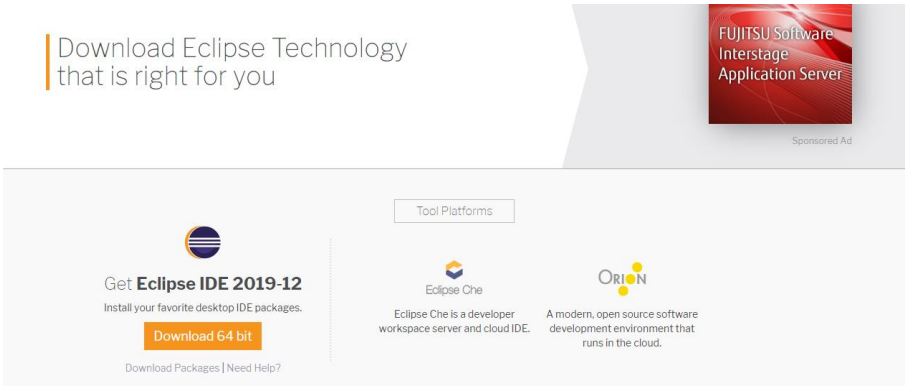
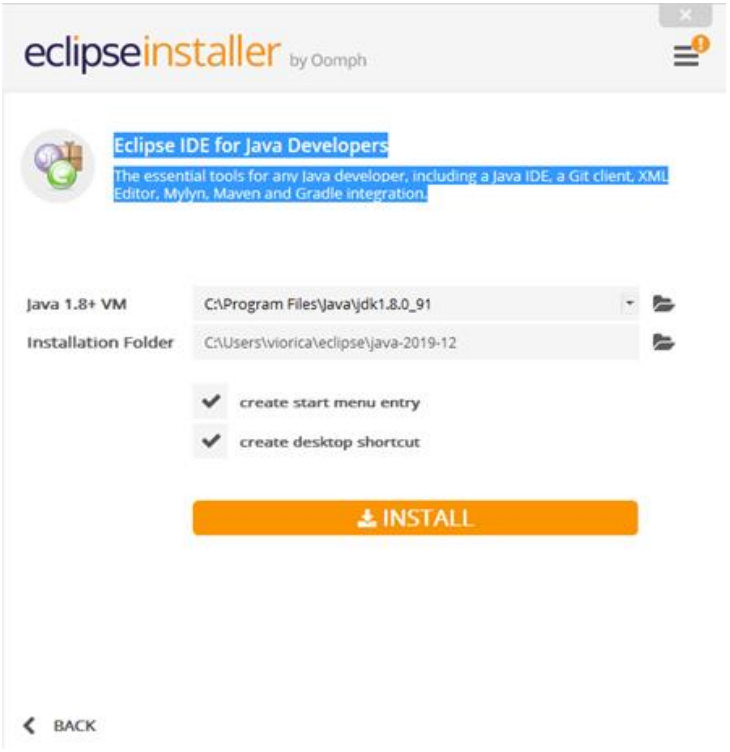
Laborator 1- Bazele Programarii Orientate pe Obiect

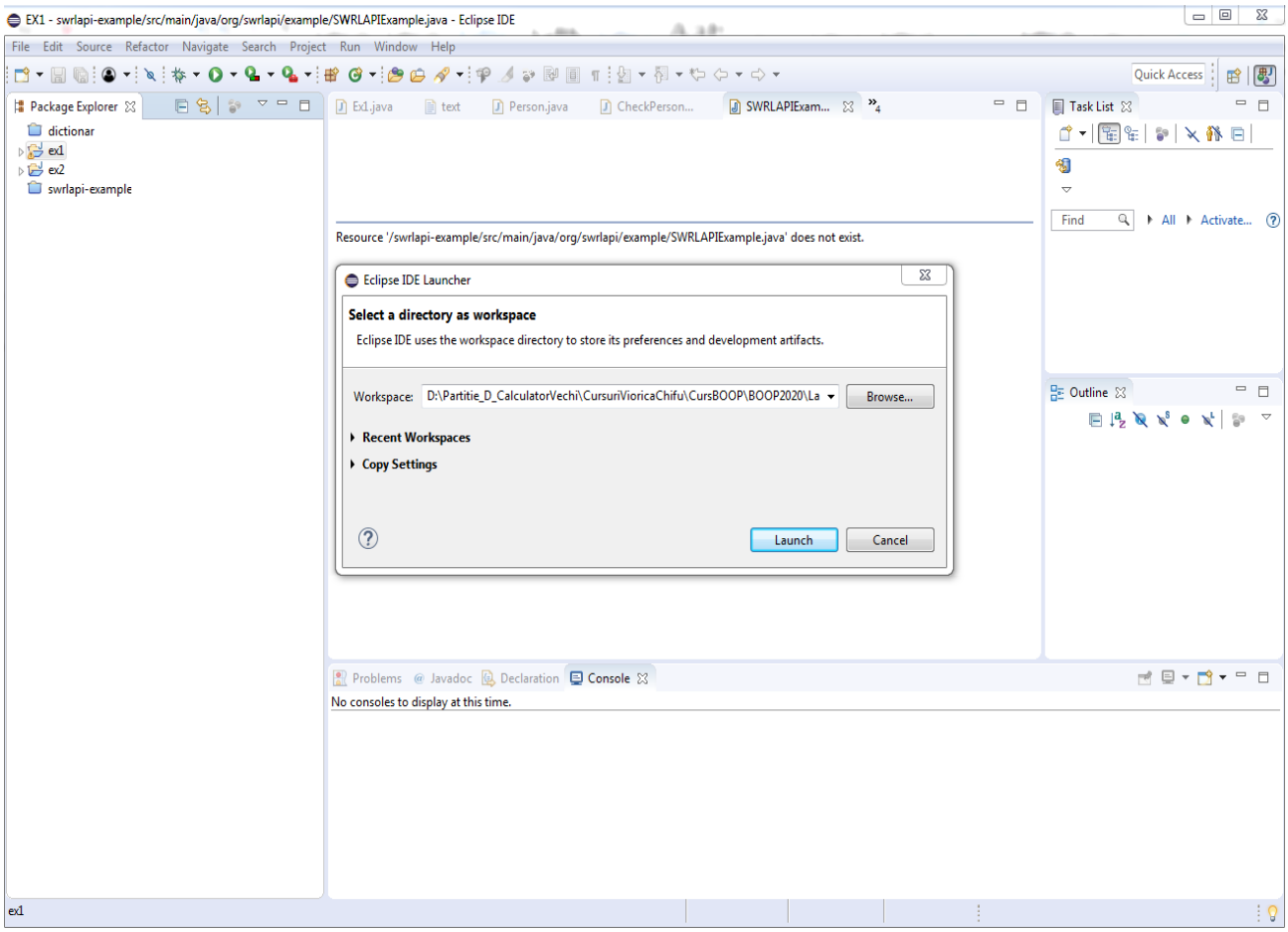
1. Mediul de lucru Eclipse

Este un mediu integrat pentru dezvoltarea aplicatiilor in java sau alte limbaje de programare. Eclipse poate fi facut download de la adresa: http://www.eclipse.org/downloads/ (vezi Figura 1). Pentru a instala mediul **Eclipse** pe sistemul de operare Windows se lanseaza in executie executabilul, se selecteaza **Eclipse IDE for Java Developers** si se instaleaza (vezi Figura 2).





