# Computer Programming
# Lecture 7

Hung-Yu Wei
Department of Electrical Engineering
National Taiwan University

# Arrays

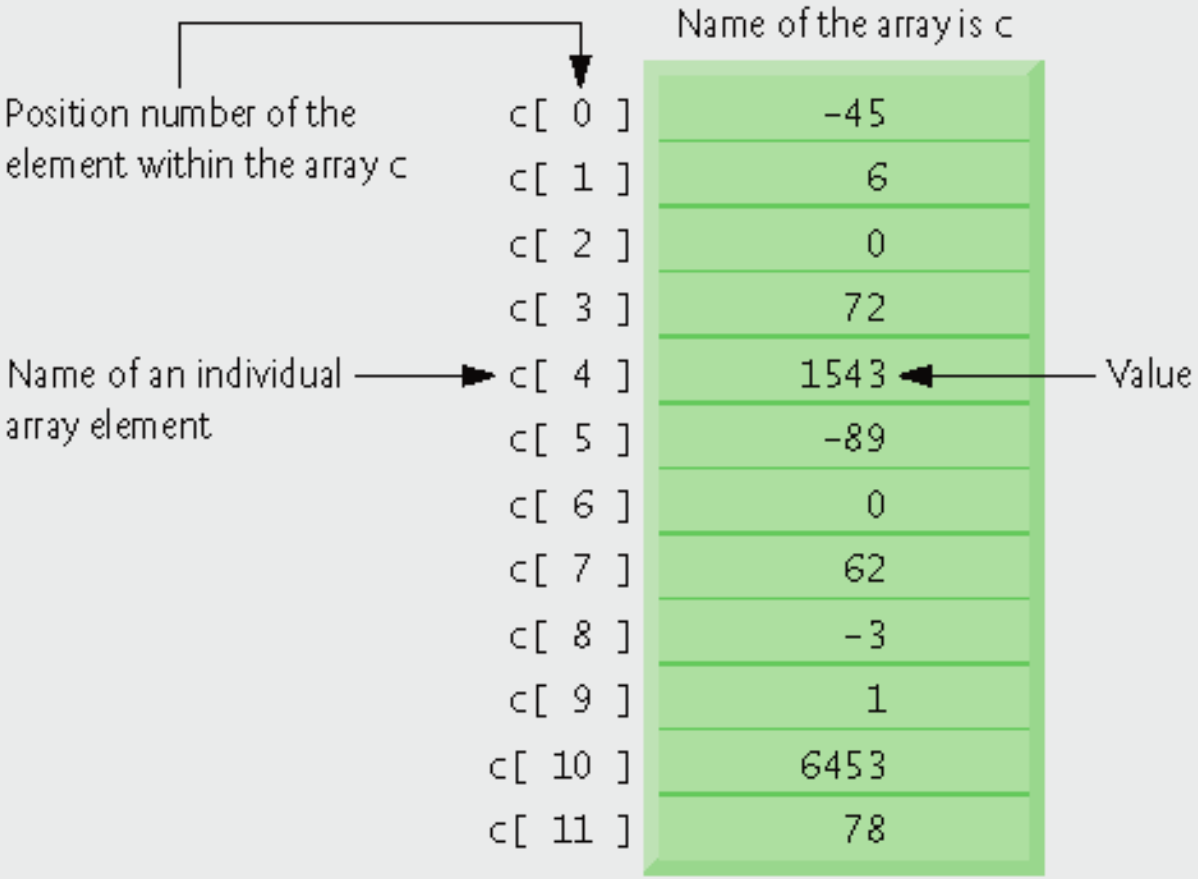
- A useful data structure
- Use consecutive memory space to store a series of data
- Structure to store related data items
- Two types
  - Pointer-based arrays (C-like)
    - X[5]={2,4,6,8,10};
  - Arrays as objects (C++)
    - *vector* in C++ Standard Template Class Library

