Sort () Function in C++ STL

C++ STL provides a built-in function sort() that sorts a vector or array (items with random access).

Syntax to sort an Array:

```
sort(arr, arr+n);

Here, arr is the name or base address of the array
and, n is the size of the array.
```

Syntax to sort a Vector:

```
sort(vec.begin(), vec.end());

Here, vec is the name of the vector.
```

Below program illustrate the sort function:

```cpp
// C++ program to demonstrate default behaviour of sort() in STL.
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int arr[] = {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};// Sorting Array
    int n = sizeof(arr)/sizeof(arr[0]);
    sort(arr, arr+n);
    cout << "Array after sorting is : \n";
    for (int i = 0; i < n; ++i)
        cout << arr[i] << " ";
    // Sorting Vector
    vector<int> vec = {1,2,4,5,3};
    sort(vec.begin(), vec.end());
    cout << "\nVector after sorting is : \n";
    for (int i = 0; i < vec.size(); ++i)
        cout << vec[i] << " ";
    return 0;
}

```

Output :

```
Array after sorting is :
0 1 2 3 4 5 6 7 8 9
Vector after sorting is :
1 2 3 4 5
```

So by default, sort() function sorts an array in ascending order.

How to sort in descending order?

The sort() function takes a third parameter that is used to specify the order in which elements are to be sorted. We can pass "greater()" function to sort in descending order. This function does comparison in a way that puts greater element before.

```cpp

// C++ program to demonstrate descending order
// sort using greater<>().
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int arr[] = {1, 5, 8, 9, 6, 7, 3, 4, 2, 0};
    int n = sizeof(arr)/sizeof(arr[0]);
    sort(arr, arr+n, greater<int>());
    cout << "Array after sorting : \n";
    for (int i = 0; i < n; ++i)
        cout << arr[i] << " ";
    return 0;
}
```

Output:

```
Array after sorting :

9 8 7 6 5 4 3 2 1 0
```

How to sort in particular order?

We can also write our own comparator function and pass it as a third parameter.

```cpp
// A C++ program to demonstrate STL sort() using our own comparator
#include<bits/stdc++.h>
using namespace std;
struct Interval       // An interval has start time and end time
{
    int start, end;
};// Compares two intervals according to staring times.
bool compareInterval(Interval i1, Interval i2)
{
    return (i1.start < i2.start);
}
int main()
{
    Interval arr[] =  { {6,8}, {1,9}, {2,4}, {4,7} };
    int n = sizeof(arr)/sizeof(arr[0]);
    // sort the intervals in increasing order of start time
    sort(arr, arr+n, compareInterval);
    cout << "Intervals sorted by start time : \n";
    for (int i=0; i<n; i++)
       cout << "[" << arr[i].start << "," << arr[i].end << "] ";
    return 0;
}

```

Output:

```
Intervals sorted by start time :

[1,9] [2,4] [4,7] [6,8]
```