Programare Avansata pe Obiecte
Laborator 2

Pahontu Bogdan-Ionut

E-mail: pahontubogdan@gmail.com

Contents

1. Understanding arrays

1.1.Declaration / Initialization

- Three ways to declare an array:
  - `Integer[] arr1 = {22, 33, 44};`
  - `String arr2[] = {"one", "two", "three"};`
  - `String arrn3[] = new String[]{"first string", "second string"};`
    `String arrn4[] = new String[10]; // without initialization`

1.2. Accessing the elements

- Looping using index

Example:

```
Integer[] arr1 = {22, 33, 44};

// Looping by index
for (int i = 0; i < arr1.length; i++) {
    System.out.println("arr1["+i+"]: " + arr1[i]);
}
```

- Looping using foreach

```
String arr2[] = {"one", "two", "three"};

// Looping using foreach
for (String s : arr2) {
    System.out.println("arr: " + s);
}
```

1.3.Bi-dimensional arrays

Example:

```
int[][] biArr2 = { { 1, 2 }, { 3, 4 } };

for (int i = 0; i < 2; i++)
    for (int j = 0; j < 2; j++)
        System.out.println("arr[" + i + "][" + j + "] = "
                + biArr2[i][j]);
```

Task 1:

1. Create packages lab2 → tasks →task1.
2. Create a class with two methods:
    a. Public static void main method;
    b. One method that will print a table of integers that is given as parameter (the table can has various sizes);

Examples of tables:

```
Integer table1[][] = { { 1, 2 }, { 3, 4 } };
Integer table2 [][] = { { 1, 2 }, { 3, 4 }, {56,65,33,7} };
```

Task 2:

1. Create packages lab2 → tasks →task2.
2. Create a class with two methods:
    a. Public static void main method;
    b. One method that will check if a square matrix is symmetric or not;

A square matrix is said to be **symmetric matrix** if the transpose of the matrix is same as the given matrix

```
1 2 3

2 1 4  → The matrix is symmetric

3 4 3


3 5 8
```

```
3 4 7  → The matrix is not symmetric
8 5 3
```

1.4.Using Arrays class

- Arrays class is used in order to do the following tasks with an array in Java:
  - Fill an array with a particular value;
  - Sort an array;
  - Binary search in sorted array.
- The most common methods:
  - `Arrays.toString(arr);`
  - `Arrays.fill(array,value);`
  - `Arrays.sort(arr);`
  - `Arrays.binarySearch(array,"text");`

Task 3:

1. Create packages lab2 → tasks →task3.
2. Create a class with two methods:
   a. Public static void main method;
   b. One method that will find an item into an array of integers and will copy in another array all the elements from the index of found item to the end of the array.
      i. The method will have two parameters: the initial array and the item that need to be found;
      ii. Print the result;
      iii. Hint: Search for other functions in Arrays class that may help;

Example:

```
Integer arr[] = { 1, 5, 6, 9, 12 ,22 ,7 ,20 };
Integer item = 9;
```

```
Integer result[] = { 9, 12 ,22 ,7 ,20 };
```

2. Classes

2.1.Access modifiers

- **public** - The method can be called from any class.
- **private** - The method can only be called from within the same class.
- **protected** - The method can only be called from classes in the same package or subclasses.
- **( ) - Default (Package Private)** - The method can only be called from classes in the same package.

Visibility	Public	Protected	*Default*	Private
From the same class	Yes	Yes	Yes	Yes
From any class in the same package	Yes	Yes	Yes	No
From a subclass in the same package	Yes	Yes	Yes	No
From a subclass outside the same package	Yes	Yes, *through inheritance*	No	No
From any non-subclass class outside the package	Yes	No	No	No

Figure 1 - Access modifiers

2.2. Constructors

- Constructors are used when creating a new object.
- This process is called ***instantiation*** because it creates a new instance of the class.

- A constructor is called when we write **new** followed by the name of the class we want to instantiate.
- When Java sees the new keyword, it allocates memory for the new object. Java also looks for a constructor and calls it.
- A constructor is typically used to **initialize instance variables**.

```
public class Main {
    public static void main(String args[]){
        Person p1 = new Person("John", 23);
        p1.printPerson();

        Person p2 = new Person();
        p2.setName("Ian");
        p2.setAge(33);
        p2.printPerson();

    }
}
```

2.3. Default constructors

- Every class in Java has a constructor whether you code one or not.
- If you don't include any constructors in the class, Java will create one for you without any parameters. This Java-created constructor is called the **default constructor**.

2.4. Static methods and fields

- Static methods don't require an instance of the class. They are shared among all users of the class.
- You can think of statics as being a member of the single class object that exist independently of any instances of that class.

Type	Calling	Legal?	How?
Static method	Another static method or variable	Yes	Using the classname
Static method	An instance method or variable	No	
Instance method	A static method or variable	Yes	Using the classname or a reference variable
Instance method	Another instance method or variable	Yes	Using a reference variable

Figure 2 - Static methods and fields

Task 4:

1. Create packages lab2 → tasks →task4.
2. Create a class Shape that has one public static member named “counter” that has value equals to 5 and two other public members “name” and “color”;
3. Implement a constructor with two parameters in which the counter will be incremented by 1 and the fields “name” and “color” will be initialized
4. Implement a function that will print all the fields for a Shape object;
5. Create a class Main in which you will create 3 instances of Shape class;
6. Add all the instances into an array.
7. Loop through the array and print the Shapes info;
8. Print the value of “counter” using Shape.counter (class name, not instance) after instantiating the objects

2.5. Initialization blocks

- Order of initialization
    - If there is a superclass, initialize it first (we’ll cover this rule in the next chapter)

- Static variable declarations and static initializers in the order they appear in the file.
- Instance variable declarations and instance initializers in the order they appear in the file.
- The constructor.