Documentatie Tema 1

Calculator polinoame

Negrea Malina-Adela

Grupa 30224

Profesor laborator: Pop Claudia

Cuprins

1. Obiectivul temei
2. Analiza problemei
3. Proiectare
4. Implementare
5. Testare
6. Rezultate
7. Concluzii
8. Bibliografie
9. Obiectivul temei

**Obiectiv principal**

Propuneti, proiectati si implementati un sistem de procesare a polinoamelor de o singura variabila cu coeficienti intregi.

**Obiective secundare**

* Cazuri de utilizare si scenarii de utilizare – capitolul 2
* Alegerea structurilor de date – capitolul 3
* Impartirea pe clase – capitolul 3
* Dezvoltarea algoritmilor – capitolul 4
* Implementarea solutiei – capitolul 4
* Testare – capitolul 5

1. Analiza problemei

Cazuri de utilizare:

* User-ul introduce doua polinoame fara o ordine anume a termenilor si posibil mai multi termini cu acelasi grad, iar rezultatul este un polinom ordonat dupa grad, ce contine cel mult un termen de un anumit grad
* Suma/Diferenta dintre primul si al doilea polinom
* Impartirea/Inmultirea celor doua polinoame
* Integrarea primului polinom
* Derivarea primului polinom
* Resetarea campurilor celor doua polinoame

Scenarii de utilizare:

Utilizatorul introduce pe rand in zonele corespunzatoare primul si al doilea polinom in format LaTeX, iar apoi apasa pe butonul corespunzator operatiei dorite. Rezultatul apare in caseta sa. Daca se doreste efectuarea unei alte operatii pe aceleasi polinome se apasa pe butonul corespunzator, fiind afisat nou rezultat.

In cazul in care utilizatorul doreste sa efectueze operatii pe alte polinoame poate apasa butonul Clear, ce reseteaza campurile, sau modifica manual campurile.

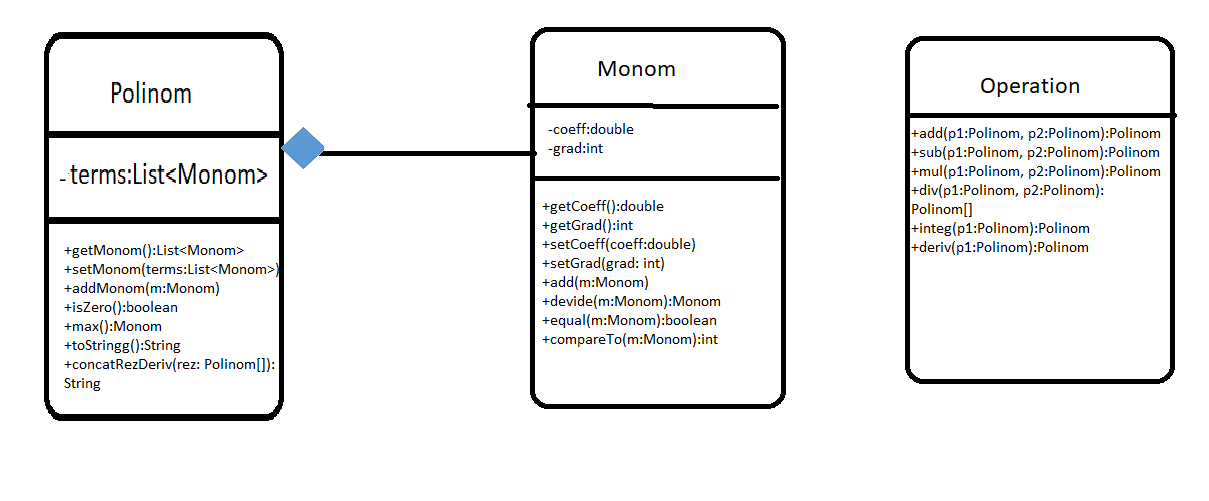
Pentru inchiderea ferestrei utilizatorul poate apasa pe butonul X din coultul dreapta sus sau apasa pe butonul Close.

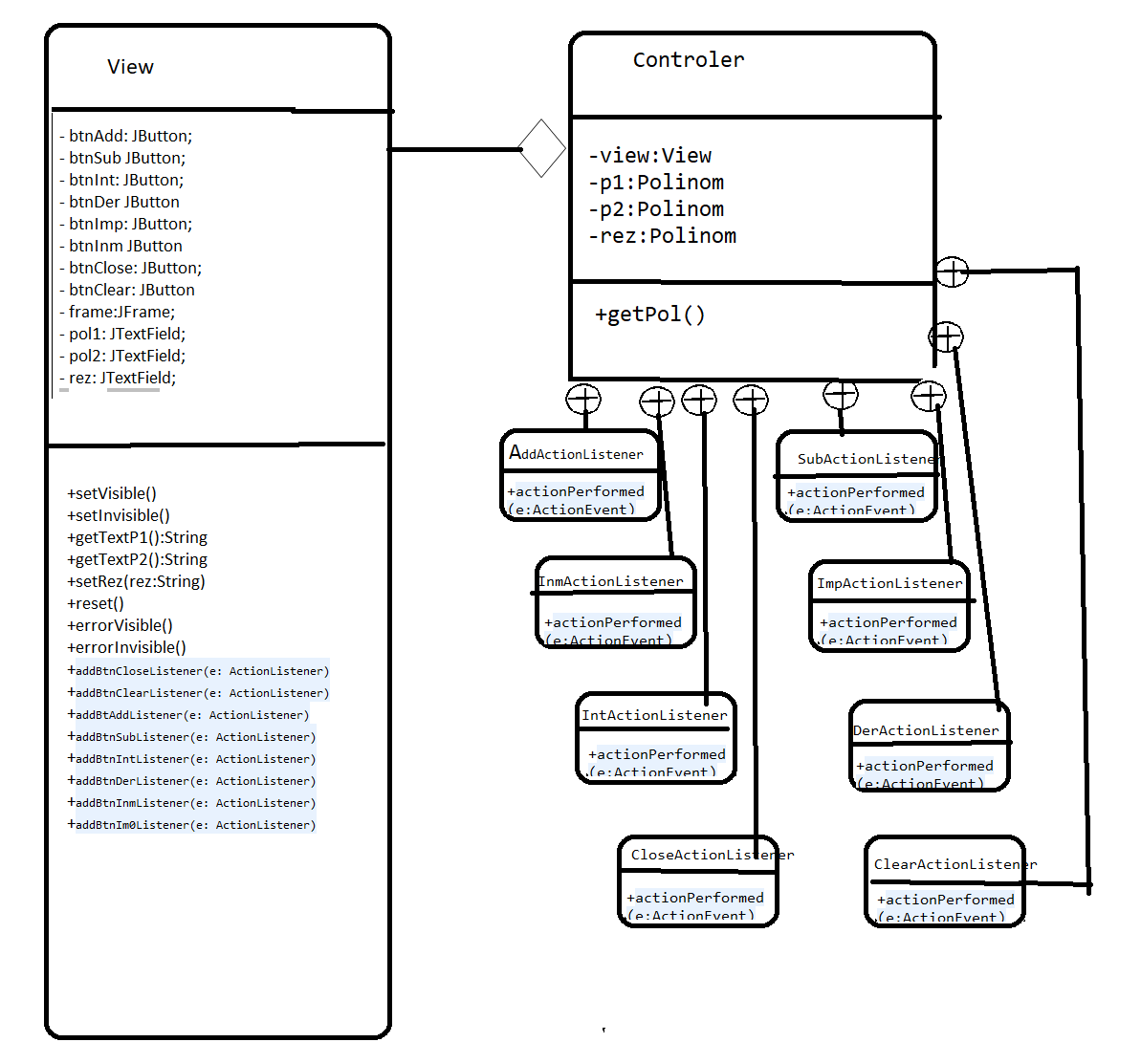
Pentru functionarea corecta a aplicatiei facem urmatoarele presupuneri legate de datele de intarare:

