Sorting a Character Array

C++ STL provides a function sort that sorts a vector or array. It generally takes two parameters, the first one being the point of the array/vector from where the sorting needs to begin and the second parameter being the length up to which we want the array/vector to get sorted.

But in cases where you want to sort the elements lexicographically then you can provide the third parameter which is optional.

By default, the sort() function sorts the elements in ascending order.

Example: Default behaviour of sort() in STL.

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
	char arr[] = { 'j', 'i', 'h', 'g', 'f', 'e', 'd', 'c', 'b', 'a' };
	int n = sizeof(arr) / sizeof(arr[0]);

	/*Here we take two parameters, the beginning of the array and the length n
upto which we want the array to be sorted*/

	sort(arr, arr + n);

	cout << "\nArray after sorting using default sort is : \n";
	for (int i = 0; i < n; ++i)
		cout << arr[i] << " ";

	return 0;
}
```

How to sort in descending order?

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

Example: To demonstrate descending order sort using greater<>().

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
	char arr[] = { 'a', 'x', 's', 'e', 'w', 'q', 'p', 'i', 'y', 'b' };
	int n = sizeof(arr) / sizeof(arr[0]);

	sort(arr, arr + n, greater<char>());

	cout << "Array after sorting : \n";
	for (int i = 0; i < n; ++i)
		cout << arr[i] << " ";

	return 0;
}
```