Programming Fundamentals

Arrays and Array lists

Week 6| Iresh Bandara

ROBERT GORDON UNIVERSITY ABERDEEN
TEF Gold
INFORMATICS INSTITUTE OF TECHNOLOGY

Learning Outcomes

- On completion of this lecture, students are expected to be able to:
  - Identify the need for arrays in a java program.
  - Construct java applications using Arrays and Array-Lists.
  - Develop java programs with 2D arrays.

What is an Array?

- An array is a group of variables of the same data type and referred to by a common name.
- An array is a block of consecutive memory locations that hold values of the same data type.

Why Do We Need Arrays?

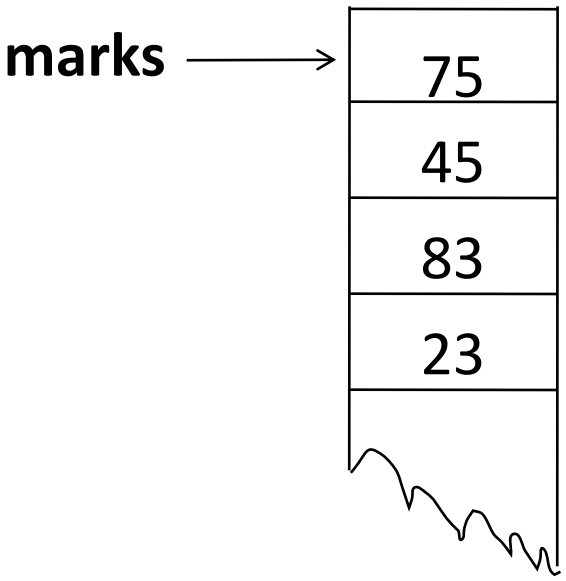
- You might come across a situation where you need to store similar type of values for a large number of data items.

Example:

- To store the marks of all the students at a university, you need to declare thousands of variables. In addition, each variable name needs to be unique. To avoid such situations, you can use arrays.

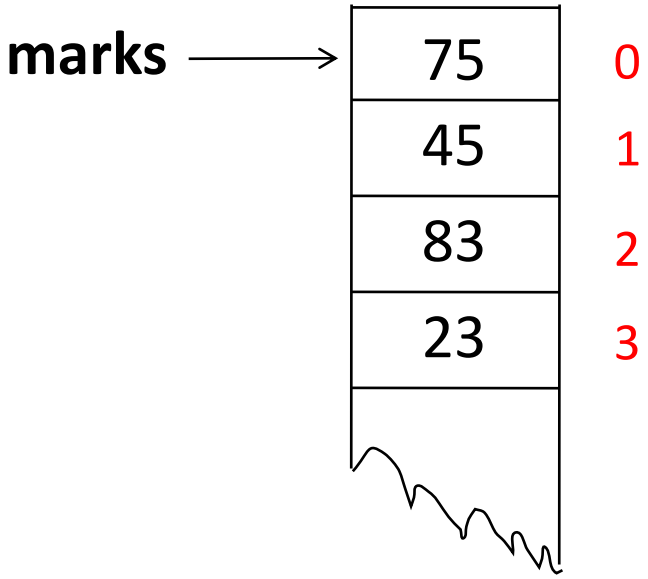
Example

- To store the marks of 1000 students, you can declare an array, call **marks**, of size 1000 and can store the marks of as many students.



More about Arrays

- Individual locations are called array's elements.
- Each location is identified by a number, called index or subscript.
- An index can have any integer value **from 0 to array's length – 1**.



Array Index

- Specific array elements are referred to by using array's name and the element's index.
  - index is written within square brackets following array's name)

  For example: marks[2]
- Indices start from 0; therefore the 1st element of an array a is referred to as marks[0] and the *n*-th element as marks[n−1].

Array Length

- In Java, the length of an array is **fixed** at the time of its creation.
- Java interpreter checks the values of indices at run time and throws ArrayIndexOutOfBoundsException if an index is negative or if it is greater than the length of the array − 1.

Arrays as Objects

- In Java, an array is an object.
- An array can hold elements of any data type; primitive or any class type.
- If the type of its elements are anyType, then the type of the array is anyType[ ].
- Recall: String args[]

How to Create an Array?