* In cele doua campuri sunt introduce siruri cu semnificatie de polinom
* Este folosita doar variabila x
* Fiecare termen are un grad( inclusive termenul liber)
* Coeficientii polinoamelor introduse sunt intregi

1. Proiectare

Diagrama UML





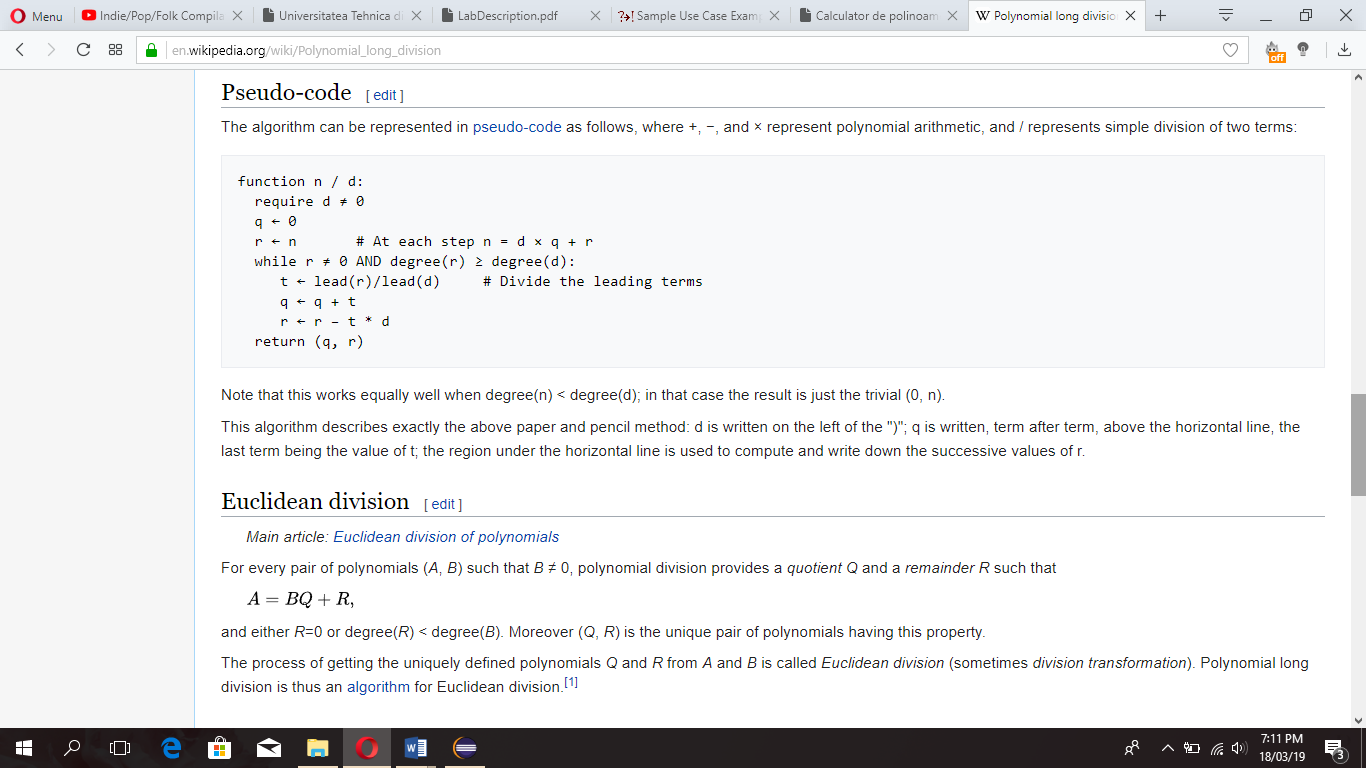
Aplicatia este impartita in trei pachete:

Clasele Controler si View se afla in packge-ul edu.utcluj.lic.TP.Assigment1.GUI, clasele Monom si Polinom se afla in packed-ul edu.utcluj.lic.TP.Assigment1.Model, iar clasa Operation in packge-ul edu.utcluj.lic.TP.Assigment1.Logic.

Pe langa tipurile simple de date sunt folosite colectiile. Fiecare polinom este o lista de monoame, astfel operatiile cu acestea( i.e. polinoame) fiind facilitate. Listele sunt de tipul List<Monom> sunt implemnetate cu ArrayList<Monom>.

Intefata grafica are un aspect simplu, usor de folosit, cu doua campuri pentru introducerea polinoamelor, un camp pentru afisarea rezultatului si butoane separate pentru fiecare operatie.

Dintre toate operatiile efecutuate cu polinoame, impartire a doua polinoame este cea mai dificila. Pestru aceasta este folosit algoritmul polynomial long division prezentat in pseudo-codul de mai jos.

 In cazul operatiei de impartire se verifica de asemenea ca polinoamele introduce de utilizator sa aiba grade corespunzatoare( gradul numaratorului sa fie mai mare decat gradul numitorului). In caz contrar este afisat un mesaj de eroare, iar operatia nu se realizeaza.

1. Implementare

Programul contine 6 clase de baza si o clasa main.

