GeeksforGeeks

Related Articles Save for later

Advantages of vector over array in C++

Difficulty Level : Medium ● Last Updated : 30 Oct, 2018

We have already discussed arrays and vectors. In this post, we will discuss advantages of vector over normal array.

Advantages of Vector over arrays :

1. Vector is ***template class*** and is ***C++ only construct*** whereas arrays are ***built-in language construct*** and present in both C and C++.
2. Vector are implemented as ***dynamic arrays*** **with** ***list interface*** whereas arrays can be implemented as ***statically or dynamically*** with ***primitive data type*** interface.

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int array[100]; // Static Implementation
    int* arr = new int[100]; // Dynamic Implementation
    vector<int> v; // Vector's Implementation
    return 0;
}
```

3. **Size of arrays are *fixed*** whereas the **vectors are *resizable*** i.e they can grow and shrink as vectors are allocated on heap memory.

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int array[100]; // Static Implementation

    cout << "Size of Array " << sizeof(array) / sizeof(array[0]) << "\n";

    vector<int> v; // Vector's Implementation

    // Inserting Values in Vector
    v.push_back(3);
    v.push_back(4);
    v.push_back(5);

    cout << "Size of vector Before Removal=" << v.size() << "\n";

    // Output Values of vector
    for (auto it : v)
        cout << it << " ";

    v.erase(v.begin() + 2); // Remove 3rd element

    cout << "\nSize of vector After removal=" << v.size() << "\n";

    // Output Values of vector
    for (auto it : v)
        cout << it << " ";

    return 0;
}
```

Output:

```
Size of Array 100
Size of vector Before Removal=5
1 2 3 4 5
Size of vector After removal=4
1 2 4 5
```

4. Arrays **have to be *deallocated explicitly*** if defined dynamically whereas vectors are ***automatically de-allocated*** from heap memory.

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int* arr = new int[100]; // Dynamic Implementation
    delete[] arr; // array Explicitly deallocated

    vector<int> v; // Automatic deallocation when variable goes out of scop
    return 0;
}
```

5. Size of array **cannot be determined** if **dynamically allocated** whereas Size of the vector can be determined in **O(1) time**.
6. When arrays are passed to a function, a **separate parameter for size is also passed** whereas in case of passing a vector to a function, there is no such need as **vector maintains variables which keeps track of size of container at all**

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int* arr = new int[100]; // Dynamic Implementation

    cout << "Size of array= ";
    cout << sizeof(arr) / sizeof(*arr) << "\n"; // Pointer cannot be used t
    // block pointed by it
    return 0;
}
```

Output:

```
Size of array= 2
```

7. When array becomes full and new elements are inserted; **no reallocation is done implicitly** whereas When vector becomes larger than its capacity, reallocation is done implicitly.
8. **Arrays *cannot be returned unless dynamically allocated* from a function** whereas v**ectors *can be returned* from a function**.

```
// Program to demonstrate arrays cannot be returned
#include <bits/stdc++.h>
using namespace std;

int* getValues()
{

    int arr[10]; // Array defined locally
    for (int i = 0; i < 10; i++) // Putting Values in array
        arr[i] = i + 1;

    return arr; // returning pointer to array
}

// main function
int main()
{

    int* array; // pointer of int type

    array = getValues(); // Call function to get arr

    for (int i = 0; i < 10; i++) { // Printing Values
        cout << "*(array + " << i << ") : ";
        cout << *(array + i) << endl;
    }

    return 0;
```

Output:

```
warning: address of local variable 'arr' returned [-Wreturn-local-ad
Segmentation Fault (SIGSEGV)
```

```
// Program to demonstrate vector can be returned
#include <bits/stdc++.h>
using namespace std;

// Function returning vector
vector<int> getValues()
{

    vector<int> v; // Vector defined locally
    for (int i = 0; i < 10; i++) // Inserting values in Vector
        v.push_back(i + 1);

    return v; // returning pointer to array
}

// main function
int main()
{

    vector<int> get;

    get = getValues(); // Call function to get v

    // Output Values of vector
    for (auto it : get)
        cout << it << " ";

    return 0;
}
```

```
1 2 3 4 5 6 7 8 9 10
```

9. Arrays cannot be copied or assigned directly whereas Vectors can be copied or assigned directly.

```
#include <bits/stdc++.h>
using namespace std;

// main function
int main()
{
    vector<int> v; // Vector defined locally
    for (int i = 0; i < 10; i++)
        v.push_back(i + 1);

    vector<int> get;

    get = v; // Copying vector v into vector get

    cout << "vector get:\n";
    for (auto it : get)
        cout << it << " ";

    int arr[10];
    for (int i = 0; i < 10; i++) // Putting Values in array
        arr[i] = i + 1;

    int copyArr[10];

    copyArr = arr; // Error

    return 0;
}
```

Output:

```
vector get:
1 2 3 4 5 6 7 8 9 10

error: invalid array assignment
     copyArr=arr;
```

Like 0

Previous

Python | Count the Number of matching characters in a pair of string

Next

Java AWT | MouseInfo and PointerInfo

RECOMMENDED ARTICLES

Page : 1 2 3

01 vector::front() and vector::back() in C++ STL
22, Dec 17

02 vector::push_back() and vector::pop_back() in C++ STL
22, Dec 17

03 vector::operator= and vector::operator[ ] in C++ STL
09, Jan 18

04 vector::crend() & vector::crbegin() with example
17, Jan 18

05 vector :: cbegin() and vector :: cend() in C++ STL
01, May 18

06 How to flatten a Vector of Vectors or 2D Vector in C++
07, Apr 20

07 vector::empty() and vector::size() in C++ STL
22, Dec 17

08 vector::begin() and vector::end() in C++ STL
22, Jan 18

Article Contributed By :

Abhishek rajput
@Abhishek rajput

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Improved By : nikhil_jsk

Article Tags : cpp-array, cpp-vector, Arrays, Data Structures, Difference Between

Practice Tags : Data Structures, Arrays

Improve Article Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments