***CALCULATOR PENTRU POLINOAME***

***Materie: Tehinici de Programare***

***Elev: Crecan Dragos***

***Grupa: 30223***

***Specializare: CTI RO, an 2***

*Cuprins*

[*Obiectivul temei* 2](#_Toc35361509)

[*Analiza problemei* 2](#_Toc35361510)

[*Proiectare* 4](#_Toc35361511)

[*Implementare* 5](#_Toc35361512)

[*Rezultate* 8](#_Toc35361513)

[Concluzii 11](#_Toc35361514)

[*Bibliografie* 11](#_Toc35361515)

# *Obiectivul temei*

Principalul obiectiv al temei este implementarea unui calculator de polinoame, functional.

Pentru a putea indeplini acest obiectiv a fost necersa imparitirea lui in mai multe obiective secundare si anume:

* Transpunerea polinoamelor si a monoamelor din lumea reala sub forma unor clase.
* Crearea unor expresii regulate si a unor sabloane pentru generearea polinoamelor dintr-un string primit ca si input
* Implementarea operatiei de adunare
* Implementarea operatiei de scadere
* Implementarea operatiei de inmultire
* Implemenatarea operatiei de impartire
* Implementarea operatiei de derivare
* Implementarea operatiei de integrare
* Crearea unei interfete grafice, pentru utilizatorul aplicatiei

# *Analiza problemei*

Pentru a intelege bine problema trebuie sa facem apel la cuunostintele de baza accumulate din matematica, astfel vom defini urmatoarele notiuni:

În [matematică](https://ro.wikipedia.org/wiki/Matematic%C4%83), un **polinom** este o [expresie](https://ro.wikipedia.org/w/index.php?title=Expresie_matematic%C4%83&action=edit&redlink=1) construită dintr-una sau mai multe [variabile](https://ro.wikipedia.org/wiki/Variabil%C4%83) și [constante](https://ro.wikipedia.org/wiki/Constant%C4%83_matematic%C4%83), folosind doar operații de [adunare](https://ro.wikipedia.org/wiki/Adunare), [scădere](https://ro.wikipedia.org/wiki/Sc%C4%83dere), [înmulțire](https://ro.wikipedia.org/wiki/%C3%8Enmul%C8%9Bire_(matematic%C4%83)) și ridicare la putere constantă pozitivă întreagă.

Polinoamele sunt construite din [termeni](https://ro.wikipedia.org/w/index.php?title=Termen_(matematic%C4%83)&action=edit&redlink=1) numiți [monoame](https://ro.wikipedia.org/w/index.php?title=Monom&action=edit&redlink=1), care sunt alcătuite dintr-o constantă (numită [coeficient](https://ro.wikipedia.org/wiki/Coeficient)) înmulțită cu una sau mai multe [variabile](https://ro.wikipedia.org/wiki/Variabil%C4%83). Fiecare variabilă poate avea un [exponent](https://ro.wikipedia.org/w/index.php?title=Exponent&action=edit&redlink=1) constant întreg pozitiv. Exponentul unei variabile dintr-un monom este egal cu [gradul](https://ro.wikipedia.org/w/index.php?title=Gradul_unui_polinom&action=edit&redlink=1) acelei variabile în acel monom.

Un monom fără variabile se numește *monom constant*, sau doar *constantă*. Gradul unui termen constant este 0. Coeficientul unui monom poate fi orice număr, inclusiv fracții, numere iraționale sau negative. Un polinom construit cu o singură variabilă se numește **univariat**.

Toate polinoamele de o variabilă sunt echivalente cu un polinom de forma:

∑akxk

unde k apartine intervalului [0, n] si n este gradul polinomului, iar sirul an reprezinta coeficientii polinomului.

Proprietati ale polinoamelor:

* Suma a doua polioame este tot un polinom
* Diferenta a doua polinomae, este tot o suma, in care unul dintre polinoame isi schimba semnul coeficientilor
* Produsul a doua polinoame este tot un polinom
* Impartirea a doua polinoame va rezulta alte doua polinome numite cat si rest
* Derivarea unui polinom reprezinta tot un polinom
* Integrarea unui polinom reprezinta tot un polinom

***Modelare***

Un prim pas al procesului de modelare il reprezinta privirea polinoamelor ca fiind o lista de monoame, deoarece cu acestea este mult mai usor de operat, fiind caracterizate doar de grad si coefficient.

Astfel, pentru a implementa operatiile dorite la nivel de polinom, acestea ar trenui sa fie functionale la nivel de monom si extinse ulterior.

***Utilizare (use-case)***

Pentru a folosi aplicatia, utilizatorul are la dispozitie interfata grafica a aplicatiei (GUI).

La rularea aplicatiei, utilizatorul va intalni o fereastra cu 6 butoane, fiecare reprezentand una dintre cele 6 operatii carcateristice ale polinoamelor. La apasarea oricarui buton pe ecarn va aparea denumirea operatiei selectate si semnul mathematic asignat acestei operatii.

