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

typedef vector <int> vi;
typedef pair <int, int> ii;
typedef vector <string> vs;
typedef vector <ii> vii;

const int MAX = 1e5;

struct Node {
        int v;
        Node() {}
        Node(int v) : v(v) {}

};

Node ope(Node A, Node B) {
        if (A.v < B.v) return A;
        return B;
}

struct SegmentTree {
        Node t[2 * MAX]; int n;
        //El arreglo incial es t[i+n]

        void build() {
                for (int i = n - 1; i > 0; i--)
                        t[i] = ope(t[i << 1], t[i << 1 | 1]);
                }

        void modify(int p, Node val) {
                for (t[p += n] = val; p >>= 1; )
                        t[p] = ope(t[p << 1], t[p << 1 | 1]);
        }

        Node get(int l, int r) { //[l,r)
                Node ansl, ansr;
                ansl = ansr = Node(1 << 30); //Inicializar con valor nulo
                for (l += n, r += n; l < r; l >>= 1, r >>= 1) {
                        if (l & 1) ansl = ope(ansl, t[l++]);
                        if (r & 1) ansr = ope(t[--r], ansr);
                }
                return ope(ansl, ansr);
```

```
        }
};

SegmentTree ST;

int main() {
        int n;
        cout << "\nIngresar n elementos: ";
        cin >> n;
        ST.n = n;

        for (int i = 0; i < n; i++) {
                cout << "\nElemento " << i<<": ";
                cin >> ST.t[n + i].v;
        }

        for (int i = 0; i < n; i++) {
                cout << ST.t[n + i].v << " ";

        }
        ST.build();
        cout << "\n";
        int q, l, r;
        cout << "\nIngresar numero de consultas: ";
        cin >> q;


        for (int i = 0; i < q; i++)
        {
                cout << "\nIngresar indice l: ";
                cin >> l;
                cout << "\nIngresar índice r: ";
                cin>>r;
                cout << "\nResultado: "<<ST.get(l, r + 1).v;
                cout << "\n";
        }
        return 0;
}
```