

```

1 #include ".././bits/stdc++.h"
2 #include "monoid.hpp"
3 // http://tsutaj.hatenablog.com/entry/2017/03/29/204841
4 // 点更新区間min 再帰セグ木 0-indexed
5 // TODO: 非再帰, template
6 // verified: http://judge.u-aizu.ac.jp/onlinejudge/review.jsp?rid=3380815
7 template <typename M>
8 class SegmentTree
9 {
10     using T = typename M::type;
11     int n;
12     std::vector<T> node;
13
14 public:
15     // v を基に初期化
16     SegmentTree(const std::vector<T> &v)
17     {
18         // n は v.size() 以上の最小の2冪
19         n = 1;
20         while (n < int(v.size()))
21             n *= 2;
22         node.resize(2 * n - 1, M::id());
23
24         // i の子 -> 2*i+1, 2*i+2 , 親 -> (i-1)/2
25         for (int i = 0; i < int(v.size()); i++)
26             node[i + n - 1] = v[i];
27         for (int i = n - 2; i >= 0; i--)
28             node[i] = M::op(node[2 * i + 1], node[2 * i + 2]);
29     }
30     // Monoid::id 初期化
31     SegmentTree(int _n)
32     {
33         // n は v.size() 以上の最小の2冪
34         n = 1;
35         while (n < _n)
36             n *= 2;
37         node.resize(2 * n - 1, M::id());
38     }
39     // x 番目を val に更新
40     void update(int x, T val)
41     {
42         x += n - 1;
43         node[x] = val;
44         while (x > 0)
45         {
46             x = (x - 1) / 2;
47             node[x] = M::op(node[2 * x + 1], node[2 * x + 2]);
48         }
49     }
50     // v[x] を M::op(v[x], val) に更新
51     void add(int x, T val)
52     {
53         x += n - 1;
54         node[x] = M::op(node[x], val);
55         while (x > 0)
56         {
57             x = (x - 1) / 2;
58             node[x] = M::op(node[2 * x + 1], node[2 * x + 2]);
59         }
60     }
61     // [a, b) の op
62     // k := 今居るノード
63     T query(int a, int b, int k = 0, int l = 0, int r = -1)
64     {
65         if (r < 0)
66             r = n;
67         if (r <= a || b <= l)
68             return M::id();
69         if (a <= l && r <= b)
70             return node[k];
71
72         T vl = query(a, b, 2 * k + 1, l, (l + r) / 2);
73         T vr = query(a, b, 2 * k + 2, (l + r) / 2, r);
74         return M::op(vl, vr);
75     }
76 };
77

```