Alegerea unei operatii care necesita 2 polinoame (adunare, scadere, inmultire, impartire) pe ecran vor aparea 2 campuri in care utilizatorul va introduce cele doua polinoame dupa un anumit tipar, iar cand ultilizatorul va selecta o operatie care necesita doar un polinom, in interfata grafica a ramane un singur camp pentru introducerea datelor. La apasarea butonului “Submit”, daca datele introduse de catre utilizator sunt valide, se va efectua operatia selectata si va aparea un nou camp cu rezultatul operatiei.

Tiparul care trebuie respectat la introducerea datelor este urmatorul:

coeficient\*x^exponent +/- coeficient\*x^exponent …

# *Proiectare*

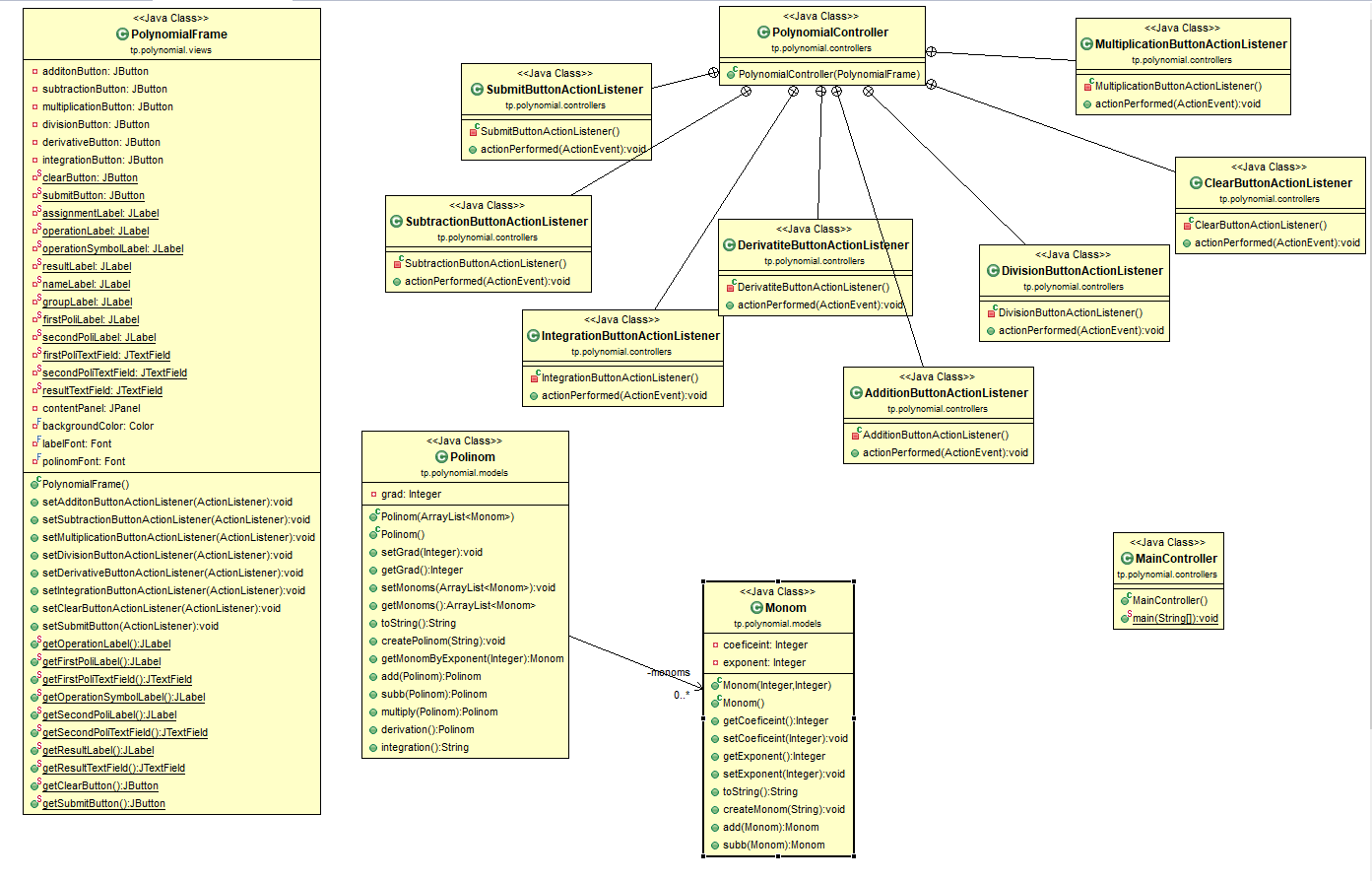
***Decizii de proiectare***

Pentru a proiecta aceasta aplicatie, am ales sa utilizez design pattern-ul Model-View-Controller in care proiectul este imparitit in 3 pachete mari

* Models: aici vor intra clasele care modeleaza monoamele si polinoamele
* Views: aici vom avea doar clase care tin de interfata grafica, se creeaza un frame al aplicatiei si se alege pozitia ficarui obiect in acest frame
* Controllers: aici vor fi implenetate controller-ele aplicatiei, atat cele care decid ceea ce se va intampla cand utilizatorul interactioneaza cu GUI-ul aplicatiei cat si controller-ul principal al aplicatiei, cel care deschide frame-ul aplicatiei.

