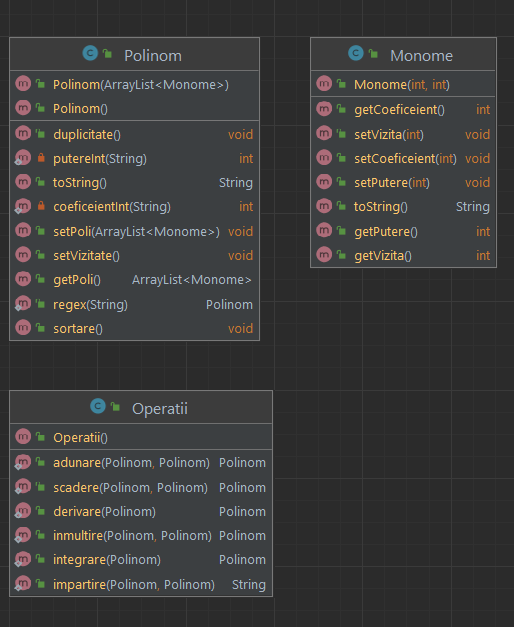
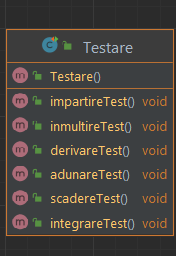


Diagrama pachet view: Diagrama pachet testing:



# Implementare

Tema dispune de 4 pachete care contin 10 clase normale si o clasa pentru testare. Pachetul testing va fi exemplificat si detaliat in capitolul urmator. Pachetul model dispune de trei clase Monome, Polinom si Operatii.

Calsa Monome are trei attribute: coefficient, vizita si putere, fiecare dintre acestea sunt de tip int. Clasa dispune de getere si setere pentru fiecare dintre attribute, constructori si o metoda de suprascriere a metodei toString.

Clasa Polinom are un singur atribut: un ArrayList de monoame. Aeasta dispune de setere si getere cat si de constructori. Pe langa aceste metode se poate remarca prezenta unei metode suprascrise de toString, O metoda de setare a atributului vizita a monoamelor din ArrayList, denumita setVizitate. O alta metoda implementata in aceasta clasa este metoda de sortare care ia Polinomul si il ordoneaza descrescator in functie de atributul putere al monoamelor din ArrayList. O alta metoda regasita aici este metoda denumita duplicitate, care verifica si face o stranger a monoamelor ccare au atributul putere egal. O alta metoda regasita in cadrul acestei clase este metoda regex, care primeste un string si il converteste intr-un ArrayList si un polinom nou. Aceasta convertire se face cu ajutorul parsarii coeficientilor dati de sirul de charactere din variabila matcher, cu ajutorul metodelor coeficientInt si putereInt care iau prima, respective a doua valoare din matcher si il transforma in atributul coefficient, putere a monomului.

Clasa Operatii este una dintre cele mai complexe clase din cadrul temei. In ea se regasesc sase metode fiecare corespunde unei operatii in ordine, aceste se numesc al fel ca operatiile: adunare, scader, inmultire, derivare, integrare, impartire.

Metoda adunare apeleaza pentru fiecare din cele doua argumente metoda setVizitate, se creaza un nou arraylist si un nou monome pentru prelucrearea datelor. Se parcurge cu un foreach primul argument, intr-un foreach embricat se parcurge al doilea argument, se cauta care monoame care au aceasi putere pentru a le pune in lista finala o singura data. Se marcheaza prelucrarea acestora prin setarea atributului vizita, se creaza monomul nou, se adauga in lista si se reseteaza monomul. Dupa identificarea monomelor cu puteri egale se parcurg individual fiecare dintre cele doua polinoame si de adauga in lista fieareelement neprelucrat in for-urile embricate. Se sorteaza polinomul nou rezultat prin apelarea metodei sortare, regasita in clasa polinom. Se returneaza polinomul.

