TEHNICI DE PROGRAMARE

Procesarea polinoamelor

Gozar-Manu Ariana

An II, CTI - romana

Grupa 30225

**CERINTA:**

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

1. Obiectivul temei

Obiectivul temei consta in implementarea unei aplicatii software, intr-un limbaj

de programare orientat pe obiect si anume, Java. Aceasta aplicatie trebuie sa poata efectua operatii pe polinoame de o singura variabila (Exemplu: adunare, scadere, inmultire, derivare, valoarea intr-un punct dat, etc.)

1. Analiza problemei, modelare, scenariu, cazuri de utilizare

Implementarea polinoamelor poate fi o problema foarte complexa, deoarece reprezinta un capitol complex si important al matematicii. Dezvoltarea unei aplicatii pe baza acestei teme trebuie abordata sistematic, deoarece ni se ofera o gama larga de operatii care pot fi implementate, unele de un grad mai mare de dificultate, ceea ce ne face nevoiti sa luam decizii optime in abordarea lor, gasind cea mai facila metoda .

Pentru inceput, ar trebui sa colectam cat mai multe materiale din domeniul pe care

trebuie sa-l studiem pentru a putea observa atat problemele care pot interveni, cat si datele ce vor duce la crearea unei idei despre structura bloc a aplicatiei.

Astfel, vom observa ca un polinom poate fi privit ca un „obiect” caracterizat de

numarul sau de coeficienti sau gradul sau si coeficientii propriu-zisi. Operatiile pot fi vazute ca metode aplicate obiectelor. De aici, observam ca aceasta aplicatie poate fi realizata prin prisma programarii orientate pe obiect, utilizand limbajul de programare sugerat – JAVA.

Putem sa incepem sa definim partile componente ale aplicatiei abia dupa ce structura este definita in mare parte. De asemenea, se pot observa proprietatile, atributele si functionalitatile acestora.

In proiectul meu, am ales sa implementez cateva operatii de baza: suma a doua polinoame, diferenta lor, produsul, calculul valorii intr-un punct dat si aflarea radacinii. Algoritmul implementat contine si metoda de derivare, care este necesara metodei de gasire a radacinii dupa principiul lui Newton, insa aceasta nu este reprezentata in interfata grafica. Pentru calculul radacinii am ales sa suprascriu metoda de gasire a ei, deoarece pentru polinoame pana la gradul patru exista formule concrete. In schimb, nu am facut clase copil pentru polinomul de grad trei si patru pentru ca formulele acestora sunt foarte complicate si nu consider ca mi-ar imbunatati tehnica de programare. Acest motiv este valabil si pentru faptul ca nu am reprezentat grafic metoda de derivare si nu am abordat metoda de integrare a unui polinom si cea de divizare a doua polinoame. Scopul principal al acestei teme nefiind testarea capacitatilor noastre matematice.

Utilizatorul are acces la introducerea coeficientilor unui polinom, generarea lui sub forma „Xn + Xn-1 + ...”, calcularea polinomului intr-o valoare data, suma a doua polinoame, diferenta, produsul lor si aflarea radacinii pentru polinomul ales. Sistemul preia datele introduse de utilizator, care sunt reprezentate de coeficienti si creaza polinomul, facand si celelalte operatii cand sunt accesate butoanele ce le declanseaza. Utilizatorul mai poate introduce valoarea in care doreste sa fie calculate polinoamele si are optiunea de a alege polinomul pe care doreste sa se aplice operatiile dorite. In cazul in care utilizatorul nu introduce date corecte se vor genera mesaje de eroare cu indicatii pentru rezolvarea lor.

1. Proiectare

Am construit o clasa Interfata, care contine doua metode, dintre care una este

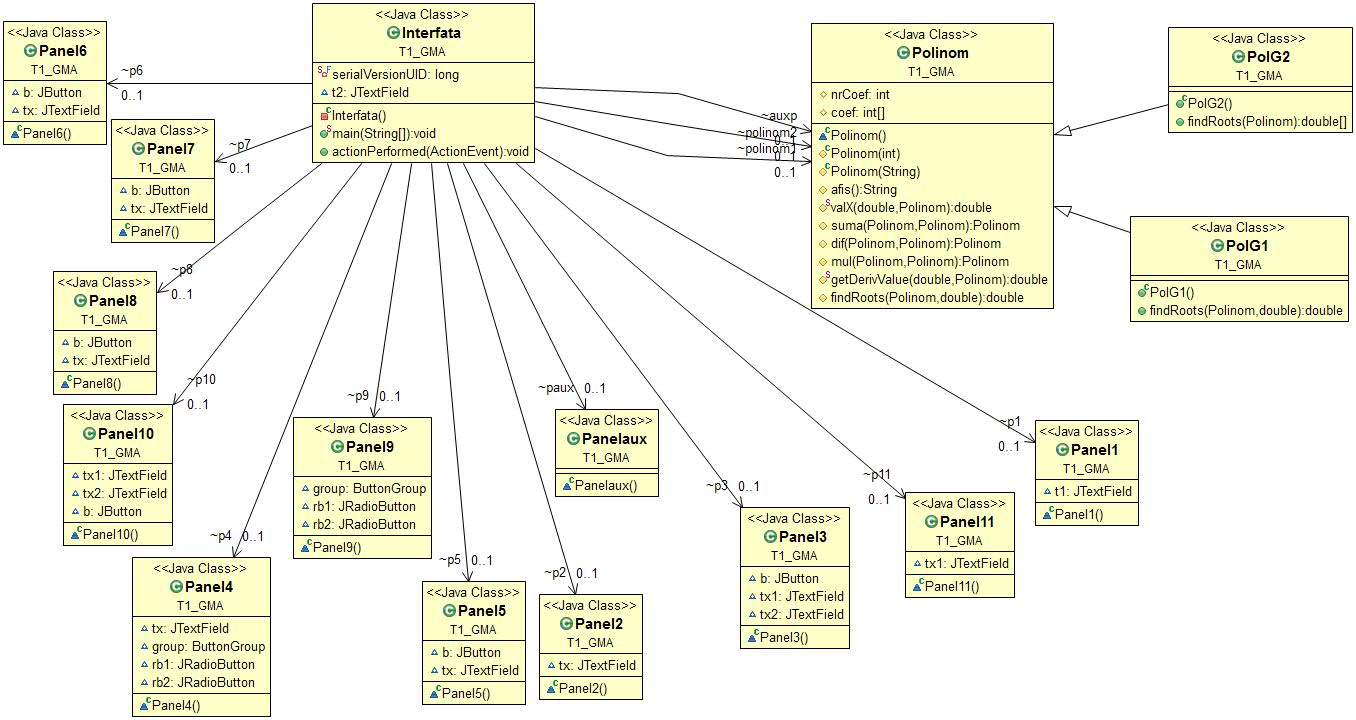
Public void main(String args[]), in care se creaza si seteaza interfata ca fiind vizibila. Cealalta metoda este Public void actionPerformed(ActionEvent e), in care se trateaza fiecare actiunea a butoanelor. Clasa Interfata colaboreaza cu toate clasele Panel si cu cele in care sunt tratate cazurile pentru polinoamele cu formule consacrate pentru aflarea radacinii. Rolul acestei clase este de a ajuta la preluarea polinoamelor introduse de catre utilizator si de a prelucra comenzile corespunzatoare operatiilor ce se doresc a fi efectuate.

