

Std Code Library(Temp version1.0)

tingyx

Nanjing University of Science and Technology

August 15, 2024

Contents

分块	2
二分图	5
Persistent Data Structure	9
Tree Dp	12
树状数组	20
数位 DP	21
math	22
图论	27
Segment Tree	29
串串	48

分块

1. 分块 I(区间修改, 区间查询)

```
1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 int n, m, sq;
13 int v[N], bl[N], tag[N], ans[N];
14
15 void update(int a, int b){
16     for (int i = a; i <= min(bl[a] * sq, b); ++i) {
17         ans[bl[a]] -= (v[i] ^ tag[bl[a]]);
18         v[i] ^= 1;
19         ans[bl[a]] += (v[i] ^ tag[bl[a]]);
20     }
21     if (bl[a] != bl[b]){
22         for (int i = (bl[b] - 1) * sq + 1; i <= b; ++i) {
23             ans[bl[b]] -= (v[i] ^ tag[bl[b]]);
24             v[i] ^= 1;
25             ans[bl[b]] += (v[i] ^ tag[bl[b]]);
26         }
27     }
28     for (int i = bl[a] + 1; i <= bl[b] - 1; ++i){
29         tag[i] ^= 1;
30         ans[i] = sq - ans[i];
31     }
32 }
33
34 ll query(int a, int b){
35     ll res = 0;
36     for (int i = a; i <= min(bl[a] * sq, b); ++i) {
37         res += (v[i] ^ tag[bl[a]]);
38     }
39     if (bl[a] != bl[b]){
40         for (int i = (bl[b] - 1) * sq + 1; i <= b; ++i) {
41             res += (v[i] ^ tag[bl[b]]);
42         }
43     }
44     for (int i = bl[a] + 1; i <= bl[b] - 1; ++i) res += ans[i];
45     return res;
46 }
47
48 void solve(){
49     cin >> n >> m;
50     sq = sqrt(n);
51     for (int i = 1; i <= n; ++i) bl[i] = (i - 1) / sq + 1;
52     for (int i = 1; i <= m; ++i){
53         int op, l, r;
54         cin >> op >> l >> r;
55         if (op == 0) update(l, r);
56         else cout << query(l, r) << endl;
57     }
58 }
59
60 signed main(){
61     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
62     int t = 1;
63     //cin >> t;
64     while(t--) solve();
65     return 0;
66 }
```

2. 分块 II

给出一个长为 n 的数列，以及 n 个操作，操作涉及区间加法，询问区间内小于某个值 x 的元素个数。

```
1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 int n, blo;
13 int v[N], bl[N], atag[N];
14 vector<int> ve[2005];
15
16 void reset(int x)
17 {
18     ve[x].clear();
19     for (int i = (x - 1) * blo + 1; i <= min(x * blo, n); i++)
20         ve[x].push_back(v[i]);
21     sort(ve[x].begin(), ve[x].end());
22 }
23
24 void add(int a, int b, int c)
25 {
26     for (int i = a; i <= min(bl[a] * blo, b); i++)
27         v[i] += c;
28     reset(bl[a]);
29     if (bl[a] != bl[b])
30     {
31         for (int i = (bl[b] - 1) * blo + 1; i <= b; i++)
32             v[i] += c;
33         reset(bl[b]);
34     }
35     for (int i = bl[a] + 1; i <= bl[b] - 1; i++)
36         atag[i] += c;
37 }
38
39 int query(int a, int b, int c)
40 {
41     int ans = 0;
42     for (int i = a; i <= min(bl[a] * blo, b); i++)
43         if (v[i] + atag[bl[a]] < c)
44             ans++;
45     if (bl[a] != bl[b])
46         for (int i = (bl[b] - 1) * blo + 1; i <= b; i++)
47             if (v[i] + atag[bl[b]] < c)
48                 ans++;
49     for (int i = bl[a] + 1; i <= bl[b] - 1; i++)
50     {
51         int x = c - atag[i];
52         ans += lower_bound(ve[i].begin(), ve[i].end(), x) - ve[i].begin();
53     }
54     return ans;
55 }
56
57 int main()
58 {
59     cin >> n;
60     blo = sqrt(n);
61     for (int i = 1; i <= n; i++)
62         cin >> v[i];
63     for (int i = 1; i <= n; i++)
64     {
65         bl[i] = (i - 1) / blo + 1;
66         ve[bl[i]].push_back(v[i]);
67     }
68     for (int i = 1; i <= bl[n]; i++)
69         sort(ve[i].begin(), ve[i].end());
70     for (int i = 1; i <= n; i++)
```

```

71     {
72         int f, a, b, c;
73         cin >> f >> a >> b >> c;
74         if (f == 0)
75             add(a, b, c);
76         if (f == 1)
77             printf("%d\n", query(a, b, c * c));
78     }
79     return 0;
80 }

