CUSTOM COMPARATOR

24 September 2023 09:22

23/09/2023

# Custom Comparator

✅ CUTOM COMPARATOR

C++ STL Concept

↳ 1. SORT A VECTOR: When you want to sort a vector using C++ STL sort data structure then syntax is

```
// SORT A VECTOR
vector<int> v;

// Syntax of sorting
sort(v.begin(), v.end());
```

increasing order sorting by *default*

2. SORT A ARRAY: When you want to sort a vector using C++ STL sort data structure then syntax is

```
// SORT A ARRAY:
int arr[];

// Syntax of sorting
sort(arr, arr+n);

/*
Where
- arr is a base address of array
- n is length of array
*/
```

increasing order sorting by *default*

## 1. SORT A VECTOR:

```
// ✅CUTOM COMPARATOR 01: SORT A VECTOR
#include <algorithm>
```

## 1. SORT A VECTOR:

```cpp
// ✅CUTOM COMPARATOR 01: SORT A VECTOR
#include <algorithm>
#include <iostream>
#include <vector>

using namespace std;

// Printing Method
void print(vector<int> &v) {
  for (int i = 0; i < v.size(); ++i) {
    cout << v[i] << " ";
  }
  cout << endl;
}

// Custom Comparator ki return value always true ya false hoti hai
bool myComparator1(int &a, int &b) {
    return a > b; // decreasing order sorting
}

bool myComparator2(int &a, int &b) {
    return a < b; // increasing order sorting
}

int main() {
  vector<int> v = {44, 55, 22, 11, 33};

  cout<<"Vector"<<endl;
  print(v);

  cout<<"Increasing order sorting by default"<<endl;
  sort(v.begin(), v.end());
  print(v);

  cout<<"Decreasing order sorting by my comparator 1"<<endl;
  sort(v.begin(), v.end(), myComparator1);
  print(v);

  cout<<"Increasing order sorting by my comparator 2"<<endl;
  sort(v.begin(), v.end(), myComparator2);
  print(v);

  return 0;
}

/*
Vector
44 55 22 11 33
Increasing order sorting by default
11 22 33 44 55
Decreasing order sorting by my comparator 1
55 44 33 22 11
Increasing order sorting by my comparator 2
11 22 33 44 55
*/
```

```cpp
// Custom comparator for decreasing order sorting
bool myComparator1(int &a, int &b) {
    return a > b;
}


// Custom comparator for increasing order sorting
bool myComparator2(int &a, int &b) {
    return a < b;
}
```

## 2. SORT VECTOR OF VECTOR:

*Declaration Syntax of 2D vector*

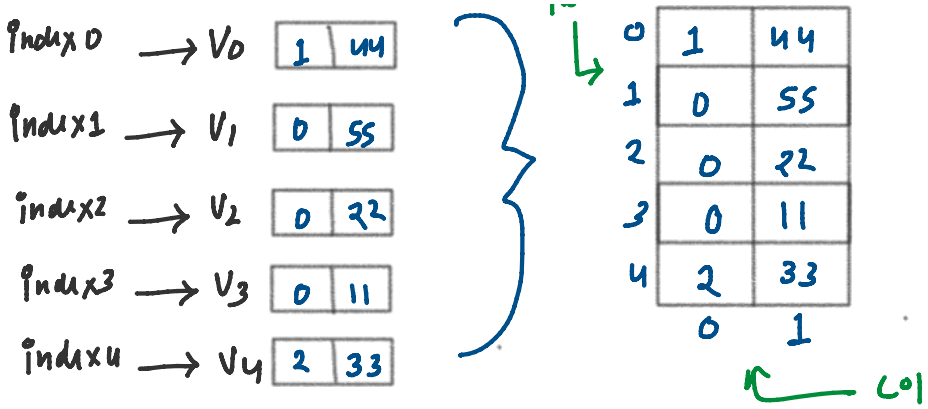
vector< vector<int> > v;