1. Clasa View – contine descrierea interfetei grafice, avand pur rolul de a o trasa.
2. Clasa Controler – reprezinta mediul de interactiune dintre actiunile utilizatorului si actiunile programului. Aceasta traduce actiunile utilizatorului prin intermediul evenimentelor, in cadrul clasei fiind implementati ascultatorii butoanelor din interfata grafica.
3. Clasa Polinom – creaza polinoame sub forma de monoame, entitati fundamentale. Aceasta asigura adaugarea corecta a noilor elemnte in polinom astfel incat termenii sa fie ordonati descrescator in functie de grad, iar fiecare termen de un anumit grad apare cel mult o data.
4. Clasa Monom – clasa fundamentala a proiectului ce contine gradul precum si coeficientul intreg al unui termen. In aceasta clasa este implementata operatia de adunare a doua monoame.
5. Clasa Operation – clasa ce contine metodele ce implementeaza operatiile de adunare, scadere, imultire, impartire de polinoame, respective integrarea si derivara unui polinom.
   1. Clasa View

Clasa contine un JFrame pe care este construita intreaga interfat. Acesta nu are un layout predefinit, folosinduse coordinate absolute pentru pozitionarea coponentelor in fereasta. Fereastra se deschide automat in centrul ecranului.

Clasa contine pe langa label-uri, 3 campuri TextField, doua pentru introducerea polinoamelor si unul pentru afisarea rezultatului, cel din urma nefiind modificabil de catre utilizator, 6 butoane aferente celor 6 operatii posibile , precum si un buton de resetare si un buton de inchidere a ferestrei. In plus, exista si un mesaj de eroare (un JLabel), ce este vizibil doar in cazul in care utilizatorul introduce date incorecte pentru operatia de impartire.

Principalele metode in clasa au rolul de a prelua datele introduce de utilizator, a reseta campurile si a face vizibil sau invizibil mesajul de eroare. Clasa nu descrie explicit modul de tratare a evenimentelor, de aceasta ocupandu-se metode din clasa Controler.

* 1. Clasa Controler

Clasa Controler controleaza actititatea integului program. Aceasta interactioneaza atat cu interfata grafica cat si cu zona de logica a programului dirijand functionarea aplicatiei.

Clasa are ca atribute un obiect de tip View sit trei obiecte de tip polinom reprezentand cele doua polinoame introduce de utilizator si rezultatul ultimei operatii.

Clasa controler are opt clase interne ce au rolul de a implementa modul de tratare a evenimentelor. Cele opt clase corespund celor opt butoane ale interfetei grafice.

Clasa AddActionListener implementeaza interfata ActionListener. La apasarea butonului Add cele doua polinoame introduce de utilizator sunt preluate si trimise ca parametru metodei statice add din clasa Operation. Rezultatul este mai apoi convertit intr-un string si apoi afisat in interfata grafica.

Clasa SubActionListener implementeaza interfata ActionListener. La apasarea butonului Subtract introduce de utilizator sunt preluate si trimise ca parametru metodei statice sub din clasa Operation. Rezultatul este mai apoi convertit intr-un string si apoi afisat in interfata grafica.

Clasa InmActionListener implementeaza la randul ei interfata ActionListener si are aceeasi actiune precum metodele din clasele prezentate mai sus cu deosebirea ca trateaza apasarea butonului Multiplication si apeleaza metoda statica mul din Operation.

Clasa ImpActionListener implementeaza interfata ActionListener. La apasarea butonului Devision aplicatia preia polinoamele introduse de utilizator si verifica daca gradul numaratorului este mai mare decat cel al numitorului. Daca da apeleaza metoda statica div din Operation, iar apoi afiseaza rezultatul adus in forma dorita de string de catre metoda concatRezDiv din clasa Polinom.

Clasele DerActionListener si IntActionListener implementeaza interfata ActionListener si trateaza apasarea butonului Derivation respectiv Integral. Prima apeleaza metoda deriv, iar cea de-a doua metoda integ, ambele statice si ambele din clasa Operation, avand ca argument polinomul introdus de catre utilizator din primul camp. Rezultatul este afisat sub forma de string.

Pe langa aceste clase interne, clasa Controler are o metoda private getPol() ce are rolul de a prelua cele doua polinoame introduce de catre utilizator si de a le converti in obiecte de tip Polinom cu ajutorul constructorului din clasa Polinom.

4.3 Clasa Polinom

Aceasta clasa implementeaza efectiv idea de polinom prin intermediu unei liste de monoame. Ea are trei contrucori publici, unul fara parametrii ce are rolul de a creea un ArrayList<Monom> gol, un al doilea ce este folosit pentru a creea un polinom copie a celui trimis ca si parametru, si un al treilea ce primeste ca si parametru un string si folosind un Regex corespunzator imparte stringul in monoame ce le adauga polinomului.

Singurul atribut al clasiei Polinom este terms ce reprezinta un List<Monom>, acesta fiind obiectul pe care sunt implementate toate operatiile. Pentru acest atribut sunt definite getter si setter.

Principala metoda din acesta clasa este addMonom ce are rolul de a aduga un nou termen la lista de monoame. Pentru acest lucru se verifica mai intai daca exista in lista un termen cu acelasi grad ca si mononul ce trebuie adaugat. Daca da atunci coeficientul sau este adaugat coeficientului termenului existent. In cazul in care coeficientul termenului devine 0 acesta este scos din lista. In cazul in care monomul nu se afta in lista, acesta este adaugat la sfarsitul acesteia iar mai apoi lista este reordonata dupa gradele termenilor. Cu ajutorul acestei metode ne asiguram ca un termen de un grad apare cel mult o data si ca lista este ordonata, astfel operatiile fiind mai usor de efectuat, iar rezultatul fiind lizibil.

Metoda toStringg returneaza un string construit din monoamele polinomului pentru a avea acelasi format ca cel introdus de catre utilizator.

Metoda concatRezDiv are un rol asemanator cu cea anterioara, cu deosebirea ca acesta contruieste un string din cele doua elemente ale parametrului, catul si restul operatiei de impartire, pentru a arata conform unui rezultat al impartirii a doua polinoame.

Alte doua metode sunt isZero ce imi returneaza false in cazul in care exista cel putin un element in list ce are coeficientul diferit de zero si true in caz contrar, si metoda max ce returneaza monomul cu cel mai mare grad.

* 1. Clasa Monom

Aceasta este clasa fundamentala a aplicatiei. Acesta are un singur contructor ce instantiaza cele doua attribute ale clasei, si anume coeff, un double, si grad, un int. Avand in vedere ca cele doua atribute sunt private, sunt definite pentru ambele getter si setter.

Metoda add aduna un double trimis ca paramentru la coeficientul obiectului, iar metoda divide creeaza un nou monom ce il trimite ca parametru, acesta fiind rezultatul impartirii monomului actual cu un monom trimis ca parametru.

