Object Oriented Programming – SCJ2153

Array of Primitives

Associate Prof. Dr. Norazah Yusof

Introduction to Array

- Array is an object and is used to store a list of data of similar types.
- Array element can be of type:
  - primitive data types (such as int, double, float or char) or
  - objects (of type class).
- The size of an array (or the array length) must be declared before it can be used.
- Once an array is created, its size becomes permanent and can be obtained through array constant, length.
- The JVM stores the array in an area of memory called heap

Declare and create an array

- Before an array can be used, it must first be declared
- Declare and create array is similar to declare and create any other type of object.
- Declare and create array can be done in two steps:
  - Step 1: declare an array reference
  - Step 2: create an array
- Declare and create array can be done in one step.
- Declare, create, and initialize array can be done using the shorthand notation

Declare and create an array in two steps

Format of Step 1: Syntax to declare an array reference

```
datatype[] arrayRefVar;
```

or

```
datatype arrayRefVar[];  // This style is allowed,
                         // but not preferred
```

Format of Step 2: Syntax to create an array

```
arrayRefVar = new datatype[arraySize];
```

Example

```
double[] myMarks;
myMarks = new double[10];
```

Declare and create an array in one step

- Syntax to declare and create the array:

```
datatype[] arrayRefVar = new datatype[arraySize];
```

or

```
datatype arrayRefVar[] = new datatype[arraySize];
```

- Example

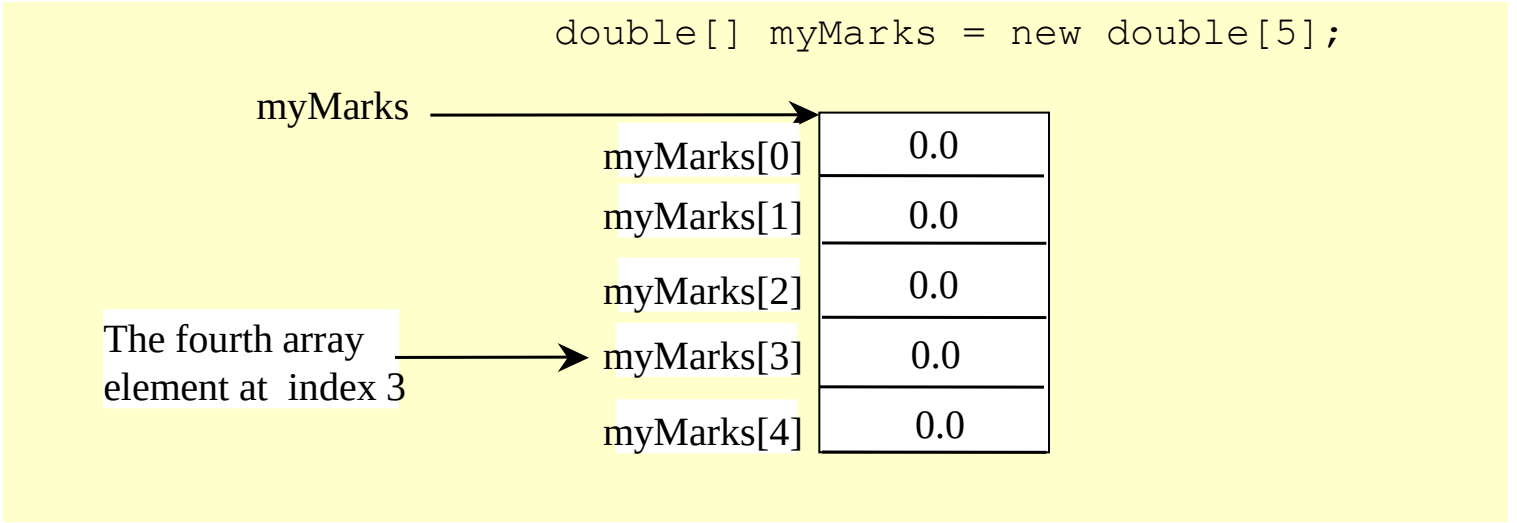
```
double[] myMarks = new double[5];
```

or

```
double myMarks[] = new double[5];
```

Array's Subscript/Index

- Array's elements numbered beginning with zero
- Can legally use any subscript from 0 through `myMarks.length-1`
- Subscript/index:
  - is an integer contained within square brackets
  - indicates one of array's variables or elements



