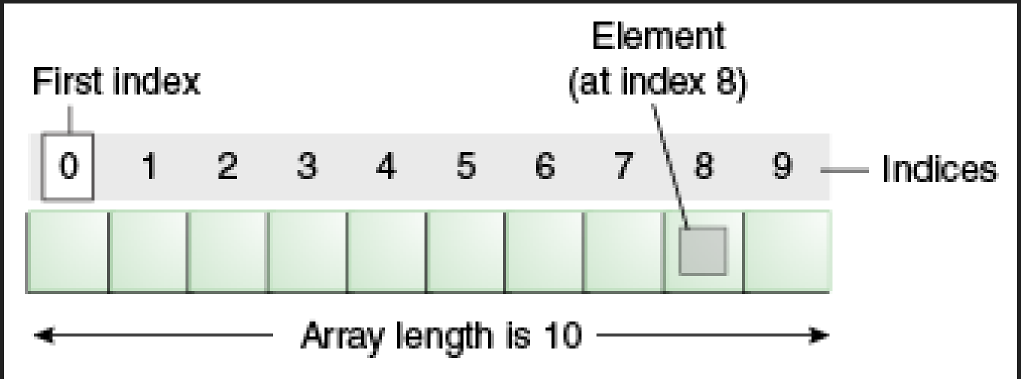
Java Array

CSX3002/ITX2001 Object-Oriented Concepts and Programming
CS4402 Selected Topic in Object-Oriented Concepts
IT2371 Object-Oriented Programming

Java Array

- A container objet that holds a fixed number of values of a single type.
- The length of an array is established when it is created.
- After array creation, its length is fixed (static memory allocation).



Declaring Array Variables(1)

Syntax

```
dataType[] arrayRefVar;
or
dataType arrayRefVar[];
```

Declaring array reference variable

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

Creating array

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

Declaration and creating array can be combined into one statement.

Declaring Array Variables(2)

Example#1

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

or

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

Example#2

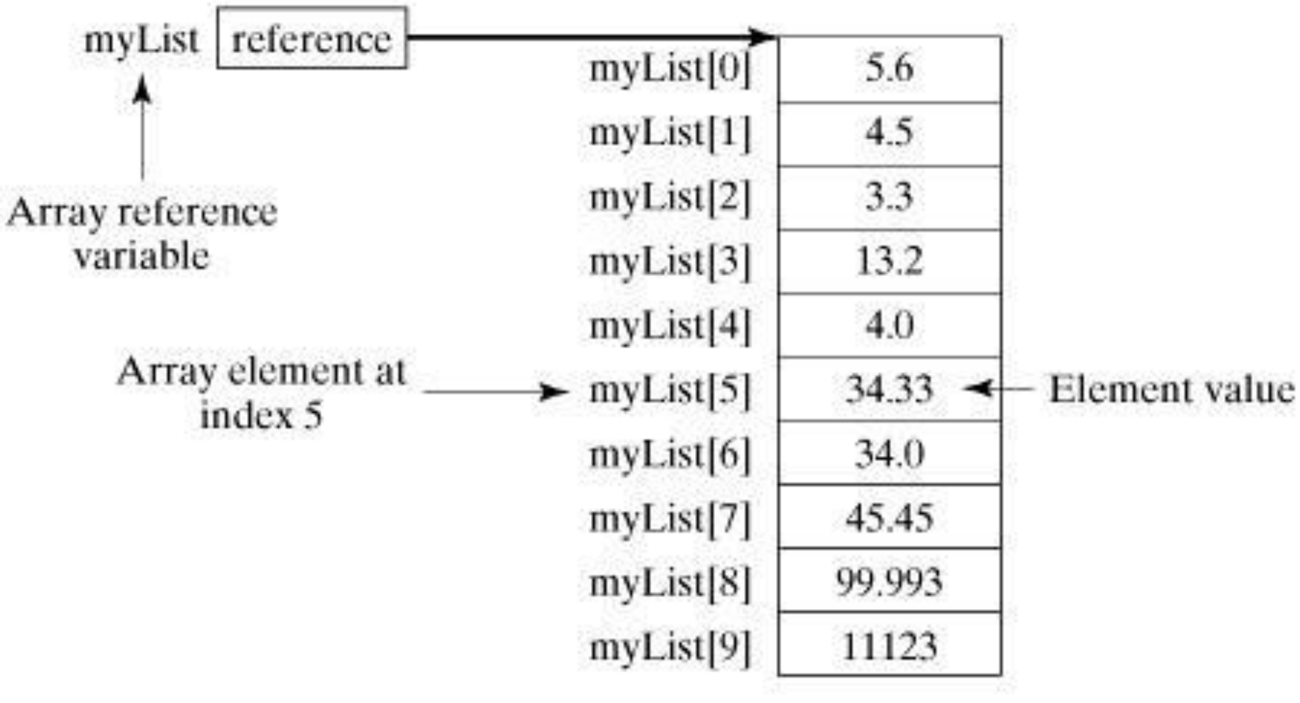
```
String[] arrOfString = {“ABC”,”DAD”,”FAT”,”ART”};
```

Example#3

```
boolean[] arrOfBoolean = new boolean[4];
```

Declaring Array Variables(3)

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



myList[0]	5.6
myList[1]	4.5
myList[2]	3.3
myList[3]	13.2
myList[4]	4.0
myList[5]	34.33
myList[6]	34.0
myList[7]	45.45
myList[8]	99.993
myList[9]	11123

Processing Arrays

- for or for..each loop oftenly used in processing array elements
- Size of an array is known via arrayRefVar.length

```
double[] arrayOfDouble = new double[4];
arrayOfDouble[0] = 2.5;
arrayOfDouble[1] = 1.7;
arrayOfDouble[2] = 0.5;
arrayOfDouble[3] = 6.75;

for (int i = 0; i < arrayOfDouble.length; i++)
   System.out.println(arrayOfDouble[i]);

for (double eachE : arrayOfDouble)
   System.out.println(eachE);
```

Passing Arrays to Methods(1)

- Passing an array to a method as an argument, actually the copy of the reference is passed
- Only arrayRefVar(no bracket,[ ]) is passed as argument

```
public static double max(double[] arr){
  double highest = arr[0];
  for (int i=1; i<arr.length; i++){
     if(arr[i] > highest)
         highest = arr[i];
   }
   return highest;
}
```

Passing Arrays to Methods(2)

```
// call a method by passing arrayRefVar as argument
System.out.println(max(myList));
```

```
// call a method by passing anonymous array as argument
double biggest = max(new double[]{4.5, 7, 9.23,-5});
```

Caution!

- Because the copy of the reference is passed as an argument, the arrayRefVar used in the method will refer to the same content of an array.
- If you make any change of an array in the method, it will effect the original array as well.

```
public static void swapEnd(int[] arr){
    int temp = arr[0];
    arr[0] = arr[arr.length-1];
    arr[arr.length-1] = temp;
}
```

ArrayList

- ArrayList class provides dynamic array for storing elements
- It is like an array but there is no size limit.

```
import java.util.*;
public class ArrayListExample{
   public static void main(String args[]){
       ArrayList<String> list=new ArrayList<String>();
       //Creating arraylist
       list.add("Mango");//Adding object in arraylist
       list.add("Apple");
       list.add("Banana");
       list.add("Grapes");
       //Traversing list through for-each loop
       for(String fruit:list)
           System.out.println(fruit);
   }
}
```

Reference

- Java Tutorials https://docs.oracle.com/javase/tutorial/java/nutsandbolts/index.html
- https://www.javatpoint.com/java-tutorial