Metoda compareTo compara doua monoame in functie de grad.

Metoda equals determina daca doua monoame sunt egale, daca au atat gradele cat si coeficientii egali.

Metoda equals suprascrie metoda din Object, iar metoda compareTo implementeaza metoda din Comparable ( clasa Monom implementeaza interfata Comparable<Monom>).

* 1. Clasa Operation

Clasa operation contine implementarea operatiilor ce se efectueaza pe polinoame. Acesta contine doar metode statice, astfel ca nu contine atribute.

Metoda add aduna doua polinoame astfel: creeaza un nou poliniom ce are aceleasi atribute ca primul polinom, iar apoi adauga pe rand monoamele din p2 cu ajutorul metodei addMonom astfel un termen de un anumit grad apare cel mult o data, iar termenii sunt ordonati dupa grad.

Metoda sub se comporta la fel ca metoda add cu deosebirea ca coeficientii monomului p2 sunt inmultiti cu -1 inainte de a fi adunati.

Metoda deriv calculeaza derivate unui polinom astfel: creaza un polinom gol, iar apoi adauga pe rand monoamele polinomului parametru modificate corespunzator, coeficientul devine vechiul coeficient inmultit cu vechiul grad, iar din grad se scade o unitate.

Metoda integ calculeaza integrala dintr-un polinom astfel: creeaza un polinom gol, iar apoi adauga pe rand monoamele polinomului parametru modificate corespunzator, coeficientul devine vechiul coeficient impartit cu noul grad, iar din grad la grad se adauga o unitate.

Metoda mul calculeaza produsul a doua polinoame astfel: creeaza un polinom gol la care adauga pe rand produsul dintre fiecare monom al primului polinom cu fiecare monom al celui de-al doilea polinom.

Metoda div efectueaza impartirea intre doua polinoame conform algoritmului prezentat la sectiunea 3, returnand un array de doua polinoame.

