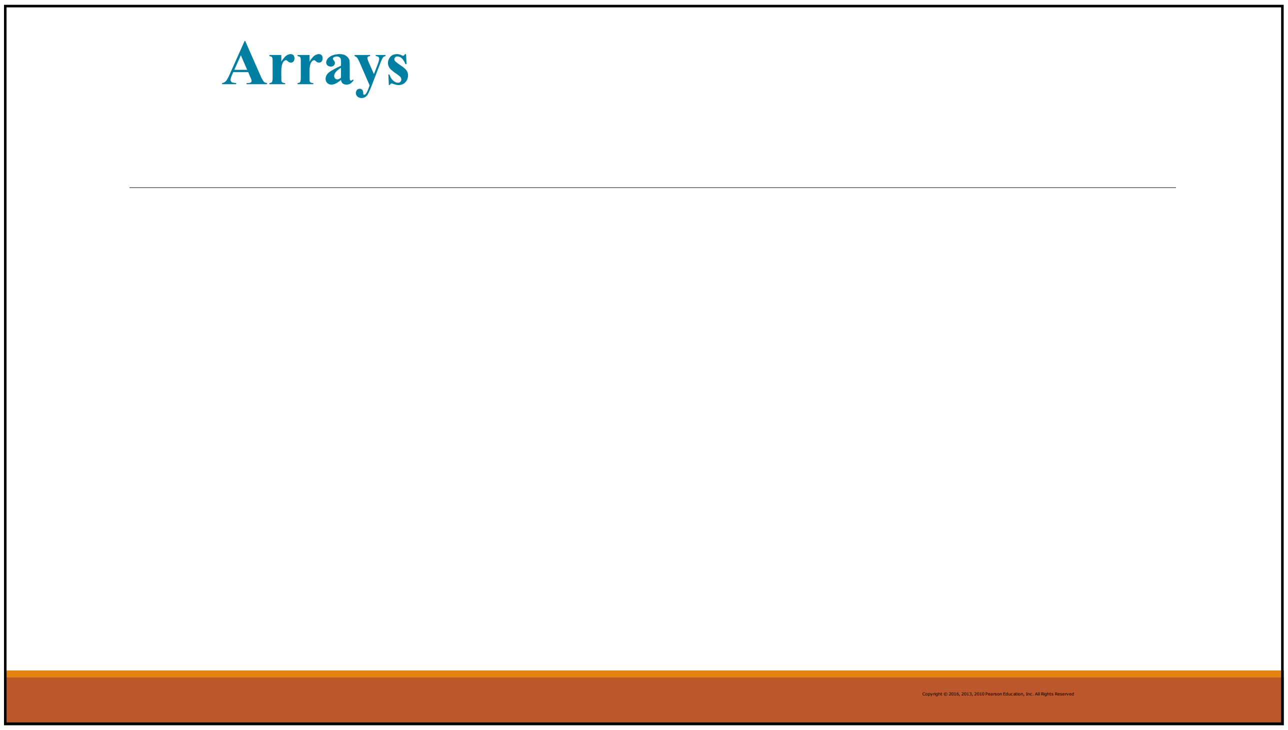
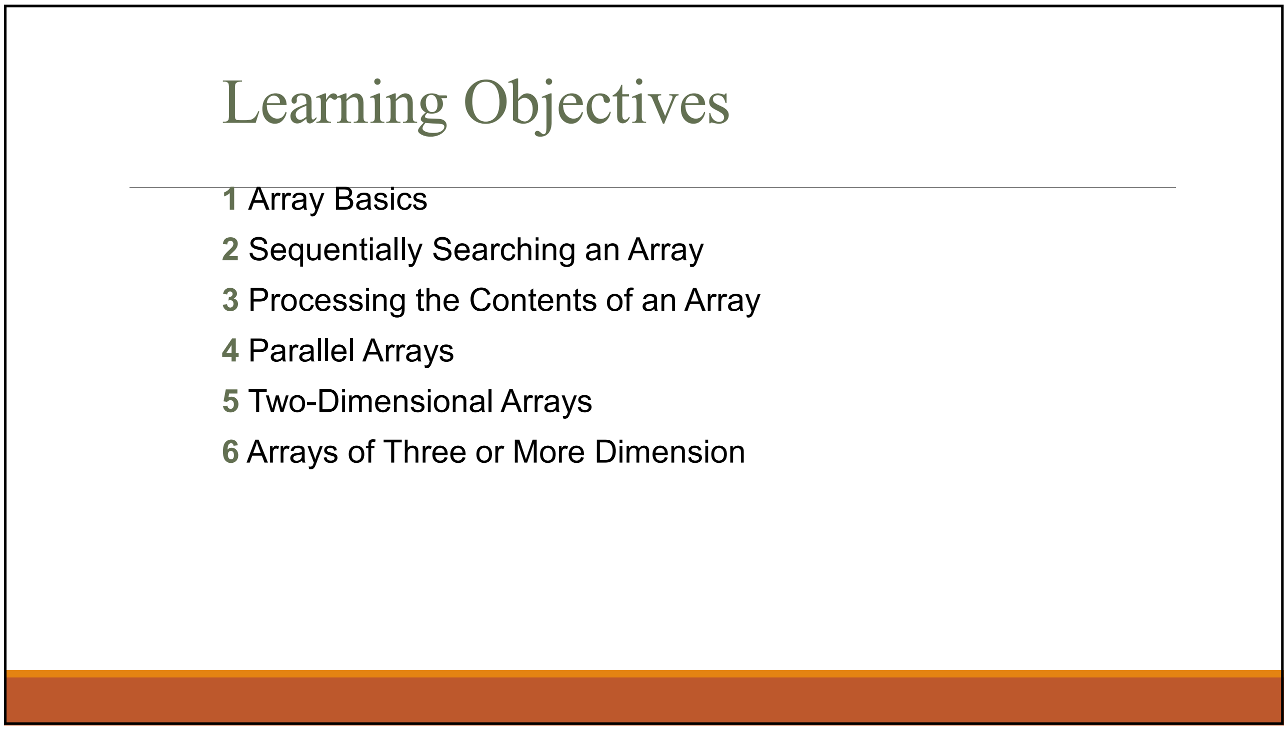