- Array declaration:

```
anyType [] arrName;
```

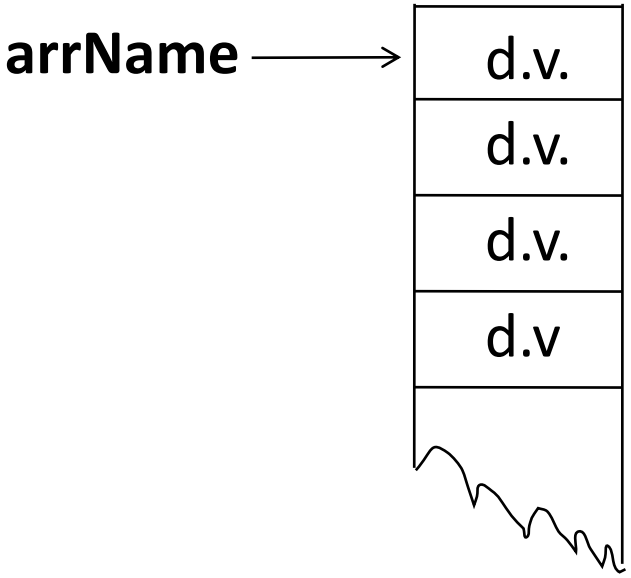
- Like other objects, the declaration creates only a reference, initially set to null. An array must be created before it can be used.
- One way to create an array:

```
arrName = new anyType [length];
```

What really happens…

- When an array is created, space is allocated to hold its elements and the elements get the default values.

```
anyType [] arrName = new anyType [length];
```



Some Examples

```
int [] scores = new int[10];

String [] words = new String[100];

double [] gasPrices;
gasPrices = new double[12];

Rectangle [] recList = new Rectangle[5];
```

length 10, all elements are set to 0

Length 100 , all elements are set to null

length 5, all elements are set to null

Create an Array: Method 2

- The other way to create an array is:

```
anyType [] arrName = {values separated
                      by commas};
```

- Here array is declared and initialized in one statement.
- Examples:

```
String [ ] cities = {"Atlanta","Boston“}

double [ ] gasPrices = {3.05,3.17, 3.59};
```

How to get the Array Length?

- The length of an array is determined when that array is created.
- The length is either given explicitly or comes from the length of the {…} initialization list.
- The length of an array **arrName** is referred to in the code as **arrName.length**
- length is a public property (not a method) in an array object.

Initialization of Array Elements

- Unless specific values are given in a {…} list, all the elements are initialized to the default value:

  0 for numbers,

  false for booleans

  null for objects or Strings.

Traverse through an Array

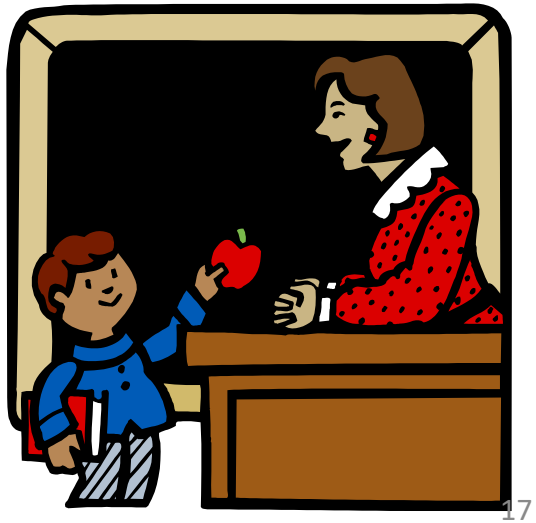
- An array can be traversed through using any loop: (for/while/do-while)

```
double [] gasPrices = {3.05,3.17,3.59};

for(int c=0; c<gasPrice.length; c++)
{
      ...

}
```

Exercise: Ms. White's Test Score Analyzer

- Problem:

  Mr. White needs to analyze students' chemistry test performance. He needs a program to display a name and let him enter the test score. Then it should compute and display the average. Finally, it should give a report with names, scores, and deviation from the average.



Step to be followed

1. Create a class to called ArrayTest, with the main method.
2. Define **`students`** array to hold names, array **`scores`** to hold test scores.
3. For each student in array
   a) display name & prompt.
   b) read double, store in array **`scores`**.
4. Compute **`overall average`**, display it.
5. For each student in array,
   a) Display name, test score, difference between that score and **`overall average`**

Output

- Display the name in a prompt
- Read the score
- Compute average
- Print summary, including deviation from mean

```
Test Analysis – enter scores:
Aardvark, James     : 92
Biffkirk, Sue       : 79
Crouse, Nancy       : 95
. . .
Average = 87.39

Summary …
```

Passing Arrays to Methods

- When an array is passed to a method, only its reference is passed.
- A copy of the array is NOT created in the method and the elements of the original array are accessible in the method's code.
- Example:

```
public void test_array_pass(int []a)
{

...

}
```

Task One

- Define a method that returns the index of the smallest element in an array of real numbers that is pass as a parameter.
- Write necessary coding to call the above method (from the main method of the same class).