Pe langa folosirea acestui design-pattern, am decis sa implementez si clase de test, pentru a testa functionalotatea metodelor aplicate pe monoame si extinderea lor ulterioara asupra polinoamelor. Pentru testarea metodelor se vor folosi testele unitare, realizate cu JUnit.

***Diagrama UML***

******

# *Implementare*

***Pachetul Models***

***Clasa Monom***

Clasa Monom are ca si variabile de instanta coeficientul monomului (Integer coeficient) si exponentul acestuia (Integer exponent). Clasa reprezinta abstractizarea in cod a monoamelor din lumea reala.

In aceasta clasa se regasesc doi constructori, cu si fara parametrii, getters si stters pentru fiecare variabila de instanta si metodele toString, createMonom, add si subb pe care le voi detalia mai jos:

***public String toString()***

* Suprascrierea metodei toString din clasa Object
* Am folosit aceasta metoda pentru a fi mult mai usor sa afisam monomul in forma in care dorim.

***public void createMonom(String string)***

* Aceasta metoda este folosita pentru a prelucra input-ul primit de la user, astfel incat sa corespunda cerintelor clasei Monom

***public Monom add(Monom monom)***

* Aceasta metoda va returna un monom ce reprezinta suma a doua monaome (cel care apleaza metoda si cel care este dat ca si parametru)
* Se verifica daca cele monoame au exponenti egali si in caz afirmativ se creeaza un nou monom avand exponentul celor doua monaome si coeficientul sgal cu suma coeficientilor celor doi termini implicate in suma, in cazul in care cele doua monoame nu exponentii egali, metoda va returna null.

***public Monom sub(Monom monom)***

* ***Aceasta metoda va returna diferenta celor doua monoame***
* ***Diferenta celor doua monoame va fi returnata sub forma unui nou monom***
* ***Princpiul de functionare este cel de la metoda precedemta, se verifica daca opertia poate fi efectuata (monomale au acelasi exponent), in caz afiramtiv s eva returna monomul ce reprezinta diferenta, iar in caz negative se va returna null***

***Clasa Polinom***

Aceasta clasa este reprezentarea in cod a uni polinom din lumea reala, matematica. Ca si variabile de instanta pentru aceasta clasa am cosiderat ca este sufficient sa avem doar gradul polinomului (Integer) si un ArrayList<Monom> care sa reprezinte lista de monoamen care alcatuiesc polinomul.

Clasa polinom are si ea doi constructori, cu si fara parametrii, getters, stetters si metodele toString, createPolinom, getMonomByExponent, add, sub, multiply, divide, derivatate, integrate detaliate mai jos.

***public String toString()***

* Spurascrierea metodei toString() din clasa Objecct
* Folosita la afisarea polinomului in forma dorita de catre developer

***public void createPolinom(String string)***

* Aceasta metoda primeste ca si paramateru un string pe care il compara cu o expresie de tip regex pentru a putea genera polinomul corespunzator input-ului primit de la user.

***public Monom getMonomByExponent(Integer exponent)***

* Aceasta metoda primeste ca si parametru exponentul monomului pe care dorim sa il accesam din lista de monoame ale polinomului.
* Ea returneaza monomul cu exponentul dat ca si parametru, iar in cazul in care nu exista un astfel de monom, se va returna null

***public Polinom add(Polinom polinom)***

* Aceasta metoda primeste ca si paramateru un polinom
* Ea returneaza un nou polinom care reprezinta suma celor doua polinoame, cel care apeleaza metoda si cel primit ca si parametru

***public Polinom sub(Polinom polinom)***

* Aceasta metoda primeste ca si parametru un polinom
* Ea returneaza un nou polinom care reprezinta diferenta dintre polinomul care apeleaza metoda si cel primit ca si parametru de catre metoda

***public Polinom multiply(Polinom polinom)***

* Aceasta metoda primeste ca si parametru un polinom
* Ea returneza un nou polinom care reprezinta rezultatul inmultirii polinomului care apeleza metoda sic el primit ca si parametru in semnatura metodei.

***public Polinom derivation()***

* Aceasta metoda nu primeste niciun parametru in semnatura metodei, deoarece ea se aplica unui singur polinom si anume polinomului care apeleaza metoda.
* E ava returna un nou polinom ce reprezinta derivarea in functie de x a polinomului care a apaleat metoda.

***public String integration()***

* Aceasta metoda nu primeste niciun parametru in semnatura metodei, deoarece ea se aplcai unui singur polinom si anume polinomului care apleleaza metoda.
* E ava returna un sting, ce reprezinta integrarea polinomului care apleaza metoda, am ales sa returnez strin si nu un polinom care are coeficinti float deoarece si in lumea reala, in 99% din cazuri este folosita scrierea (1/2)\*x, nu 0.5\*x.