```

3. 线性 RMQ

```

1  /**
2   * 线性 RMQ
3   */
4
5  #include <bits/stdc++.h>
6  #include <limits>
7  #define endl '\n'
8  using namespace std;
9  typedef long long ll;
10
11 const int N = 5e4 + 10;
12 const int M = 2e4 + 10;
13 const int L = 80;
14 const int mod = 998244353;
15 const int inf32 = 0x3f3f3f3f;
16 const ll inf64 = 4e18;
17
18 int a[N + M];
19 int highbit[M];
20 int stmax[M][L], stmin[M][L];
21 int premax[M][L], premin[M][L];
22 int sufmax[M][L], sufmin[M][L];
23 int quemax[M][L], quemin[M][L];
24 int stackmax[L], stackmin[L];
25
26 void solve(){
27     int n, q;
28     cin >> n >> q;
29     int B = int(log2(n)); // 块的大小
30     int S = (n - 1) / B + 1; // 块的个数
31     for (int b = 0; b < S; ++b) stmin[b][0] = inf32;
32     for (int i = 0; i < n; ++i){
33         cin >> a[i];
34         stmin[i / B][0] = min(stmin[i / B][0], a[i]);
35         stmax[i / B][0] = max(stmax[i / B][0], a[i]);
36     }
37     for (int b = S - 1; b >= 0; b--){
38         for (int k = 1; b + (1 << k) - 1 < S; ++k){
39             stmin[b][k] = min(stmin[b][k - 1], stmin[b + (1 << (k - 1))][k - 1]);
40             stmax[b][k] = max(stmax[b][k - 1], stmax[b + (1 << (k - 1))][k - 1]);
41         }
42     }
43     for (int b = 0; b < S; ++b){
44         int be = b * B;
45         premin[b][0] = premax[b][0] = a[be];
46         for (int k = 1; k < B; ++k){
47             premin[b][k] = min(premin[b][k - 1], a[be + k]);
48             premax[b][k] = max(premax[b][k - 1], a[be + k]);
49         }
50         sufmin[b][B - 1] = sufmax[b][B - 1] = a[be + B - 1];
51         for (int k = B - 2; k >= 0; --k){
52             sufmin[b][k] = min(sufmin[b][k + 1], a[be + k]);
53             sufmax[b][k] = max(sufmax[b][k + 1], a[be + k]);
54         }
55     }
56     for (int b = 0; b < S; ++b){
57         int be = b * B;
58         int spmin = 0, nowmin = 0;
59         int spmax = 0, nowmax = 0;

```

```

60     for (int i = 0; i < B; ++i){
61         while (spmin && a[be + stackmin[spmin]] > a[be + i]) nowmin ^= 1 << stackmin[spmin--];
62         while (spmax && a[be + stackmax[spmax]] < a[be + i]) nowmax ^= 1 << stackmax[spmax--];
63         quemin[b][i] = (nowmin ^= 1 << (stackmin[++spmin] = i));
64         quemax[b][i] = (nowmax ^= 1 << (stackmax[++spmax] = i));
65     }
66 }
67 for (int i = 2; i <= S; ++i) highbit[i] = highbit[i >> 1] + 1;
68
69 while(q --) {
70     int l, r;
71     cin >> l >> r;
72     l--, r--;
73     int L = l / B, R = r / B;
74     int li = l % B, ri = r % B;
75
76     int mn = inf32, mx = 0;
77     if(L == R) {
78         mn = min(mn, a[l + __builtin_ctz(quemin[R][ri] >> li)]);
79         mx = max(mx, a[l + __builtin_ctz(quemax[R][ri] >> li)]);
80     }
81     else {
82         mn = min(mn, sufmin[L][li]);
83         mn = min(mn, premin[R][ri]);
84         mx = max(mx, sufmax[L][li]);
85         mx = max(mx, premax[R][ri]);
86         int len = R - L - 1;
87         int k = highbit[len];
88         if(len) {
89             mn = min(mn, stmin[L + 1][k]);
90             mn = min(mn, stmin[R - (1 << k)][k]);
91             mx = max(mx, stmax[L + 1][k]);
92             mx = max(mx, stmax[R - (1 << k)][k]);
93         }
94     }
95     cout << mx - mn << endl;
96 }
97 }
98
99 signed main(){
100     ios::sync_with_stdio(false);
101     cin.tie(nullptr);
102     int t = 1;
103     //cin >> t;
104     while(t--) solve();
105     return 0;
106 }

