

```

1 #include "../bits/stdc++.h"
2 // binary trie に対し、全要素にある値を xor する操作を追加したもの
3 // verified: https://atcoder.jp/contests/dwacon5th-final-open/submissions/4650380
4 class BinaryTrie
5 {
6     using T = unsigned;
7     // bit 数
8     static const int B = 32;
9     struct Node
10    {
11        // 要素数
12        int cnt;
13        // xor する値
14        T lazy;
15        Node *ch[2];
16        Node() : cnt(0), lazy(0)
17        {
18            ch[0] = nullptr, ch[1] = nullptr;
19        }
20    };
21
22    Node *root;
23
24    void eval(Node *t, int b)
25    {
26        if ((t->lazy >> (T)b) & (T)1)
27            std::swap(t->ch[0], t->ch[1]);
28        if (t->ch[0])
29            t->ch[0]->lazy ^= t->lazy;
30        if (t->ch[1])
31            t->ch[1]->lazy ^= t->lazy;
32        t->lazy = 0;
33    }
34
35    // Node t 以下に 値 val を追加
36    Node *add(Node *t, T val, int b = B - 1)
37    {
38        if (!t)
39            t = new Node;
40        t->cnt += 1;
41        if (b < 0)
42            return t;
43        eval(t, b);
44        bool f = (val >> (T)b) & (T)1;
45        t->ch[f] = add(t->ch[f], val, b - 1);
46        return t;
47    }
48    // Node t 以下から値 val を削除
49    Node *sub(Node *t, T val, int b = B - 1)
50    {
51        // 要素が無ければエラー
52        assert(t);
53        t->cnt -= 1;
54        if (t->cnt == 0)
55            return nullptr;
56        if (b < 0)
57            return t;
58        eval(t, b);
59        bool f = (val >> (T)b) & (T)1;
60        t->ch[f] = sub(t->ch[f], val, b - 1);
61        return t;
62    }
63    // Node t 以下の k-th element を取得
64    T get(Node *t, int k, int b = B - 1)
65    {
66        assert(t);
67        if (b < 0)
68            return 0;
69        eval(t, b);
70        int m = t->ch[0] ? t->ch[0]->cnt : 0;
71        return k < m ? get(t->ch[0], k, b - 1) : get(t->ch[1], k - m, b - 1) | ((T)1 << (T)b);
72    }
73    // Node t 以下で val と xor するときの最小値取得
74    T getMinimum(Node *t, T val, int b = B - 1)
75    {
76        assert(t);
77        if (b < 0)
78            return 0;
79        eval(t, b);
80        bool f = (val >> (T)b) & (T)1;
81        f ^= !t->ch[f];
82        return getMinimum(t->ch[f], val, b - 1) | ((T)f << (T)b);
83    }
84    // Node t 以下の val 未満の要素数
85    int countLessThan(Node *t, T val, int b = B - 1)
86    {
87        if (!t || b < 0)
88            return 0;
89        eval(t, b);
90        bool f = (val >> (T)b) & (T)1;
91        return (f && t->ch[0] ? t->ch[0]->cnt : 0) + countLessThan(t->ch[f], val, b - 1);
92    }
93
94 public:
95     BinaryTrie() : root(nullptr) {}

```

```

96     int size() const
97     {
98         return root ? root->cnt : 0;
99     }
100    bool empty() const
101    {
102        return !root;
103    }
104    // val を集合に1つ追加
105    void insert(T val)
106    {
107        root = add(root, val);
108    }
109    // val を集合から1つ削除(なければRE)
110    void erase(T val)
111    {
112        root = sub(root, val);
113    }
114    // 集合内の最大値
115    T maximumElement(T mask = 0)
116    {
117        return getMinimum(root, ~mask);
118    }
119    // 集合内の最小値
120    T minimumElement(T mask = 0)
121    {
122        return getMinimum(root, mask);
123    }
124    // 集合内で val 以上の要素の最小index
125    int lowerBound(T val)
126    {
127        return countLessThan(root, val);
128    }
129    // 集合内で val より大きい要素の最小index
130    int upperBound(T val)
131    {
132        return countLessThan(root, val + 1);
133    }
134    // k-th element(0-indexed) を取得
135    T get(int k)
136    {
137        assert(0 <= k && k < size());
138        return get(root, k);
139    }
140    // val の個数
141    int count(T val)
142    {
143        if (!root)
144            return 0;
145        Node *t = root;
146        for (int i = B - 1; i >= 0; i--)
147        {
148            eval(t, i);
149            t = t->ch[(val >> (T)i) & (T)1];
150            if (!t)
151                return 0;
152        }
153        return t->cnt;
154    }
155    // 全ての要素を val で xor した値に変更
156    void xorAllElements(T val)
157    {
158        if (root)
159            root->lazy ^= val;
160    }
161 };
162

```