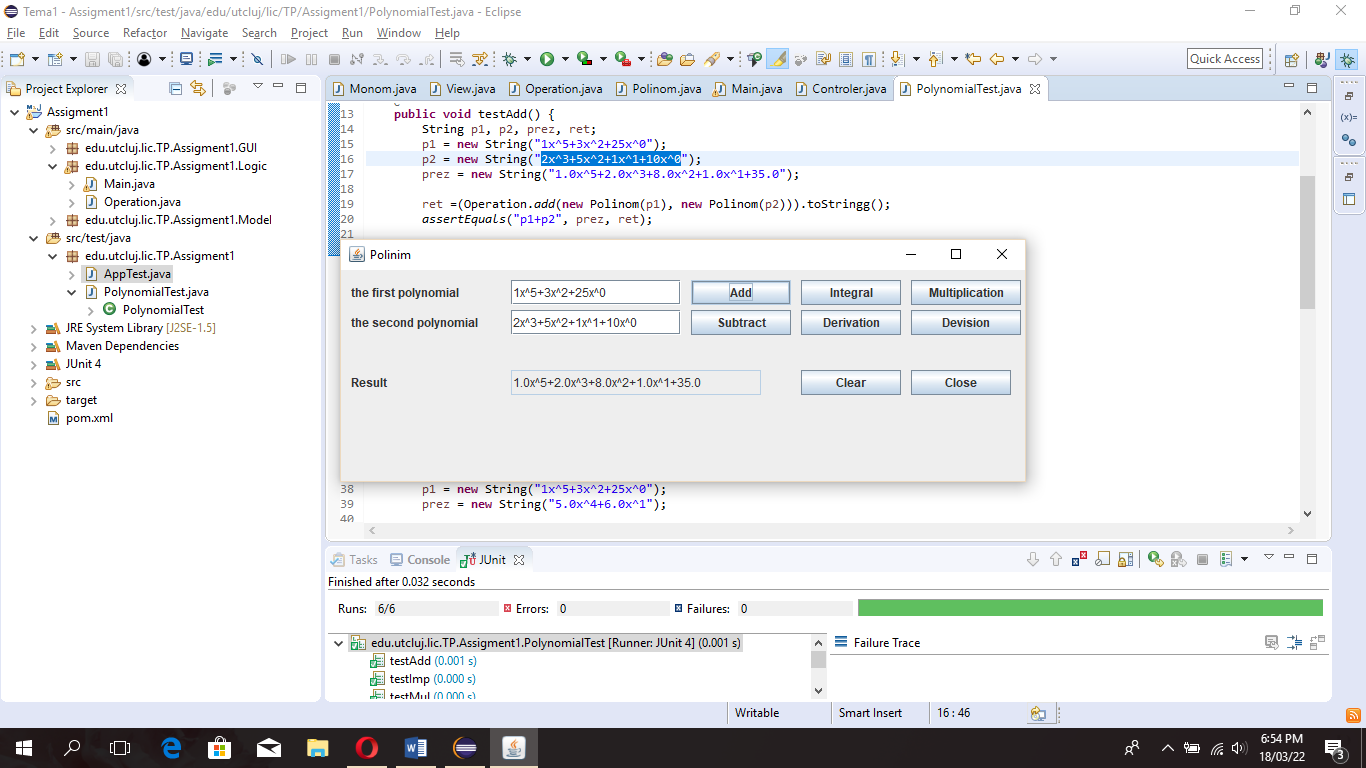
1. Testare

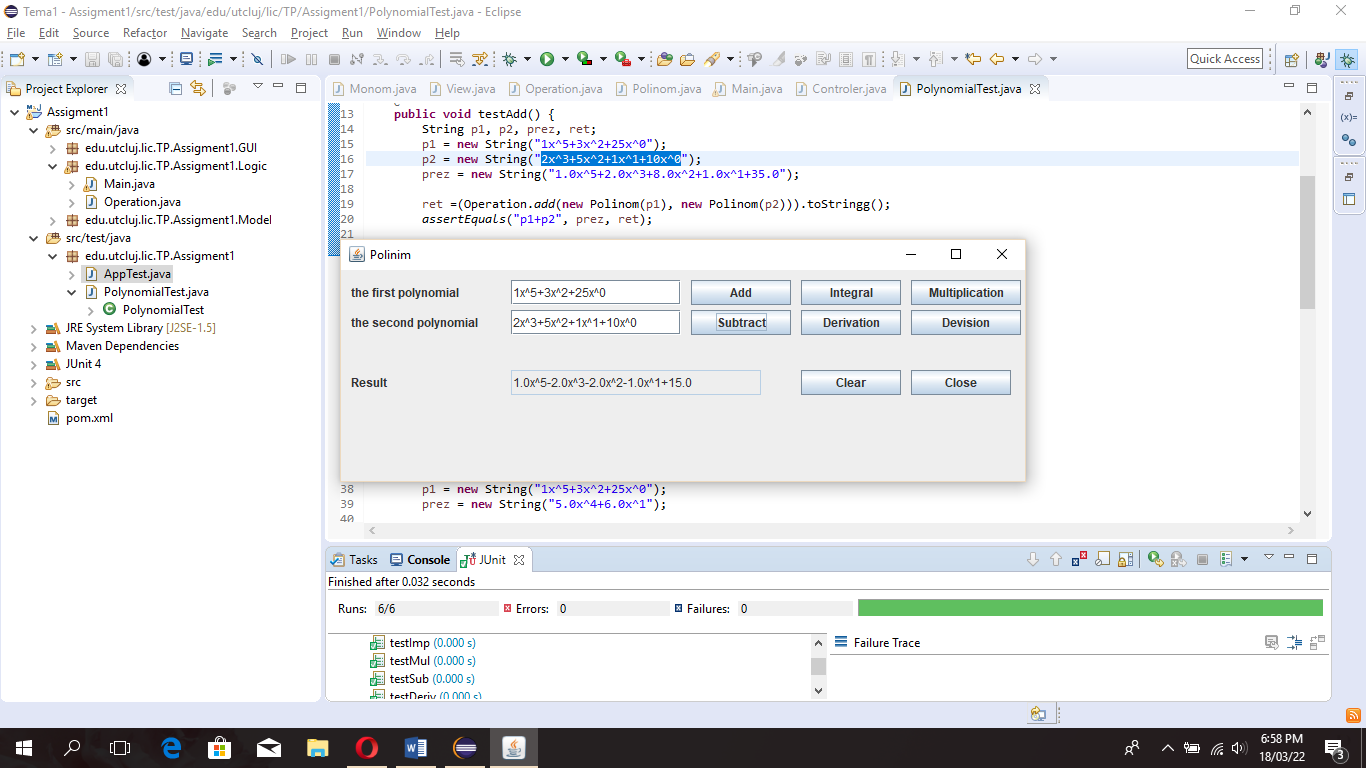
Testarea aplicatiei s-a realizat cu ajutorul Junit-ului.

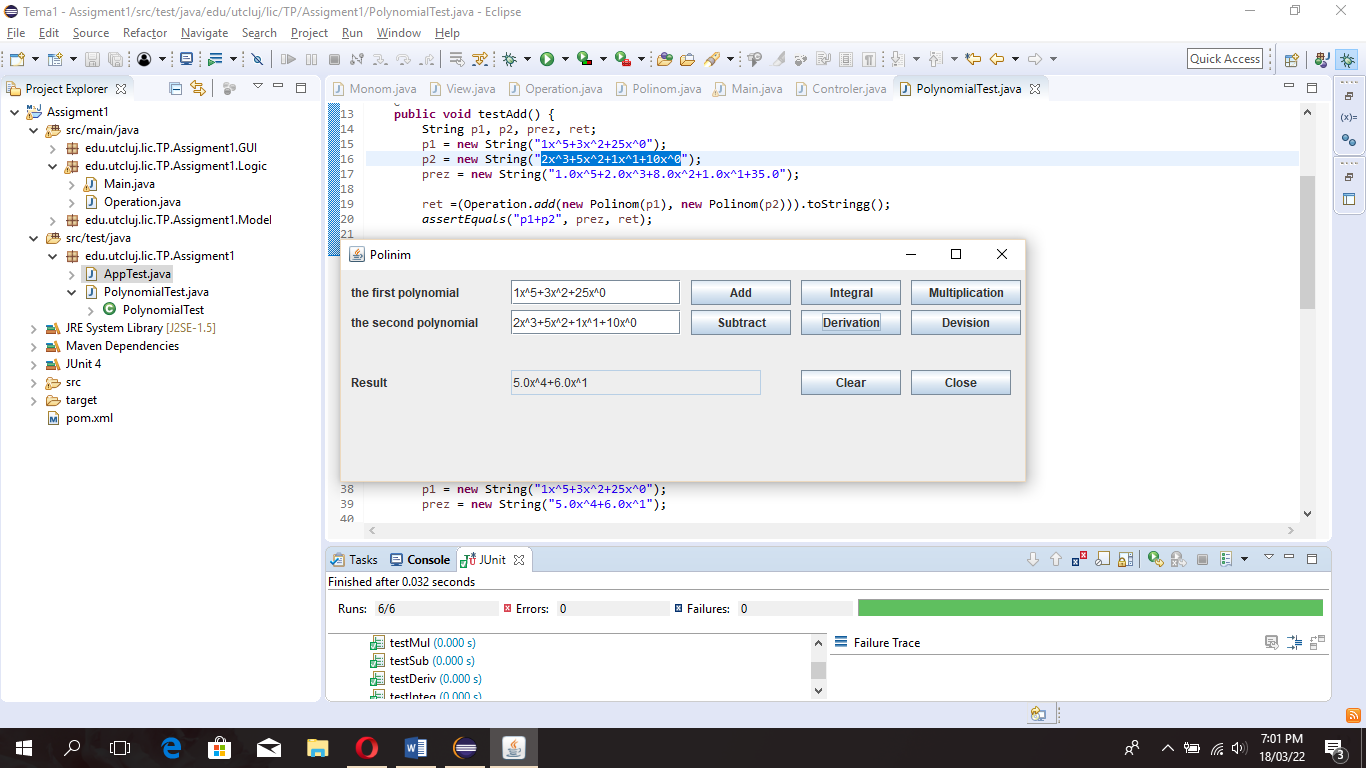
Pentru aceasta a fost creata clasa PolynomialTest. Acesta contine sase metode aferente celor sase operatii ce se port efecutua. Pentru toate operatii( mai putin impartirea pentru un rezultat mai scurt), s-au folosit aceleasi date de intarare, rezultatul asteptat fiind comparat cu cel real.