Returning Arrays from Methods

- The return type of a method that returns an array with anyType elements is designated as anyType[].
- Example:

```
int[] test_return_array()
{
      int[] myArray = new int[3];
      return myArray;
}
```

Task Two

- Define a method that separates a given sentence (as a parameter) into words by one space.
- Note: The method should return the separated words.

Array of Objects

- Array elements are not limited to primitive data types.
- Arrays can also hold objects of any given class.
- Elements of an Object array are initially set to null.
- Once the objects are assigned, the array holds references to the objects.
- Each object-type element must be initialized before it is used.

Storing Objects in an Array

- Example:

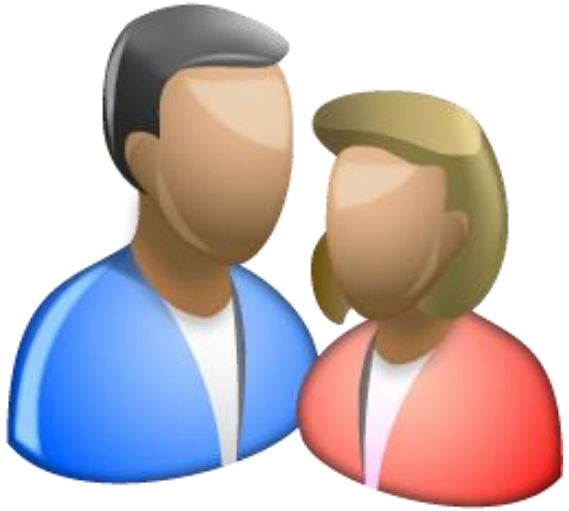
Array is created; all three elements are set to null

```
Rectangle [] design = new Rectangle[3];
...
design[0] = new Rectangle(5,2,“red”);
design[1] = new Rectangle(10,3,“blue”);
design[2] = new Rectangle(20,10,“yellow”);
```

Now all three elements are initialized

Task Three

- Define a **Person** class that maintain the name, gender and age information.
- Declare an constructor to initialize the data.
- Declare methods to get name, gender and age.



Task Three cont…

- Create a class call **People** to maintain a list of Person objects.
- Declare an array to hold Person objects.
- Write a constructor to create the Person array to the size given as a parameter.
- Declare a method to set values to a Person object in the array.
- Declare another method to find the

  name and the gender of the oldest

  Person in the group.

Advantages of Using an Array

1. You can refer to a large number of elements by just specifying the index number and the array name.
2. Arrays make it easy to do calculations in a loop.

Two-Dimensional Array (2-D)

- 2-D arrays are used to represent tables, matrices, game boards, etc.
- One way to create a 2-D array is:

```
type arrName[][] = new type[rows][cols];
```

- Example:

```
//2-D array of char with 5 rows & 7 cols:

char[][] letterGrid = new char[5][7];
```

Create a 2-D Array: Method 2

- Like 1-D arrays there is another way to create 2-D arrays:
- Example:

```
//2-D array of int with 2 rows and 3 cols:

int [][] sample = {{ 1, 2, 3 },
                   { 4, 5, 6 }};
```

Elements in a 2-D Array

- An element of a 2-D array is addressed using a pair of indices, “row” and “column.”

```
arrName[r][c]
```

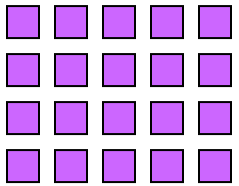
- For example:

```
letterGrid[1][2] = 'x';
```

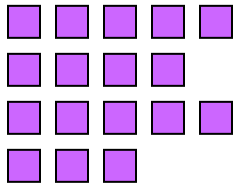
Types of 2-D Array

- Java allows “ragged” arrays, in which different rows have different lengths.

Rectangular array:



“Ragged” array:



More about 2-D Arrays

- In a two-dimensional array,

1. You need to allocate memory for only the first dimension.
2. You can allocate the remaining dimensions separately.
3. When you allocate memory to the second dimension, you can also allocate different number to each dimension.

```
int ragArr[][] = new int[3][];
ragArr[0] = new int[1];
ragArr[1] = new int[4];
```

Getting the 2-D Array Length

- In Java, a 2-D array is basically a 1-D array of 1-D arrays. Each row is stored in a separate block of consecutive memory locations.
- If **`sample`** is a 2-D array;
  - **`sample.length`** is the number of rows.
  - **`sample[n-1].length`** is the length of the n^{th} row.