```

二分图

1.Graph Coloring I(奇环判断染色)

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 3e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 vector<int> G[N];
13
14 bool col[N], err;
15 int st[N], pos[N], top;
16
17 void dfs(int x, int fa, int c){
18     if (!err && pos[x] && pos[x] < top && col[x] != c){
19         err = 1;

```

```

20     cout << top - pos[x] + 1 << endl;
21     for (int i = pos[x]; i <= top; ++i)
22         cout << st[i] << " \n"[i == top];
23 }
24 if (err || pos[x]) return;
25 st[++top] = x;
26 pos[x] = top;
27 col[x] = c;
28 for (auto y : G[x]){
29     if (y == fa) continue;
30     dfs(y, x, c ^ 1);
31 }
32 --top;
33 }
34
35 void solve(){
36     int n, m;
37     cin >> n >> m;
38     for (int i = 1; i <= m; ++i){
39         int u, v;
40         cin >> u >> v;
41         G[u].push_back(v);
42         G[v].push_back(u);
43     }
44     dfs(1, 0, 0);
45     if (!err){
46         cout << 0 << endl;
47         for (int i = 1; i <= n; ++i) cout << col[i] << " \n"[i == n];
48     }
49 }
50
51 signed main(){
52     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
53     int t = 1;
54     //cin >> t;
55     while(t--) solve();
56     return 0;
57 }

```

2. 增广路算法 (矩阵游戏)

矩阵进行行列交换，能否把对角线都变成 1

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 struct augment_path
13 {
14     vector<vector<int>> g;
15     vector<int> pa; // 匹配
16     vector<int> pb;
17     vector<int> vis; // 访问
18     int n, m;      // 两个点集中的顶点数量
19     int dfn;        // 时间戳记
20     int res;        // 匹配数
21
22     augment_path(int _n, int _m) : n(_n), m(_m)
23     {
24         assert(0 <= n && 0 <= m);
25         pa = vector<int>(n, -1);
26         pb = vector<int>(m, -1);
27         vis = vector<int>(n);
28         g.resize(n);
29         res = 0;

```

```

30     dfn = 0;
31 }
32
33 void add(int from, int to)
34 {
35     assert(0 <= from && from < n && 0 <= to && to < m);
36     g[from].push_back(to);
37 }
38
39 bool dfs(int v)
40 {
41     vis[v] = dfn;
42     for (int u : g[v])
43     {
44         if (pb[u] == -1)
45         {
46             pb[u] = v;
47             pa[v] = u;
48             return true;
49         }
50     }
51     for (int u : g[v])
52     {
53         if (vis[pb[u]] != dfn && dfs(pb[u]))
54         {
55             pa[v] = u;
56             pb[u] = v;
57             return true;
58         }
59     }
60     return false;
61 }
62
63 int solve()
64 {
65     while (true)
66     {
67         dfn++;
68         int cnt = 0;
69         for (int i = 0; i < n; i++)
70         {
71             if (pa[i] == -1 && dfs(i))
72             {
73                 cnt++;
74             }
75         }
76         if (cnt == 0)
77         {
78             break;
79         }
80         res += cnt;
81     }
82     return res;
83 }
84 };
85
86 void solve()
87 {
88     int n;
89     cin >> n;
90     augment_path ap(n, n);
91     for (int i = 0; i < n; ++i)
92         for (int j = 0; j < n; ++j){
93             int x;
94             cin >> x;
95             if (x) ap.g[i].push_back(j);
96         }
97     int res = ap.solve();
98     if (res == n){
99         cout << "Yes" << endl;
100        return;