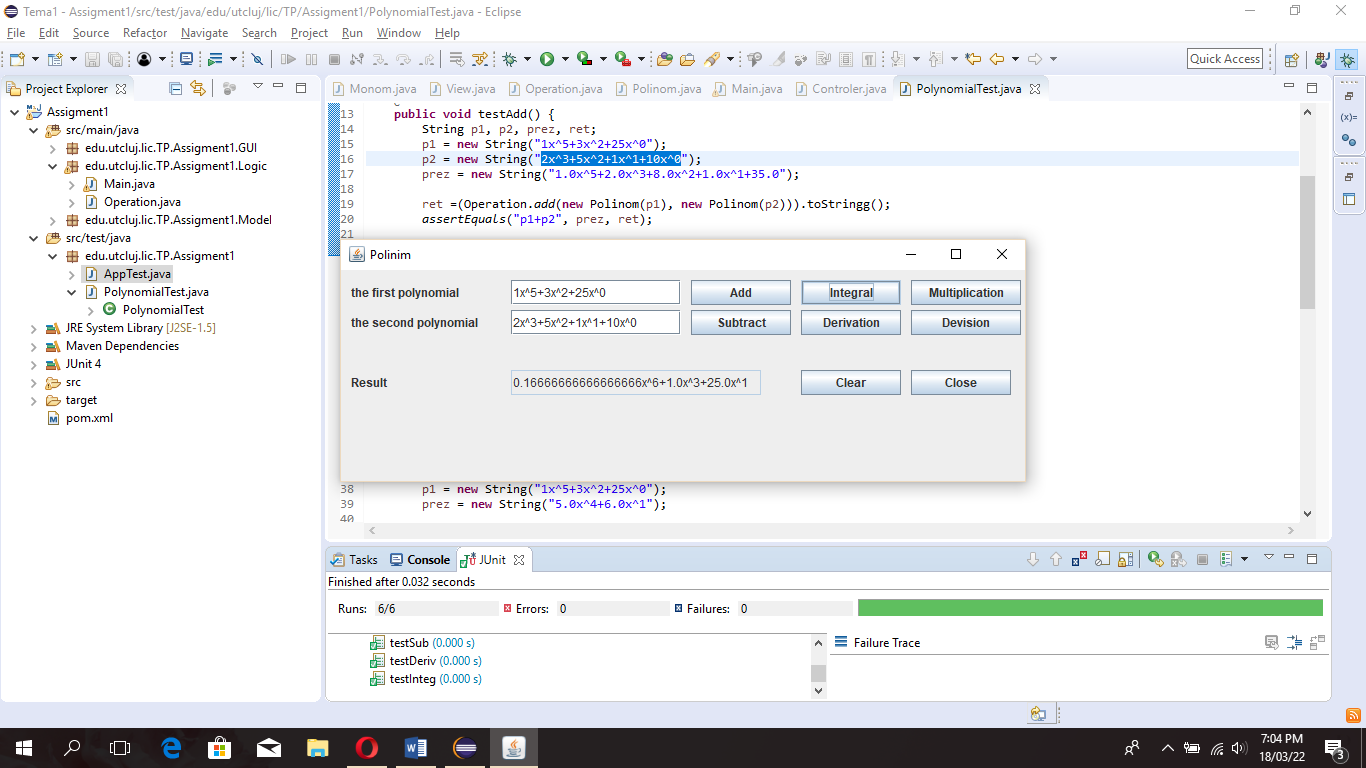
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ce se testeaza | Date de intrare | Rezultat asteptat | Rezultat efectiv | Pass/ Fail |
| Operatia de adunare | P1= 1x^5+3x^2+25x^0  P2= 2x^3+5x^2+1x^1+10x^0 | 1.0x^5+2.0x^3+8.0x^2+  1.0x^1+35.0 | 1.0x^5+2.0x^3+8.0x^2+  1.0x^1+35.0 | Pass |
| Operatia de scadere | P1= 1x^5+3x^2+25x^0  P2= 2x^3+5x^2+1x^1+10x^0 | 1.0x^5-2.0x^3-2.0x^2-1.0x^1+15.0 | 1.0x^5-2.0x^3-2.0x^2-1.0x^1+15.0 | Pass |
| Operatia de derivate | P1= 1x^5+3x^2+25x^0  P2= 2x^3+5x^2+1x^1+10x^0 | 5.0x^4+6.0x^1 | 5.0x^4+6.0x^1 | Pass |
| Operatia de integrare | P1= 1x^5+3x^2+25x^0  P2= 2x^3+5x^2+1x^1+10x^0 | 0.166666666  66666666x^6+  1.0x^3+25.0x^1 | 0.16666666  666666666x^6+  1.0x^3+25.0x^1 | Pass |
| Operatia de inmultire | P1= 1x^5+3x^2+25x^0  P2= 2x^3+5x^2+1x^1+10x^0 | 2.0x^8+5.0x^7+  1.0x^6+16.0x^5+  15.0x^4+53.0x^3+  155.0x^2+25.0x^1+250.0 | 2.0x^8+5.0x^7+1.0x^6+16.0x^5+15.0x^4+53.0x^3+155.0x^2+25.0x^1+250.0 | Pass |
| Operatia de  impartire | P1= 1x^3-4x^2+2x^1-3x^0  P2= 1x^1+2x^0 | 1.0x^2-6.0x^1+14.0+(-31.0)/(1.0x^1+2.0) | 1.0x^2-6.0x^1+14.0+(-31.0)/(1.0x^1+2.0) | Pass |

1. Rezultate

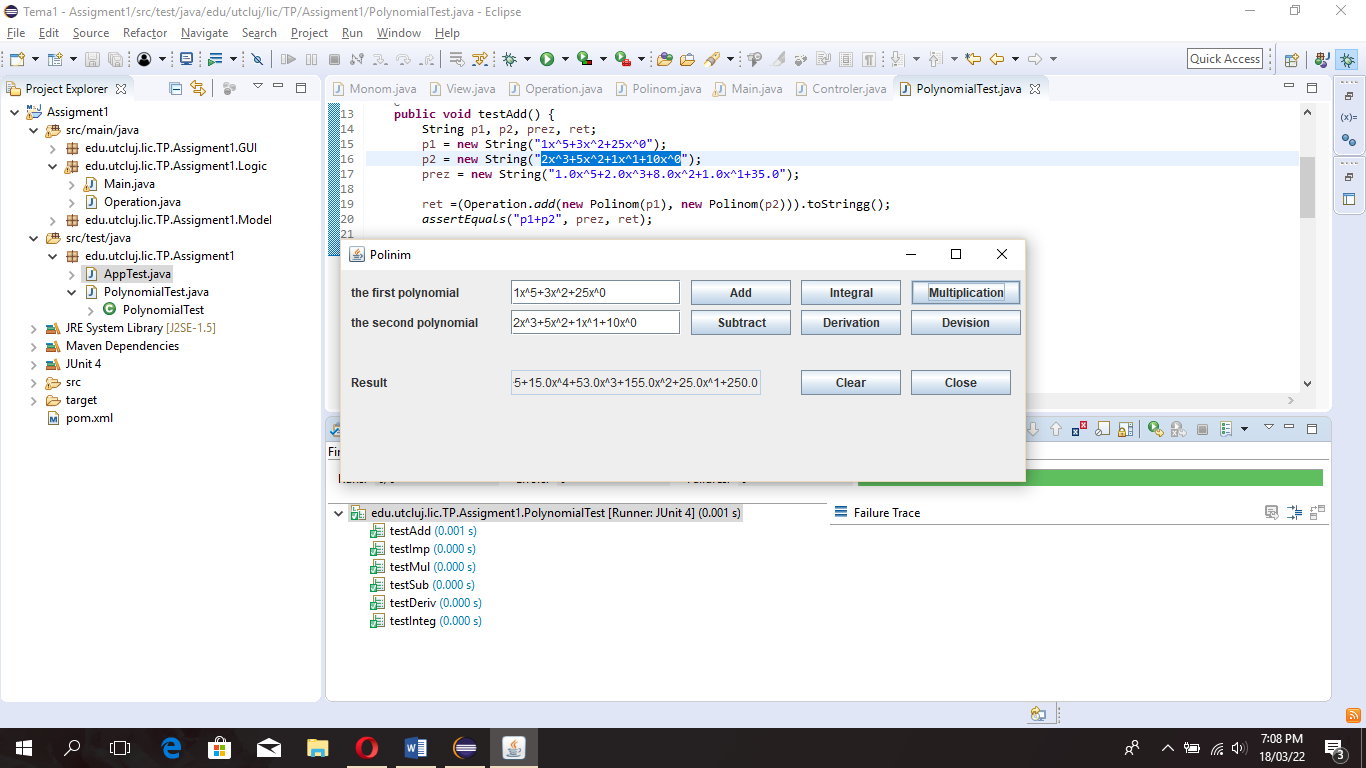
Operatia de adunare:

