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:

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

Declaration:
double[] scores;

The entire array has a single name

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];
```

Size of array:
scores.length

10
Arrays - Initialization

Declaration: The entire array has a single name

```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:

```java
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`

- 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’
What gets printed?

```java
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

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...

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
```
Processing Arrays using for-loops:
3) write a for-loop to print the values in the \textbf{vowel} array (going \textit{forward})

4) write a for-loop to print the values in the \textbf{vowel} array (going \textit{backward})
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;
```
Initializer Lists

• Alternative way to declare, instantiate, and initialize an array. For example:

```c
int[] ratings = {4, 3, 3, 1, 4, 2, 1, 0, 3, 4};
char[] grades = {'A', 'B', 'C', 'D', 'F'};
```

• **NOTE:**
  - the `new` operator is **not** used
  - size of array is determined by the number of items listed
  - can only be used in the array declaration

try this with the **vowel** array
The “for-each” Loop

• A simple way of processing every array element:

```java
for (double score : scores)
    System.out.println(score);
```

**NOTE:**
• Only appropriate when processing all array elements starting at index 0
• It can't be used to set the array values

try this with the `vowel` array
Another example

```java
String[] animals = {"dog", "cat", "mouse", "fox"};
for (String word : animals)
    System.out.println("The " + word + " ate the cake");

for (String word : animals)
    for (String otherWord: animals)
        System.out.println("The " + word + " ate the " + otherWord);
```
Try this: Use the “for each” loop to scan through an array of `int` containing ratings (range: 0 - 4) and count up how many 4’s.

```csharp
int[] ratings = {4, 3, 3, 1, 4, 3, 1, 0, 3, 4};
```
Try this: Repeat, but now count up the 0’s, 1’s,... 4’s – Use a separate array for this

```c
int[] ratings = {4, 3, 3, 1, 4, 3, 1, 0, 3, 4};
```
More array examples (see textbook):

- `BasicArray.java`
- `Primes.java`
- `ReverseOrder.java`
- `LetterCount.java`
import java.util.Scanner;

public class ReverseOrder
{
    //---
    //  Reads a list of numbers from the user, storing them in an
    //  array, then prints them in the opposite order.
    //---
    public static void main (String[] args)
    {
        Scanner scan = new Scanner (System.in);

        double[] numbers = new double[10];

        System.out.println ("The size of the array: " + numbers.length);

        continue
continue

```java
for (int index = 0; index < numbers.length; index++)
{
    System.out.print ("Enter number " + (index+1) + ": ");
    numbers[index] = scan.nextDouble();
}

System.out.println ("The numbers in reverse order:"/MPL/);

for (int index = numbers.length-1; index >= 0; index--)
    System.out.print (numbers[index] + "  ");
}```
Sample Run

The size of the array: 10
Enter number 1: 18.36
Enter number 2: 48.9
Enter number 3: 53.5
Enter number 4: 29.06
Enter number 5: 72.404
Enter number 6: 34.8
Enter number 7: 63.41
Enter number 8: 45.55
Enter number 9: 69.0
Enter number 10: 99.18
The numbers in reverse order:
99.18  69.0  45.55  63.41  34.8  72.404  29.06  53.5  48.9  18.36

places numbers in an array, then prints them out **backward**

... alternatively, we could place the numbers in the array **backward** and then print them **forward**
Another example: Computing letter frequency counts

Sample run:

Enter a sentence:
In Casablanca, Humphrey Bogart never says "Play it again, Sam."

A: 0    a: 10
B: 1    b: 1
C: 1    c: 1
D: 0    d: 0
E: 0    e: 3

Let’s write a program to do this
import java.util.Scanner;

public class LetterCount
{
    public static void main (String[] args)
    {
        final int NUMCHARS = 26;

        Scanner scan = new Scanner (System.in);

        int[] upper = new int[NUMCHARS];
        int[] lower = new int[NUMCHARS];

        char current; // the current character being processed
        int other = 0; // counter for non-alphabetics

        continue
continue
System.out.println("Enter a sentence:");
String line = scan.nextLine();

// Count the number of each letter occurrence
for (int ch = 0; ch < line.length(); ch++)
{
    current = line.charAt(ch);
    if (current >= 'A' && current <= 'Z')
        upper[current-'A']++;
    else
        if (current >= 'a' && current <= 'z')
            lower[current-'a']++;
        else
            other++;
}

// Print the results
System.out.println();
for (int letter=0; letter < upper.length; letter++)
{
    System.out.print((char)(letter + 'A'));
    System.out.print(": "+ upper[letter]);
    System.out.print("\t\t"+(char)(letter + 'a'));
    System.out.println(": "+ lower[letter]);
}

System.out.println();
System.out.println("Non-alphabetic characters: "+ other);
Sample Run

Enter a sentence:
In Casablanca, Humphrey Bogart never says "Play it again, Sam."

A: 0     a: 10
B: 1     b: 1
C: 1     c: 1
D: 0     d: 0
E: 0     e: 3
F: 0     f: 0
G: 0     g: 2
H: 1     h: 1
I: 1     i: 2
J: 0     j: 0
K: 0     k: 0
L: 0     l: 2
M: 0     m: 2
N: 0     n: 4
O: 0     o: 1
P: 1     p: 1
Q: 0     q: 0
R: 0     r: 3
S: 1     s: 3
T: 0     t: 2
U: 0     u: 1
V: 0     v: 1
W: 0     w: 0
X: 0     x: 0
Y: 0     y: 3
Z: 0     z: 0

Non-alphabetic characters: 14