```



```

101     }
102     cout << "No" << endl;
103 }
104
105 signed main()
106 {
107     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
108     int t = 1;
109     cin >> t;
110     while (t--)
111         solve();
112     return 0;
113 }

```

3. 宿舍的假期 (简略版 AG)

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 struct augment_path
13 {
14     vector<vector<int>> g;
15     vector<int> pa; // 匹配
16     vector<int> pb;
17     vector<int> vis; // 访问
18     int n, m; // 两个点集中的顶点数量
19     int dfn; // 时间戳记
20     int res; // 匹配数
21
22     augment_path(int _n, int _m) : n(_n), m(_m)
23     {
24         assert(0 <= n && 0 <= m);
25         pa = vector<int>(n, -1);
26         pb = vector<int>(m, -1);
27         vis = vector<int>(n);
28         g.resize(n);
29         res = 0;
30         dfn = 0;
31     }
32
33     void add(int from, int to)
34     {
35         assert(0 <= from && from < n && 0 <= to && to < m);
36         g[from].push_back(to);
37     }
38
39     bool dfs(int v)
40     {
41         vis[v] = dfn;
42         for (int u : g[v])
43         {
44             if (pb[u] == -1)
45             {
46                 pb[u] = v;
47                 pa[v] = u;
48                 return true;
49             }
50         }
51         for (int u : g[v])
52         {
53             if (vis[pb[u]] != dfn && dfs(pb[u]))
54             {
55                 pa[v] = u;
56                 pb[u] = v;

```

```

57         return true;
58     }
59 }
60 return false;
61 }
62
63 int solve()
64 {
65     while (true)
66     {
67         dfn++;
68         int cnt = 0;
69         for (int i = 0; i < n; i++)
70         {
71             if (pa[i] == -1 && dfs(i))
72             {
73                 cnt++;
74             }
75         }
76         if (cnt == 0)
77         {
78             break;
79         }
80         res += cnt;
81     }
82     return res;
83 }
84 };
85
86 void solve()
87 {
88     int n;
89     cin >> n;
90     augment_path ap(n, n);
91     for (int i = 0; i < n; ++i)
92         for (int j = 0; j < n; ++j){
93             int x;
94             cin >> x;
95             if (x) ap.g[i].push_back(j);
96         }
97     int res = ap.solve();
98     if (res == n){
99         cout << "Yes" << endl;
100        return;
101    }
102    cout << "No" << endl;
103 }
104
105 signed main()
106 {
107     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
108     int t = 1;
109     cin >> t;
110     while (t--)
111         solve();
112     return 0;
113 }

```

Persistent Data Structure

1. 区间第 K 小

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;

```

```

10  const ll inf64 = 5e18;
11
12  int root[N];
13  int a[N], b[N];
14  struct PersistentTree{
15      int ls[N << 5], rs[N << 5], sum[N << 5], tot = 0;
16      //建树
17      inline void build(int & rt, int l, int r){
18          rt = ++tot;
19          if (l == r) {
20              sum[rt] = 0;
21              return;
22          }
23          int mid = l + r >> 1;
24          build(ls[rt], l, mid);
25          build(rs[rt], mid + 1, r);
26          sum[rt] = sum[ls[rt]] + sum[rs[rt]];
27      }
28      //单点修改
29      inline void update(int & rt, int old, int l, int r, int p, int k){
30          rt = ++tot;
31          ls[rt] = ls[old], rs[rt] = rs[old], sum[rt] = sum[old] + k;
32          if (l == r) return;
33          int mid = l + r >> 1;
34          if (p <= mid) update(ls[rt], ls[old], l, mid, p, k);
35          else update(rs[rt], rs[old], mid + 1, r, p, k);
36      }
37      //区间查询
38      inline int query(int old, int rt, int l, int r, int kth){
39          if (l == r) return ls[rt];
40          int mid = (l + r) >> 1;
41          int res = sum[ls[rt]] - sum[ls[old]];
42          if (kth <= res) return query(ls[old], ls[rt], l, mid, kth);
43          else return query(rs[old], rs[rt], mid + 1, r, kth - res);
44      }
45  }T;
46
47  void solve(){
48      int n, m;
49      cin >> n >> m;
50      for (int i = 1; i <= n; ++i) {
51          cin >> a[i];
52          b[i] = a[i];
53      }
54      sort(b + 1, b + 1 + n);
55      int cnt = 1;
56      for (int i = 2; i <= n; ++i){
57          if (b[i] != b[cnt])
58              b[++cnt] = b[i];
59      }
60      T.build(root[0], 1, cnt);
61      for (int i = 1; i <= n; ++i){
62          int p = lower_bound(b + 1, b + cnt + 1, a[i]) - b;
63          T.update(root[i], root[i - 1], 1, cnt, p, 1);
64      }
65      for (int i = 1; i <= m; ++i){
66          int l, r, k;
67          cin >> l >> r >> k;
68          int idx = T.query(root[l - 1], root[r], 1, cnt, k);
69          cout << b[idx] << endl;
70      }
71  }
72
73  signed main(){
74      ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
75      int t = 1;
76      //cin >> t;
77      while(t--) solve();
78      return 0;
79  }