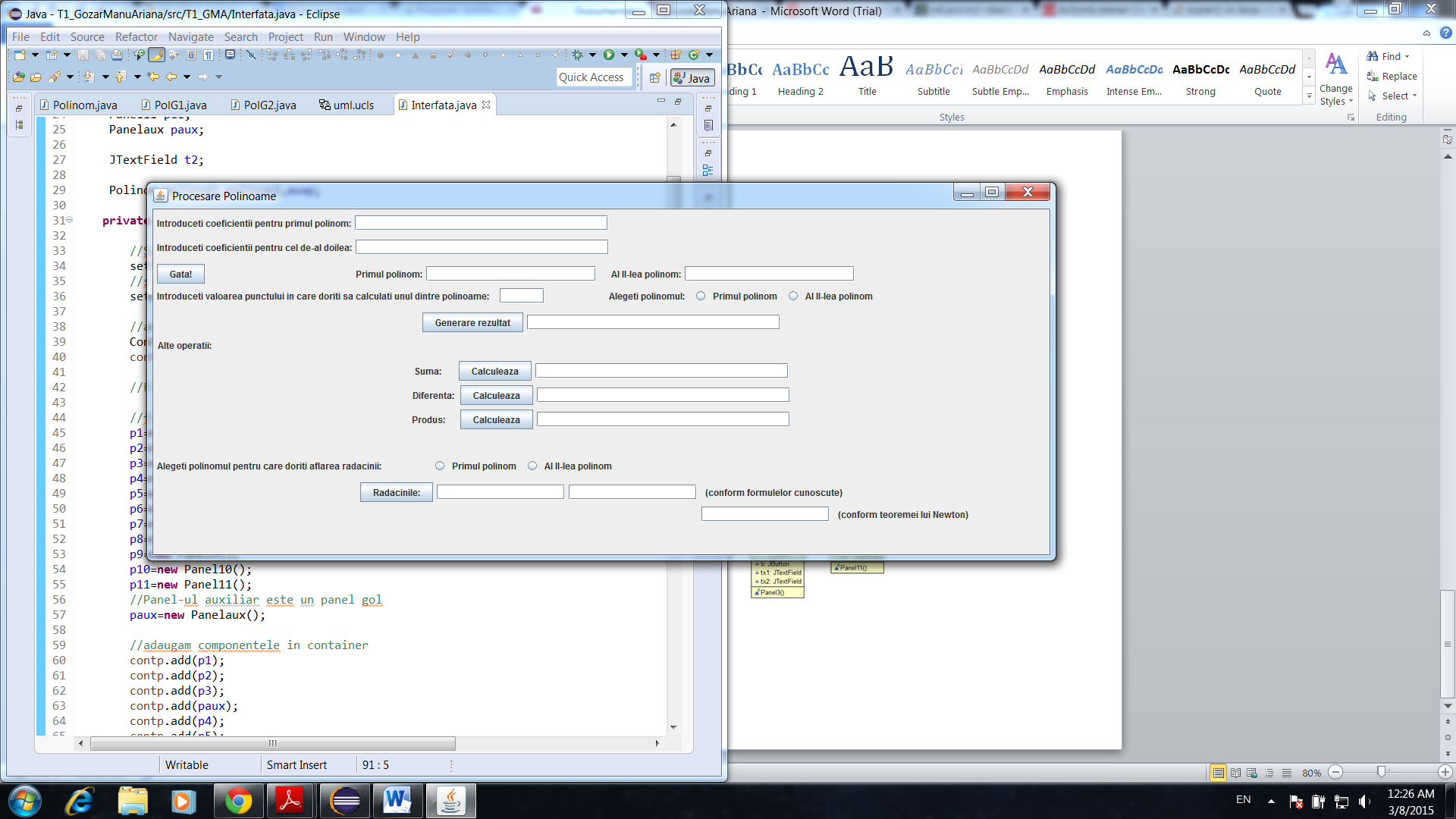
Legate de clasa Interfata, sunt, dupa cum am mentionat si mai sus, clasele Panelx, cu x de la 1 la 11 si unul auxiliar. Acestea sunt aplicate pe fiecare linie a Container-ului, pentru a crea un nou Layout pentru asezarea cat mai usoara a componentelor, reprezentate de butoane, sectiuni in care poti sa scrii si texte.

Clasa Polinom contine trei constructuri. Primul constructor este lipsit de cod deoarece este util pentru apelarea clasei parinte de catre copii prin apelul super(); fara argument. Al doilea constructor, protected Polinom (int nrCf), este utilizat pentru crearea unui polinom cu un numar de coeficienti dati, avand valoarea 0. In final, cel de-al treilea constructor, protected Polinom (String coefS), creaza un polinom cu coeficienti extrasi dintr-un sir de caractere. Acesti ultimi doi constructori colaboreaza cu clasa Interfata. Protected String afis() afiseaza un polinom, fie datde utilizator, fie unul obinut in urma realizarii unei operati, de variabila X. Protected static double valX(double x, Polinom p) calculeaza valoarea polinomului intr-un punct x real. Protected Polinom dif(Polinom p1, Polinom p2) calculeaza diferenta a doua polinoame, protected Polinom dif(Polinom p1, Polinom p2) calculeaza suma a doua polinoame, iar Protected Polinom mul(Polinom p1,Polinom p2) realizeaza inmultirea lor. Protected static double getDerivValue(double x, Polinom p) genereaza derivata unui polinom, intr-un punct dat. Aceasta metoda este necesara metodei Protected double findRoots(Polinom p, double startValue), care calculeaza radacina unui polinom dupa ratinamentul lui Newton.

