Array class in C++

The introduction of array class from C++11 has offered a better alternative for C-style arrays. The advantages of array class over C-style array are :-

- Array classes knows its own size, whereas C-style arrays lack this property. So when passing to functions, we don't need to pass size of Array as a separate parameter.
- With C-style array there is more risk of array being decayed into a pointer. Array classes don't decay into pointers
- Array classes are generally more efficient, light-weight and reliable than C-style arrays.

Operations on array :-
1. at() :- This function is used to access the elements of array.
2. get() :- This function is also used to access the elements of array. This function is not the member of array class but overloaded function from class tuple.
3. operator[] :- This is similar to C-style arrays. This method is also used to access array elements.

```
// C++ code to demonstrate working of array,
// to() and get()
#include<iostream>
#include<array> // for array, at()
#include<tuple> // for get()
using namespace std;
int main()
{
    // Initializing the array elements
    array<int,6> ar = {1, 2, 3, 4, 5, 6};

    // Printing array elements using at()
    cout << "The array elements are (using at()) : ";
    for ( int i=0; i<6; i++)
    cout << ar.at(i) << " ";
    cout << endl;

    // Printing array elements using get()
    cout << "The array elements are (using get()) : ";
    cout << get<0>(ar) << " " << get<1>(ar) << " ";
    cout << get<2>(ar) << " " << get<3>(ar) << " ";
    cout << get<4>(ar) << " " << get<5>(ar) << " ";
    cout << endl;

    // Printing array elements using operator[]
    cout << "The array elements are (using operator[]) : ";
    for ( int i=0; i<6; i++)
    cout << ar[i] << " ";
    cout << endl;

    return 0;

}
```

Output:

```
The array elemets are (using at()) : 1 2 3 4 5 6
The array elemets are (using get()) : 1 2 3 4 5 6
The array elements are (using operator[]) : 1 2 3 4 5 6
```

4. front() :- This returns the first element of array.

5. back() :- This returns the last element of array.

```
// C++ code to demonstrate working of
// front() and back()
#include<iostream>
#include<array> // for front() and back()
using namespace std;
int main()
{
    // Initializing the array elements
    array<int,6> ar = {1, 2, 3, 4, 5, 6};

    // Printing first element of array
    cout << "First element of array is : ";
    cout << ar.front() << endl;

    // Printing last element of array
    cout << "Last element of array is : ";
    cout << ar.back() << endl;

    return 0;

}
```

Output:

```
First element of array is : 1
Last element of array is : 6
```

6. size() :- It returns the number of elements in array. This is a property that C-style arrays lack.

7. max_size() :- It returns the maximum number of elements array can hold i.e, the size with which array is declared. The size() and max_size() return the same value.

```
// C++ code to demonstrate working of
// size() and max_size()
#include<iostream>
#include<array> // for size() and max_size()
using namespace std;
int main()
{
    // Initializing the array elements
    array<int,6> ar = {1, 2, 3, 4, 5, 6};

    // Printing number of array elements
    cout << "The number of array elements is : ";
    cout << ar.size() << endl;

    // Printing maximum elements array can hold
    cout << "Maximum elements array can hold is : ";
    cout << ar.max_size() << endl;

    return 0;

}
```

Output:

```
The number of array elements is : 6
Maximum elements array can hold is : 6
```

8. swap() :- The swap() swaps all elements of one array with other.

```cpp
// C++ code to demonstrate working of swap()
#include<iostream>
#include<array> // for swap() and array
using namespace std;
int main()
{

    // Initializing 1st array
    array<int,6> ar = {1, 2, 3, 4, 5, 6};

    // Initializing 2nd array
    array<int,6> ar1 = {7, 8, 9, 10, 11, 12};

    // Printing 1st and 2nd array before swapping
    cout << "The first array elements before swapping are : ";
    for (int i=0; i<6; i++)
    cout << ar[i] << " ";
    cout << endl;
    cout << "The second array elements before swapping are : ";
    for (int i=0; i<6; i++)
    cout << ar1[i] << " ";
    cout << endl;

    // Swapping ar1 values with ar
    ar.swap(ar1);

    // Printing 1st and 2nd array after swapping
    cout << "The first array elements after swapping are : ";
    for (int i=0; i<6; i++)
    cout << ar[i] << " ";
    cout << endl;
    cout << "The second array elements after swapping are : ";
    for (int i=0; i<6; i++)
    cout << ar1[i] << " ";
    cout << endl;

    return 0;

}
```

Output:

```
The first array elements before swapping are : 1 2 3 4 5 6
The second array elements before swapping are : 7 8 9 10 11 12
The first array elements after swapping are : 7 8 9 10 11 12
The second array elements after swapping are : 1 2 3 4 5 6
```

9. empty() :- This function returns true when the array size is zero else returns false.

10. fill() :- This function is used to fill the entire array with a particular value.

```
// C++ code to demonstrate working of empty()
// and fill()
#include<iostream>
#include<array> // for fill() and empty()
using namespace std;
int main()
{

    // Declaring 1st array
    array<int,6> ar;

    // Declaring 2nd array
    array<int,0> ar1;

    // Checking size of array if it is empty
    ar1.empty()? cout << "Array empty":
        cout << "Array not empty";
    cout << endl;

    // Filling array with 0
    ar.fill(0);

    // Displaying array after filling
    cout << "Array after filling operation is : ";
    for ( int i=0; i<6; i++)
        cout << ar[i] << " ";

    return 0;

}
```

Output:

```
Array empty
Array after filling operation is : 0 0 0 0 0 0
```

Recommended Posts:

Implementation of Array class in JavaScript
How to create a dynamic 2D array inside a class in C++ ?
How to convert a class to another class type in C++?
Find original array from encrypted array (An array of sums of other elements)
Find an element in array such that sum of left array is equal to sum of right array
std::any Class in C++
std:: valarray class in C++
std::string class in C++
Structure vs class in C++
std::hash class in C++ STL
C++ String Class and its Applications | Set 2
Difference between namespace and class

Virtual base class in C++

Friend class and function in C++

C++ string class and its applications

Improved By : siddharthx_07

Article Tags : Arrays C C++ cpp-array CPP-Library STL

Practice Tags : Arrays STL C CPP

4

1.4

☐ To-do ☐ Done

Based on **18** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Load Comments