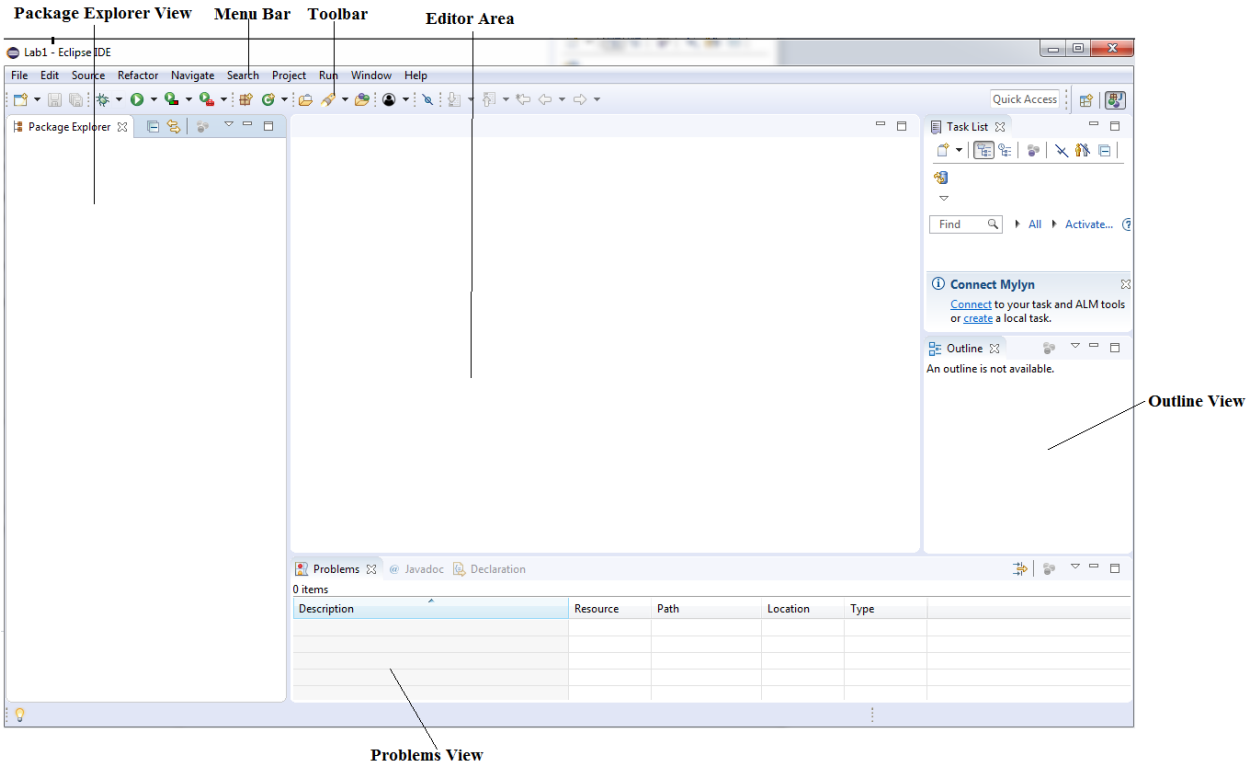
Lansare in executie a mediului Eclipse. Când se laseaza in executie Eclipse pentru prima data, vi se va solicita sa selectati un director ca si spațiu de lucru. Aceasta inseamna ca toate aplicatiile java pe care le veti dezvolta vor fi salvate in acel director. Puteți accepta directorul care este propus implicit sau puteti alege un alt director ca si spatiu de lucru.



Componentele unei ferestre Eclipse. Componentele principale ale unei ferestre Eclipse sunt:

- Views
- Editors (all appear in one editor area)
- Menu Bar
- Toolbar

În general, view-rile sunt utilizate pentru a vizualiza metadatele proiectului. Package Explorer ne permite sa vizualizam fișierele java din proiect și Editor Area este folosit pentru a edita un fișier java. Componentele din Menu Bar si Toolbar reprezintă comenzi care pot fi declanșate de un utilizator final.



Meniurile disponibile in **Menu Bar** sunt:

- File menu - permite deschiderea unor fișiere pentru editare, inchiderea, salvarea conținutului editat, precum si redenumirea de fișierele. De asemenea, mai permite importul și exportul conținutului spațiului de lucru și închiderea mediului de lucru Eclipse.
- Edit menu - prezintă elemente precum copy & paste sau find/repace
- Source menu - este vizibil numai atunci când este deschis un editor java. Prezintă o serie de elemente de meniu utile.
- Refactor – perminte sa redenumiti o clasa, semnatura unei metode, etc.
- Navigate menu - vă permite să localizați rapid resurse și să navigați la ele.
- Search menu - prezintă elemente care vă permit să căutați în spațiul de lucru printre fișierele care conțin date specifice.
- Project menu - Elementele de meniu legate de construirea unui proiect
- Run menu - Elementele de meniu din meniul Run îți permit să pornești un program în modul rulare sau în modul de depanare. De asemenea, prezintă elemente de meniu care vă permit să depanați codul.
- Window menu - vă permite să deschideți și să închideți vizualizări și perspective. Vă permite, de asemenea, să deschideți dialogul Preferințe
- Help menu - Meniul de ajutor poate fi utilizat pentru a deschide fereastra de ajutor, vizualizarea Eclipse Marketplace-ului sau pentru a instala noi plugin-uri. Elementul de meniu About Eclipse IDE vă oferă informații despre versiune.

Crearea unui proiect java in Eclipse. Se poate face folosind **New Java Project wizard** in 3 moduri:

- Dând click pe meniul File și alegand New ⟶ Java Project.