Metoda scadere apeleaza pentru fiecare din cele doua argumente metoda setVizitate, se creaza un nou ArrayList si un un nou monome pentru prelucrarea datelor. Se parcurge cu doua for-uri embricate, se cauta monoamele cu putere egala si se scad coeficientii, se salveaza in lista, se reseteaza monomul, si se marcheaza prelucrarea monoamelor. Dupa identficarea monoameluc cu puteri egale se parcurg individual fiecare dintre cele doua polinoame si se adauga in lista fiecare element neprelucrat in for-urile embricate. Monoamele neprelucrate se adauga cu coeficientul negat in cazul celui de-al doilea polinom si cu coefficient normal daca fac parte din primul monom.

Metoda inmultire apeleaza metoda setVizitate, se creaza un now arrayList, In care se tot adauga monoame. Se folosesc doua for-uri embricate intre elementele carora se face un produs cartezian, astfel fiecare elment din primul polinom ajunge sa fie inmultit cu fiecare element din al doilea polinom. Se apeleaza metoda duplicitate regasita in clasa polinom pentru a se restrange monomele care au aceasi putere in polinomul rezultat in urama inmultirii. Se sorteaza prin intermediul metodei sortare.

Metoda derivare apeleaza metoda setVizitate pentru un polinom. Se creaza un arraylist nou pentru stocarea rezultatului, si un monom petnru prelucrarea datelor. Intr-un for se creaza monoame noi decrementand cu unu puterea monomului nou si inmultind coeficientul cu puterea nedecrementata. Dupa terminarea for-ului se apeleaza metoda de sortare: sortare. Se returneaza polinomul.

Metoda integrare apleeaza metoda setVizitate pentru un polinom. Se creaza un arraylist nou pentru stocare rezultatului si un monom pentru prelucrarea datelor. Intr-un for se creaza monoame noi impartind coeficientul cu puterea si incrementad putearea cu unu. Dupa terminarea for-ului se apeleaza metoda de sortare: sortare. Se returneaza polinomul.

Metoda impartire apelaeaza pentru argumentele sale metoda de sortare: sortare. Declara doua noi polinoame, prin intermediul unui if va face diferenta intre care polinom are gradul mai mare, salvandu-l in polinomul p si cel cu grad mai mic in polinomul q. Se decalra trei noi monoame in care se vor salva monoamele cu puterea cea mai mare din polinomul p si din polinomul q, prin apelarea geterului getPoli care retureneaza arraylistul si prin apelarea metodei specific arraylistlui .size care returneaza numarul de elemnte plus unu. Se declara doua noi liste de tip arraylist pentru prelucrarea datelor. Se foloseste o instructiune repetitiva de tip while pentru a se prelucra atata timp cat polinomul p are gradul mai mare decat polinomul q. Se face un monom nou a carui coefficient vor fi coeficientele impartite a celor doua monoame cu cel mai mare grad in polinoamele p si q, puterea acestuia o sa fie diferenta dintre cele doua puteri. Se adauga in cele doua liste, o lista se transforma in polinom, se apeleaza inmultirea intre polinomul nou creat si polinomul q. Polinomul nou rezultat va fi salvat in polinomul B, se va apela metoda scadere intre polinomul p si polinomul B si se va salva in polinomul p. Se va reinitialize cu monomul cu cel mai mare grad in p, se va da clear la lista care formeaza polinomul auxiliar si se va re peta procesul. Dupa termianrea instructiunii repetitive de tip while, se va face o interogare de tip if unde se vor compara gradele polinoamelor, iar daca sunt egale se va mai face o impartire cu monomul auxiliar urmate de inmultire si scadere. Astefl in urma acestora in lista avem catul impartirii si restul acesteia. Se mai foloseste inca o interogare pentru a trimite stringul necesar in functie de polinomul de rest care poate sau nu sa fie null.