Arrays



Learning Objectives

1 Array Basics
2 Sequentially Searching an Array
3 Processing the Contents of an Array
4 Parallel Arrays
5 Two-Dimensional Arrays
6 Arrays of Three or More Dimension

8.1 Array Basics (1 of 9)

An **array** allows you to store a group of items of the same data type together in memory

– Why? Instead of creating multiple similar variables such as **employee1, employee2, employee3** and so on…

– It's more efficient to create just one variable

◦ **Declare String employees[50]**

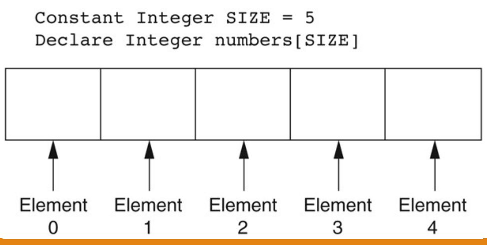
◦ **Declare Real salesAmounts[7]**

◦ The number in the [ ] is the size of the array

1 Array Basics (2 of 9)

- The storage locations in an array are **elements**
- Each element of the array has a unique number called a **subscript** that identifies it – the subscript starts at 0 in most languages.

Figure 8-1 Array subscripts



```
Constant Integer SIZE = 5
Declare Integer numbers[SIZE]
```

1 Array Basics (3 of 9)

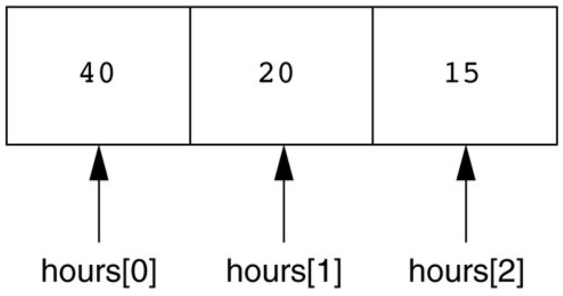
Assigning values can be done individually using a subscript…

