Table of Contents

[1. Tablouri 2](#_Toc2511237)

[1.1. Declarare 2](#_Toc2511238)

[1.2. Accesarea elementelor 2](#_Toc2511239)

[1.3. Clasa ajutătoare / utilitară: Arrays 3](#_Toc2511240)

[2. Clase 4](#_Toc2511241)

[2.1. Modul de definire al unei clase 4](#_Toc2511242)

[2.2. Importarea de pachete 5](#_Toc2511243)

[2.3. Modificatorii de acces 5](#_Toc2511244)

# Tablouri

## Declarare

Sunt trei opțiuni:

1. După tipul variabilele se adaugă paranteze pătrate

Integer[] arr = {4, 6, 7};

1. După numele variabilei se adaugă paranteze pătrate

String arrs[] = {**"cuvant unu"**, **"cuvant doi"**, **"cuvant trei"**};

1. Pe partea de inițializare se adaugă și new de tip cu paranteze pătrate. Pentru că este laborioasă a devenit inutilizabilă în ultimele versiuni, cu o singură excepție

*// old fashion, old school*String arrn[] = **new** String[]{**"new string"**, **"second string"**};

String arrn[] = **new** String[10]; *// fara initializare*

Plus declararea in-line a unor variabile

Arrays.*toString*(**new** Integer[]{1, 2, 3});

Declararea unui tablou multidimensional:

Integer[][] a = {{1, 2, 3}, {4, 5, 6}};  
Integer[][] b = {{1, 2, 3}, {4, 5, 6}};

## Accesarea elementelor

Cea mai simpla accesare se face folosind indexul elementului dorit.

Tablourile folosesc notarea primului element cu indexul 0.

Integer[] arr = {4, 6, 7};

arr[i] …

// parcurgere folosind index

**for** (**int** i = 0; i < arr.**length**; i++) {  
 System.***out***.println(**"i : "** + arr[i]);  
}

// parcurgere folosind foreach

**for** (Integer i : arr) {  
 System.***out***.println(**"arr: "** + i);  
}

O varianta de parcurgere a unui tablou bi-dimensional

**for** (**int** i = 0; i < a.**length**; i++) {  
 Integer[] row = a[i];  
 System.***out***.println(**"row: "** + row);  
 **for** (**int** j = 0; j < row.**length**; j++) {  
 System.***out***.println(**"row["** + j + **"] = "** + row[j]);  
 System.***out***.println(**"b["** + i + **", "** + j + **"] = "** + b[i][j]);  
  
 }  
}

Task - 1

Să se realizeze un pachet com.fmi.lab2.arrays, în pachetul creat să se definească clasa ArraysMain. Aceasta va conține o metoda public static void main(String[] args) în care să va definiți un tablou uni-dimensional ce va conține elemente de tip Integer.

Realizați o metoda statică ce implementează imprimarea la consolă a elementelor tabloului. Numele / semnătura metodei poate să fie prinArrays(Integer[] array) sau prinArrays(Object[] array).

## Clasa ajutătoare / utilitară: Arrays

**import** java.util.Arrays;

Este o clasă ajutătoare ce permite realizarea unor operațiuni simple cu tablouri.

Exemple de operațiuni: transformarea in șir de caractere(String), sortarea lor, umplerea cu elemente predefinite, căutarea unui element.

Arrays.toString – transformă un tablou uni dimensional in String folosind metoda toString a fiecărui element

Arrays.sort – sortează tabloul în ordinea ascendentă bazată pe ordinea naturala. Obs există și posibilitatea de a se adăuga un Comparator și de a se defini o ordine particulară.

Arrays.binarySearch - căuta un element într-un tablou, tabloul trebuie să fie sortat în prealabil

Arrays.fill – umple un tablou cu același element. Obs este vorba de referință la elementul respectiv.