***Pachetul Views***

***Clasa PolynomialFrame***

Aceasta clasa reprezinta transpunerea in cod a GUI-ului aplicatiei. Aici sunt descrise toate componentele cu care utilizatorul va interactiona de-a lungul uilizarii aplicatiei.

Componentele pe care utilizicatorul le va intalni sunt:

* Butoane (JButton) pentru cele 6 operatii si pentru clear (stergerea datelor introduse din ambele filed-uri pentru polinoame)
* Text filed-uri (JTextFiled) pentru introducerea polinoamelor, aceastea sunt vizibile doar in momentul in care operatia este selectata si inca un text field in care este afiast rezultatul operatiei, acesta este facut astfel incat sa nu fie editat din exterior (de catre utilizator)
* Etichete (JLabel) cu scop ajutator, pentru a-l ajuta pe utilizator ce operatie se efectuaeaza, atat cu text (exemplu: addition), cat si cu simbol matematic (exemplu: +)
* Culoarea de fundal, am ales un fundal mov deschis
* Fonturile folosite, 2 la numar, pentru afisarea etichetelor si pentru afisarea polinoamelor

Toate componentele sunt declarate ca variabile private ale aceste clase si au create getters si stters pentru a putea fi folosite mai departe in pachetul controller.

Butoanelor le sunt create si metode de adaugare a ActionListener-urilor, iar aceste ActionListener-uri sunt descrise in pachetul controllers.

Tot in aceasta clasa, este setata dimensiunea frame-ului aplicatiei, dar si a elementelor ce compun acest frame, cat si pozitionarea lor in frame. Frame-ul este setat sa nu poate fi redimensionat, deoarece asa am considerat ca este sufficienta aceasta dimensiune.

***Pacehtul Controllers***

***Clasa PolynomialController***

Aceasta clasa are rolul de descrie ceea ce se va inatmpla la interactiunea utilizatorului cu aplicatia, in momentul incare acesta apasa butoanelele aplicatiei, sau introduce date si asa mai departe.

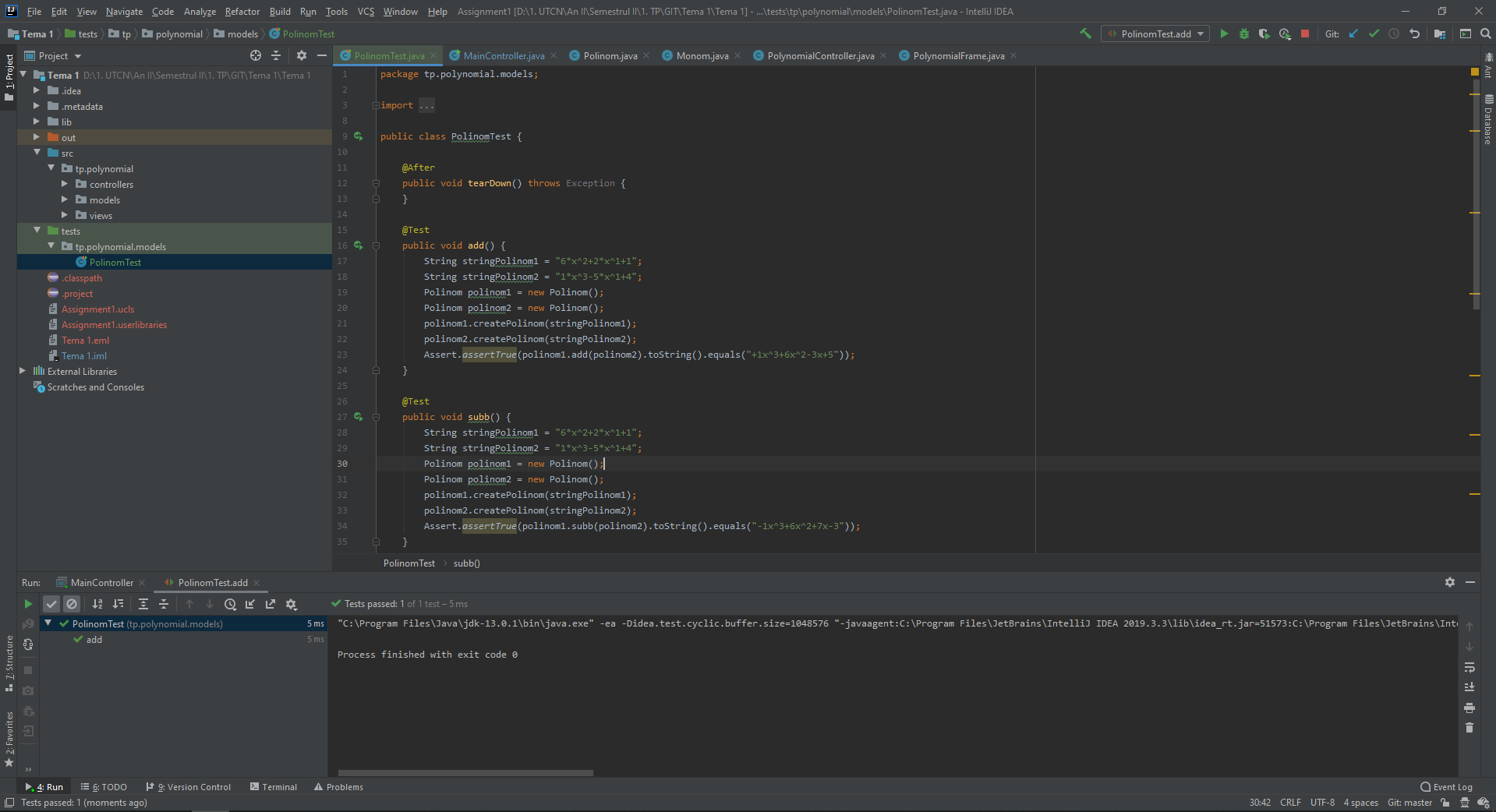
El poate efectua urmatoatele actiuni:

* Apasarea butonului “+”, corespunzator operatiei de adunare, acest lucru va face ca in frame sa apara campurile text pentru cei doi termini (polinoamele) implicati in adunare.
* Apasarea butonului “-“ corespunzator operatiei de scadere, care va avea acelasi efeact ca si cel produs de catre apasarea butonui +.
* Apasarea butonului “\*”, pentru operatia de inmultire, acelsi effect ca si cel al butoanelor de mai sus
* Apasarea butonului “/”, pentru imparitre, efect identic cu cel de mai sus
* Apsarea butonului “ |’ ” care corespunde operatiei de dervivare, efect foarte asemanator cu cel de mai sus, difera prin faptul ca la derivare este nevoir de un singur polinom.
* Apsarea butonului “S|” corespunzator integrarii polinomului, acelasi efectu cu el al operatiei de derivare.
* Apasarea fieacaruia din butoanele de mai sus va genera si un label care va indica in text operatia de efectuat, si un label care va indica simbolul matematica al operatiei.
* Apasarea butonului “Clear” care va avea ca efect stergea datelor introduse in campurile celor doua polinoame.
* Apasarea butonului “Submit” care in functie de operatia selectata, va incepe prin validarea datelor introduse, in cazul in care ele sunt bune, se va genera automau campul care va contine rezultatul operatiei dorite, in caz contrar, va inrosi campul care contine date eroane, pentru a-l atentiona pe utilizator ca nu a respectat formatul droit.