Pachetul view are este format dintr-o singura clasa care va reprezenta interfata grafica a sistemului denumite tot view. Aceasta va dispune de doua textfield-uri in care se vor afisa rezultatele in urma apelarii operatiilor, de doua textarea-uri unde vor fi introduce polinoamele de la tastatura dupa modelul specific, de sase butaone care vor fi folosite pentru apelarea operatiilor asupra polinoamelor introduce ulterior si de patru label-uri care ajuta cu orientarea si organizarea.

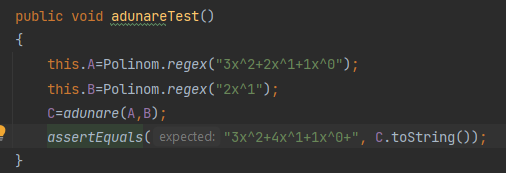
Pe langa acestea se regasesc si referinte spre pachetul de controller pentru a face lagatura intre aceastea. Pentru afisarea interfetei se face din constructor. Aceasta clasa dispune si de doua getere pentru textarea-uri pentru a prelua si prelucra ulterior textul din aceastea. Dispune si de doua setere pentru textfield-uri prin care se incarca in acestea textul trimis ca parameter prin cadrul argumentului de tip string.

Pachetul controller dispune de sase clase asemanatoare, denumite Controller si se adauga numele operatiei. Aceastea au ca si atribut un obiect de tip view prin care se face legatura in clasa view din pachetul view. Aceste clase au o metoda de tip actionPerformed prin care se face operatia necesara si se returneaza string-ul necesar afisarii. Prin intermediul geterelor din view se folosesc string-urile si se transforma in polinoame cu ajutorul metodei regex, intr-unul dintre polinoame se salveaza rezultatul operatiei si cu ajutorul seterelor se afiseaza in textarea-uri. Pentru clasele de controller pentru derivare sau pentru integrare se face separate o apleare a operatiei pentru fiecare dintre cele doua polinoame introduse initial.

Astfel aceasta este implementarea intreaga a acestui calculator polinomial.

# Rezultate

In cadrul testarii s-a folosit Junit, am setat versiunea in pom.xml, am rulat pentru fiecare operatie in parte cate un test, pe un caz oarecare, aceste cazuri se pot schimba usor, doar prin schimbarea string-ului de intrare. Fiecare test poate fi rulat individual, sau pot fi rulate taote odata. A fost aleasa abordarea in care in fata fiecarei metode implementate am marcat ca este un test cu ajutorul”@Test”. Testarea prorpiu-zisa a fost realziata cu ajutoul metodei asserEquals. Rata de success a testelor este de suta la suta. Un exemplu de astfel de test se poate remarca in imaginea urmatoare, in care este reprezentat codul de testare pentru metoda de adunare a doua polinoame:

**

# Concluzii

In concluzie in urma realizarii temei, am invatat sa folosim conversie de string prin intermediul regex, am revenit asupra conceptelor invatate si le-am reimprospatat, am refacut un model architectural de tip model controller view,am recapitulat metoda de testare cu ajutorul JUnit, am invatat sa avem un proiect structurat. Am reveneit peste operatiile de baza folosite la polinoame, am recapitulat conceptele matematice, pentru a implementa cerintele( operatiile pe polinoame).

# Bibliografie

<https://dsrl.eu/courses/pt/>

<https://cadredidactice.ub.ro//sorinpopa/files/2018/10/L1_diagrame_use_case.pdf>

https://www.google.com/search?q=exemple+de+impartire+cu+polinoame&sxsrf=APq-WBuzcU7CenE1BgjH5DJjtDs7Ia6m9w:1646581625198&source=lnms&tbm=isch&sa=X&ved=2ahUKEwiX9\_PG6rH2AhUghf0HHaJfATUQ\_AUoAXoECAEQAw&biw=870&bih=876&dpr=1.1#imgrc=YWvtVLDSkx6u8M