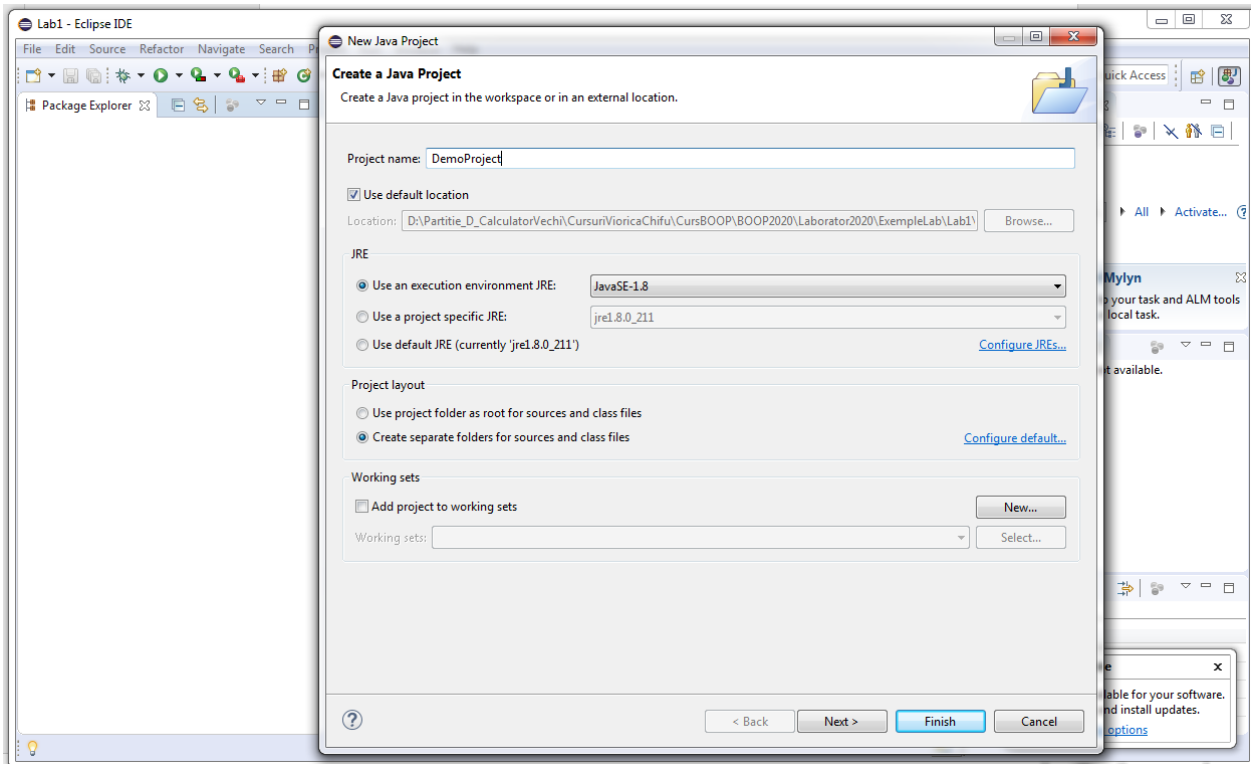
- Dând click dreapta oriunde în Package Explorere și selectând New → Java Project
- Dând clic pe butonul New din ToolBar și selectând Java Project.

New Java Project wizard are doua pagini:

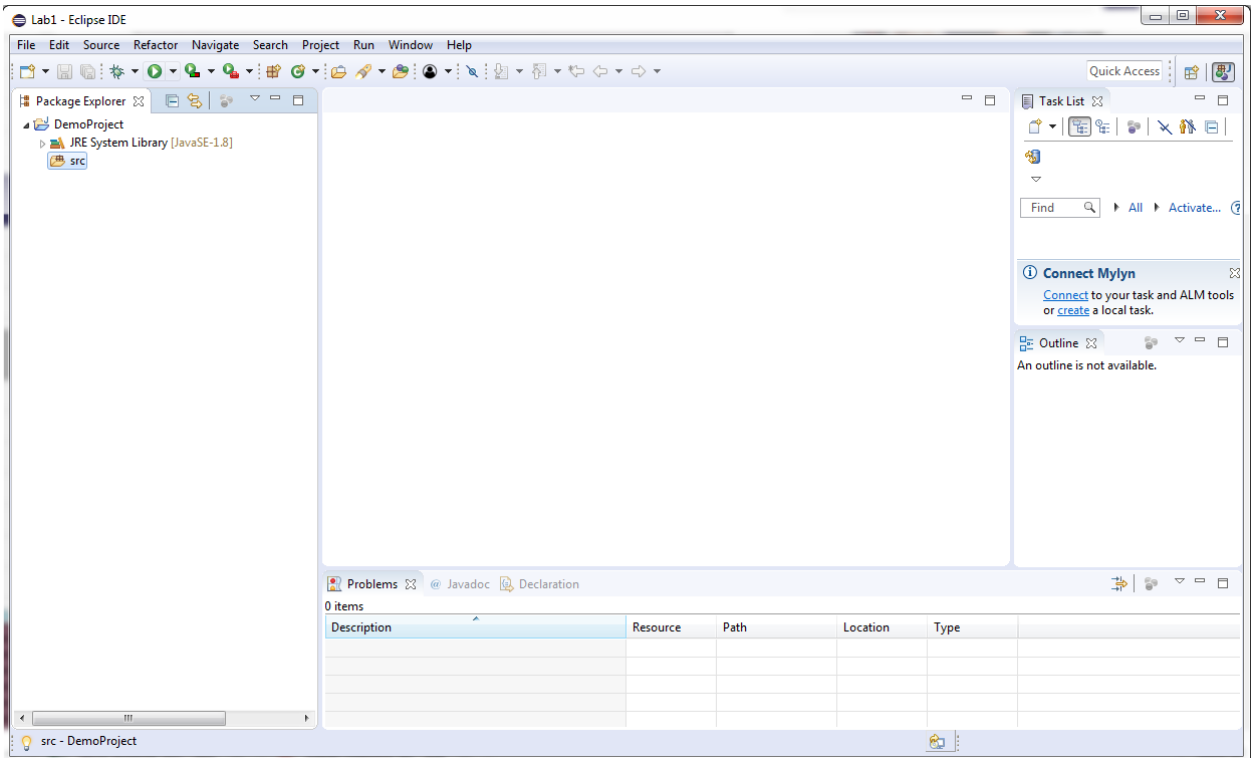
In prima pagina:

- Introduceți numele proiectului
- Selectați Java Runtime Environment (JRE) sau lăsați-l implicit
- Selectați aspectul proiectului care stabilește dacă ar exista un folder separat pentru codurile sursă și fișierele de clasă. Opțiunea recomandată este crearea de dosare separate pentru surse și fișiere de clasă.
- Puteți face clic pe butonul **Finish** pentru a crea proiectul sau faceți clic pe butonul **Next** pentru a modifica setările de construire java.

In a doua pagină puteți schimba **Java Build Settings**, cum ar fi setarea dependențelor de proiect (dacă există mai multe proiecte) și adăugarea de fișiere jar suplimentare la calea de construire.



Vizualizarea proiectului creat. Se poate face in **Package Explorer**.