Arrays.copyOfRange

Task - 2

În pachetul com.fmi.lab2.arrays, folosind clasa ArraysMain, să se creeze un tablou de elemente bidimensional pentru care să se afișeze valorile fiecărei linii. (hint Arrays.toString)

Să se realizeze un tablou unidimensional de string-uri (folosiți orice cuvinte din limba română/engleză). Căutați un cuvânt folosind binarySearch. Refaceți căutarea după ce ați sortat tabloul. ☺

# Clase

“In the real world, you'll often find many individual objects all of the same kind. There may be thousands of other bicycles in existence, all of the same make and model. Each bicycle was built from the same set of blueprints and therefore contains the same components. In object-oriented terms, we say that your bicycle is an instance of the class of objects known as bicycles. *A class is the blueprint from which individual objects are created.*”

(<https://docs.oracle.com/javase/tutorial/java/concepts/class.html> )

## Modul de definire al unei clase

În cea mai simplă formă, o clasă *publică,* folosește numele fișierului în care se află.

**package** com.fmi.lab2.animals;  
  
*/\*\*  
 \* the most simple class  
 \*/***public class** UnknownAnimal {  
   
 *// fields and methods*}

Obs. Există și clase imbricate ( inner class)

Structura unei clase este de genul

Ex.

Pachet

**package** com.fmi.lab2.animals;

Importuri (opțional)

**public class** Cat {  
  
 **private boolean shortHair**;  
  
 **private** Integer **size**; *// possible values 1, 2, 3 => small, medium, big - enum in the future* Integer **propDefault**;  
  
 **public void** meow() {  
 System.***out***.println(**"cat - meow"**);  
 }  
}

Definiția clasei

O clasa aparține unui pachet (unul singur, există clase cu același nume in pachete diferite însă ele sunt considerate clase diferite)

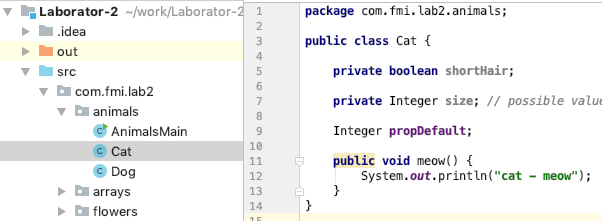
De obicei prima sau a doua linie in fișier este de forma

**package** com.fmi.lab2.animals;

reprezentând pachetul in care o clasă este declarată. Acest pachet trebuie sa corespunda cu structura de pachete în care se află clasa.

În exemplul de mai sus clasa Cat se află în pachetul

com.fmi.lab2.animals



## Importarea de pachete

**import** java.util.Arrays;

**import** java.util.\*;

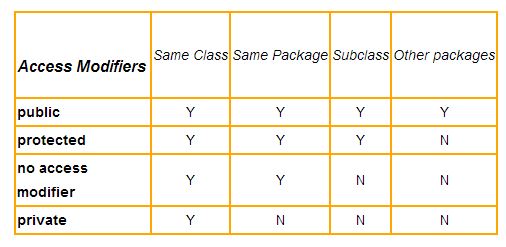
Se poate importa o singură clasă (ex. Arrays) sau toate clasele unui pachet prin folosirea \*.

Wildcard-ul \* ține loc doar de numele claselor nu și al pachetelor.

## Modificatorii de acces

Sunt patru categorii de modificatori:

* Default
* Public
* Private
* Protected



*Task - 3*

Se va crea un pachet com.fmi.lab2.animals, în cadrul lui se vor crea clasele Cat, Dog and AnimalsMain.

Clasa Cat va avea

3 câmpuri shortHair, size (una protected și una private, prima boolean a doua integer) și propDefault fără nici un modificator de acces si de tipul String

Din clasa AnimalsMain se vor accesa câmpurile disponibile, pentru cele ne-disponibile se va schimba tipul de modificator de acces.

In pachetul com.fmi.lab2 se va crea o clasa Main de unde se vor accesa câmpurile definite “default”.

*Task - 4*

Se va crea un pachet com.fmi.lab2.shapes, în cadrul lui se vor crea clasele Circle, Rectangle și Square.

Fiecare clasă va avea 4 câmpuri: name(String), dimension1(Integer), dimension2(Integer), secretField(String). Fiecare câmp va avea un tip de modificator de acces.

În pachetul com.fmi.lab2 se va realiza o clasă ShapesMain.

Din clasa ShapesMain se vor accesa câmpurile disponibile, pentru cele ne-disponibile se va schimba tipul de modificator de acces.