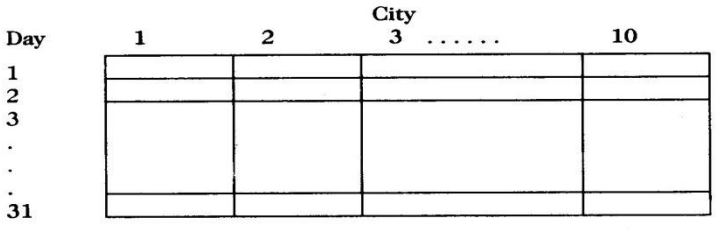
Traverse through a 2-D Array

- A 2-D array can be traversed using nested loops:

```
int [][] sample = {{ 1, 2, 3 },
                   { 4, 5, 6 }};

 for(int r=0; r<sample.length; r++) {
    for(int c=0; c<sample[r].length; c++) {
          ...   //process sample[r][c]

    }
 }
```

Exercise: Weather Analyzer

- The daily maximum temperatures recorded in 10 cities during the month of January (for all 31 days)
- Write a program to read the table elements into a two-dimensional array temperature, and to find the city and day corresponding to
  - the highest temperature and
  - the lowest temperature.



ArrayLists

- ArrayLists are commonly used instead of arrays, because they expand automatically when new data is added to them.
- ArrayLists can hold only Objects and not primitive types (eg: int).

How to Create a ArrayList

- You must import either import java.util.ArrayList; or import java.util.*;

- Create a ArrayList with default initial size

```
ArrayList v = new ArrayList();
```

- Create a ArrayList with an initial size

```
ArrayList v = new ArrayList(300);
```

Common ArrayList Methods

Method	Description
v.add(o)	Adds Object o to ArrayList v
v.add(i,o)	Inserts Object o at index i, shifting elements up
v.clear()	removes all elements from ArrayList v
v.firstElement()	Returns the first element.
v.get(i)	Returns the object at int index i.
v.lastElement()	Returns the last element.
v.remove(i)	Removes the element at position i, and shifts all following elements down.
v.set(i,o)	Sets the element at index i to o.
v.size()	Returns the number of elements in ArrayList v.

Advantages of ArrayList over Arrays

1. It is convenient to use ArrayList to store objects.
2. An ArrayList can be used to store a list of objects that may vary in size.
3. We can add and delete objects from the list as and when required.

Disadvantages of ArrayList

- A major constraint in ArrayList is that simple data types cannot be directly stored; It can be only stored objects.
- To convert simple types to objects use the *wrapper classes.*

Summary

- An array is a group of variables of the same data type and referred to by a common name.
- Specific array elements are referred to by using array's name and the element's index.
- In Java, the length of an array is fixed at the time of its creation.
- When an array is passed to a method, only its reference is passed.
- 2-D arrays are used to represent tables, matrices, game boards, etc.
- ArrayLists are commonly used instead of arrays, because they expand automatically when new data is added to them.

test analyzer

```
import java.util.Scanner;

public class ArrayTest {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        // Step 1: Declare and initialize the students and scores arrays
        String[] students = {"Alice", "Bob", "Charlie", "David", "Emma"};
        double[] scores = new double[students.length];

        // Step 2: For each student in array
        for (int i = 0; i < students.length; i++) {
            // Step 3a: Display name & prompt.
            System.out.print("Enter the test score for " + students[i] + ": ");

            // Step 3b: Read double, store in array scores.
            scores[i] = input.nextDouble();
        }

        // Step 4: Compute overall average, display it.
        double sum = 0;
        for (double score : scores) {
            sum += score;
        }
        double average = sum / scores.length;
        System.out.println("The overall average is " + average);

        // Step 5: For each student in array
        for (int i = 0; i < students.length; i++) {
            // Step 5a: Display name, test score, difference between that score and overall average
            System.out.print(students[i] + " scored " + scores[i] + ", which is ");
            System.out.printf("%.2f", Math.abs(scores[i] - average));
            System.out.println(" points away from the overall average.");
        }
    }
}
```

Thank you

ROBERT GORDON UNIVERSITY ABERDEEN

TEF Gold

INFORMATICS INSTITUTE OF TECHNOLOGY