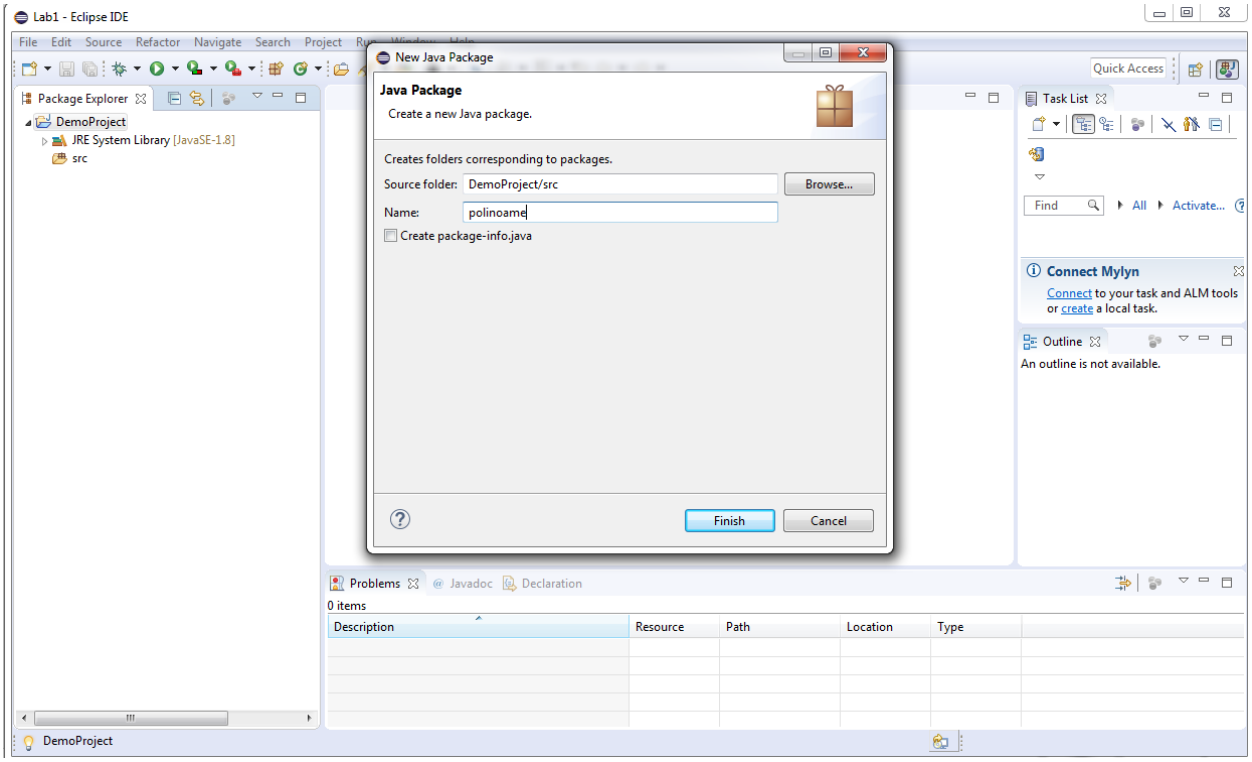


Crearea unui package in Eclipse. Se poate face folosind **New Java Package** wizard in trei moduri:

- Dand click pe meniul File și selectând New → Package
- Dand click dreapta în Package Explorer și selectând New → Package.
- Dand click dreapta pe pictograma package in Tool Bar.

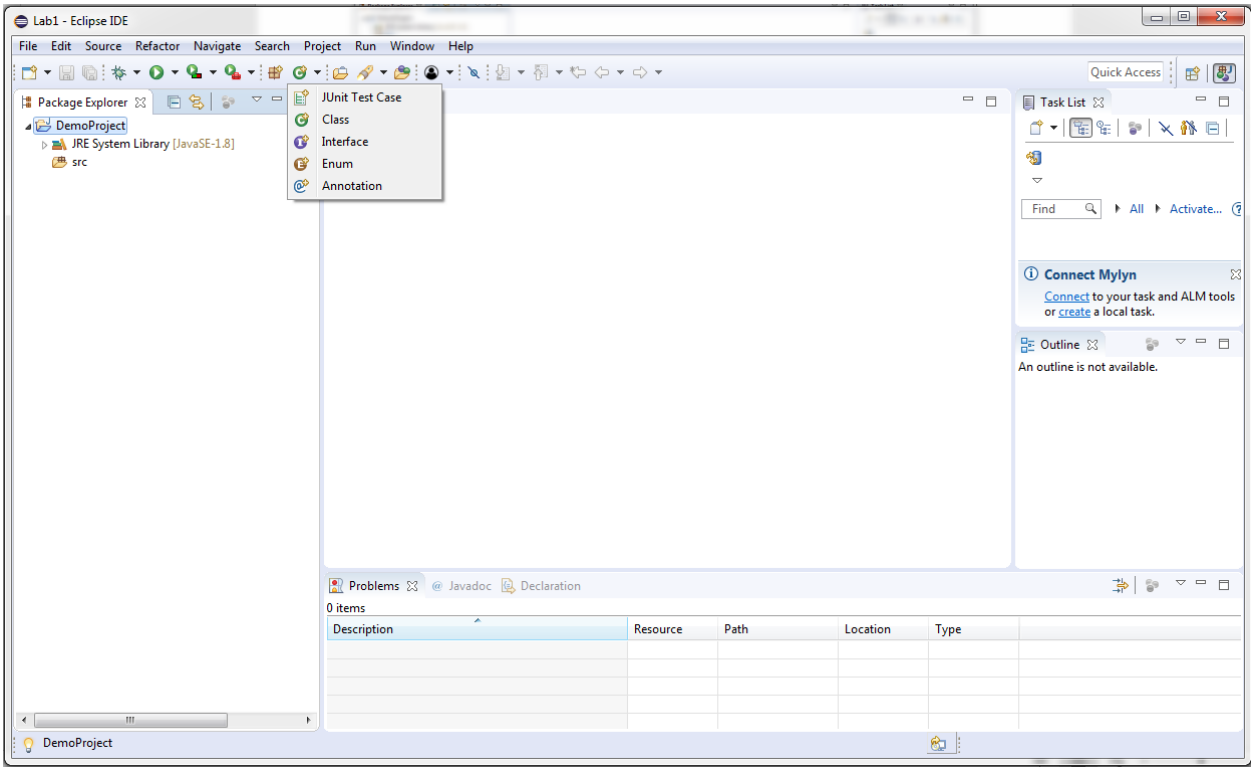
Pasii urmati pentru a crea un package cu **New Java Package**

- Introduceți / confirmați numele directorului sursa.
- Introduceți numele pachetului.
- Faceți click pe butonul **Finish**.



Crearea unei clase in Eclipse. Se poate face folosind **Java Class wizard** in trei moduri.