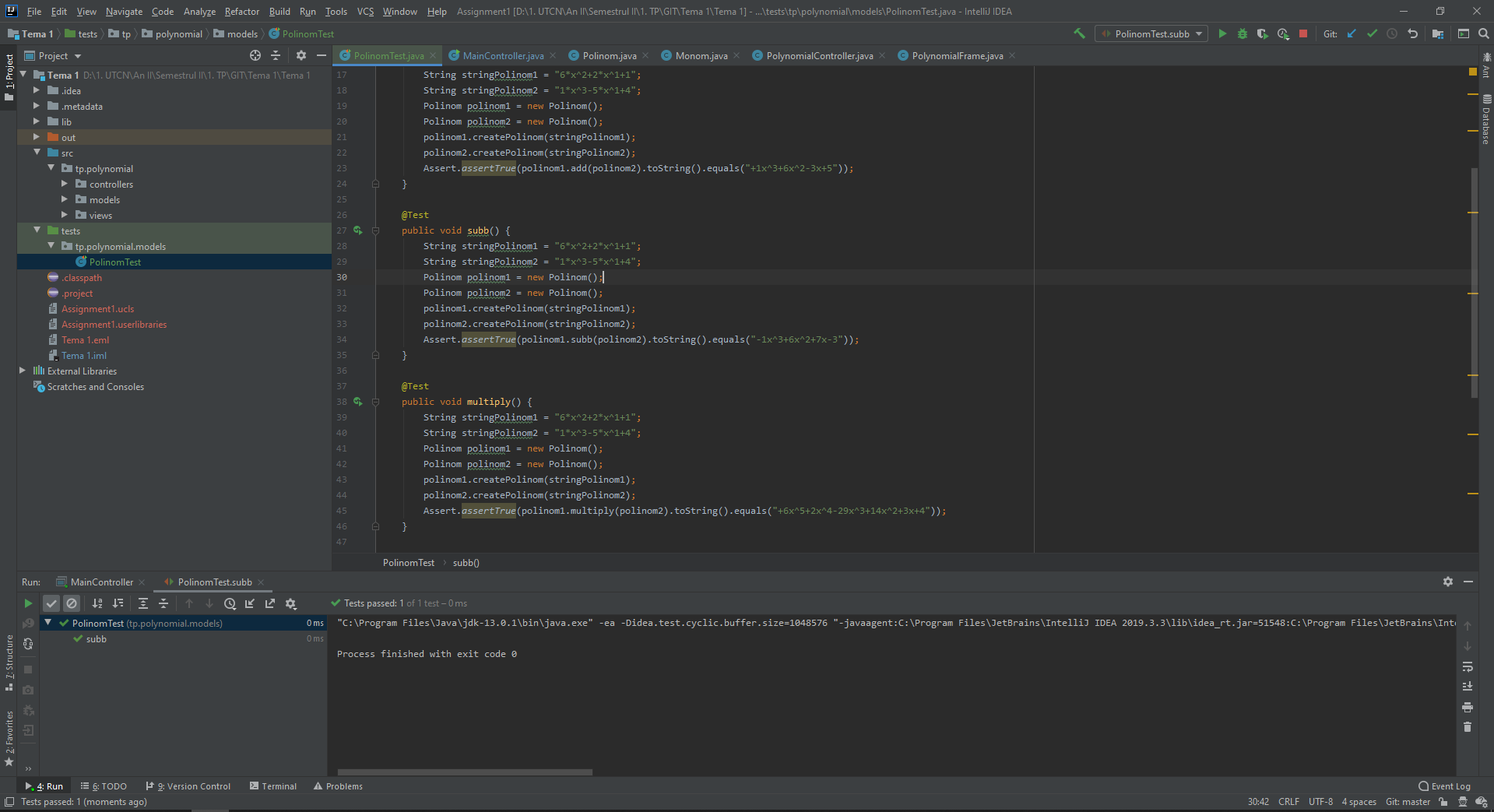
# *Rezultate*

Pentru testarea metodelor am folosit Junit, mai exact Junit 4 din IDE-ul IntelliJ.

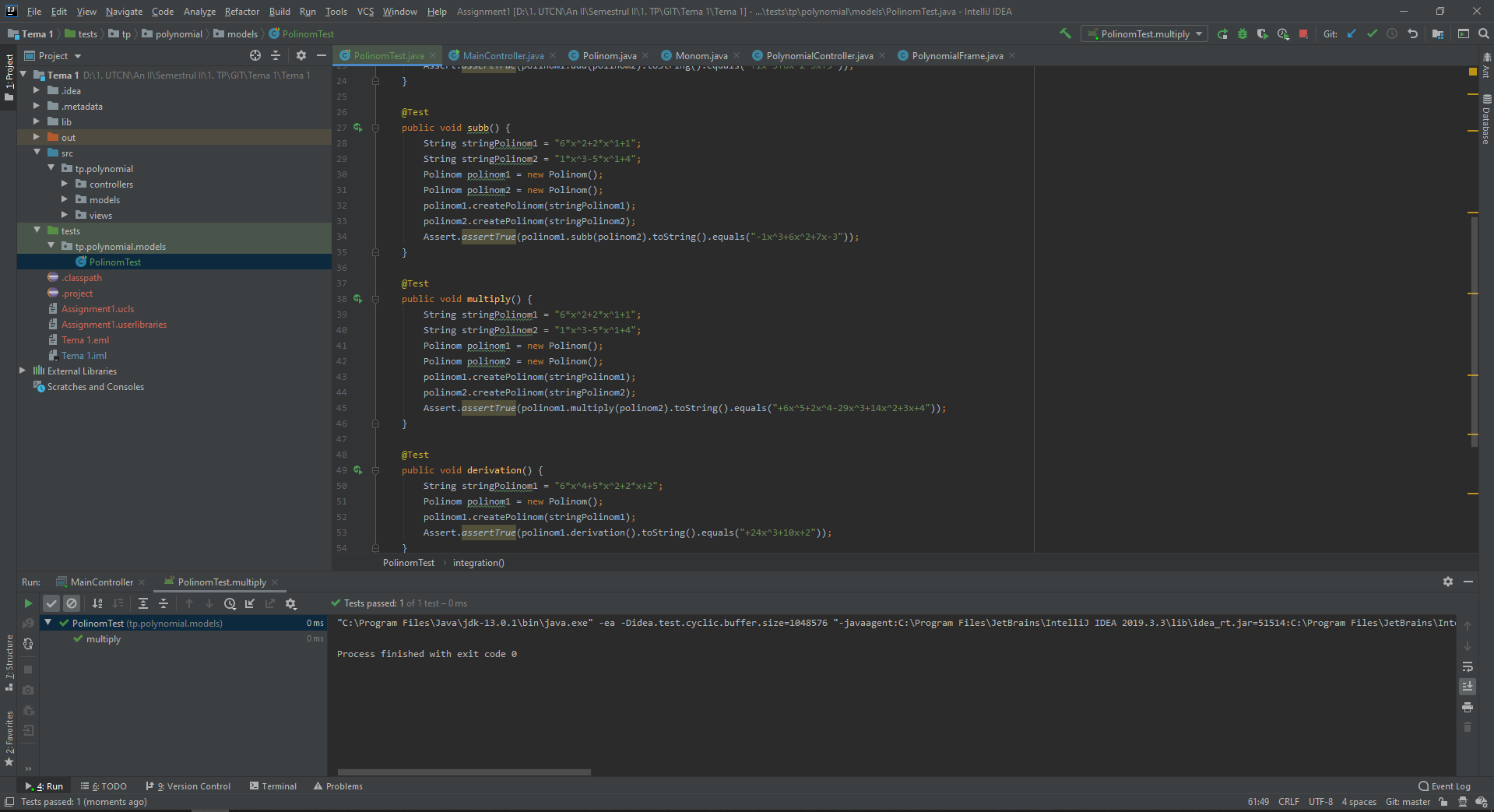
Testul efectuat pentru operatia de adunare a polinoamelor