# Arrays

- Terms
  - Element
    - stored data
  - Index (or subscript)
    - Help you identify data
- To refer to an element
  - Specify array name and position number (index)
  - Syntax

    arrayname[ position number ]
  - First element at position 0
- N-element array c

  `c[ 0 ], c[ 1 ] … c[ n - 1 ]`
  - Nth element as position N-1


Name of the array is c
Position number of the element within the array c
Name of an individual array element
Value
c[ 0 ] -45
c[ 1 ] 6
c[ 2 ] 0
c[ 3 ] 72
c[ 4 ] 1543
c[ 5 ] -89
c[ 6 ] 0
c[ 7 ] 62
c[ 8 ] -3
c[ 9 ] 1
c[ 10 ] 6453
c[ 11 ] 78

# Operations of an array

- Array elements are like other variables
  - Assignment, printing for an integer array **c**

```
c[0]= 3;
cout << c[0];
x = c[6]/2;
```

- Can perform operations inside subscript

  **c[ 5 - 2 ]** same as **c[3]**

# 6.3 Declaring Arrays

- When declaring arrays, specify
  - Name
  - Type of array
    - Any data type
  - Number of elements
  - *type arrayName*`[` *arraySize* `];`

```
int c[ 10 ];  // array of 10 integers
float d[ 3284 ]; // array of 3284 floats
```

- Declaring multiple arrays of same type
  - Use comma separated list, like regular variables

```
int b[ 100 ], x[ 27 ];
```

# 6.4 Examples Using Arrays

- Initializing arrays
  - *for* loop
    - Set each element
  - Initializer list
    - Specify each element when array declared

      ```
      int n[ 5 ] = { 1, 2, 3, 4, 5 };
      ```

    - Not enough initializers: rightmost elements are set to 0
    - Too many initializers: syntax error
  - To set every element to same value

    ```
    int n[ 5 ] = { 0 };
    ```

  - If array size omitted, initializers determine size

    ```
    int n[] = { 1, 2, 3, 4, 5 };
    ```

    - 5 initializers, therefore 5 element array

```
1  // Fig. 6.3: fig06_03.cpp
2  // Initializing an array.
3  #include <iostream>
4  #include <iomanip>
5  using namespace std;
6
7  int main()
8  {
9     int n[ 10 ]; // n is an array of 10 integers
10
11    // initialize elements of array n to 0
12    for ( int i = 0; i < 10; i++ )
13       n[ i ] = 0; // set element at location i to 0
14
15    cout << "Element" << setw( 13 ) << "Value" << endl;
16
17    // output each array element's value
18    for ( int j = 0; j < 10; j++ )
19       cout << setw( 7 ) << j << setw( 13 ) << n[ j ] << endl;
20 } // end main
```

**Fig. 6.3** | Initializing an array's elements to zeros and printing the array. (Part 1 of 2.)

```
1  // Fig. 6.4: fig06_04.cpp
2  // Initializing an array in a declaration.
3  #include <iostream>
4  #include <iomanip>
5  using namespace std;
6
7  int main()
8  {
9     // use initializer list to initialize array n
10    int n[ 10 ] = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };
11
12    cout << "Element" << setw( 13 ) << "Value" << endl;
13
14    // output each array element's value
15    for ( int i = 0; i < 10; i++ )
16       cout << setw( 7 ) << i << setw( 13 ) << n[ i ] << endl;
17 } // end main
```

g. 6.4 | Initializing the elements of an array in its declaration. (Part 1 of 2.)

# const

- Declare a constant variable
  - Cannot be modified later
  - Read-only variable
- Example

  **const** int arraySize=10;
  - Declare variable arraySize as a constant integer
- Common errors
  - Not assigning a value when declare a constant variable
  - Assigning a value in executable statements

# Tips: Define array size with const

- Step1:Use const to define array size
- Step2:Use loop to set array elements
- Using constant variables to specify array size is more scalable
  - Easy to track your program
- Example
  - Create an array that contains 10 even numbers
    - 2,4,6,8,10,….,18,20

```
1  // Fig. 6.5: fig06_05.cpp
2  // Set array s to the even integers from 2 to 20.
3  #include <iostream>
4  #include <iomanip>
5  using namespace std;
6
7  int main()
8  {
9     // constant variable can be used to specify array size
10    const int arraySize = 10;
11
12    int s[ arraySize ]; // array s has 10 elements
13
14    for ( int i = 0; i < arraySize; i++ ) // set the values
15       s[ i ] = 2 + 2 * i;
16
17    cout << "Element" << setw( 13 ) << "Value" << endl;
18
19    // output contents of array s in tabular format
20    for ( int j = 0; j < arraySize; j++ )
21       cout << setw( 7 ) << j << setw( 13 ) << s[ j ] << endl;
22 } // end main
```

Fig. 6.5 | Generating values to be placed into elements of an array. (Part 1 of 2.)