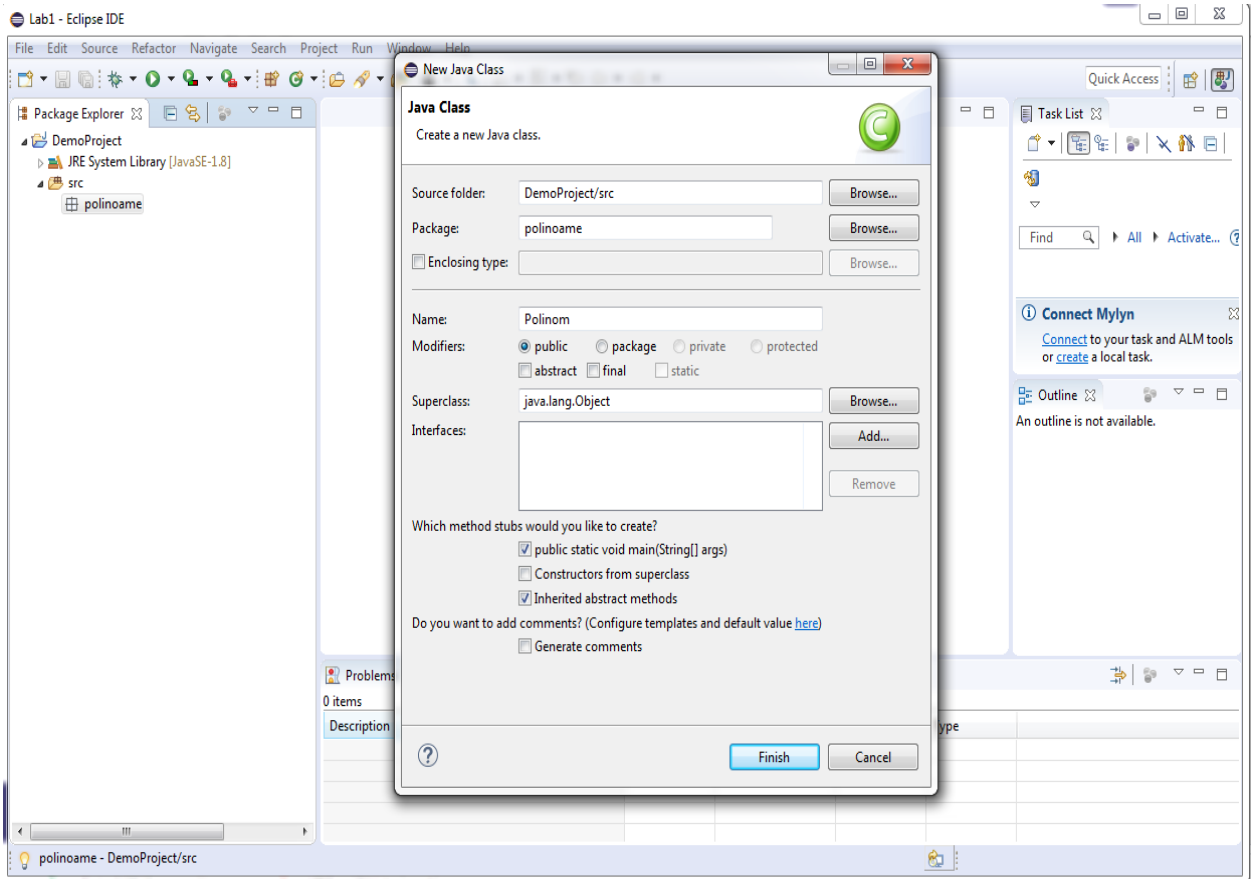
- Dand click pe meniul **File** și selectând New → Class.
- Dand click dreapta în **Package Explorer** și selectând **New** → Class.
- Dand click pe drop down button-ul corespunzator din Tool Bar si selectand apoi Class

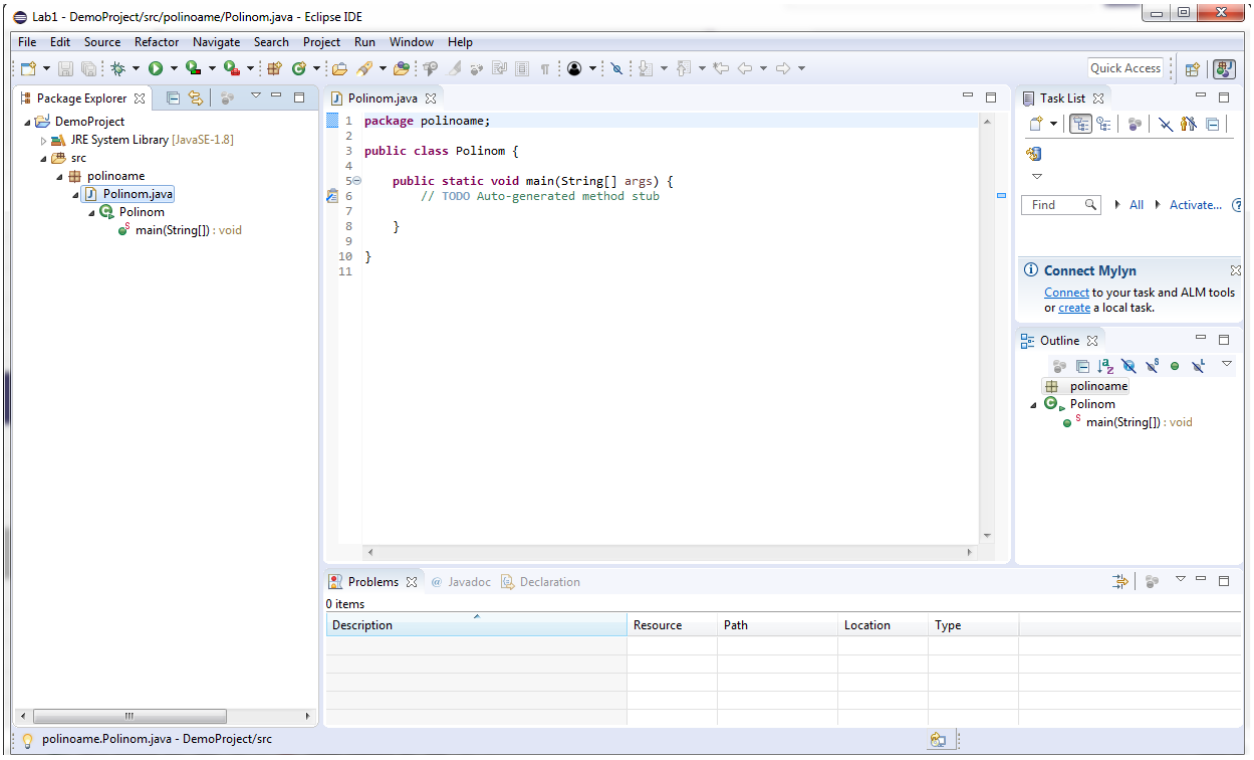


Pasii urmati pentru a crea un package cu **New Java Class wizard**

- Asigurați-vă că folderul sursă și pachetul sunt corecte.
- Introduceți numele clasei.
- Selectați modificatorul de clasă corespunzător.
- Introduceți numele super clasei sau faceți clic pe butonul Browse pentru a căuta o clasă existentă.
- Faceți clic pe butonul Adăugare pentru a selecta interfețele implementate de această clasă.
- Examinați și modificați casetele de validare referitoare la butoanele și comentariile metodelor.

Clasa nou creata va aparea in Package Explorer si va putea fi modificate in **Editor Area**.


