segment tree

```cpp
/*
 * Created by Dipta Das on 14-11-2018
 * Title: Segment Tree
 * Problem Link:
 * Editorial: http://www.shafaetsplanet.com/?p=1557
 * Solution:
 * Comments:
*/
int arr[mx];
int tree[mx * 3];
void init(int node, int b, int e)
{
    if (b == e) {
        tree[node] = arr[b];
        return;
    }
    int Left = node * 2;
    int Right = node * 2 + 1;
    int mid = (b + e) / 2;
    init(Left, b, mid);
    init(Right, mid + 1, e);
    tree[node] = tree[Left] + tree[Right];
}
int query(int node, int b, int e, int i, int j)
{
    if (i > e || j < b)
        return 0;
    if (b >= i && e <= j)
        return tree[node];
    int Left = node * 2;
    int Right = node * 2 + 1;
    int mid = (b + e) / 2;
    int p1 = query(Left, b, mid, i, j);
    int p2 = query(Right, mid + 1, e, i, j);
    return p1 + p2;
}
void update(int node, int b, int e, int i, int newvalue)
{
    if (i > e || i < b)
        return;
    if (b >= i && e <= i) {
        tree[node] = newvalue;
        return;
    }
    int Left = node * 2;
    int Right = node * 2 + 1;
    int mid = (b + e) / 2;
    update(Left, b, mid, i, newvalue);
    update(Right, mid + 1, e, i, newvalue);
    tree[node] = tree[Left] + tree[Right];
}
int main()
{
    int n;
    cin >> n;
    for (int i = 1; i <= n; ++i) cin >> arr[i];
    init(1, 1, n);
    update(1, 1, n, 2, 0);
    cout << query(1, 1, n, 1, 3) << endl;
    update(1, 1, n, 2, 2);
```

```
    cout << query(1, 1, n, 2, 2) << endl;

    return 0;
}
```