# Example: const

```
1  // Fig. 6.6: fig06_06.cpp
2  // Using a properly initialized constant variable.
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8     const int x = 7; // initialized constant variable
9
10    cout << "The value of constant variable x is: " << x << endl;
11 } // end main
```

```
The value of constant variable x is: 7
```

**Fig. 6.6** | Initializing and using a constant variable.

# Example: summation of array elements

```
1  // Fig. 6.8: fig06_08.cpp
2  // Compute the sum of the elements of the array.
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8     const int arraySize = 10; // constant variable indicating size of array
9     int a[ arraySize ] = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
10    int total = 0;
11
12    // sum contents of array a
13    for ( int i = 0; i < arraySize; i++ )
14       total += a[ i ];
15
16    cout << "Total of array elements: " << total << endl;
17 } // end main
```

```
Total of array elements: 849
```

**Fig. 6.8** | Computing the sum of the elements of an array.

# Example: Array as counters

```
1  // Fig. 6.10: fig06_10.cpp
2  // Roll a six-sided die 6,000,000 times.
3  #include <iostream>
4  #include <iomanip>
5  #include <cstdlib>
6  #include <ctime>
7  using namespace std;
8
9  int main()
10 {
11    const int arraySize = 7; // ignore element zero
12    int frequency[ arraySize ] = {}; // initialize elements to 0
13
14    srand( time( 0 ) ); // seed random number generator
15
16    // roll die 6,000,000 times; use die value as frequency index
17    for ( int roll = 1; roll <= 6000000; roll++ )
18       frequency[ 1 + rand() % 6 ]++;
19
20    cout << "Face" << setw( 13 ) << "Frequency" << endl;
21
```

**Fig. 6.10** | Die-rolling program using an array instead of `switch`. (Part 1 of 2.)

```
22      // output each array element's value
23      for ( int face = 1; face < arraySize; face++ )
24         cout << setw( 4 ) << face << setw( 13 ) << frequency[ face ]
25            << endl;
26   } // end main
```

```
Face    Frequency
   1      1000167
   2      1000149
   3      1000152
   4       998748
   5       999626
   6      1001158
```

**Fig. 6.10** | Die-rolling program using an array instead of `switch`. (Part 2 of 2.)

# Example

- Using array to summarize survey results
  - In a survey, respondents will grade based on a scale of 1~10
  - 40 respondents
  - Summarize the results

```
1  // Fig. 6.11: fig06_11.cpp
2  // Poll analysis program.
3  #include <iostream>
4  #include <iomanip>
5  using namespace std;
6
7  int main()
8  {
9     // define array sizes
10    const int responseSize = 40; // size of array responses
11    const int frequencySize = 11; // size of array frequency
12
13    // place survey responses in array responses
14    const int responses[ responseSize ] = { 1, 2, 6, 4, 8, 5, 9, 7, 8,
15       10, 1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7,
16       5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };
17
18    // initialize frequency counters to 0
19    int frequency[ frequencySize ] = {};
20
21    // for each answer, select responses element and use that value
22    // as frequency subscript to determine element to increment
23    for ( int answer = 0; answer < responseSize; answer++ )
24       frequency[ responses[ answer ] ]++;
```

**Fig. 6.11** | Poll analysis program. (Part 1 of 2.)

```
25
26       cout << "Rating" << setw( 17 ) << "Frequency" << endl;
27
28       // output each array element's value
29       for ( int rating = 1; rating < frequencySize; rating++ )
30          cout << setw( 6 ) << rating << setw( 17 ) << frequency[ rating ]
31             << endl;
32    } // end main
```

```
Rating        Frequency
     1                2
     2                2
     3                2
     4                2
     5                5
     6               11
     7                5
     8                7
     9                1
    10                3
```

**Fig. 6.11** | Poll analysis program. (Part 2 of 2.)