Clasa Polinom este mostenita de alte doua clase PolG1 si PolG2, in care este suprascrisa sau supraincarcata metoda de gasire a radacinii, findRoots, avand metode de gasire a radacinii specifice pentru polinomul de grad unu si doi. De asemenea, exista aceste formule specifice si pentru polinoamele de grad trei si patru, insa motivul pentru care nu le-am implementat este mentionat mai sus. Prin crearea acestor doua clase se doreste exemplificarea conceptului foarte important din programarea orientata pe obiect – mostenirea.

**DIAGRAMA UML:**





Dupa cum se observa, aplicatia permite citirea de la tastatura a doua polinoame de orice grad, cu coeficienti intregi si efectuarea catorva operatii matematice pe baza acestora. Pentru a folosi polinomul in operatii utilizatorul trebuie sa introduca coeficinetii cu spatiu intre ei, fara a lasa spatiu dupa ultimul coeficient. Primul coeficient introdus va fi coeficientul punctului cu gradul cel mare. La final, se apasa butonul „Gata!” pentru ca aplicatia sa preia coeficientii si sa genereze polinomul dorit, totodata afisandu-l sub forma mai reprezentativa. Daca se doreste calcularea valorii intr-un punct, textele din aplicatie sunt destul de sugestive pentru realizarea ei. Se introduce o valoare reala in JTextField-ul corespunzator si se alege polinomul pentru care se doreste calcularea rezultatului, fapt realizat cu ajutorul JRadioButton-urilor, care au proprietatea de grupare, adica nu pot fi activate ambele in acelasi timp. Pentru generarea rezultatului se apasa butonul „Generare rezultat”, iar o valoare reala va aparea in JTextField-ul din dreptul sau.

Daca se doreste suma, diferenta sau inmultirea polinoamelor nu trebuie decat sa se apese pe butonul din dreptul fiecarui text indicator, iar rezultatul va fi, din nou, afisat intr-un JTextField.

In spatele interfetei se afla algoritmi destul de simpli, insa complecsi din punct de vedere al tratarii erorilor ce pot sa apara. Cea mai complexa metoda din Interfata este cea de actionPerformed in care se ia, pe rand, fiecare buton, si se scrie cod pentru ceea ce se va intampla dupa accesarea lui. Interfata este realizata cu un Container, nu cu un Frame. Container-ul apartine unei clase din java.awt.\*. Acest obiect este o fereastra abstracta care contine componente AWT sau altele. Pentru a pozitiona cat mai bine componentele, am folosit tot un obiect din clasa apartinatoare lui java.awt.\*, GridLayout, care imparte Container-ul ca o matrice, iar pe fiecare linie sunt plasate alte Layout-uri, fiind de tipul FlowLayout, din aceeasi clasa.

In metoda main a clasei Interfata se instantiaza un obiect de tipul Interfata si se seteaza vizibila reprezentarea grafica a aplicatiei. In metoda actionPerformed sunt accesate JButton-uri, JTextField-uri si JLabel-uri si clasele Panel. Aceste componente fac parte dintr-o clasa ce apartine lui javax.swing.\*. JButton creeaza un buton si are un constructor cu un string ca parametru. String-ul devine textul afişat pe buton. Când un utilizator face click pe un buton, butonul genereaza un eveniment de tip ActionEvent. Acest eveniment este trimis la un listener care este inregistrat ca şi un ActionListener pe acel buton. Când actionPerformed(evt) este apelat de către buton, parametrul, evt, conţine informaţii despre eveniment. JLabel este cea mai simplă componentă. Un obiect de tip JLabel afişează o linie de text. Textul nu poate fi editat de utilizator. Clasa JTextField este o componenta care contine text ce poate fi editat de utilizator. Acesta are o singura linie de text. Un lucru avantajos este faptul ca aceasta poate fi setata Read-only, astfel utilizatorul poate doar sa citeasca textul nu sa il si modifice, creand posibile erori. Clasa JTextField are un constructor public JTextField(int columns), unde columns este un ȋntreg care specifică numărul de caractere care trebuie să fie vizibile. JRadioButton sunt asemanatoare cu cele de tip JCheckBox. Acestea au doua stari: selectate si neselectate. Uilizatorul schimba starea unui radio button facand click pe el. Starea acestuia este reprezentata de o valoare booleana (true daca este selectat sau false in caz contrar).

Algoritmi:

Pentru a sparge sirurile de caractere extrase din JTextField-uri, un lucru necesar, pe care se bazeaza toata aplicatia, se cauta spatiul din sir si se substrage un subsir de la spatiul anterior pana la cel nou gasit. Subsirul rezult se converteste intr-un intreg. In tot acest timp se contorizeaza cate substring-uri s-au format, astfel avem numarul de coeficineti. In continuare se intorc coeficinetii in vector, adica cel de pe pozitia cel mai mare se pune pe pozitia cea mai mica. La fel si ceilalti coeficienti interschimba cu cei aflati la indecsi opusi, pentru a reprezenta gerafic ca primul coeficient este valoarea din fata valorii X cu grad cel mai mare.

In vederea afisarii polinoamelor sub forma lor stiuta, continand valoarea necunoscuta X, se iau cazuri in metoda afis() si se concateneaza sirurile. De exemplu, daca urmatorul coeficient este de valoare pozitiva, se concateneaza valoarea rezultatului curenta si cu semnul „+”, altfel nu, pentru a se scrie automat semnul negativ de la coeficient. Daca este ultimull coeficient, atunci nu se mai concateneaza nimic. Variabila rezultat este un sir de caracter fara niciun element la inceput.

Pentru calculul valorii in punctul x , avem o variabila rezultat de tip double initializata la inceput cu 0. Intr-o structura repetitiva care parcurge intreg vectorul coeficientilor I se atribuie variabilei rezultat valoarea lui x la puterea numarului coeficientului corespunzator, inmultit cu coeficientul propriu-zis. Metoda primeste ca parametrii un polinom si o valoare reala si returneaza rezultatul, fiind tot o valoare reala.

In metoda calcului sumei a doua polinoame, sunt tratate doua cazuri. Principiul general al algoritmului abordatat este adunarea coeficient cu coeficient, incepand de la cel mai mic, spre cel mai mare, al celor doua polinoame. Deoarece cei doi vectori pot fi de dimensiuni diferite se verifica care este cel mai lung, astfel, intr-o structura repetitiva se incrementeaza un contor pana la lungimea celui mai mic polinom, adunandu-se coeficientii corespunzatori, doi cate doi, iar dupa aceea in polinomul rezultat, care este un obiect de tip Polinom, se introduc coeficientii ramasi de la polinomul cu lungimea cea mai mare. Bineinteles, polinomul rezultat va fi initializat la inceput cu ajutorul constructorului Polinom(int nrCof), avand ca parametru intreg transmis numarul coeficientilor polinomului celui mai lung. Coeficientii acestui vector vor fi initalizati cu 0, unde se va putea plasa noua valoare primita dupa realizarea sumei dintre coeficientii polinoamelor transmise ca parametrii.