Operatia de scadere:

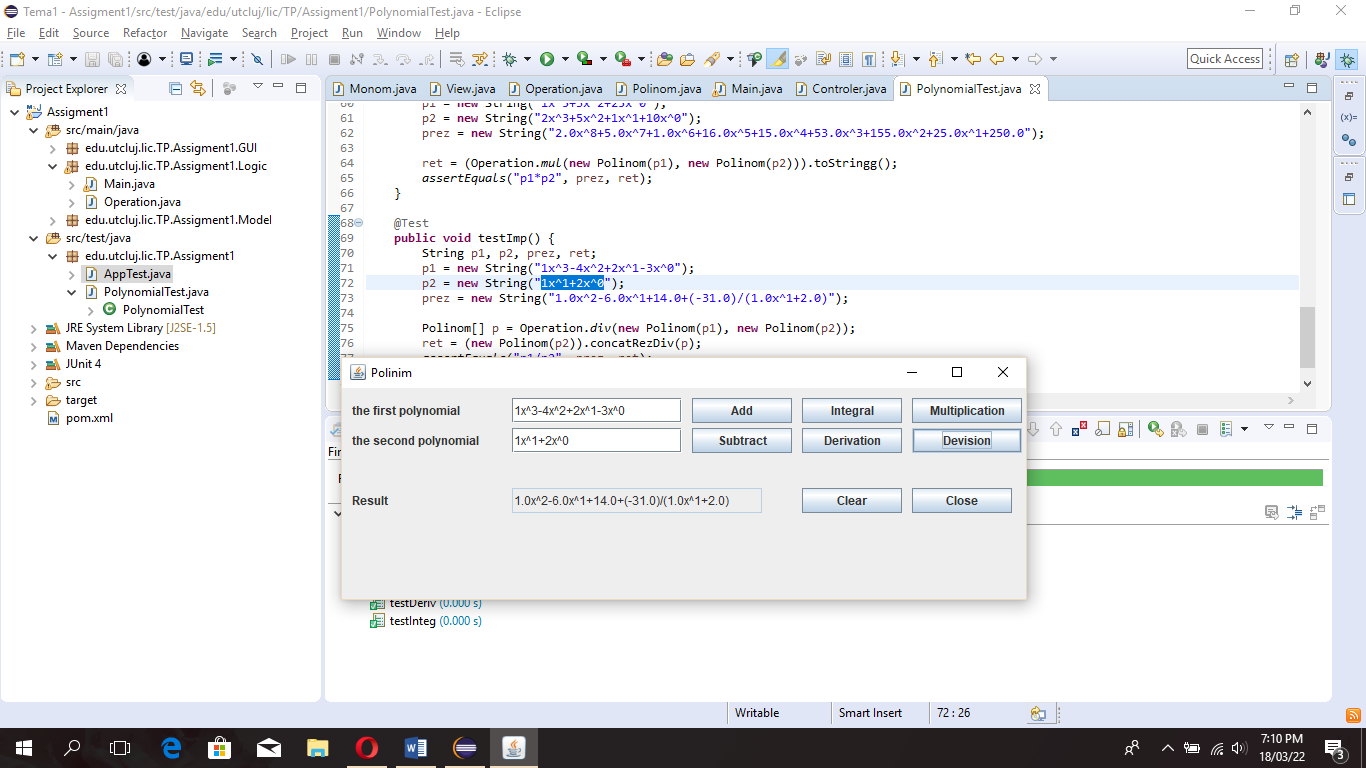
Operatia de derivare:

