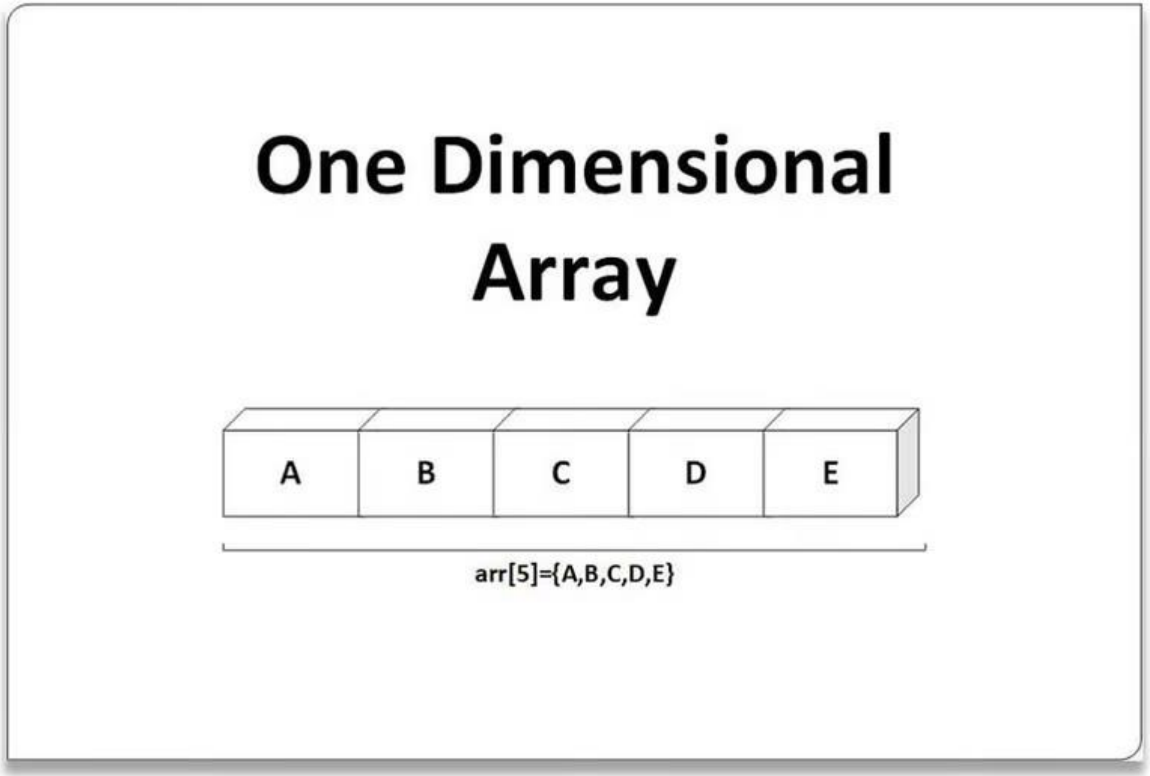
Fundamentals of Computer Programming

CS-110

Course Instructor: Dr. Momina Moetesum

One-Dimensional Arrays

Week 9



One Dimensional Array

arr[5]={A,B,C,D,E}

Learning Objectives

01

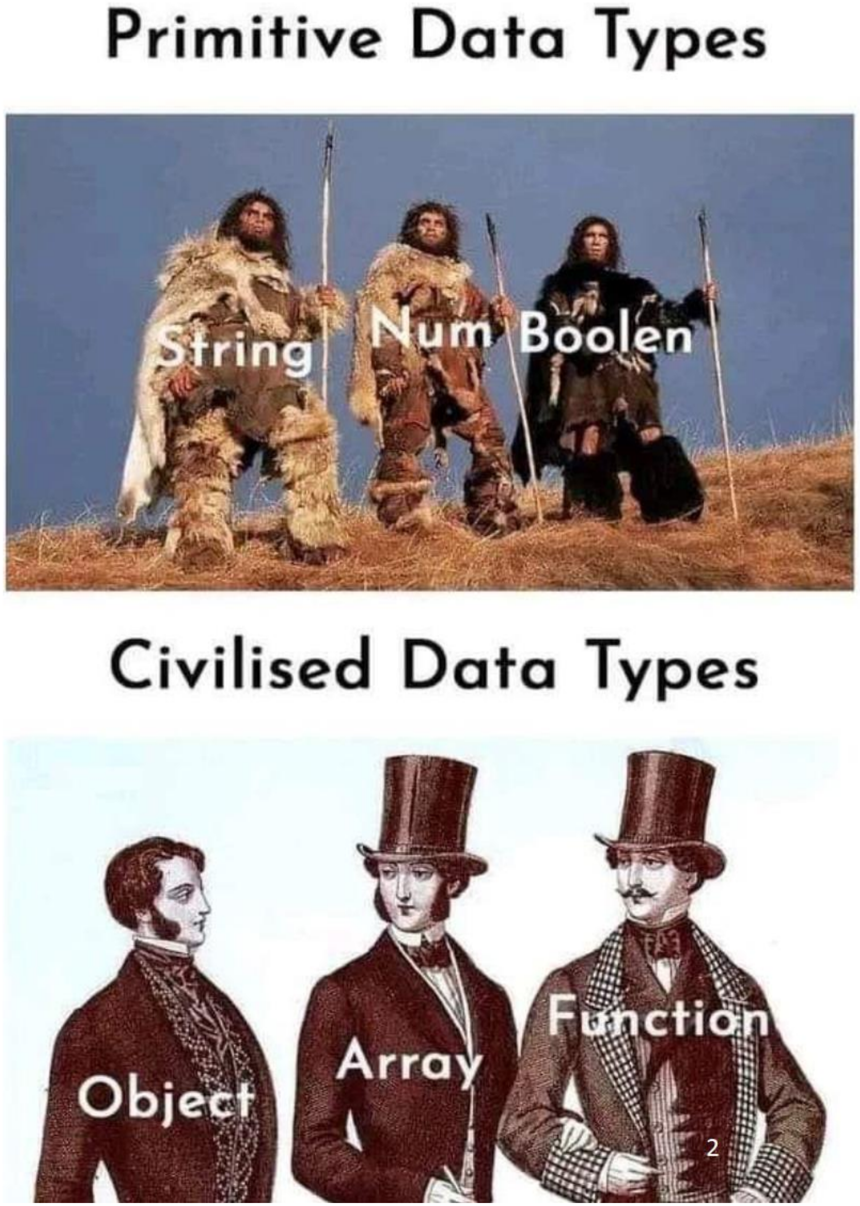
To understand how to declare and initialize a one-dimensional array