La fel ca si in metoda calcularii sumei se abordeaza si algoritmul aflat in metoda calcularii diferentei. Si aceasta metoda primeste ca si parametrii doua polinoame initalizate cu coeficienti cu valoarea 0, avand ca valoare intreaga transmisa constructorului numraul coeficientilor polinomului cu grad cel mai mare. Din nou intr-o structura repetitiva de tip for incrementata pana la lungimea polinomului cel mai scurt se scad coeficientii doi cate doi iar pentru a completa variabila rezultat, se introduc si coeficientii ramasi de la polinomul cu gradul cel mai mare.

La inmultire, daca numarul de coeficienti este 1 si acel coeficient dat este 0, rezultatul primeste direct instantierea Polinom(1) pentru a putea fi reprezentata valoarea 0 in reprezentarea grafica a aplicatiei, altel aceasta valoare nula nu apare. Daca polinoamele au un grad pozitiv, rezultal, de tip Polinom, este initializat ca avand numarul de coficientul suma dintre coficientii celor doua polinoame transmise ca parametrii. Cu ajutorul a doua structuri repetitive de tip for imbricate se calculeaza inmultirea pe principiul calcului produselor din fiecare doi coeficienti, iar apoi suma acelora a cui grad este egal.

Metoda „auxiliara”, getDerivValue, este folosita pentru a scrie cat mai optim algoritmul de gasire a radacinii unui polinom de o variabila cu metoda lui Newton. In getDerivValue se calculeaza, de fapt, valoarea derivatei unui polinom intr-o valoare data ca parametru. Pentru derivare se inmulteste exponentul cu coeficientul corespunzator inmultite, la randul lor, cu valoarea x la valoarea exponentului decrementata.

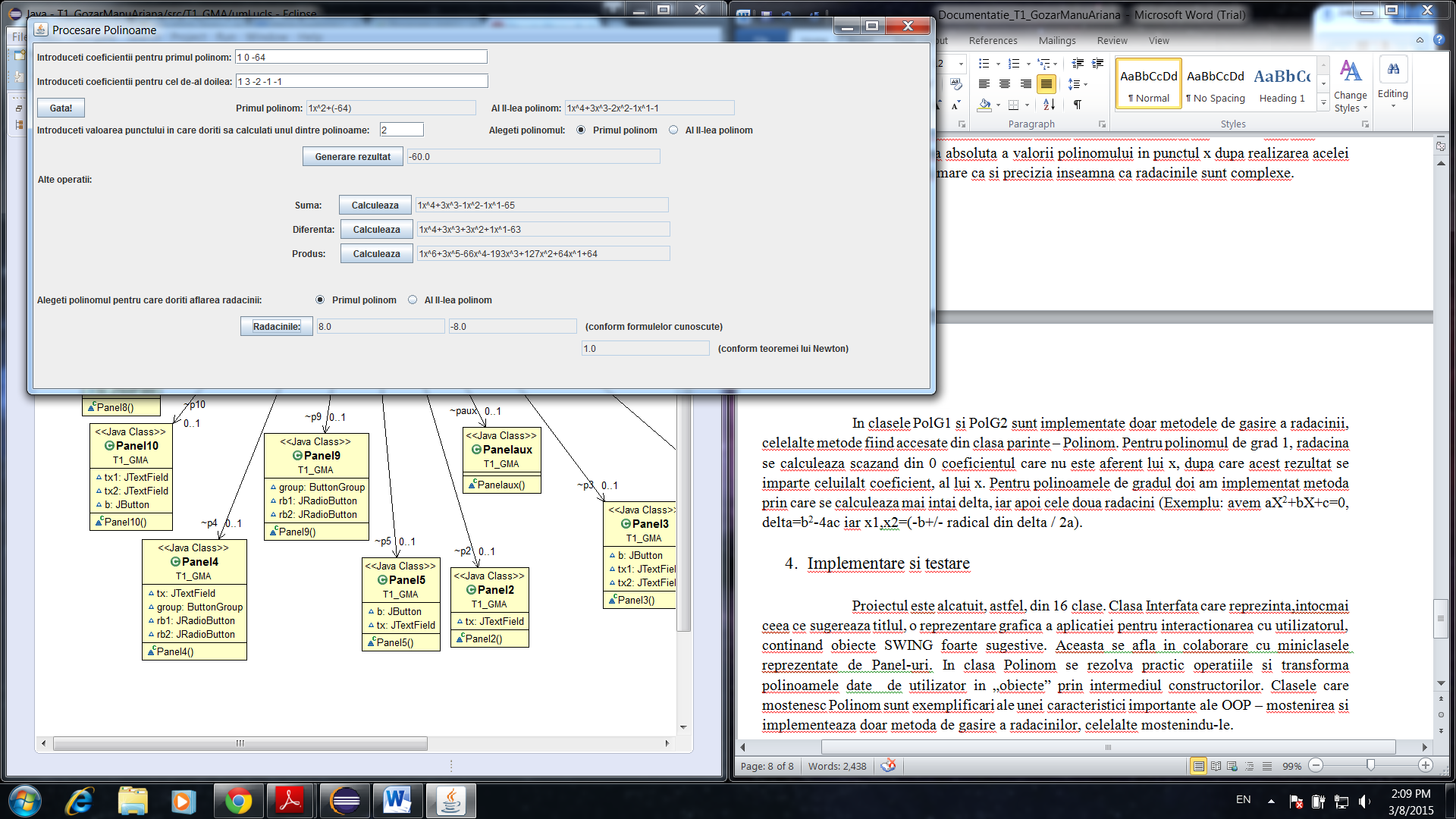
Metoda findRoots gaseste o radacina a unui polinom in functie de o valoare de start. Daca dai o valoare de start pozitiva este cel mai posibil sa obtii o radacina pozitiva, iar daca e negativa o radacina negativa daca exista. Metoda lui Newton implementata cauta cea mai apropiata radacina de valoarea de inceput, startValue. Acest algoritm necesita in primul rand o precizie, adica cate zecimale sa fie afisate dupa virgula. Precizia este data de forma 1e-x. In implementarea mea am folosit doar precizia 1e-1, adica o zecimala dupa virgula. Intr-o structura repetitiva, de 500 de iteratii se calculeaza valoarea x, care la inceput ia valoarea de start, din care se scade valoarea polinomului in punctul x impartita la valoarea derivatei in punctulul x. Acest proces se repeta pana cand valoarea absoluta dintre elementul curent si cel anterior este mai mica ca si precizia. Daca valoarea absoluta a valorii polinomului in punctul x dupa realizarea acelei structuri repetitive este mai mare ca si precizia inseamna ca radacinile sunt complexe.