Lab1 - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
DemoProject
JRE System Library [JavaSE-1.8]
src
polinoame
New Java Class
Java Class
Create a new Java class.
Source folder: DemoProject/src
Package: polinoame
Enclosing type:
Name: Polinom
Modifiers: public package private protected
abstract final static
Superclass: java.lang.Object
Interfaces:
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Browse...
Add...
Remove
Finish
Cancel
Task List
Find All Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
An outline is not available.
Problems
0 items
Description
polinoame - DemoProject/src


Lab1 - DemoProject/src/polinoame/Polinom.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
DemoProject
JRE System Library [JavaSE-1.8]
src
polinoame
Polinom.java
Polinom
main(String[]) : void
Polinom.java
1 package polinoame;
2
3 public class Polinom {
4
5 public static void main(String[] args) {
6 // TODO Auto-generated method stub
7
8 }
9
10 }
11
Task List
Find All Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
polinoame
Polinom
main(String[]) : void
Problems
Javadoc
Declaration
0 items
Description
Resource
Path
Location
Type
polinoame.Polinom.java - DemoProject/src

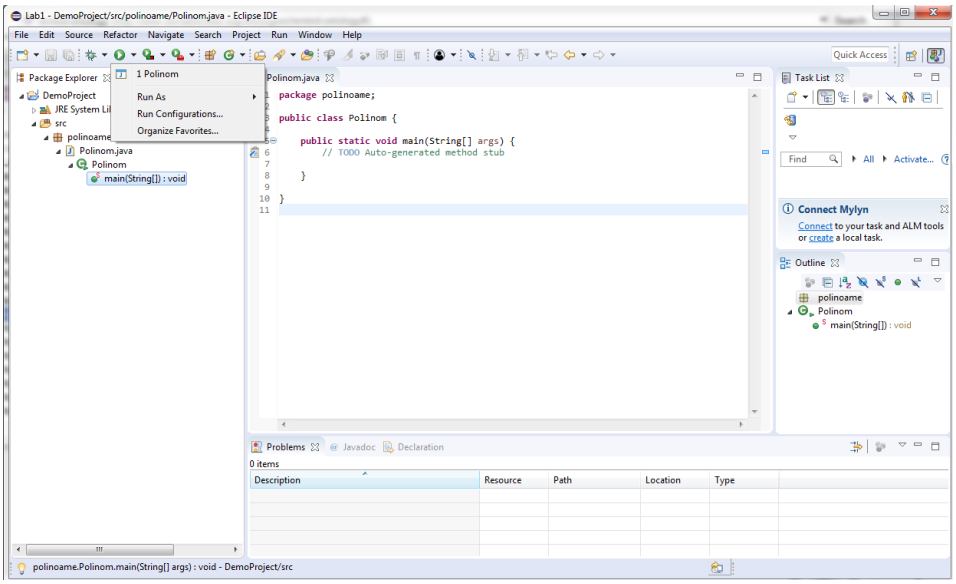
Rularea unei aplicatii Java.

1) În Package Explorer

- Faceți click dreapta pe clasa java care conține metoda main ()
- Selectați Run As → Java Applications.

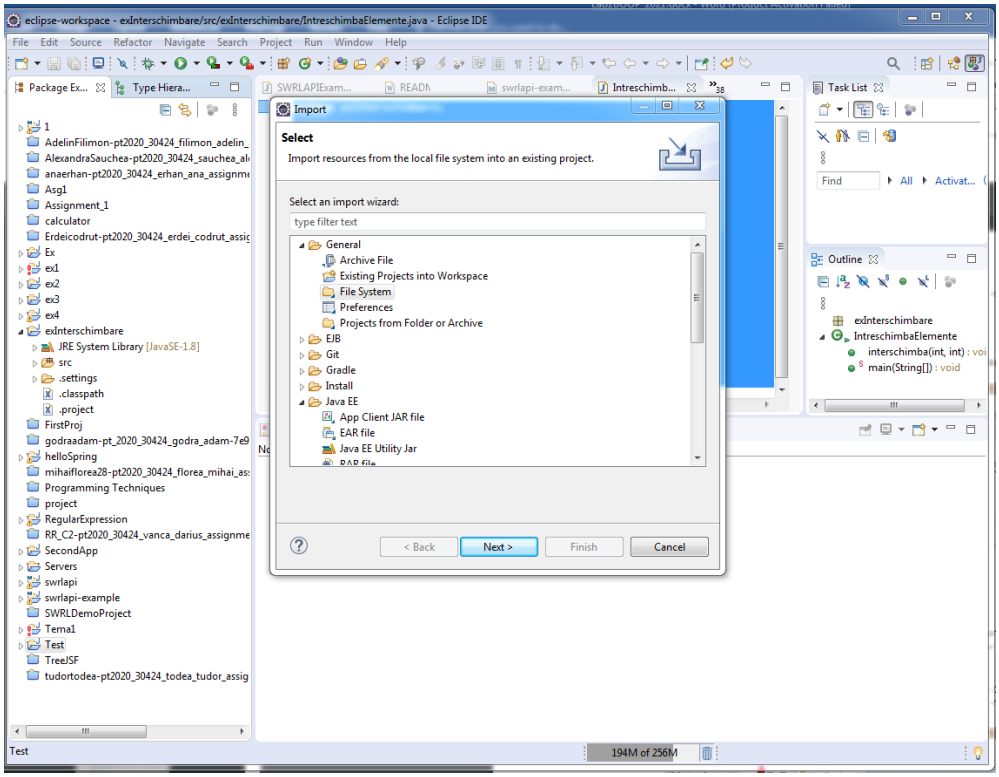
2) folosind combinati de taste CTRL+F11 sau selectand din Menu Bar, Run

3) Selectand din **ToolBar** menu icon-ul corespunzator

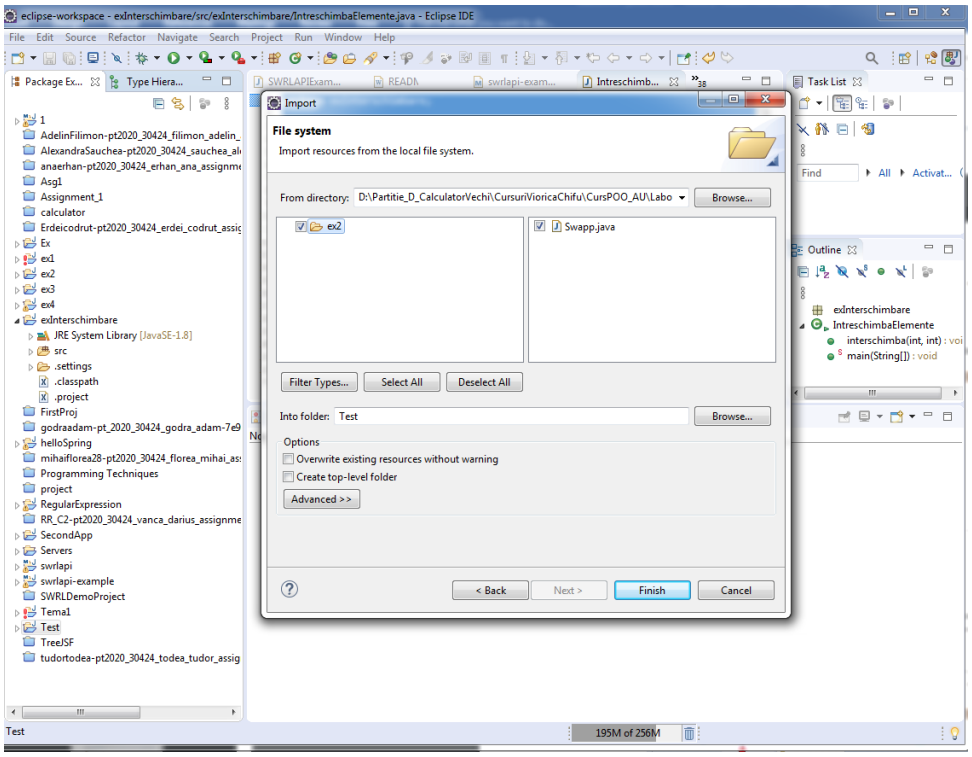


2. Crearea unei aplicații Java pe baza unor fișiere sursă existente

Se creaza un proiect nou. Se selecteaza meniul File->Import->General->File System.