02

To understand how to access an array item

03

To practice how to pass an array in a function

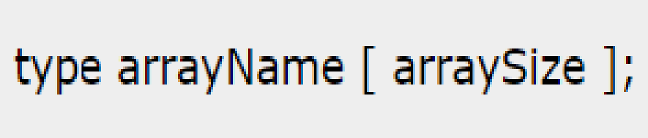


Arrays in C++

- C++ provides a data structure, **the array**, which stores a fixed-size sequential collection of elements of the same type.
- An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.
- Instead of declaring individual variables, such as number0, number1, ..., and number 99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables.
- A specific element in an array is accessed by an index.
- All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

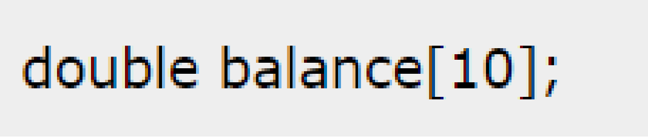
Declaring an Array

To declare an array in C++, the programmer specifies the type of the elements and the number of elements required by an array as follows:



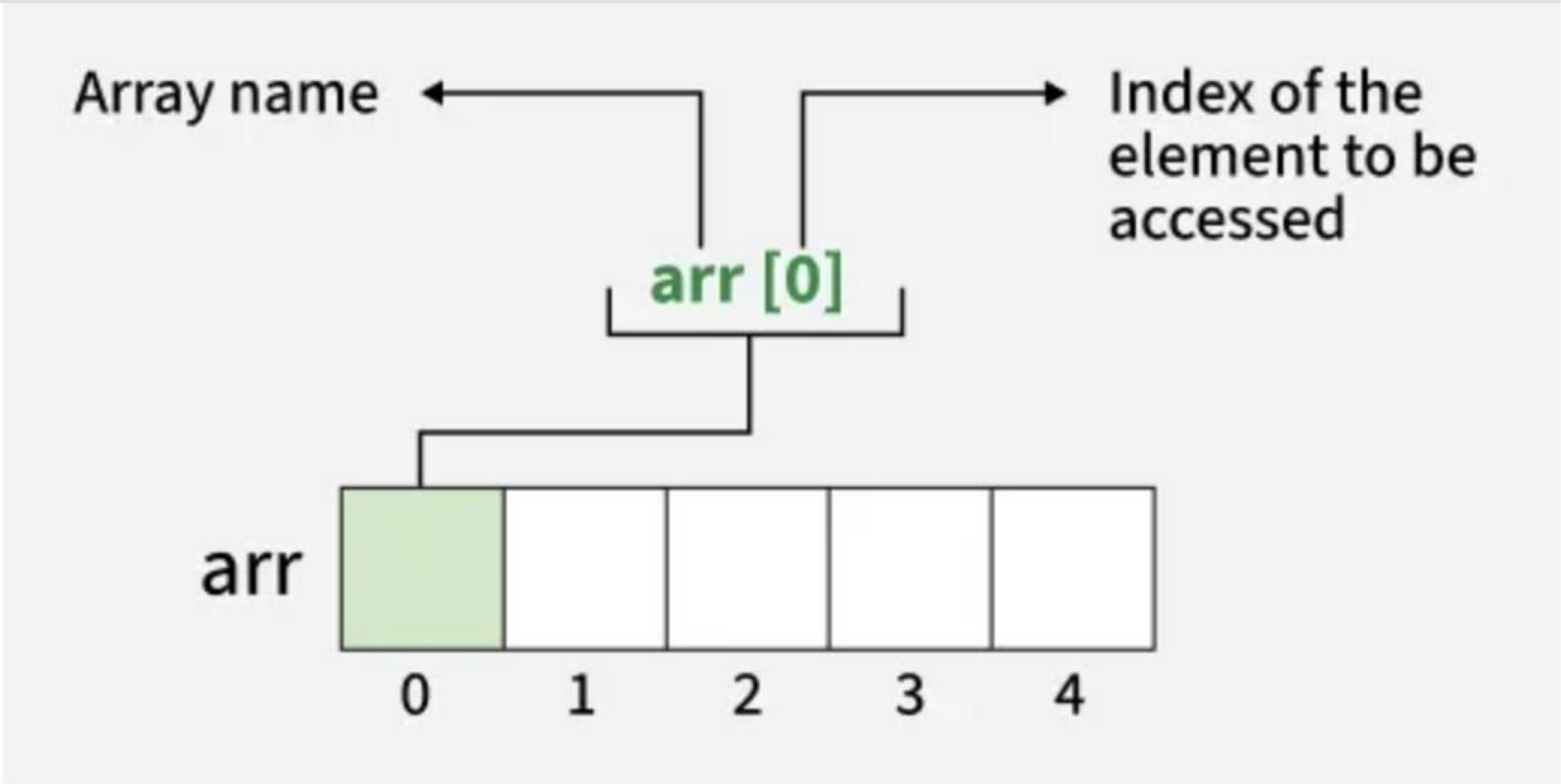
```
type arrayName [ arraySize ];
```

This is called a single-dimension array. The **arraySize** must be an integer constant greater than zero and **type** can be any valid C++ data type. For example, to declare a 10-element array called balance of type double, use this statement:



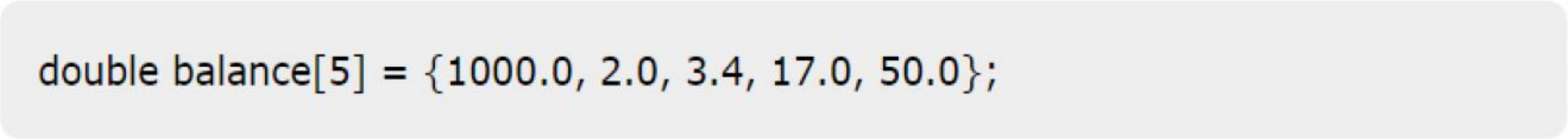
```
double balance[10];
```

Declaring an Array



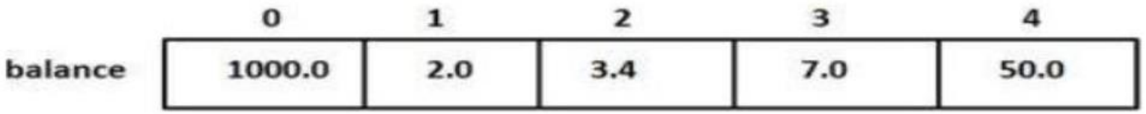
Initializing Arrays

You can initialize C++ array elements either one by one or using a single statement as follows:



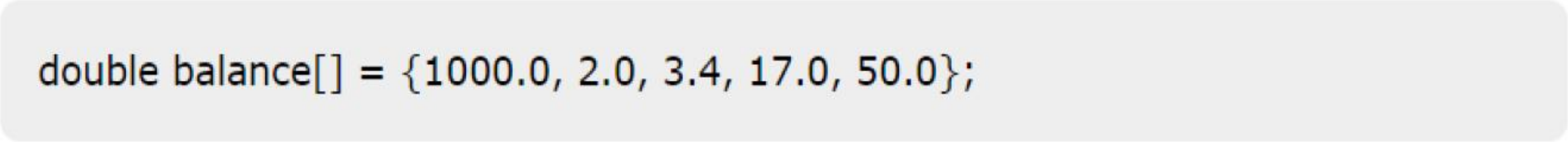
```
double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

The number of values between braces { } can not be larger than the number of elements that we declare for the array between square brackets [ ].



	0	1	2	3	4
balance	1000.0	2.0	3.4	7.0	50.0

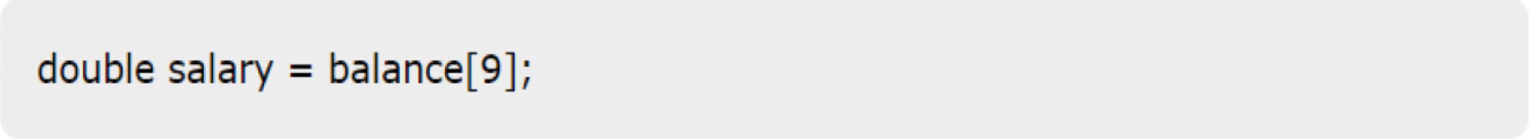
If you omit the size of the array, an array just big enough to hold the initialization is created. Therefore, if you write:



```
double balance[] = {1000.0, 2.0, 3.4, 17.0, 50.0};
```

Assigning and Accessing a single Element

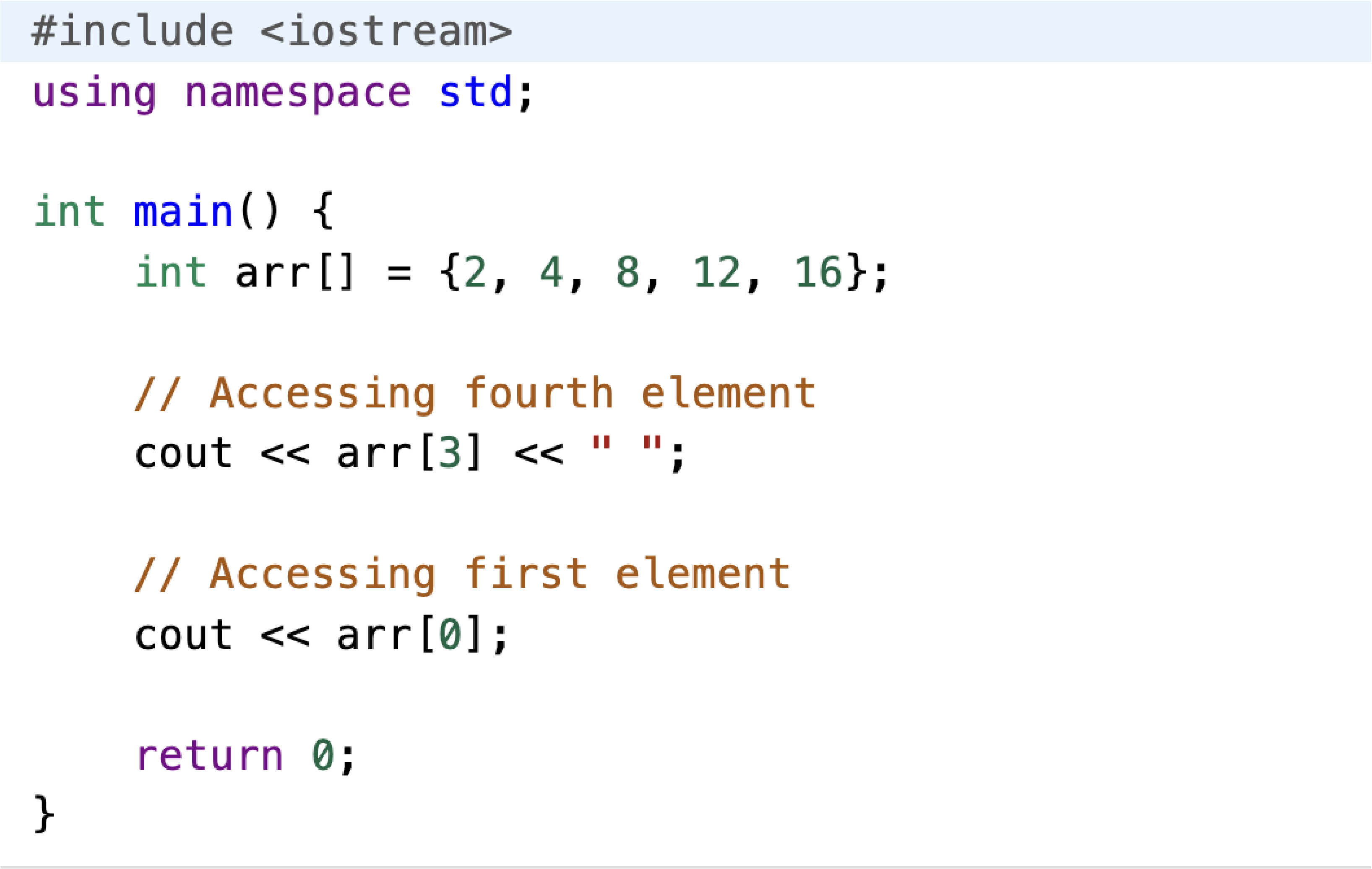
An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array. For example:



```
double salary = balance[9];
```

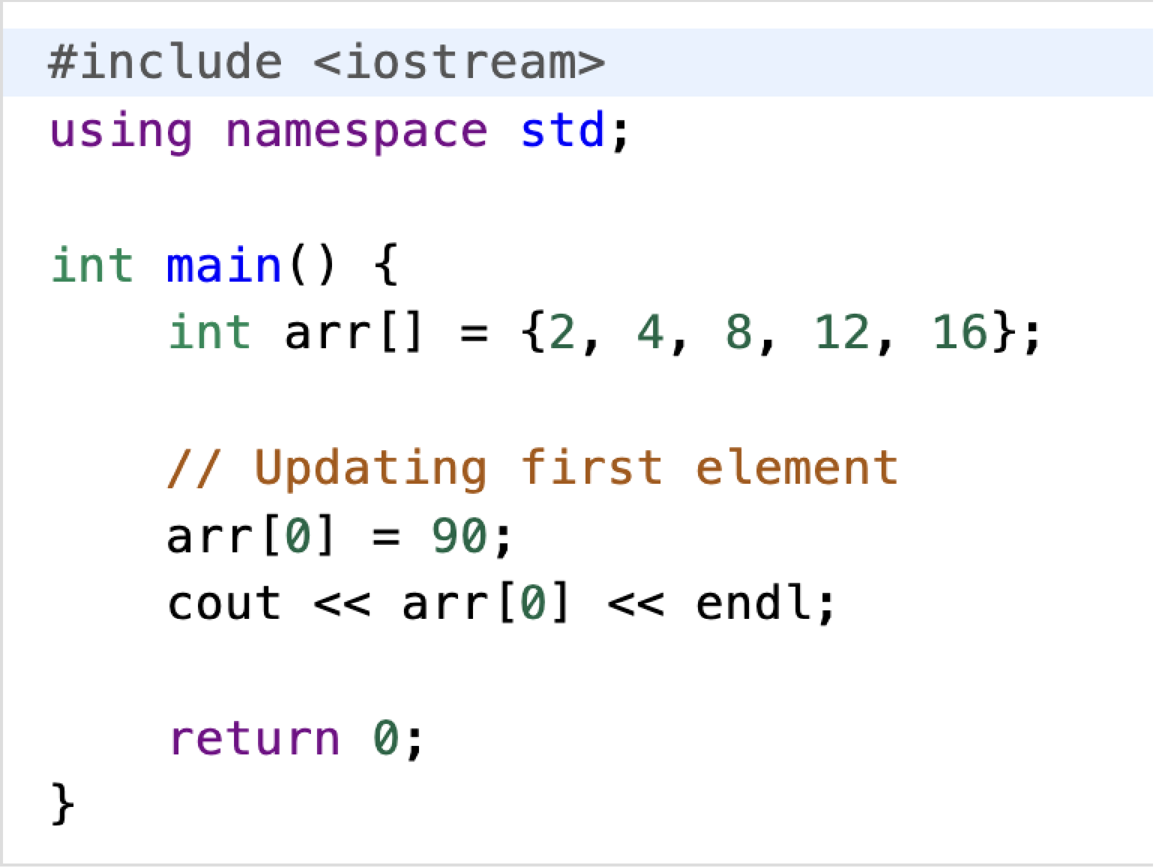
The above statement will take 10th element from the array and assign the value to salary variable.

Example

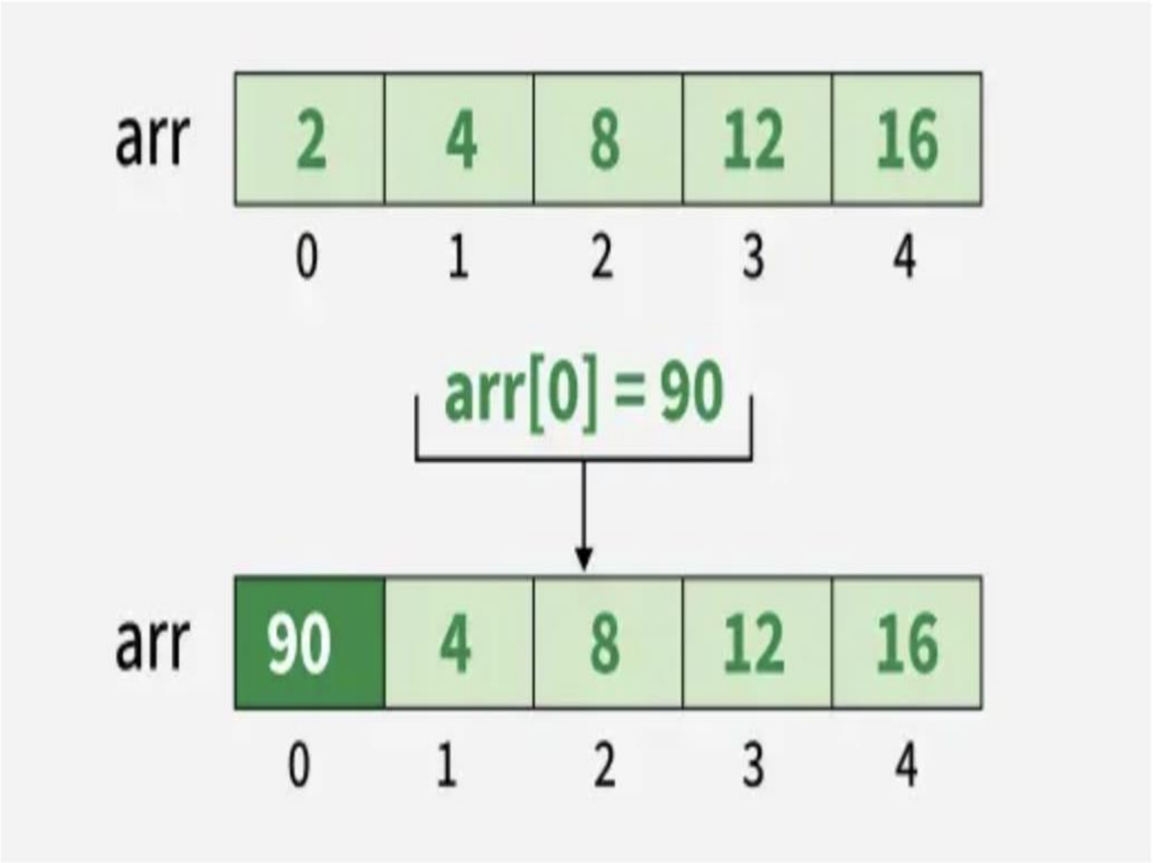


```cpp
#include <iostream>
using namespace std;

int main() {
    int arr[] = {2, 4, 8, 12, 16};

    // Accessing fourth element
    cout << arr[3] << " ";

    // Accessing first element
    cout << arr[0];