Set numbers[0] = 20
Set numbers[1] = 30
Set numbers[2] = 40
Set numbers[3] = 50
Set numbers[4] = 60

But, it is much more efficient to use a loop to step through the array

1 Array Basics (4 of 9)

Figure Contents of the `hours` array



1 Array Basics (5 of 9)

Arrays can be initialized to 0 or specific values

Declare String days[7] = "Sunday", "Monday", "Tuesday", Wednesday", "Thursday", "Friday", "Saturday"

Array bounds checking should be performed to avoid use of an invalid subscript

Days[7] = "Saturday" is invalid because there is no 7 index

- A common error is running a loop one time more than is necessary, exceeding the bound of the array
- Off-by-one Error

1 Array Basics (6 of 9)

- Partially Filled Array
  - Sometimes an array is only partially filled
  - To avoid processing the unfilled elements, you must have an accompanying integer variable that holds the number of items stored in the array.
  - When the array is empty, 0 is stored in this variable
  - The variable is incremented each time an item is added to the array
  - The variable's value is used as the array's size when stepping through the array.

1 Array Basics (7 of 9)

```
Constant Integer SIZE = 100
Declare Integer values[SIZE]
Declare Integer count = 0
Declare Integer number
Declare Integer Index

Display "Enter a number, or -1 to quit."
Input number
While (number != -1 AND count < SIZE)
    Set values[count] = number
    Set count = count + 1
    Display "Enter a number, or -1 to quit."
    Input number
End While

Display "Here are the values you entered:"
For index = 0 To count - 1
    Display values[index]
End For
```

The count variable holds the number of items stored in the array.

Partially Filled Array Example

1 Array Basics (8 of 9)

- Optional Topic: The **For Each** Loop
  - Some languages provide a For Each loop
  - It works with an array, iterating once for each array element
  - During each iteration, the loop copies an element's value to a variable.

1 Array Basics (9 of 9)

```
Constant Integer SIZE = 5
Declare Integer numbers[SIZE] = 5, 10, 15, 20, 25
Declare Integer num

For Each num In numbers
    Display num
End For
```

For Each Example

2 Sequentially Searching An Array

A sequential search algorithm is a simple technique for finding an item in a string or numeric array

- Uses a loop to sequentially step through an array
- Compares each element with the value being searched for
- Stops when the value is found or the end of the array is hit

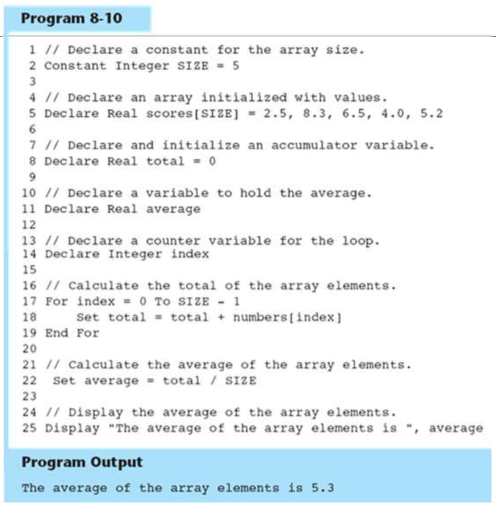
```
Set found = False
Set index = 0
While found == False AND index <= SIZE -1
    If (array[index] == searchValue Then
        Set found = True
    Else
        Set index = index + 1
    End If
End While
```

3 Processing the Contents of an Array (1 of 3)

Totaling the values in an array and calculating average

- Loops are used to accumulate the values
- Then, the total is simply divided by the size

Example



Program 8-10

```
// Declare a constant for the array size.
Constant Integer SIZE = 5

// Declare an array initialized with values.
Declare Real scores[SIZE] = 2.5, 8.3, 6.5, 4.0, 5.2

// Declare and initialize an accumulator variable.
Declare Real total = 0

// Declare a variable to hold the average.
Declare Real average

// Declare a counter variable for the loop.
Declare Integer index

// Calculate the total of the array elements.
For index = 0 To SIZE - 1
    Set total = total + numbers[index]
End For

// Calculate the average of the array elements.
 Set average = total / SIZE

// Display the average of the array elements.
Display "The average of the array elements is ", average
```