```

2. 最大异或和

```
1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 6e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 int a[N], s[N];
13 struct Trie
14 {
15     int cnt, rt[N], ch[N * 33][2], val[N * 33];
16     void insert(int o, int lst, int v)
17     {
18         for (int i = 28; i >= 0; i--)
19         {
20             val[o] = val[lst] + 1; // 在原版本的基础上更新
21             if ((v & (1 << i)) == 0)
22             {
23                 if (!ch[o][0])
24                     ch[o][0] = ++cnt;
25                 ch[o][1] = ch[lst][1];
26                 o = ch[o][0];
27                 lst = ch[lst][0];
28             }
29             else
30             {
31                 if (!ch[o][1])
32                     ch[o][1] = ++cnt;
33                 ch[o][0] = ch[lst][0];
34                 o = ch[o][1];
35                 lst = ch[lst][1];
36             }
37         }
38         val[o] = val[lst] + 1;
39     }
40
41     int query(int o1, int o2, int v)
42     {
43         int ret = 0;
44         for (int i = 28; i >= 0; i--)
45         {
46             int t = ((v & (1 << i)) ? 1 : 0);
47             if (val[ch[o1][!t]] - val[ch[o2][!t]])
48                 ret += (1 << i), o1 = ch[o1][!t],
49                 o2 = ch[o2][!t]; // 尽量向不同的地方跳
50             else
51                 o1 = ch[o1][t], o2 = ch[o2][t];
52         }
53         return ret;
54     }
55 } st;
56
57 void solve()
58 {
59     int n, q;
60     cin >> n >> q;
61     for (int i = 1; i <= n; ++i) cin >> a[i], s[i] = s[i - 1] ^ a[i];
62     for (int i = 1; i <= n; ++i){
63         st.rt[i] = ++st.cnt;
64         st.insert(st.rt[i], st.rt[i - 1], s[i]);
65     }
66     while (q--){
67         char op;
68         cin >> op;
69         if (op == 'A'){
70             ++n;
```

```

71         cin >> a[n];
72         s[n] = s[n - 1] ^ a[n];
73         st.rt[n] = ++st.cnt;
74         st.insert(st.rt[n], st.rt[n - 1], s[n]);
75     }else{
76         int l, r, x;
77         cin >> l >> r >> x;
78         l--, r--;
79         if (l == 0) cout << max(s[n] ^ x, st.query(st.rt[r], st.rt[0], s[n] ^ x)) << endl;
80         else cout << st.query(st.rt[r], st.rt[l - 1], s[n] ^ x) << endl;
81     }
82 }
83 }
84
85 signed main()
86 {
87     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
88     int t = 1;
89     // cin >> t;
90     while (t--)
91         solve();
92     return 0;
93 }

```

Tree Dp

1. 树的平衡点

```

1 function<void(int, int)> dfs = [&](int u, int fa){
2     sz[u] = 1;
3     for (auto v : G[u]){
4         if (v == fa) continue;
5         dfs(v, u);
6         sz[u] += sz[v];
7         int mx = max(sz[u], n - sz[u]);
8         if (mx <= mn){
9             mn = mx;
10            id = min(u, id);
11        }
12    }
13 };