The Length of an Array

- Once an array is created, its size is fixed. It cannot be changed.
- The size can be determined by the constant, `length` with the following syntax:

```
arrayRefVar.length
```

- For example,

```
System.out.println(myMarks.length);
```

returns 5

Accessing Array Elements

- The array elements are accessed through the index, known as indexed variable
- Syntax:

```
arrayRefVar[index];
```

- An indexed variable can be used in the same way as a regular variable.
- For example, the value of the array element may be assigned as follows:

```
myMarks[0] =  5.6;
myMarks[1] =  myMarks[0] - 1.1;
Scanner inp = new Scanner(System.in);
myMarks[2] = inp.nextDouble();
```

Accessing Array Elements

- The array elements can be output using the `println` method.

```
System.out.println (myMarks[0]);
System.out.println (myMarks[1]);
System.out.println (myMarks[2]);
```

Declaring, creating, initializing Using the Shorthand Notation

- The syntax to declare and create the array:

```
datatype[] arrayRefVar = {arrylist1, arrylist2,…};
```

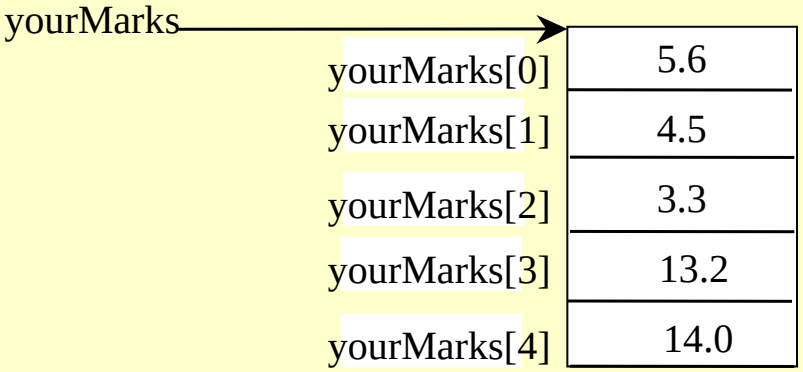
- Example

```
double[] yourMarks = {5.6, 4.5, 3.3, 13.2, 14.0};
```

Array's Subscript/Index

- The size of the array name `yourMarks` is 5.
- The array's elements numbered beginning from 0 to 4.
- The content of the array will automatically filled to each element.

```
double[] yourMarks = new double[5];
```



Using for loop

- Perform loops that vary loop control variable
  - Start at 0
  - End at one less than size of array
- Using the `length` field to control the number of elements in array.

```
1 for(int i = 0; i < myMarks.length; i++)
2    myMarks[i] += 3;
3
```

Using enhanced for loop

- Enhanced for loop (also known as for each loop) for accessing each of the array element.
- The syntax:

```
1 for (dataType elementVariable: array) {
2   // statement
3 }
```

- Example:

```
1 for(double val : myMarks)
2       System.out.println(val);
3
```

enhanced for loop (for each loop)

- Both loops are placed side by side

Conventional for loop	*Enhanced for loop*

double sample [] = {7.8,4.5,1.2,9.9};
for (int i=0; i<sample.length;i++){
System.out.println("Elements:"+ sample[i]);
}
	double sample [] = {7.8,4.5,1.2,9.9};
	for(double val : sample){
	System.out.println (val);
	}

enhanced for loop (cont.)

- One drawback of enhanced-for loop is that cannot be used if we have to add an index variable in the program code and need to increment it each time through the loop.