What are Arrays?

This lesson introduces arrays, indexing, accessing arrays and explains how to change values in an array

We'll cover the following

Definition

An *array* is a collection of *similar* data types under the same **name**.

In C++, arrays are *declared* in the following way:

```
dataType arrayName[arraySize];
```

Let's take a look at an example:

```
int arr[5];    //int is the datatype, arr is the name, 5 is the size of array
```

The above statement declares a **static** *array* of **5** *elements*, which can be accessed individually.

> **Note:** Although `arr` behaves like a *pointer*, its value **cannot** be changed as it references a *specific* region in memory.

Initializing Arrays

Now let's learn how to *initialize* an *array* using the example below:

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

int main() {
  int arr[5] = {19, 10, 5, 6, 14}; //initializing the array with 5 values
  cout << "The value of arrr[0], that is, the first value in the array is: " << arr[0] << endl;
  cout<< "The value of arrr[1], that is, the second value in the array is: " << arr[1] << endl;
  cout<< "The value of arrr[2], that is, the third value in the array is: " << arr[2] << endl;
  cout<< "The value of arrr[3], that is, the fourth value in the array is: " << arr[3] << endl;
  cout<< "The value of arrr[4], that is, the fifth value in the array is: " << arr[4] << endl;
  int arr2[] = {1,2,3,4}; //we don't specify the size and the compiler assumes a size of 4

  //we calculate size of the arr2 using the inbuilt sizeof function
  //divide the total size of the array by the size of the array element to claculate the size
  cout << "The size of arr2 is: "<<sizeof(arr2)/sizeof(arr2[0])<<endl;

}
```

Example Explanation

In the above example, in line **5** we have an *array* `arr` of size **5**. We use *indexing* to refer to the *location* of the values in an *array*. For example:

- The value **19** is present at the index **0** of the *array*. This is written as `arr[0]` = **19**. Here, by using `arr[0]` we are referring to the location of **19**.
- Similarly, the *index* of the value **10** in the *array* will be **1**. It'll be written `arr[1]` = **10**.
- The *index* of the value **5** is **2** in the *array*. So `arr[2]` = **5**.

> **Note:** Arrays have **0** as the *first* index not **1**.

- Similarly, there is another way to *initialize* an *array* where we **don't** *specify* the **size** and the *compiler* automatically assumes the *size* depending on the number of values stored, this is shown is the line **11** in above example.

Accessing Array Values

We use *indexing* to access arrays values just like we did in the example above.

Suppose you declared an *array* of **10** elements.

You can use the array members from `arr[0]` to `arr[9]`.

> **Note:** If you try to access array elements *outside of its bound*, let's say `arr[14]`, the compiler may not show any error. However, this may cause *unexpected* output (*undefined behavior*).

Look at the example below for better understanding.

```
#include <iostream>
using namespace std;
int main(){
  int arr[10] = {1,2,3,4,5,6,7,8,9,10};
  cout << "The value of arr[16] is: "<< arr[16] <<endl;
  return 0;
}
```

As you can see from the output above we get a garbage value at `arr[16]` which is an index that is *out of bounds*.

Using for loops to access array values

We can also iterate over the whole array and access any value easily using a `for` loop.

Let's take a look at the example below.

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

int main() {
  int arr[5] = {}; // no values stored in array by default
  int num = 1;
  for (int i=0; i<5; i++){
    arr[i] = num; //setting index i of array arr equal to num
    num++;        //incrementing num
    cout << "The value of arr["<<i<<"] is equal to: "<< arr[i]<<endl;
  }
  return 0;
}
```

Changing Array values

In order to change the *array* values, we first access them using *indexing* and then change the value at that specified *index*.

Take a look at the example below to better understand this concept:

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

int main() {
  int arr[5] = {19, 10, 5, 6, 14}; //initializing the array with 5 values
  cout << "The value of arr[1] originally is: "<<arr[1] <<endl;
  arr[1] = 20; //we access the index 1 which has the value 10 stored originally and then change th
  cout << "The value of arr[1] after is: " << arr[1] <<endl;
  return 0;
}
```

As you can see in the example above, originally the value of `arr[1]` is **10**. In line **7** we access the value at *index* **1** and then set it **20**. You can see that the value gets changed from the output of the code above.

These were some of the basics on arrays in C++. Let's delve into other details about arrays in the next lesson.