    return 0;
}
```

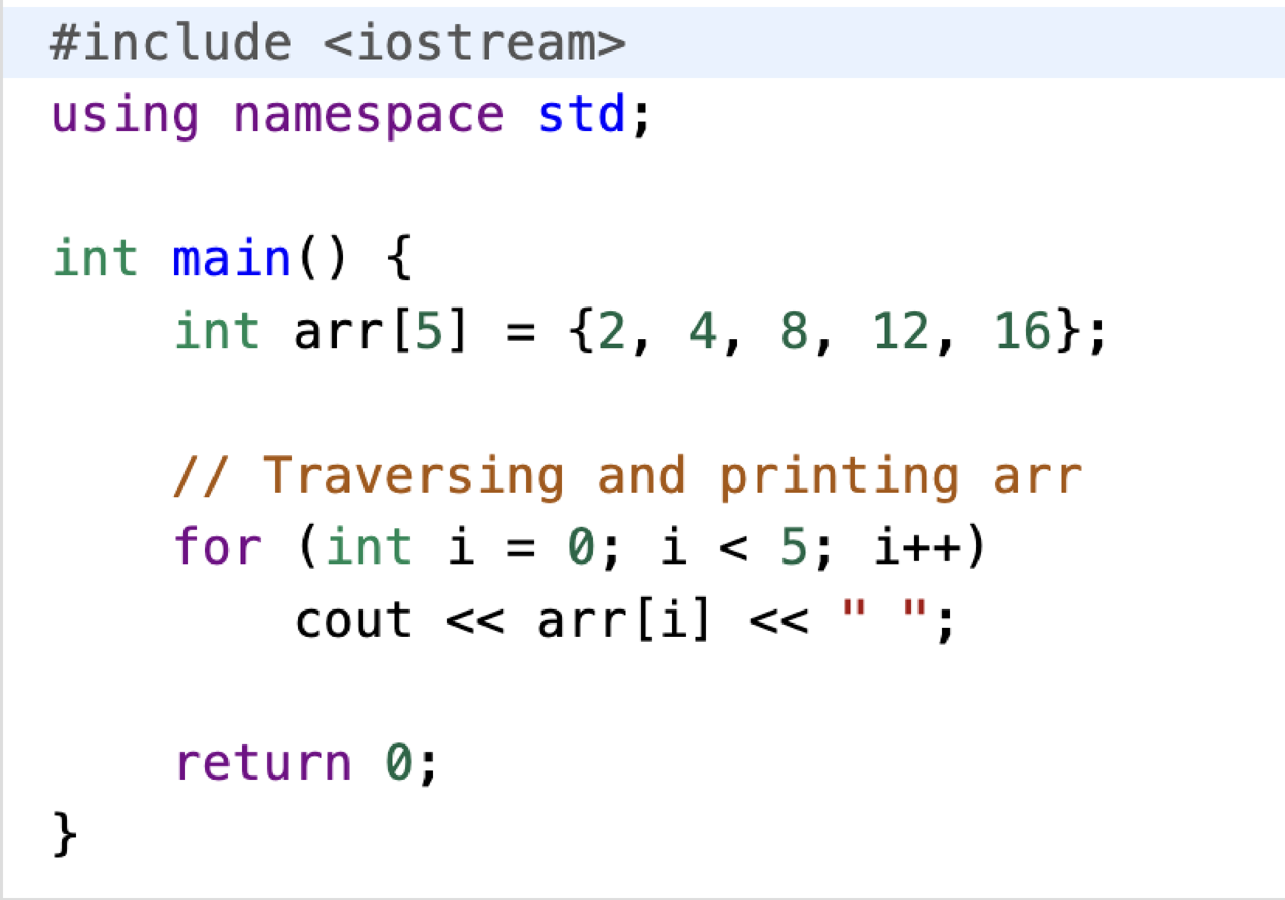
Update Array Values



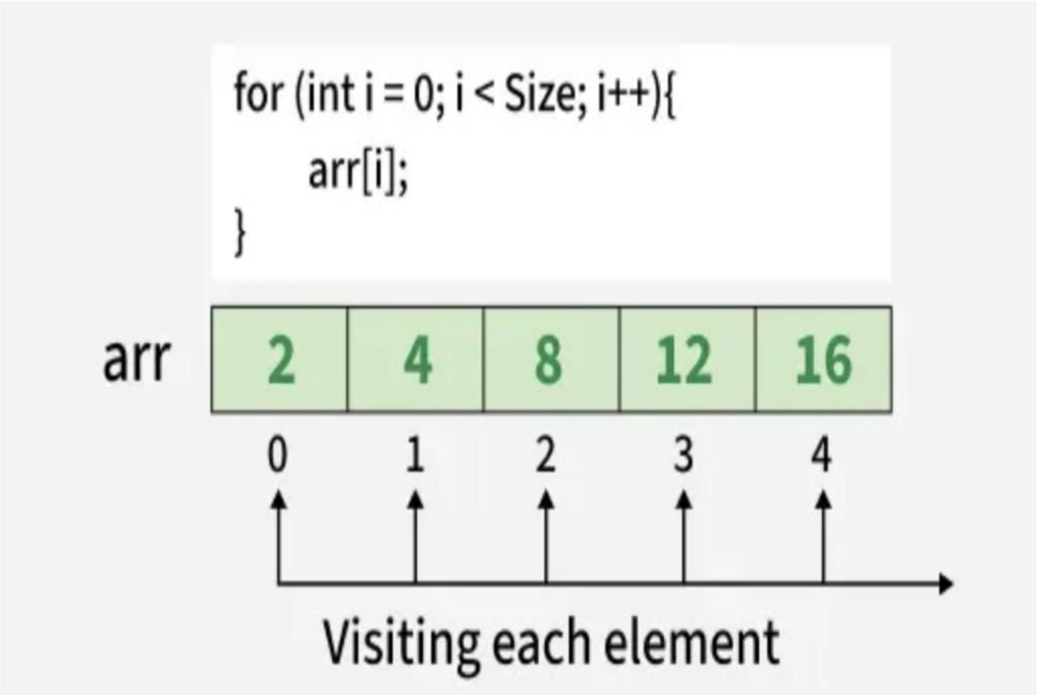
```
#include <iostream>
using namespace std;

int main() {
    int arr[] = {2, 4, 8, 12, 16};