In clasele PolG1 si PolG2 sunt implementate doar metodele de gasire a radacinii, celelalte metode fiind accesate din clasa parinte – Polinom. Pentru polinomul de grad 1, radacina se calculeaza scazand din 0 coeficientul care nu este aferent lui x, dupa care acest rezultat se imparte celuilalt coeficient, al lui x. Pentru polinoamele de gradul doi am implementat metoda prin care se calculeaza mai intai delta, iar apoi cele doua radacini (Exemplu: avem aX2+bX+c=0, delta=b2-4ac iar x1,x2=(-b+/- radical din delta / 2a).

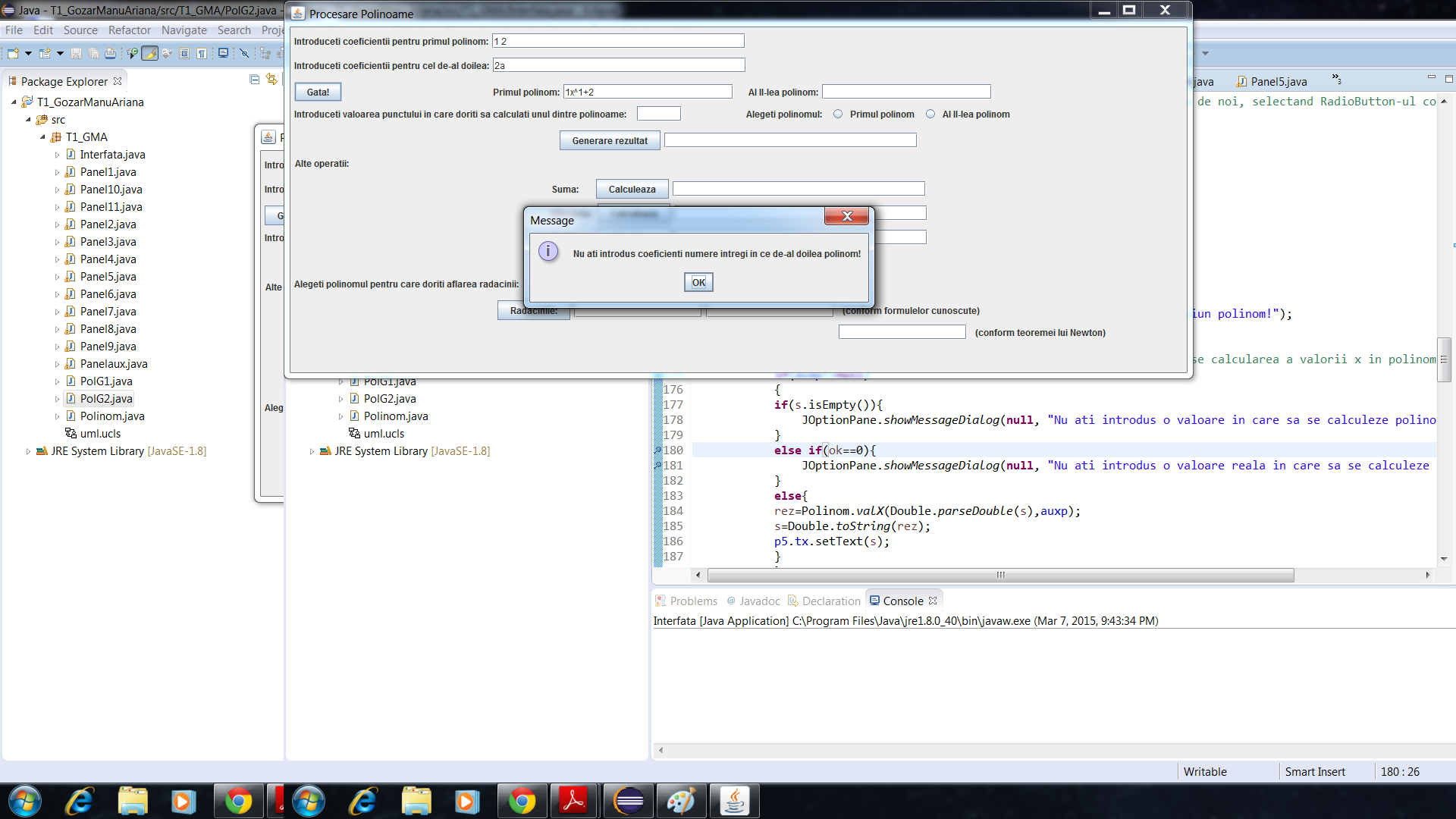
1. Implementare si testare

Proiectul este alcatuit, astfel, din 16 clase. Clasa Interfata care reprezinta,intocmai