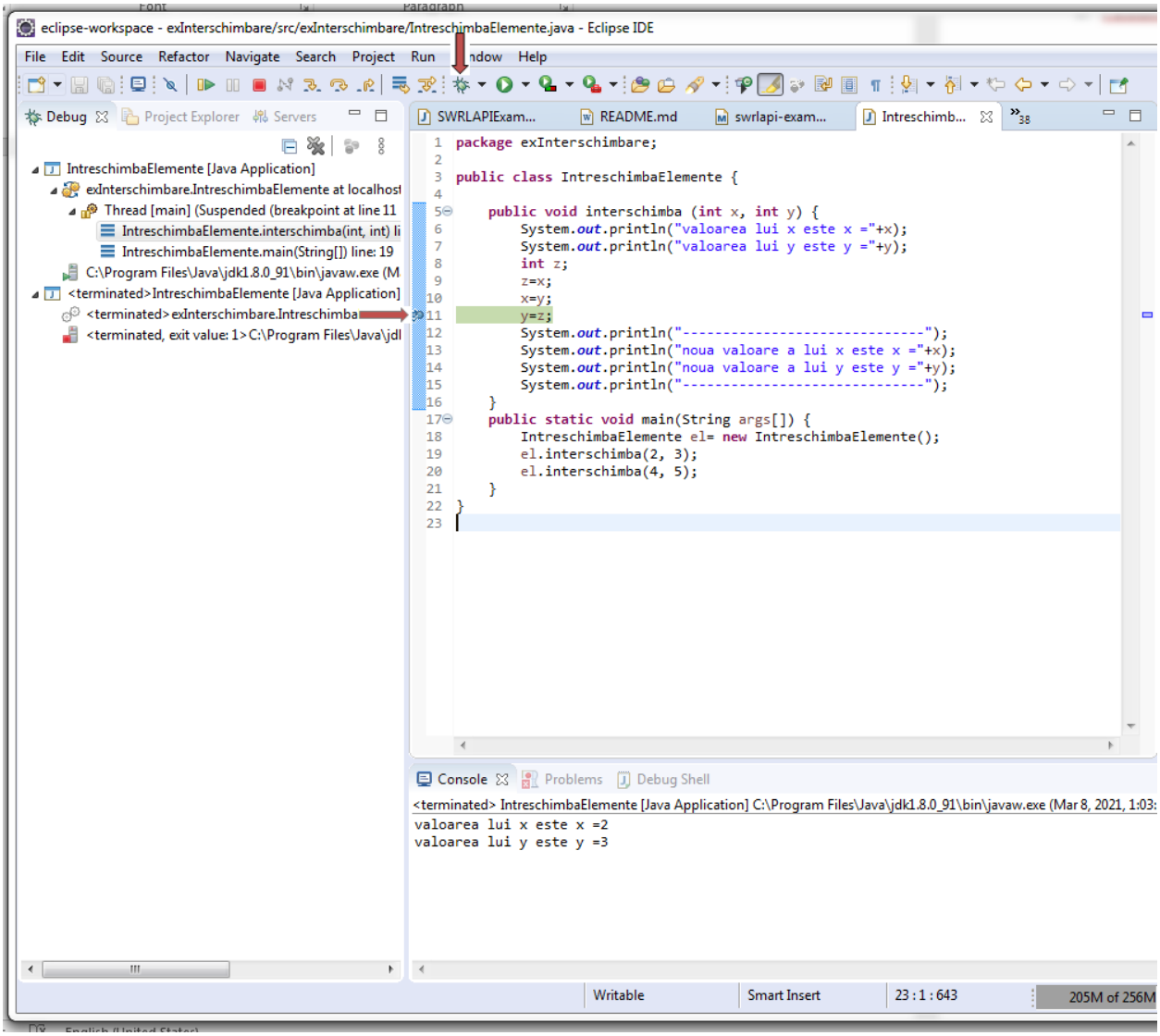


Se selecteaza directorul corespunzator și se importa în proiect toate fișierele .java din directorul selectat.

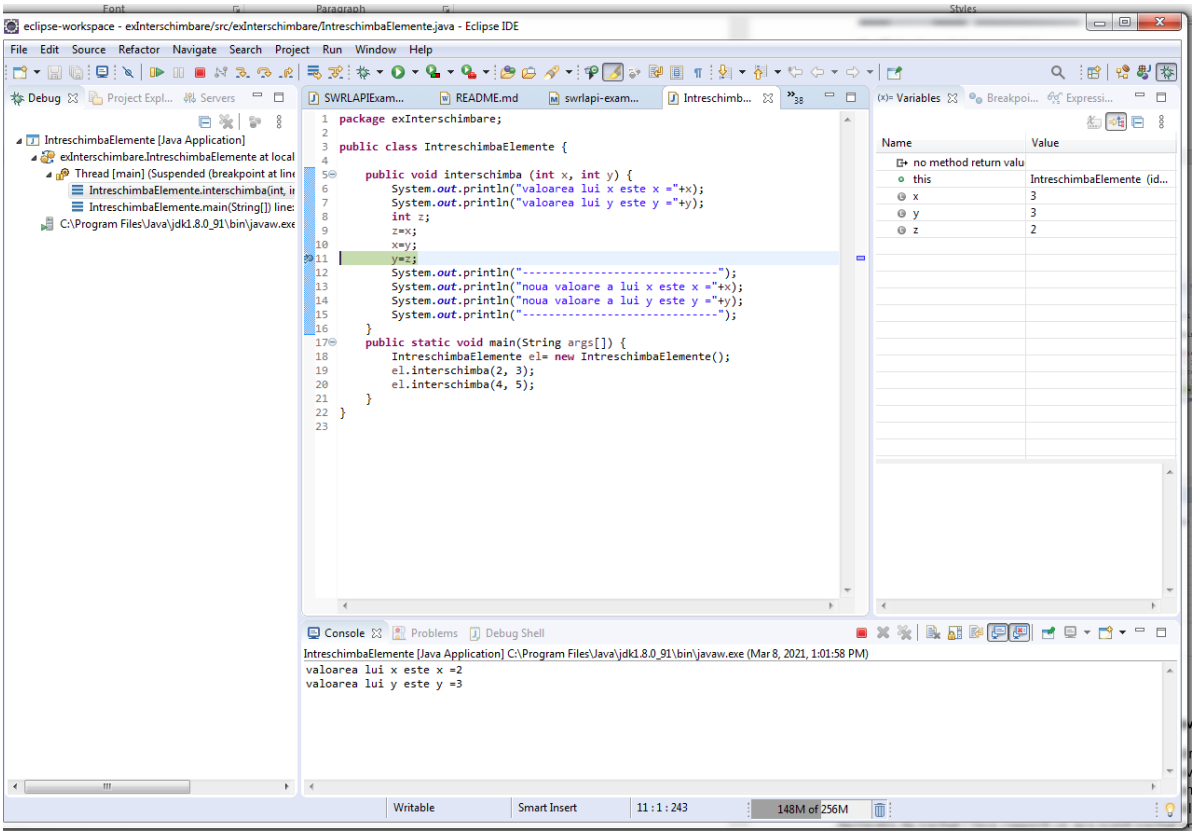


3. Folosirea debugger-ului

Debugger-ul se foloseste pentru a depana un program cu scopul de a identifica eventualele erori. Pentru depanare se plaseaza break point-uri în dreptul liniilor de cod unde vrem să oprim execuția programului cu scopul de a verifica valorile stocate in variabilele curente. Un break point se activeaza dand dublu click în dreptul acelei linii de cod, iar apoi se ruleaza programul în modul Debug:



La rulare, execuția se oprește în dreptul liniei unde am plasat break point-ul. Se deschide automat fereastra Variables unde se pot vizualiza valorile variabilelor locale.