Program Output

```
The average of the array elements is 5.3
```

3 Processing the Contents of an Array (2 of 3)

Finding the highest & lowest values in an array

- The highest
  - Create a variable to hold the highest value
  - Assign the value at element 0 to the highest
  - Use a loop to step through the rest of the elements
  - Each iteration, a comparison is made to the highest variable
  - If the element is greater than the highest value, that value is then the assigned to the highest variable
- The lowest
  - Same process, but checks if the element is less than the lowest value

3 Processing the Contents of an Array (3 of 3)

Copying an array can be done using loops

```
For index = 0 to SIZE – 1
    Set secondArray[index] = firstArray[index]
End For
```

Passing an Array as an Argument

- Usually must pass the array and the size

The module call

getTotal(numbers, SIZE)

The module header

Function Integer getTotal (Integer array[], Integer arraySize)

4 Parallel Arrays

By using the same subscript, you can establish a relationship between data stored in two or more arrays

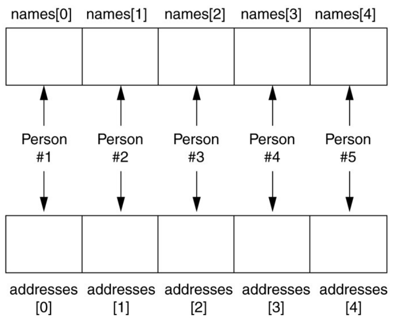


Figure The `names` and `addresses` arrays

5 Two-Dimensional Arrays (1 of 2)

A two-dimensional array is like several identical arrays put together

– Suppose a teacher has six students who take five tests

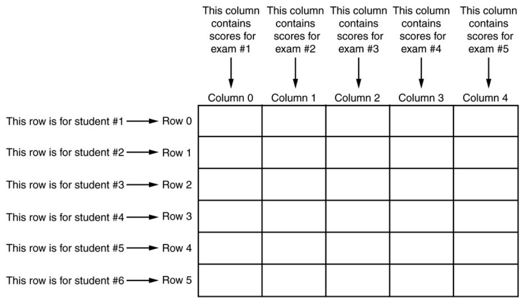

This column contains scores for exam #1
This column contains scores for exam #2
This column contains scores for exam #3
This column contains scores for exam #4
This column contains scores for exam #5
Column 0
Column 1
Column 2
Column 3
Column 4
This row is for student #1 Row 0
This row is for student #2 Row 1
This row is for student #3 Row 2
This row is for student #4 Row 3
This row is for student #5 Row 4
This row is for student #6 Row 5

Figure Two-dimensional array with six rows and five columns

5 Two-Dimensional Arrays (2 of 2)

Two size variables are required when declaring

```
Constant Integer ROWS = 3
Constant Integer COLS = 4
Declare Integer values[ROWS][COLS]
```

Accessing is done with two loops, and both subscripts

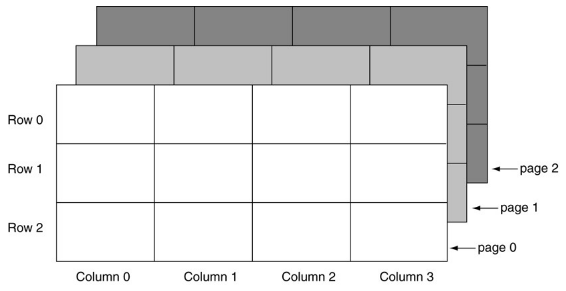
```
For row = 0 To ROWS -1
    For col = 0 To COLS – 1
        Display “Enter a number.”
        Input values[row][col]
    End For
End For
```

8 Arrays of Three or More Dimensions

Arrays can also be three or more dimensions

Declare Real seats[3][5][8]

Figure A three-dimensional array


Row 0
Row 1
Row 2
Column 0
Column 1
Column 2
Column 3
page 2
page 1
page 0