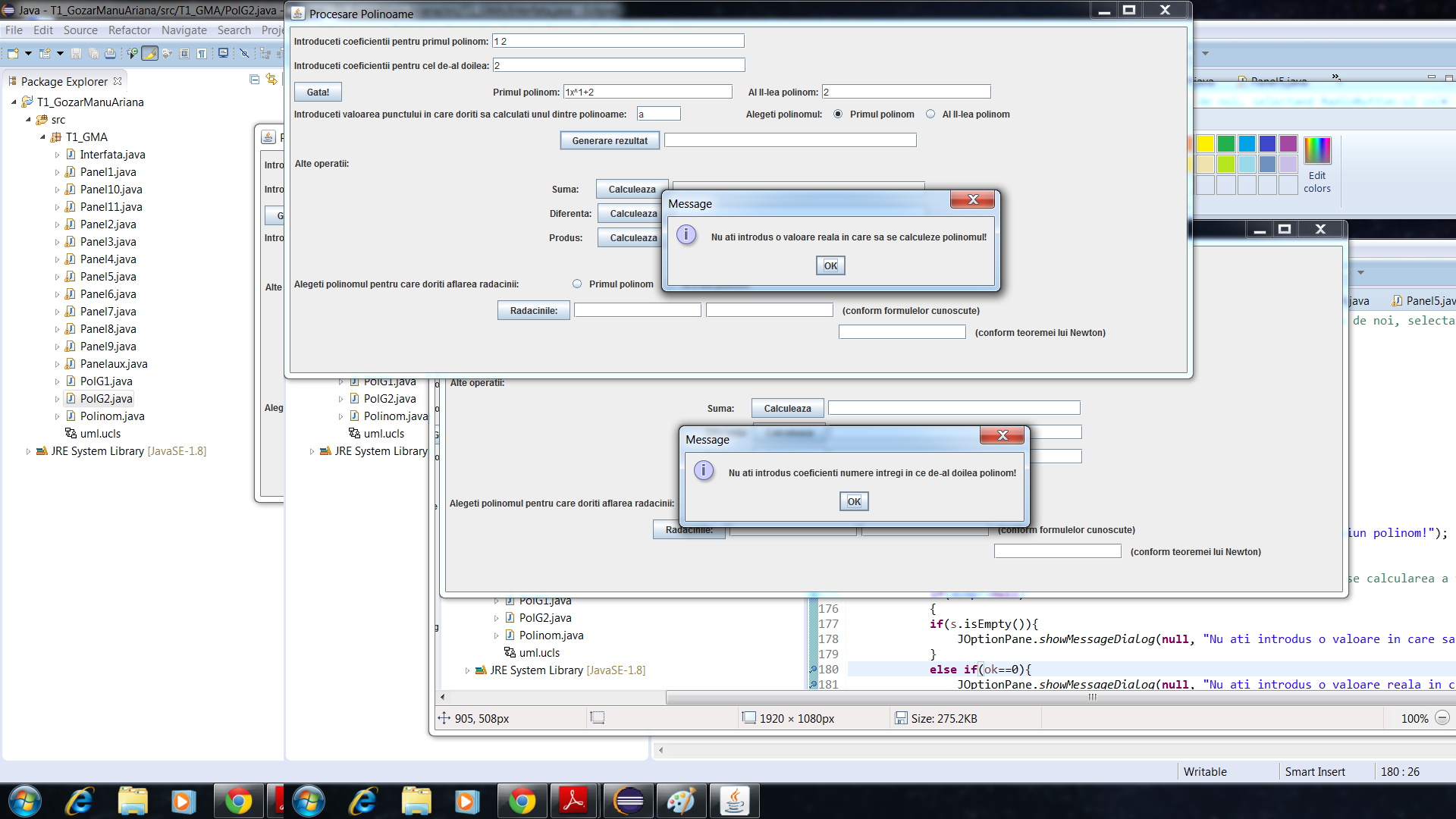
ceea ce sugereaza titlul, o reprezentare grafica a aplicatiei pentru interactionarea cu utilizatorul, continand obiecte SWING foarte sugestive. Aceasta se afla in colaborare cu miniclasele reprezentate de Panel-uri. In clasa Polinom se rezolva practic operatiile si transforma polinoamele date de utilizator in „obiecte” prin intermediul constructorilor. Clasele care mostenesc Polinom sunt exemplificari ale unei caracteristici importante ale OOP – mostenirea si implementeaza doar metoda de gasire a radacinilor, celelalte mostenindu-le.



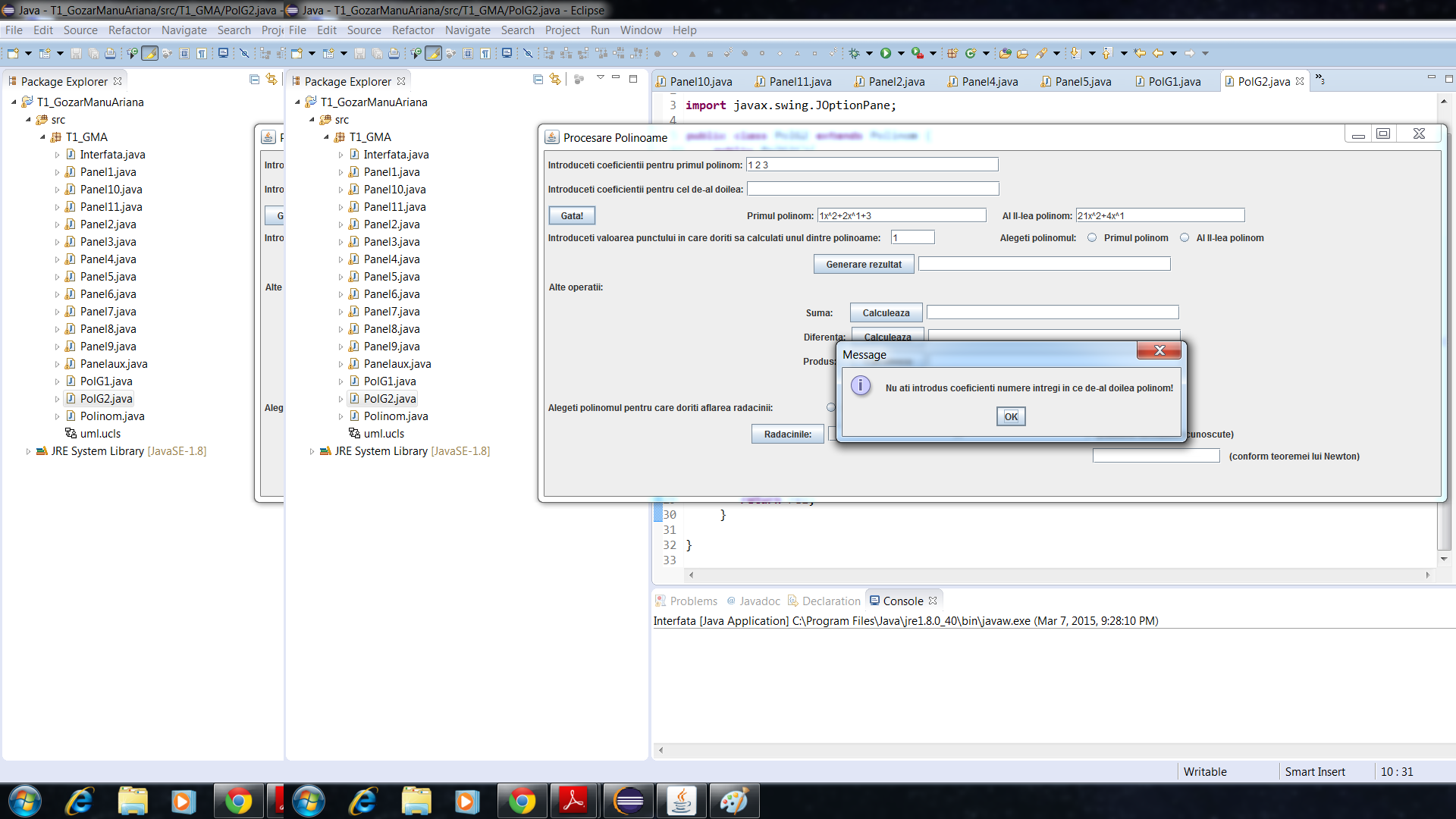
Pentru testare am realizat atat un model complet si bun, cat si unele care genereaza eroare. Erorile sunt, in mare parte rezolvate prin afisarea unei casute de avertizare. Aceste casete de erori sunt realizare cu un obiect dintr-o clasa din javax.swing.\* si anume JOptionPane.