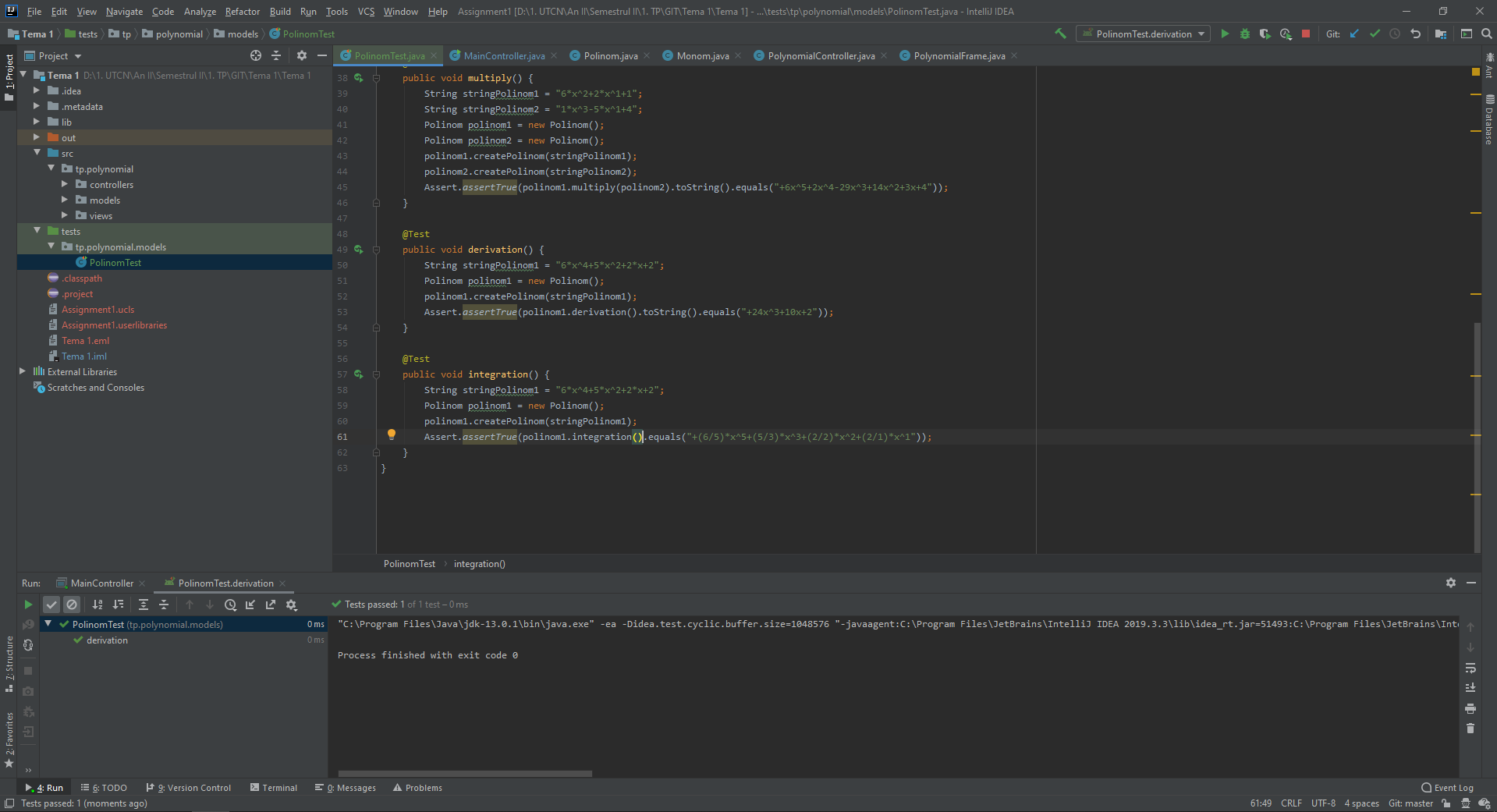
Testul efectuat pentru operatia de scadere a polinoamelor



