JOBSHEET III
ARRAY OF OBJECTS

1.1 Learning Objectives

At the end of this session, students must be able to:

1. Understand and explain the use of Array of Objects
2. Understand the logic of implementing Array of Objects in Java
3. Implement Array of Object in Java

1.2 Create, insert, and display Array of Object

Times: 50 minutes

In this session, we will have a practice of creating array of object, insert the data, and display it.

1.2.1 Labs Activity

1. Create a new project with name **ArrayOfObjects**. Create the package with name **week3**
2. Create a **Rectangle** class:

```
public class Rectangle {
    public int length;
    public int width;
}
```

3. In main method in **ArrayOfObjects** class, create an array **Rectangle** and its length is 3

```
public class ArrayOfObjects {
    public static void main(String[] args) {
        Rectangle[] rectangleArray = new Rectangle[3];
    }
}
```

4. Then insert values for each the object's attributes.

```
rectangleArray[0] = new Rectangle();
rectangleArray[0].length = 110;
rectangleArray[0].width = 30;

rectangleArray[1] = new Rectangle();
rectangleArray[1].length = 80;
rectangleArray[1].width = 40;

rectangleArray[2] = new Rectangle();
rectangleArray[2].length = 100;
rectangleArray[2].width = 20;
```

5. Print all the attributes object from ppArray as follows.

```
System.out.println("First rectangle, width: " + rectangleArray[0].width + ", length: " + rectangleArray[0].length);
System.out.println("Second rectangle, width: " + rectangleArray[1].width + ", length: " + rectangleArray[1].length);
System.out.println("Third rectangle, width: " + rectangleArray[2].width + ", length: " + rectangleArray[2].length);
```

1.2.2 Result

Compile the code and see the result if it matches with following image.

```
run:
First rectangle, width: 30, length: 110
Second rectangle, width: 40, length: 80
Third rectangle, width: 20, length: 100
BUILD SUCCESSFUL (total time: 0 seconds)
```

1.2.3 Questions

1. Based on part 1.2, does the class that are going to be used as an array of object must have attributes and methods? Please explain!
2. Does class **Rectangle** have constructor? If not, why we instantiate the object as follows?

```
rectangleArray[1] = new Rectangle();
```

3. What's the meaning of this line of code?

```
Rectangle[] rectangleArray = new Rectangle[3];
```

4. What's the meaning of these lines of code?

```
rectangleArray[1] = new Rectangle();
rectangleArray[1].length = 80;
rectangleArray[1].width = 40;
```

5. Why **ArrayOfObject** class and **Rectangle** class should be separated?

1.3 Input data into Array of Objects using Loops

Times: 50 minutes

In this part, we will update the program resulted in 1.2, so that it could receive user input and use loops to assign values of each attribute of rectangles in **ppArray**

1.3.1 Steps

1. Import scanner in **ArrayOfObjects** class below the package declaration.

```
package week3;
import java.util.Scanner;
```

2. In part 1.2 at 4th steps, change the code as follows, this allows the Scanner object to be included in loops to receive input and assign user input values to the attributes.

```
        Rectangle[] rectangleArray = new Rectangle[3];
        Scanner sc = new Scanner(System.in);

        // Assign the values for each attribute in objects
        for (int i = 0; i < 3; i++) {
            rectangleArray[i] = new Rectangle();
            System.out.println("Rectangle " + i);

            System.out.print("Input length : ");
            rectangleArray[i].length = sc.nextInt();

            System.out.print("Input width : ");
            rectangleArray[i].width = sc.nextInt();;
        }
```

3. In part 1.2 at 5th steps. Change the code as follows. This time, we will use loop to access the element of **ppArray** and print it on the console.

```
// Display the result in console
  for (int i = 0; i < 10; i++) {
      System.out.println("Rectangle " + i);
      System.out.println("width: " + rectangleArray[0].width + ", length: " + rectangleArray[0].length);
  }
```

4. Observe the result!

1.3.2 Result

Run the program and see if it matches with following result:

```
run:
Rectangle 0
Input length : 5
Input width : 6
Rectangle 1
Input length : 5
Input width : 3
Rectangle 2
Input length : 4
Input width : 8
Rectangle 0
width: 6, length: 5
Rectangle 1
width: 6, length: 5
Rectangle 2
width: 6, length: 5
Rectangle 3
width: 6, length: 5
Rectangle 4
width: 6, length: 5
Rectangle 5
width: 6, length: 5
Rectangle 6
width: 6, length: 5
Rectangle 7
width: 6, length: 5
Rectangle 8
width: 6, length: 5
Rectangle 9
width: 6, length: 5
BUILD SUCCESSFUL (total time: 10 seconds)
```