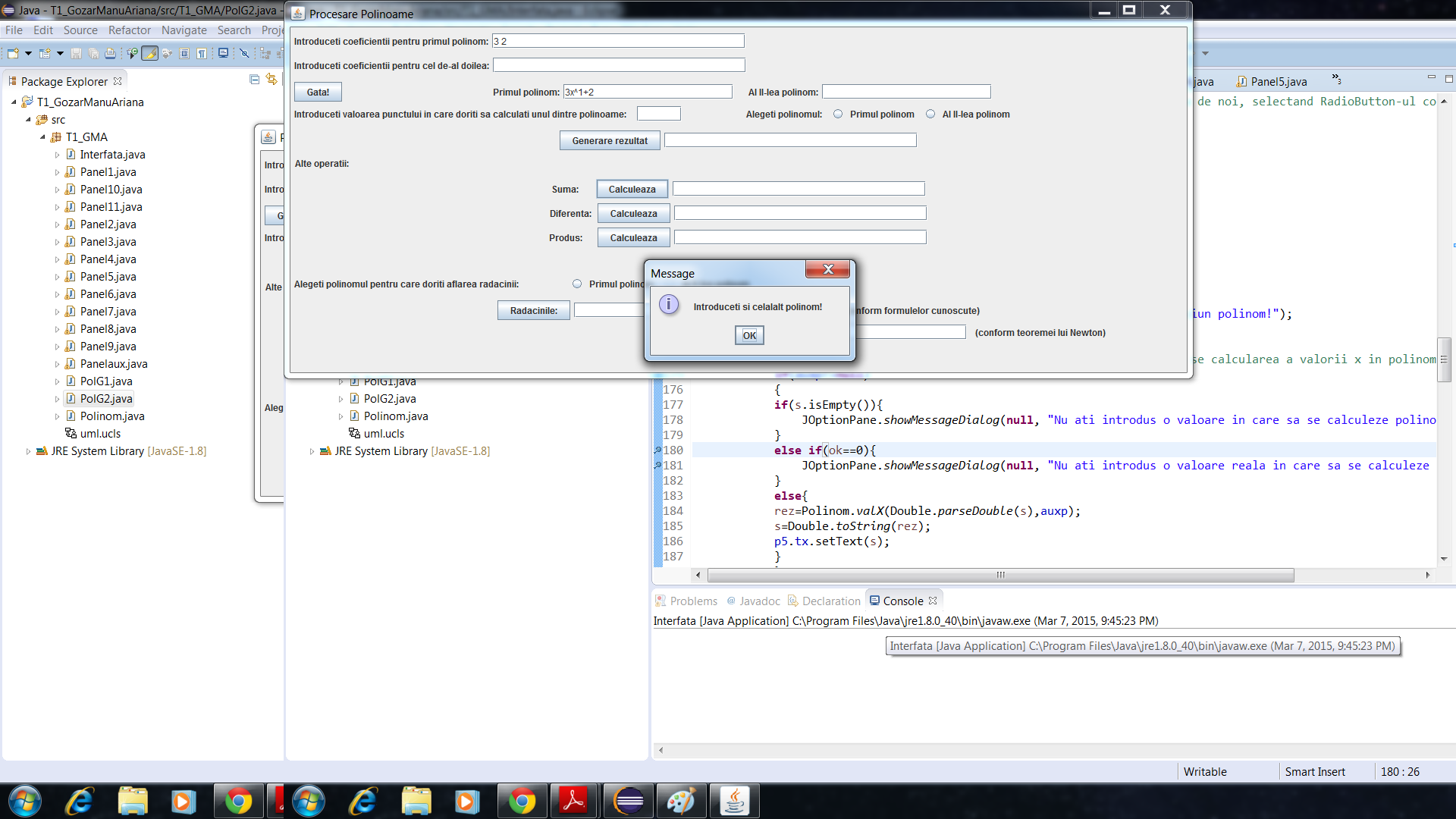
Daca in caseta de text pentru introducere a coeficientilor polinoamelor nu se introduce un caracter de tip spatiu, de tip „-” sau cifra, se va afisa un mesaj de avertizare ca nu ati introdus coeficienti numere intregi in primul sau cel de-al doilea polinom. Daca se acceseaza ok se poate relua programul si reintroduce cofiecienti adecvati.