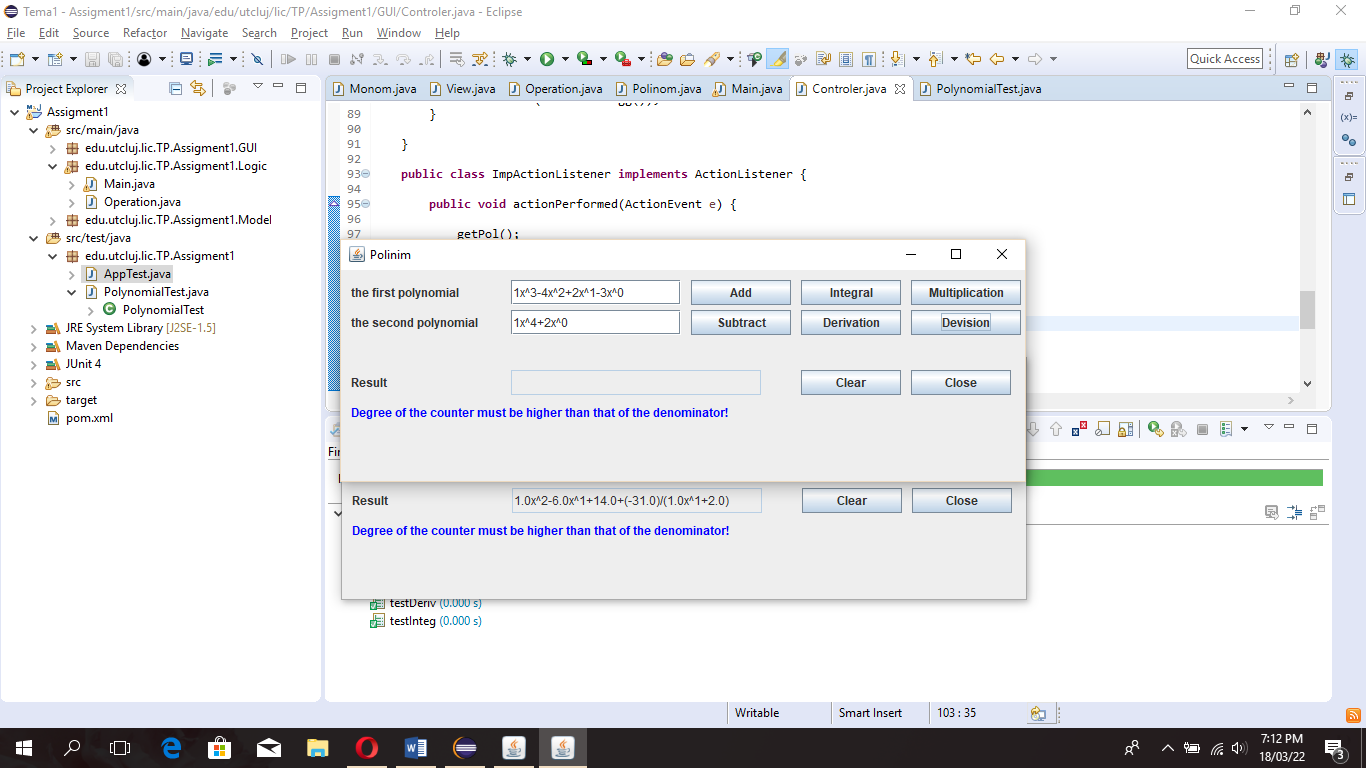
Operatia de integrare:

Operatia de inmultire:

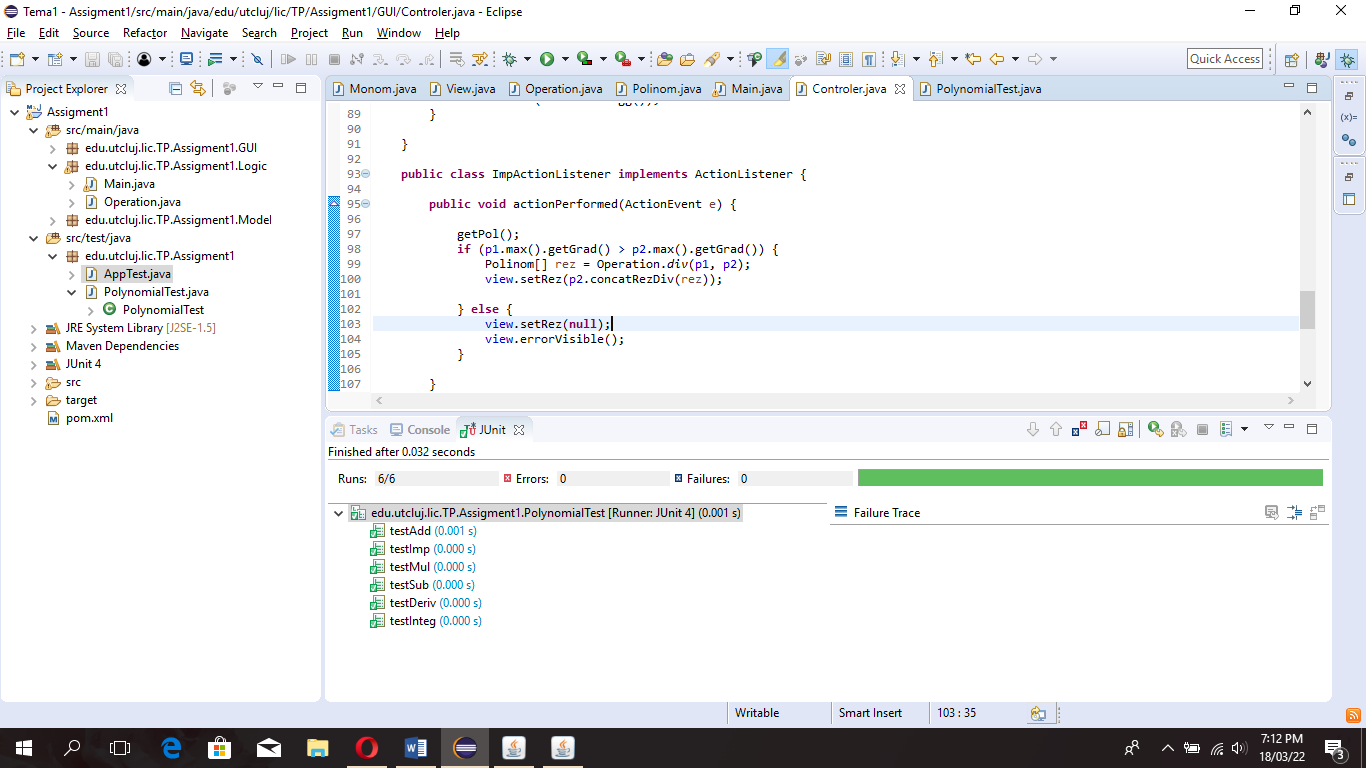


Operatia de impartire:





Junit:



Dupa cum se poate observa toate operatile se reazlizeza cu success, rezultatel fiind ordonat si lizibile. Mai mult se poate observa in cazul in care gardul numaratorului este mai mare decat gardul numitorului operatia de impartire nu se realizeaza, fiind afisat un mesaj de eroare. De asemenea se poate observa ca apliatia a trecut toate testele.

1. Conculzii

In concluzie, aplicatia este una de complexitate redusa, ce poate fi utilizata de orice user cu cunostinte minime despre polinoame ( formatul lor, operatii posibile pe acestea, etc.), fiind extrem de utila pentru verificarea de rezultate.

Realizarea temei a reamintit cu success cunostintele de baza despre limbajul Java, despre clase si obiecte si relatiile dintre ele, precum si despre constructia unei interfete grafice folosind obiecte din packge-ul javax.swing.

Pentru o dezvoltare ulterioara ar putea fi implementate medode pentru determinarea radacinilor unui polinom, pentru tratarea grafiului, o metoda de calculare a transormatei Laplace.

1. Bibliografie

1.https://en.wikipedia.org/wiki/Polynomial\_long\_division

2. <https://www.code-it.ro/calculator-de-polinoame-in-java/#_Toc445825162>

3. <https://moodle.cs.utcluj.ro/course/view.php?id=83>