1.3.3 Questions

1. Does array of object can be implemented on 2D array?
2. If yes, then please give an example. Otherwise, please explain.
3. There is a **Square** class that has an attribute **side** with integer as its data type. There will be an error when we run this code, why?

```
Square[] squareArray = new Square[100];
squareArray[5].side = 20;
```

4. Modify the code on part 1.3 so that the length of the array will be defined from user input.
5. Can we duplicate the instantiation process in array of objects? For example, we assign the object in **ppArray[i]** and **ppArray[0]**, the instantiation process of **ppArray[0]** will be done twice. What's the effect of this?

1.4 Mathematical operation in array of object's attribute

Times: 50 minutes

1.4.1 Steps

1. Still in package **week3**
2. Create a class named **Blocks**

```
public class Blocks {
    public int width, length, height;

    public Blocks(int p, int l, int t){
        length = p;
        width = l;
        height = t;
    }

    public int countVolume(){
        return length*width*height;
    }
}
```

3. In **main** function in class **ArrayBlocks,** instantiate array of **Blocks** that has size of 3

```
public class ArrayBlocks {
    public static void main(String[] args) {
        Blocks[] blArray = new Blocks[3];
    }
}
```

4. Then add these following codes to insert the value of **blArray** using its constructor.

```
    blArray[0] = new Blocks(100, 30, 12);
    blArray[1] = new Blocks(120, 40, 15);
    blArray[2] = new Blocks(210, 50, 25);
```

5. Display the volume of all blocks by calling the method **countVolume**() in loop as follows.

```
for (int i = 0; i < 3; i++) {
    System.out.println("Volume blocks - " + i + " : " + blArray[i].countVolume());
}
```

6. Run and observe the result.

1.4.2 Result

Run the program and see if it matches with following result:

```
run:
Volume blocks - 0 : 36000
Volume blocks - 1 : 72000
Volume blocks - 2 : 262500
BUILD SUCCESSFUL (total time: 0 seconds)
```

1.4.3 Questions

1. Can we have more than one constructor in one class? Please explain.
2. Create a **Triangle** class as follows.

```
public class Triangle{
        public int base;
        public int height;
}
```

   Add another constructor in this class that has parameter **int a, int t**. These represents its base and height.

3. Add method **countArea()** and **countPerimeter()** in class **Triangle**
4. In main function, instantiate array of **Triangle** objects. Assign the attributes values as follows:

0th trArray	base: 10, height: 4
1st trArray	base: 20, height: 10
2nd trArray	base: 15, height: 6
3rd trArray	base: 25, height: 10

5. Display the result of area and perimeter for each triangle by calling the method **countArea()** and **countPerimeter()**

1.5 Assignments

1. Create a program to display information about lecturers. The program should accept input for all relevant lecturer details and display them on the screen. The program consists of a **Lecturer<NoPresensi>** class with the following attributes/properties: **id (string), name (string), gender (Boolean), and age (int**). The class should include the following constructor method:

```
public Lecturer(String id, String name, Boolean gender, int age) {
        // Implementation here
}
```

   Next, create a **LecturerDemo<NoPresensi>** class to handle input and display multiple lecturer data, using the following rules:

- Use a **FOR** loop to create an array of objects.
- Use a **FOREACH** loop to display the data on the screen.

2. Add a new class called **LecturerData< NoPresensi >** with the following methods:
   a. `showAllLecturerData(Lecturer[] lecturerArray)` – Displays all lecturer data.
   b. `countLecturerByGender(Lecturer[] lecturerArray)` – Displays the number of lecturers based on gender (Male/Female).
   c. `averageLecturerAgeByGender(Lecturer[] lecturerArray)` – Displays the average age of lecturers based on gender (Male/Female).
   d. `showOldestLecturerInfo(Lecturer[] lecturerArray)` – Displays the information of the oldest lecturer.
   e. `showYoungestLecturerInfo(Lecturer[] lecturerArray)` – Displays the information of the youngest lecturer.

   All these methods must be callable and testable from the **LecturerDemo** class.