Fenwick in C++

```
#include <iostream>
#include <vector>
using namespace std;

class FenwickTree {
private:
    vector<int> fenwick;
    int n;

public:
    FenwickTree(int size) {
        n = size + 1;
        fenwick.assign(n, 0);
    }

    void add(int idx, int val) {
        idx++; // 1 based index
        while (idx < n) {
            fenwick[idx] += val;
            idx += idx & (-idx); // add last set bit
        }
    }

    int sum(int idx) {
        idx++; // 1 based index
        int ans = 0;
        while (idx > 0) {
            ans += fenwick[idx];
            idx -= idx & (-idx); // remove last set bit
        }
        return ans;
    }

    int rangeSum(int l, int r) {
        return sum(r) - sum(l - 1);
    }
};

int main() {
    vector<int> v = {1, 2, 3, 4, 5, 6, 7};
    FenwickTree tree(v.size());

    // Initialize Fenwick Tree
    for (int i = 0; i < v.size(); i++) {
        tree.add(i, v[i]);
    }

    // Query range sum [3, 5]
    cout << tree.rangeSum(3, 5) << endl; // Output: 15

    // Update index 4 with new value -3
    tree.add(4, -3);

    // Query range sum [3, 5] after update
    cout << tree.rangeSum(3, 5) << endl; // Output: 12

    return 0;
}
```

Initial Array v = {1, 2, 3, 4, 5, 6, 7}

Step 1: Build Fenwick Tree

i (0-based)	v[i]	Operation	Fenwick Tree (1-based index after update)
0	1	add(0, 1)	fenwick[1] += 1 → [0, **1**, 1, 0, 1, 0, 0, 1]
1	2	add(1, 2)	fenwick[2] += 2 → [0, 1, **3**, 0, 3, 0, 0, 1]
2	3	add(2, 3)	fenwick[3] += 3 → [0, 1, 3, **3**, 6, 0, 0, 1]
3	4	add(3, 4)	fenwick[4] += 4 → [0, 1, 3, 3, **10**, 0, 0, 1]
4	5	add(4, 5)	fenwick[5] += 5 → [0, 1, 3, 3, 10, **5**, 5, 1]
5	6	add(5, 6)	fenwick[6] += 6 → [0, 1, 3, 3, 10, 5, **11**, 1]
6	7	add(6, 7)	fenwick[7] += 7 → [0, 1, 3, 3, 10, 5, 11, **8**]

⚠ Note: This is the internal fenwick[] array. Index 0 is unused.

Step 2: Query rangeSum(3, 5)

That means: sum(5) - sum(2)

➤ **sum(5):**

idx	fenwick[idx]	sum
6	11	11
4	10	21
0	—	21

➞ sum(5) = 21

➤ **sum(2):**

idx	fenwick[idx]	sum
3	3	3
2	3	6
0	—	6

idx	fenwick[idx]	sum

➞ sum(2) = 6

➞ rangeSum(3, 5) = 21 - 6 = 15

Step 3: add(4, -3) (v[4] becomes 2)

This updates the tree:

idx	fenwick[idx] before	Change	New fenwick[idx]
5	5	-3	2
6	11	-3	8
8	— (out of range)		—

Updated fenwick[] = [0, 1, 3, 3, 10, 2, 8, 8]

Step 4: rangeSum(3, 5) again

Again: sum(5) - sum(2)

➤ sum(5):

idx	fenwick[idx]	sum
6	8	8
4	10	18
0	—	18

➞ sum(5) = 18

➤ sum(2):

idx	fenwick[idx]	sum
3	3	3
2	3	6
0	—	6

➞ sum(2) = 6

➞ rangeSum(3, 5) = 18 - 6 = 12

	✅ **Final Output:** 15 12
15 12	