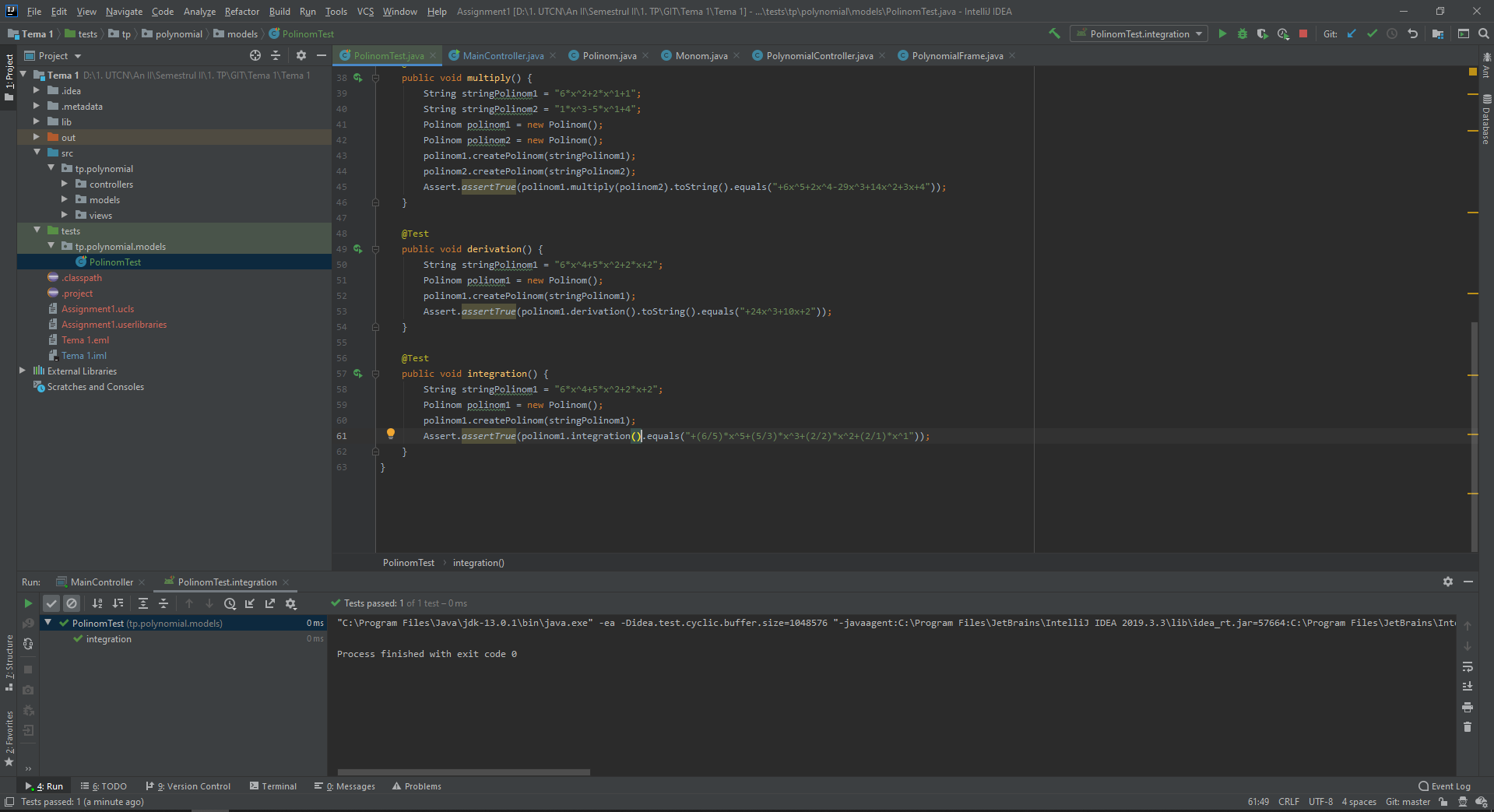
Testul efectuat pentru operatia de inmultire a polinoamelor:



Testul efectuat pentru operatia de derivare:



Testul efectuat pentru operatia de integrare:



# Concluzii

# 

Consider ca in aceasta tema am invatat sa lucreaz mult mai bine cu listele din Java si am invatat sa folosesc expresiile de tip regex, pe care nu le-am folosit deloc pana acum.

Ca si posibilitati de dezvoltare as putea enumera urmatoarele: imbunatatirea design-ului (prin alegerea unor culori mai potrivite, prin stilizarea butoanelor), transformarea aplicatiei dintr-o aplicatie desktop intr-o aplicatie web, legarea aplicatiei la o baza de data (ca utilizatorul sa poata vedea si ce operatii a efectuat in trecut, un scurt istoric), arm ai putea fi adaugata o ferseastra in care sa fie efectuata operatia pas cu pas (astfek utilizatorul nu ar vedea doar rezultatul, ar vedea si pasii care trebuie urmati pentru a ajunge la rezultatul dorit), o alta imbunartatire pe care as putea sa o adaug proiectului pe viitor ar fi realizarea simplificarii in caazul integrarii, daca se va putea.

# *Bibliografie*

<https://ro.wikipedia.org/wiki/Polinom>

<https://www.jetbrains.com/help/idea/tdd-with-intellij-idea.html#>