PEN203

C++ Arrays

C++ How to Program
Deitel & Deitel

Outline

- Arrays
- Defining Arrays
- Initializing Arrays
- Array Examples
- Passing Arrays to Functions
- Sorting Arrays
- Multiple Subscripted Arrays

Arrays

- Arrays store related data.
- The size of the array is not changed during the execution of the program.
- An array can be defined as a group of consecutive memory locations.
- To access an element of array, array name and the position number is provided.

  arrayname[position number]

Arrays

- Example: 5 element ar array.

ar[0]	12
ar[1]	-3
ar[2]	8
ar[3]	0
ar[4]	64

Arrays

- Array elements can be used like other ordinary variables.
- Examples:

```
ar[2]=18;
cout<<ar[ 1 ];
```

- Array subscript may be an operation, variable or constant.

```
ar[7-4], ar[i], ar[3]
```

Defining Arrays

- To define arrays, you need to provide name, type of array, and the number of elements.

  arType arName[numberofElements]

- Examples:

  int ar[5];

  float x[10];

Initializing Arrays

- int ar[5]={-2,0,45,-13,20};
- If you do not provide enough initializers, rightmost elements are initialized to 0.
- int ar[5]={0} initialize all elements to 0.
- If you provide more than required elements, you get syntax error.
- If you do not provide size, the number of elements in initializer list become size.

  int ar[ ]={-2,0,45};

  The size of our array is 3.

Array Examples

```
1   // Fig. 4.5: fig04_05.cpp
2   // Initialize array s to the even integers from 2 to 20.
3   #include <iostream>
4
5   using std::cout;
6   using std::endl;
7
8   #include <iomanip>
9
10  using std::setw;
11
12  int main()
13  {
14    // constant variable can be used to specify array size
15    const int arraySize = 10;
16
17    int s[ arraySize ];  // array s has 10 elements
18
19    for ( int i = 0; i < arraySize; i++ )  // set the values
20      s[ i ] = 2 + 2 * i;
21
22    cout << "Element" << setw( 13 ) << "Value" << endl;
23
```

Array Examples

```
24    // output contents of array s in tabular format
25    for ( int j = 0; j < arraySize; j++ )
26       cout << setw( 7 ) << j << setw( 13 ) << s[ j ] << endl;
27
28    return 0;  // indicates successful termination
29
30 } // end main
```

```
Element         Value
      0             2
      1             4
      2             6
      3             8
      4            10
      5            12
      6            14
      7            16
      8            18
      9            20
```

Passing Arrays to Functions

- You just need to provide array name without square brackets

```
int ar[5];
function1(ar,5);
```

- You may pass array size.
- Arrays are passed call-by-reference inherently.
- Actually array names shows the address of first element in the array.
- Function prototype example:

```
void function1(int b[], int sizeOfArray);
```

- Individual array elements are passed call-by value.

Sorting Arrays

- Sorting is an important concept in Computer Science
- Example: Bubble sort
  - You need several passes on the array
  - You compare successive pairs
    - If increasing order, no change
    - If decreasing order, elements swapped.
  - Repeat

Multiple Subscripted Arrays

- Multiple subscripted arrays can be considered as tables like matrices with rows and columns.
- int ar[3][2] = { {3,5}, {4,-1}, {7,4} };
- int ar[3][2] = {3, 5, 4, -1, 7, 4};
- Uninitialized elements set to zero
- To access an element of array, you need to specify row and column subscripts.

```
cout<<ar[ 2 ][ 1 ];
```

Multiple Subscripted Arrays

```
1  // Fig. 4.22: fig04_22.cpp
2  // Initializing multidimensional arrays.
3  #include <iostream>
4
5  using std::cout;
6  using std::endl;
7
8  void printArray( int [][ 3 ] );
9
10 int main()
11 {
12    int array1[ 2 ][ 3 ] = { { 1, 2, 3 }, { 4, 5, 6 } };
13    int array2[ 2 ][ 3 ] = { 1, 2, 3, 4, 5 };
14    int array3[ 2 ][ 3 ] = { { 1, 2 }, { 4 } };
15
16    cout << "Values in array1 by row are:" << endl;
17    printArray( array1 );
18
19    cout << "Values in array2 by row are:" << endl;
20    printArray( array2 );
21
22    cout << "Values in array3 by row are:" << endl;
23    printArray( array3 );
24
25    return 0;  // indicates successful termination
26
27 } // end main
```

Multiple Subscripted Arrays

```
28
29  // function to output array with two rows and three columns
30  void printArray( int a[][ 3 ] )
31  {
32     for ( int i = 0; i < 2; i++ ) {    // for each row
33
34        for ( int j = 0; j < 3; j++ )   // output column values
35           cout << a[ i ][ j ] << ' ';
36
37        cout << endl;  // start new line of output
38
39     } // end outer for structure
40
41  } // end function printArray
```

```
Values in array1 by row are:
1 2 3
4 5 6
Values in array2 by row are:
1 2 3
4 5 0
Values in array3 by row are:
1 2 0
4 0 0
```