    // Updating first element
    arr[0] = 90;
    cout << arr[0] << endl;

    return 0;
}
```



Traversing an Array



```cpp
#include <iostream>
using namespace std;

int main() {
    int arr[5] = {2, 4, 8, 12, 16};

    // Traversing and printing arr
    for (int i = 0; i < 5; i++)
        cout << arr[i] << " ";

    return 0;
}
```

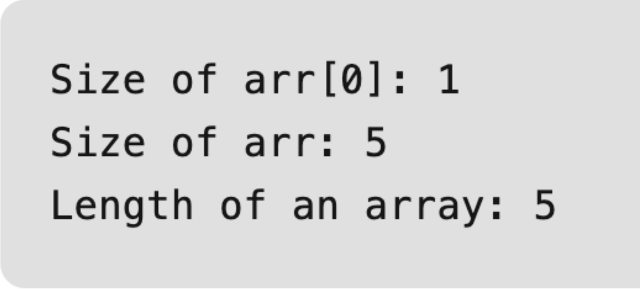


```
for (int i = 0; i < Size; i++){
    arr[i];
}
```

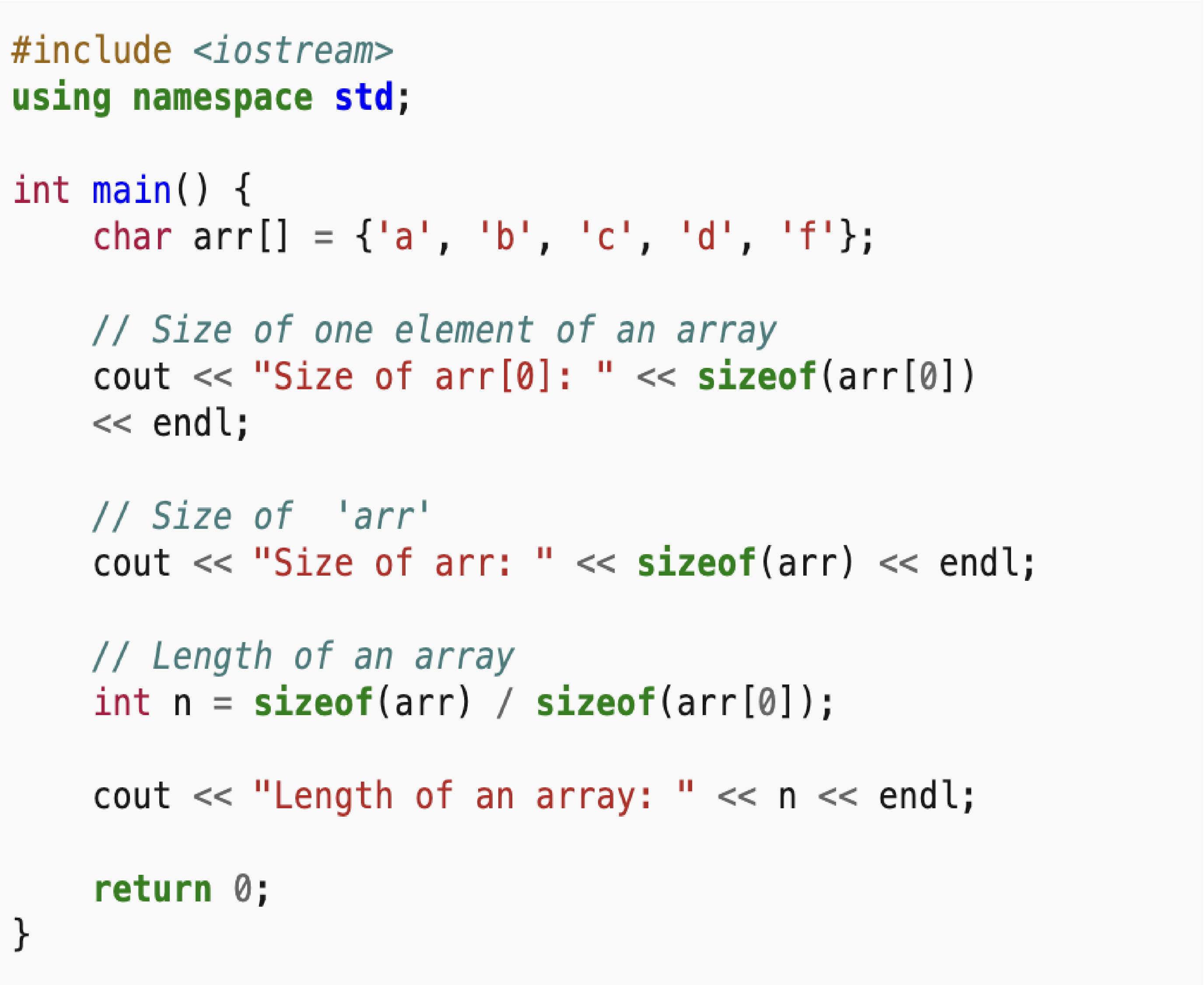
Size of the Array

The size of the array refers to the number of elements that can be stored in the array. The array does not contain the information about its size but we can extract the size using sizeof() operator.

Output

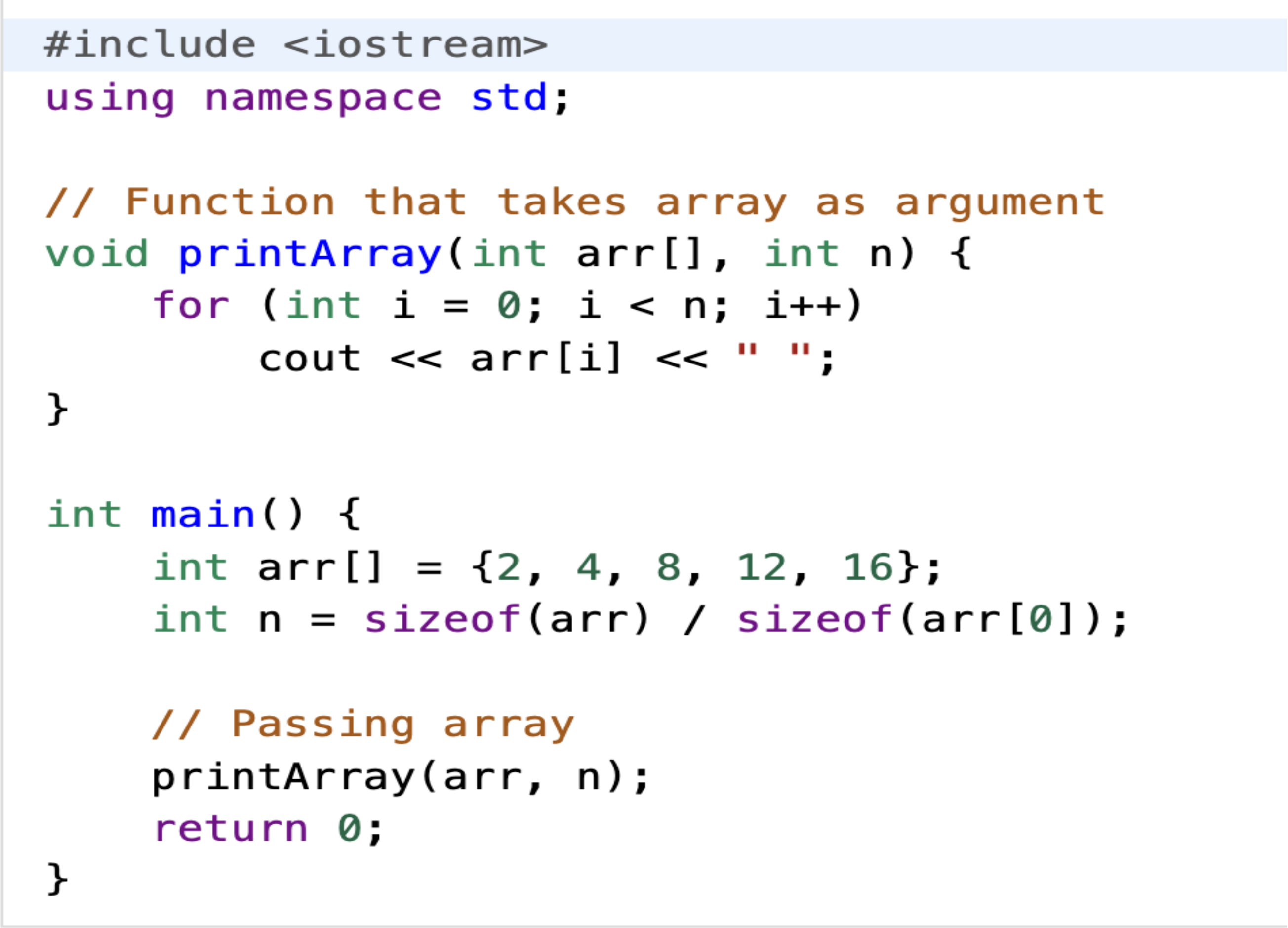