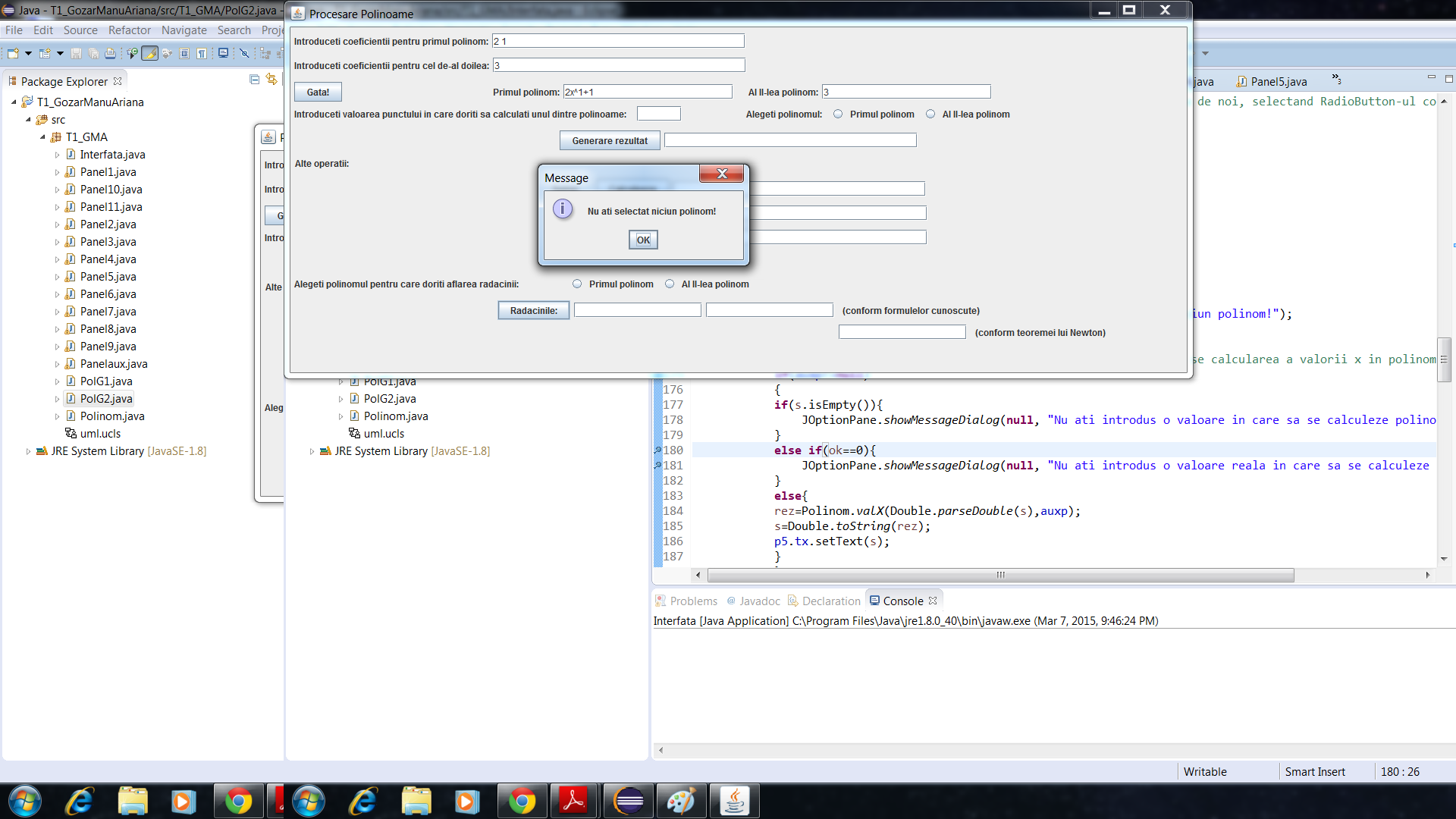
Daca se doreste calcularea valorii unui polinom intr-un punct dat, valoarea introdusa poate sa fie una reala, insa nu se pot introduce alte caractere inafara de cele ce reprezinta cifre. Astfel, la introducerea unui caracter nepermis apare o casuta de dialog cu mesajul „Nu ati introdus o valoare reala in care sa se calculeze polinomul!”. Dupa inchiderea casetei, se poate reintroduce o valoare dorita si aplicatia va continua de unde a ramas, avand ca polinoame utilizate cele generate anterior.



Daca apasati butonul „Gata!”, iar intr-o caseta text pentru introducerea coeficientilor polinoamelor nu este introdus nimic, se va genera o caseta de avertizare cu mesajul ca nu ati introdus coeficienti in polinomul respectiv. Bineinteles ca aplicatia va putea functiona si cu un polinom generat, insa se vor putea accesa doar operatiile de calcul a valorii polinomului intr-un punct x dat si gasirea radacinii. In Print Screen-ul facut, au fost generate inainte doua polinoame, iar dupa aceea am incercat sa generez doar unul, astfel pentru cel de-al doilea mi-a aparut caseta amintita, ramanand in polinomul al doilea valoarea polinomului generat anterior.



Daca se doreste calcularea unei sume, diferente sau a unui produs dintre doua polinoame, trebuie introduse ambele polinoame. Astfel, la accesarea oricarei operatii care necesita doua polinoame se va afisa un mesaj de eroare indicandu-i utilizatorului sa introduca si celalalt polinom.



Atat pentru calcularea valorii date x intr-un polinom, cat si pentru calcularea radacinii, trebuie ales unul dintre cele doua JRadioButton-uri aflat in dreptul fiecarei operatii. Pentru a selecta un JRadioButton trebuie sa apasati click pe cercul din stanga textului ce indica ceea ce veti activa odata cu activarea acelui buton. Pentru ambele operatii trebuie sa fie ales butonul pentru a selecta primul polinom sau al doilea. Daca se va accesa buton de „Generare rezultat” sau „Radacinile:” , un nou mesaj de eroare va aparea pe ecran amintndu-va ca ati uitat sa alegeti ce polinom vreti sa utilizati.

O caracteristica importanta a aplicatiei este faptul ca, daca dupa introducerea coeficientilor nu se va accesa butonul „Gata!”, aplicatia nu va rula niciuna dintre operatii din cauza ca polinoamele nu s-au creat. Instantierea se face doar la accesarea butoanelor, in general.

1. Concluzii, ce s-a invatat din tema, dezvoltari ulterioare

Tema a reprezentat o buna oportunitate de a-mi imbunatati cunostintele de JAVA

si de a incepe sa le ridic la un nivel mai inalt. De asemenea, m-a ajutat sa sa fac o recapitulare a cunostintelor acumulate de curand si sa invat lucuri noi, cum ar fi dezvoltarea unei interfete grafice lucrand doar din cod, fara generare automata. Crearea aplicatiei m-a ajutat sa inteleg, de fapt, ce se intampla in spatele unei interfete grafice si cum se utilizeaza SWING-ul.

Din punct de vedere matematic sper ca am reusit sa acopar esentialul, desi am dezvoltat doar o mica parte din acest subiect complex matematic ce tine de polinoame.

Pe viitor se pot implementa mai multe operatii. Se pot implementa si clasele pentru ecuatia de gradul trei si patru cu formulele specifice. De asemenea, pe interfatata grafica se pot reprezenta atat radacinile calculate cu formulele consacrate, car si cu metoda lui Newton, pentru polinoamele ale caror grad permit acest lucru. Se mai poate dezvolta aplicatia si din punct de vedere al erorilor care pot aparea si a mesajelor transmise utilizatorului. O reprezentare a unui grafic ar fi , de asemenea , foarte utila si „user friendly”. Cea mai de impact dezvoltare ar fi reprezentarea polinomului cu afisarea bazei si puterea acesteia.

1. Bibliografie:

* Tutoriale <http://www.youtube.com> – pentru interfata grafica
* <http://docs.oracle.com/javase/tutorial/> - pentru intelegerea termenilor