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:

The entire array has a single name

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

- **Declaration:**

  \[
  \text{double[]} \text{ scores}
  \]

  The entire array has a single name.
Arrays - Instantiation

• **Declaration:**

```java
double[] scores = new double[10];
```

The entire array has a single name

- **element type**
- **array element**
  - `scores[2]`

- **index**

0 1 2 3 4 5 6 7 8 9

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

- The size of the array is given by: `scores.length = 10`

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

• Declaration:
  ```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;
```

The entire array has a single name

- **Element type**: `double`
- **Array element**: `scores[2]` has a value of 9.4
- **Index**: `scores[2]` is at index 2

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

The size of the array is given by: `scores.length = 10`
Declaring and instantiating Arrays

• Some more examples:

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

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

char[] codes = new char[1750];

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

– The brackets of the array type can be associated with element type or with array name, so this alternative code is also ok:

```java
double prices[] = new double[500];
```

– The first format is preferable because it is more readable.
Using Arrays

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

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

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

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

scores[scores.length - 1] = 9.0;

double num = scores[(int) Math.random()*10];
```

Try this: Write some Java code to create an array

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

- declare and instantiate an array named `vowel` to hold 5 values of type `char`, then initialize its values to the vowels ‘a’, ‘e’, ‘i’, ‘o’, ‘u’
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 - 1]);
Show how `scores` values change:

```java
scores[8] = 1;
scores[5] = scores[0] + 1;
scores[scores.length - 1]) = 5.5;
```
Processing Arrays using for-loops:
1) draw a picture of the resulting array

double[] scores = new double[10];

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

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

2) show output

double[] scores = new double[10];

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

Output:
Processing Arrays using for-loops:
3) write a for-loop to print the values in the *vowel* array
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
Initializer Lists

• Alternative way to declare, instantiate, and initialize an array. For example:
  
  int[] units = {147, 323, 89, 933, 540, 269, 97, 114, 298, 476};

  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
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
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:");
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
Another example: Computing letter frequency counts

Sample run:

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

<table>
<thead>
<tr>
<th>Letter</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
</tr>
<tr>
<td>d</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
</tr>
<tr>
<td>e</td>
<td>3</td>
</tr>
</tbody>
</table>

Let’s write a program to do this
Letter Frequency counts - Algorithm

Uses two arrays: `upper` and `lower` with 26 elements each, to count the number of upper- and lowercase letters, respectively.

1. declare and instantiate `upper` and `lower` with 26 elements of type `int`
2. prompt user to "Enter a sentence:" and scan input into a String `line`
3. for each character `current` in `line` do the following:
   - if (`current` is between ‘A’ and 'Z')
     • `position ← current - 'A'` // compute distance from ‘A’
     • `upper[position] ← upper[position] + 1` // increment count for `current`
   - if (`current` is between ‘a’ and ‘z')
     • `position ← current − ‘a’`
     • `lower[position] ← upper[position] + 1`
   - // do nothing if current is not a letter.
import java.util.Scanner;

public class LetterCount
{
    //---
    //  Reads a sentence from the user and counts the number of
    //  uppercase and lowercase letters contained in it.
    //---
    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
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++;
}

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

Sample Run (continued)

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