Introduction to Arrays

CSC 1051 – Data Structures and Algorithms I
Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website:
www.csc.villanova.edu/~map/1051/

Some slides in this presentation are adapted from the slides accompanying Java Software Solutions by Lewis & Loftus
### Arrays

- An array is an ordered list of values:

```
<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>7.9</td>
<td>8.7</td>
<td>9.4</td>
<td>8.2</td>
<td>6.7</td>
<td>9.8</td>
<td>8.7</td>
<td>8.1</td>
<td>7.4</td>
<td>9.1</td>
</tr>
</tbody>
</table>
```

This array holds 10 values of type `double` that are indexed from 0 to 9.
Arrays - Declaration

**Declaration:**

The entire array has a single name

```
double[] scores;
```

0 1 2 3 4 5 6 7 ...

CSC 1051 M.A. Papalaskari, Villanova University
Arrays - Instantiation

**Declaration:**
The entire array has a single name.

**Instantiation:**
```java
double[] scores = new double[10];
```

**Element type:**
- `scores` (array name)
- `scores[2]` (array element)
- `scores.length` (size of array)

CSC 1051 M.A. Papalaskari, Villanova University
Arrays - Initialization

**Declaration:**
The entire array has a single name

**Instantiation:**
```java
double[] scores = new double[10];
```

**Initialization:**
```java
scores[0] = 7.9;
scores[1] = 8.7;
scores[2] = 9.4;
scores[3] = 8.2;
scores[4] = 6.7;
scores[5] = 9.8;
scores[6] = 8.7;
scores[7] = 8.1;
scores[8] = 7.4;
scores[9] = 9.1;
```
Declaring and instantiating Arrays

• More examples:

```java
int[] weights = new int[2000];

boolean[] flags;
flags = new boolean[20];

char[] codes = new char[1750];

double[] prices = new double[500];
```
Using Arrays

Array elements can be assigned a value, printed, or used in a calculation. Examples:

```
System.out.println("Top = " + scores[5]);

mean = (scores[0] + scores[1])/2;

scores[3] = 7 + Math.random();

scores[scores.length - 1] = 9.0;

double num = scores[rand.nextInt(10)];
```
Try this: Write some Java code to create an array

- declare and instantiate an array `ratings` that holds 5 values type `int`

<table>
<thead>
<tr>
<th>ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

- declare and instantiate an array `vowel` to hold 5 values of type `char`, then initialize its values to the vowels ‘a’, ‘e’, ‘i’, ‘o’, ‘u’

<table>
<thead>
<tr>
<th>vowel</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘a’</td>
</tr>
<tr>
<td>‘e’</td>
</tr>
<tr>
<td>‘i’</td>
</tr>
<tr>
<td>‘o’</td>
</tr>
<tr>
<td>‘u’</td>
</tr>
</tbody>
</table>

vowel.length
5
What gets printed?

System.out.println (scores[8] + 1);

System.out.println (scores[1] + scores[2]);

System.out.println (scores[1 + 2]);

System.out.println(scores[scores.length - 2]);
Show how `scores` values change:

```java
scores[4] = 1;
scores[5] = scores[0] + 1;
scores[scores.length - 2]) = 5.5;
```
Processing Arrays using for-loops:
1) draw a picture of the resulting array

```java
public void populateArray()
{
    double[] mylist = new double[10];

    for (int i = 0; i < 10; i++)
        mylist[i] = 0;

    for (int i = 0; i < 10; i++)
        mylist[i] = i;
}
```
Processing Arrays using for-loops:

2) Reversing through...

```java
double[] tinyScores = new double[5];
for (int i = 4; i >= 0; i--)
    System.out.println(tinyScores[i]);
```

Output:

```
    7.9  8.7  9.4  8.2  6.7
tinyScores 0 1 2 3 4
```
Processing Arrays using for-loops:
3) write a for-loop to print the values in the **vowel** array (going **forward**)

4) write a for-loop to print the values in the **vowel** array (going **backward**)

CSC 1051 M.A. Papalaskari, Villanova University
Bounds Checking

An array index must specify a valid element

- Example: If an array `codes` holds 100 values, it can be indexed from 0 to 99. If the value of `count` is 100, then

  ```java
  System.out.println(codes[count]);
  ```

causes an `ArrayIndexOutOfBoundsException`

- It’s common to introduce off-by-one errors when using arrays:

  ```java
  for (int index=0; index <= 100; index++)
      codes[index] = index*50 + epsilon;
  ```

CSC 1051 M.A. Papalaskari, Villanova University