```
Size of arr[0]: 1
Size of arr: 5
Length of an array: 5
```

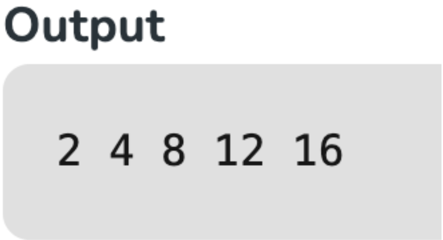


```cpp
#include <iostream>
using namespace std;

int main() {
    char arr[] = {'a', 'b', 'c', 'd', 'f'};

    // Size of one element of an array
    cout << "Size of arr[0]: " << sizeof(arr[0])
    << endl;

    // Size of  'arr'
    cout << "Size of arr: " << sizeof(arr) << endl;

    // Length of an array
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Length of an array: " << n << endl;

    return 0;
}
```

Pass Arrays to Function



```cpp
#include <iostream>
using namespace std;

// Function that takes array as argument
void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
}

int main() {
    int arr[] = {2, 4, 8, 12, 16};
    int n = sizeof(arr) / sizeof(arr[0]);

    // Passing array
    printArray(arr, n);
    return 0;
}
```



Output

```
2 4 8 12 16
```

Acknowledgment

- Content of these slides are taken from:
  - https://www.geeksforgeeks.org/
  - https://www.tutorialspoint.com/
  - https://www.programiz.com/
  - https://www.w3schools.com/