Computer Programming Using JAVA
COP 2800 - Fall 2016

Lecture 12: Java Array Cont.

By Ahmed E. Khaled
Recap to previous lectures!

- What is a Java Array?
Lecture 11 : Java Arrays

- 2D arrays.
- More examples.
Suppose we need to define an array where each item points to another array, this is a multi-D array.

```java
int[][][] A = new int[3][4];
```

This statement creates a new two-dimensional array object and sets A to point to the newly created object.
If you create an array `A = new int[3][4]`, you should think of it as a "matrix" with 3 rows and 4 columns.

But in reality, `A` holds a reference to an array of 3 items, where each item is a reference to an array of 4 `ints`.

A: 

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>12</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>-3</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>-5</td>
<td>-2</td>
<td>2</td>
<td>-9</td>
</tr>
</tbody>
</table>

(4) 

1  
0  
12  
-1

(3) 

(4) 

7  
-3  
2  
5
• The notation `int[3][4]` indicates that there are 3 arrays-of-ints in the array A, and that there are 4 ints in each array-of-ints.

• Think of it as a grid or matrix, with 3 rows and 4 columns.

```
int[][] array = {
{1, 2, 3},
{4, 5, 6},
{7, 8, 9},
{10, 11, 12}
};
```
int[][] sourceArray = new int[2][3];

for(int i=0 ; i < 2 ; i++){
    for(int j=0 ; j < 3 ; j++){
        sourceArray[i][j] = (int) (Math.random() * 10);
    }
}

for(int i=0 ; i < 2 ; i++){
    for(int j=0 ; j < 3 ; j++){
        System.out.print(sourceArray[i][j] + " ");
    }
    System.out.println();
}
More examples !
public static boolean isValid(String cardNum) {

//write your program here, that check the validity of credit card.
//deal with the card number as a string, not as number.
//valid card number is of length 8.
//the valid card ends with 83 and starts with 925.

}
... one possible solution.

```java
public static boolean isValid(String cardNum) {

    if((cardNum.length() == 8) &&
    (cardNum.startsWith("925")) &&
    (cardNum.endsWith("83"))){
        return true;
    }

    return false;
}
```
Multiply elements of an array by 2

```java
public static int[] MulTwo(int[] myarray) {

    //write your program here, that multiply elements of the array by two and return.

}
```
... one possible solution.

```java
public static int[] MulTwo(int[] myarray) {

    for (int i = 0; i < myarray.length; i++) {
        myarray[i] = myarray[i] * 2;
    }

    return myarray;
}
```
public static int[] MulTarget(int[] myarray, int target) {

    //write your program here, that multiply elements of the array by the target value and return.

}
... one possible solution.

```java
public static int[] MulTarget(int[] myarray, int target) {

    for (int i = 0; i < myarray.length; i++) {
        myarray[i] = myarray[i] * target;
    }

    return myarray;
}
```
Find minimum value of array’s elements

```java
public static int getMinValue(int[] myarray) {

    //write your program here, that returns the minimum value.

}
```
... one possible solution.

```java
public static int getMinValue(int[] myarray) {
    int minValue = myarray[0];
    for (int i = 1; i < myarray.length; i++) {
        if (myarray[i] < minValue) {
            minValue = myarray[i];
        }
    }
    return minValue;
}
```
Find average value of array’s elements

public static int getAvgValue(int[] myarray) {

    //write your program here, that returns the average value.
    //Note: average of a set of numbers is:
    // (the summation) / (number of these elements)
public static int getAvgValue(int[] myarray) {
    double sum = 0;
    double avg = 0;

    for (int i = 0; i < myarray.length; i++) {
        sum = sum + myarray[i];
    }
    avg = sum / myarray.length;
    return avg;
}
Find number of elements above average in an array

public static int countAboveAvg (int[] myarray) {

    //write your program here, that returns the number of elements with values above the average.

}
... one possible solution.

class Solution {
    public static int countAboveAvg (int[] myarray) {
        double sum = 0;
        double avg = 0;
        int counter = 0;
        for (int i = 0; i < myarray.length; i++) {
            sum = sum + myarray[i];
        }
        avg = sum / myarray.length;
        for (int i = 0; i < myarray.length; i++) {
            if (myarray[i] > avg) {
                counter = counter + 1;
            }
        }
        return counter;
    }
}
Add two arrays and return the result.

```java
public static int[] addArrays(int[] first, int[] second) {

    //write your program here, that returns the result of adding two arrays.
    //assume both of same length

}
... one possible solution.

```java
public static int[] addArrays(int[] first, int[] secound) {
    int[] result = new int[first.length];
    for (int i = 0; i < first.length; i++) {
        result[i] = first[i] + secound[i];
    }
    return result;
}
```
Reverse an array and return the new array.

```java
public static int[] reverse(int[] mylist) {
    //create new array that holds the reverse order of the mylist
    //return the new array
}
```
... one possible solution.

```java
public static int[] reverse(int[] mylist) {
    int[] newList = new int[mylist.length];

    for (int i = 0; i < mylist.length; i++)
    {
        newList[i] = mylist[mylist.length - i - 1];
    }

    return newList;
}
```