În continuare aveți posibilitatea - de a continua rularea programului (Resume F8), - de a merge la următoarea linie de cod care se execută (Step Into F5); dacă pe linia curentă este apelul unei metode, atunci opțiunea Step Into va intra în interiorul metodei. - de a merge la următoarea linie de cod din metoda curentă (Step Over F6).

4. **Recomandari de urmat in dezvoltarea aplicatiilor in Java**.

Ori de câte ori începeți un proiect nou, creați un nou director pentru fișierele sursă. Numele directorului ar trebui să fie specificat cu litere mici. Daca avem de dezvoltat o aplicatie complexa alcatuita din mai multe clase e recomandat sa clasificam clasele in pachete. Pachetele sunt directoare care conțin clase Java înrudite. Pentru aplicatiile de dimensiuni mici, se poate omite declaratia de pachet (Java creează un asa numit pachet implicit în acest caz). Se poate declara cate o singura clasa in cate un fisier (fiecare clasa va avea propriul sau fisier). Exista posibilitatea sa declarati mai multe clase intr-un fisier, dar nu e recomandabil in cadrul aplicatiilor mari. Exista medii de lucru precum NetBeans care necesita ca fiecare clasa sa fie declarata intr-un fisier separat. Pentru o clasa declarata publica, numele clasei trebuie sa coincida cu numele fisierului in care este salvata clasa.

5. **Consideratii teoretice.**

Orice aplicatie in Java este alcatuita din clase.

O clasa se declara cu cuvantul cheie **class** urmat de numele clasei si de corpul clasei scris intre acolade.

class Persoana {

}

O clasa poate contine declaratii de variabile, metode, constructori. Constructorii au acelasi nume cu numele clasei in care sunt declarati si sunt folositi pentru a instatia obiecte de tipul clasei. Intr-o calsa pot fi declarati unul sau mai multi constructori care au parametrii diferiti (poate sa difere atat atat tipul de data al parametrilor cat si numarul lor). Intr-o aplicatie java desktop trebuie sa existe o clasa care are declarata metoda **main** (). Metoda **main**() corespunde programului principal – codul scris in interiorul metodei **main**() este ceea ce se executa cand noi lansam in executie o aplicatie java.

Ex de clasa declarata in Java

```
class Persoana {
        private String nume;
        private int varsta;
        //declaratie de constructor cu parametrii
        Persoana(String n, int v){
          this.nume = n;
          this.varsta = v;
        }
        public String getNume(){
          return nume;
        }
        public int getVarsta(){
          return varsta;
        }
        public void setNume(String nume){
          this.nume=nume;
        }

        //declaratie de metoda
        public String toString(){
          return “Nume: “+nume+”, varsta “+varsta;
        }
        //declaratia metodei main ()
        public static void main (String args()){
         Persoana p= new Persoana(“Ana”, 20);
```

```
        System.out.println(p.toString());
        }
}
```

Cum putem genera numere aleatorii in java?

In java se pot genera numere aleatorii folosindu-ne de clasa **Random**.Pentru a utiliza această clasă pentru a genera numere aleatorii, trebuie mai întâi să creăm o instanță a acestei clase și apoi să invocăm metode precum nextInt (), nextDouble (), nextLong () etc. folosind acea instanță.

Putem genera numere aleatorii de tipuri întregi, flotante, duble, lungi, booleeni folosind clasa Random.

Putem transmite argumente metodelor care reprezinta limita superioara corespunzatoare numerelor care urmeaza a fi generate. De exemplu, nextInt (6) va genera numere cuprinse între 0 și 5, inclusiv 0 si 5.

Exemplu:

```
import java.util.Random;
…….
Random rand = new Random();

// Generate random integers in range 0 to 999
int rand_int1 = rand.nextInt(1000);
int rand_int2 = rand.nextInt(1000);

// Generate Random doubles
double rand_dub1 = rand.nextDouble();
double rand_dub2 = rand.nextDouble();
```

Probleme de rezolvat la laborator:

1. Scrieti o clasa java care sa afiseze la consola mesajul:" This is my first application".

2. Scrieti un program alcatuit din doua clase: clasa **Exemplu** si clasa **Instantiere.** Clasa **Exemplu** declara un atribut **x** de tip **int**, un constructor, o metoda **increment**() care incrementeaza valoarea lui **x** cu valoarea 1 si o metoda **print**() care afiseaza valoarea atributului clasei. Clasa **Instantiere** instatiaza doua obiecte de tipul **Exemplu**, apeleaza metoda de incrementare pe cele doua obiecte si metoda **print**() atat inainte cat si dupa incrementare.

3. Scrieti un program Figuri Geometrice alcatuit din urmatoarele clase: clasa **Punct**, clasa **Cerc** si clasa **FiguriGeometrice**. Clasa **Punct** declara atributele unui punct (se considera atributele punctului cele doua coordinate x, y), doi constructori, o metoda **equals,** metode de tip **get si set** si metoda **toString**. Metoda **equals** compara doua puncte daca sunt egale folosind ca si criteriu de comparatie coordonatele celor doua puncte. O metoda **get** este o metoda care returnează valoarea unei variabilă, iar o metoda **set** seste o metoda care setează valoarea acelei variabile - a se vedea exemplu de la consideratii teoretice. Metoda **toString**() permite obţinerea unui **String** cu informaţii despre obiectul pe care este apelată - a se vedea exemplu de la consideratii teoretice.

Clasa **Cerc** declara atributele unui cerc (atributele unui cer sunt centru cercului reprezenta de un punct si raza cercului), doi constructori, metoda **equals,** metoda **toString**(), metode pentru calcularea ariei si perimetrului unui cerc. Metoda **equals** compara doua cercuri daca sunt egale folosind ca si criteriu de comparatie centru cercului si raza cercului. In implementarea metodelor **equals** si **toString**() din clasa **Cerc** se vor lua in considerare metodele **equals** si **toString**(0 definite in clasa **Punct**.
Clasa **FiguriGeometrice** instatiaza obiecte de tip Punct si Cerc si apeleaza metodele celor doua clase.

4. Scrieti o aplicatie java care genereaza aleatoriu un număr întreg între 0 și 9 și tipăreste dublul acestui număr.