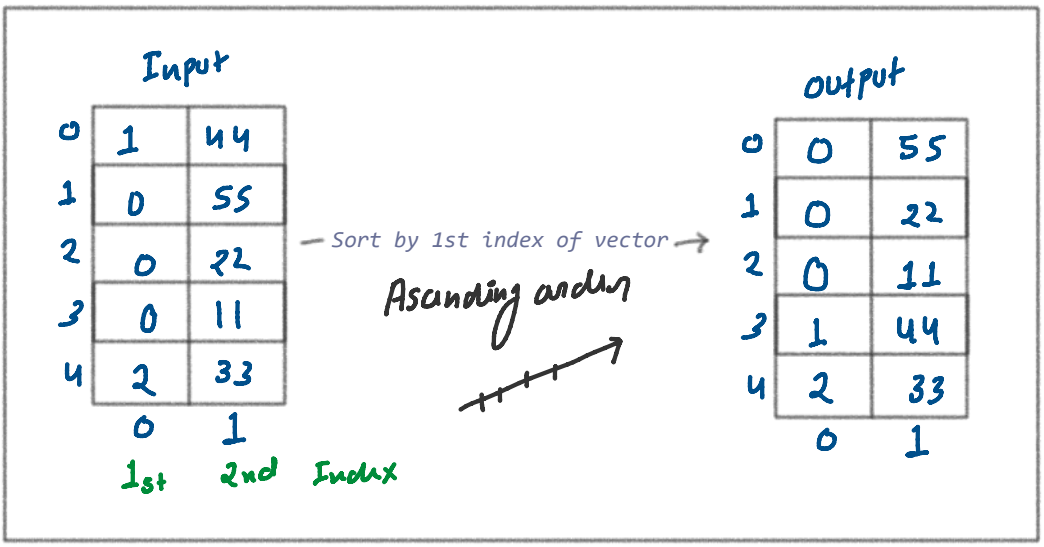
n=5

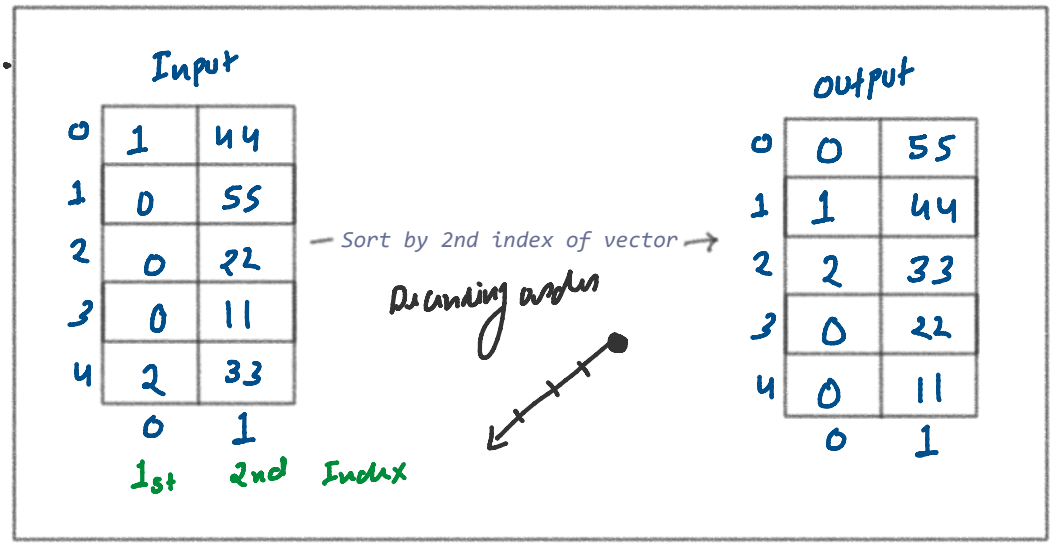
Index 0 → V0 | 1 | 44 |

Row

| | | |
|---|---|---|
| 0 | 1 | 44 |
| 1 | [illegible] | 55 |


index 0 → V0 1 44
index1 → V1 0 55
index2 → V2 0 22
index3 → V3 0 11
index4 → V4 2 33
0 1 44
1 0 55
2 0 22
3 0 11
4 2 33
0 1
Col


Input
0 1 44
1 0 55
2 0 22
3 0 11
4 2 33
0 1
1st 2nd Index
Sort by 1st index of vector
Ascending order
output
0 0 55
1 0 22
2 0 11
3 1 44
4 2 33
0 1


Input
0 1 44
1 0 55
2 0 22
3 0 11
4 2 33
0 1
1st 2nd Index
Sort by 2nd index of vector
Descending order
output
0 0 55
1 1 44
2 2 33
3 0 22
4 0 11
0 1

```cpp
// ✅CUTOM COMPARATOR 02: SORT VECTOR OF VECTOR
#include <algorithm>
#include <iostream>
#include <vector>
using namespace std;

// Printing Method

void print(vector<vector<int>> &v) {
  for (int i = 0; i < v.size(); ++i) {
    vector<int> &temp = v[i];
    int a = temp[0];
    int b = temp[1];
    cout << a << " " << b << endl;
  }
  cout << endl;
}

// Custom Comparator ki return value always true ya false hoti hai

bool myComparator1(vector<int> &a, vector<int> &b) {
    return a[0] > b[0]; // decreasing order sorting by 1st index
}

bool myComparator2(vector<int> &a, vector<int> &b) {
    return a[1] > b[1]; // decreasing order sorting by 2nd index
}

int main() {
    // vector of vector sorting
    vector<vector<int>> v;

    int n;
    cout << "Enter size:\n";
    cin >> n;

    // Taking input from user in 2D vector
    for (int i = 0; i < n; ++i) {

        int a, b;
        cout << "Enter a, b" << endl;
        cin >> a >> b;

        // Creation of 1D vector temp
        vector<int> temp;

        // Inserting element a at 0th and b at 1st index in 1D vector
        temp.push_back(a);
        temp.push_back(b);

        // Inserting 1D vector temp in 2D vector v
        v.push_back(temp);
    }

    cout << "Here are the Values" << endl;
    print(v);

    cout << "Sorted by 1st index" << endl;
    sort(v.begin(), v.end(), myComparator1);
    print(v);

    cout << "Sorted by 2st index" << endl;
    sort(v.begin(), v.end(), myComparator2);
    print(v);

    return 0;
}
```

```cpp
bool myComparator1(vector<int> &a, vector<int> &b) {
    return a[0] < b[0]; // increasing order sorting by 1st index
}

bool myComparator2(vector<int> &a, vector<int> &b) {
    return a[1] < b[1]; // increasing order sorting by 2nd index
}
```