```

2. 树的最小点覆盖 (最少的点覆盖所有边)

```

1 void dp(int u) {
2     bool fg = 0;
3     for(int i = h[u]; ~i; i = nex[i]) {
4         int j = v[i];
5         fg = 1;
6         dp(j);
7         f[u][0] += f[j][1];
8         f[u][1] += min(f[j][0], f[j][1]);
9     }
10    f[u][1] += 1;
11    if(!fg) {
12        f[u][0] = 0; f[u][1] = 1;
13    }
14 }

```

3. 树的最小支配集 (最少的点覆盖所有点)

f[i][0] 选 i 且 i 及 i 的子树都被覆盖了 f[i][1] 不选 i 且 i 被其儿子覆盖 f[i][2] 不选 i 且 i 被其父亲覆盖 (儿子可选可不选)

```

1 void dfs(int u, int fa){
2     f[u][0] = 1; f[u][1] = f[u][2] = 0;
3     bool ok = false;
4     int tmp = inf32;
5     for (auto v : G[u]){
6         if (v == fa) continue;
7         dfs(v, u);
8         f[u][2] += min(f[v][1], f[v][0]);
9         f[u][0] += min({f[v][0], f[v][1], f[v][2]});

```

```

10         if (f[v][0] <= f[v][1]){
11             ok = true;
12             f[u][1] += f[v][0];
13         }else{
14             f[u][1] += f[v][1];
15             tmp = min(tmp, f[v][0] - f[v][1]);
16         }
17     }
18     if (!ok) f[u][1] += tmp;
19 }

```

4. 树的最大独立集 (选定的任意两点之间无边)

```

1 function<void(int, int)> dfs = [&](int u, int fa)
2 {
3     dp[u][1] = h[u];
4     for (auto v : G[u])
5     {
6         if (v == fa)
7             continue;
8         dfs(v, u);
9         dp[u][0] += max(dp[v][1], dp[v][0]);
10        dp[u][1] += dp[v][0];
11    }
12 };

```

5. 树上背包 (最多不超过 m 条边)

```

1 #include <bits/stdc++.h>
2 #include <cstring>
3 #include <vector>
4 #define endl '\n'
5 using namespace std;
6 typedef long double db;
7 typedef long long ll;
8
9 const ll N = 1e2 + 10;
10 const ll mod = 998244353;
11 const ll inf32 = 0x3f3f3f3f;
12 const ll inf64 = 5e18;
13
14 vector<pair<int, int>> G[N];
15 int n, m, dp[N][N], sz[N], tmp[N];
16
17 void dfs(int u, int fa){
18     sz[u] = 1;
19     for (auto [v, w] : G[u]){
20         if (v == fa) continue;
21         dfs(v, u);
22         memset(tmp, 0, sizeof tmp);
23         for (int i = sz[u] - 1; i >= 0; --i){
24             for (int j = sz[v] - 1; j >= 0; --j){
25                 int nxt = i + j + 1;
26                 if (nxt <= m){
27                     tmp[nxt] = max(tmp[nxt], dp[u][i] + dp[v][j] + w);
28                 }
29             }
30         }
31         sz[u] += sz[v];
32         for (int i = 0; i <= sz[u] - 1; ++i)
33             dp[u][i] = max(dp[u][i], tmp[i]);
34     }
35 }
36
37 void solve(){
38     cin >> n >> m;
39     for (int i = 1; i < n; ++i){
40         int u, v, w;
41         cin >> u >> v >> w;
42         G[u].emplace_back(v, w);
43         G[v].emplace_back(u, w);
44     }

```

```

45     dfs(1, 0);
46     int ans = 0;
47     for (int i = 1; i <= m; ++i){
48         ans = max(ans, dp[1][i]);
49     }
50     cout << ans << endl;
51 }
52
53 signed main(){
54     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
55     int t = 1;
56     //cin >> t;
57     while(t--) solve();
58     return 0;
59 }

```

6.2022CCPC-A(树上背包) 爱丽丝想在公园里找到她丢失的猫。

爱丽丝想在公园里找到她丢失的猫。

公园是一棵有根的树，由 n 个顶点组成。顶点的编号从 1 到 n ，根顶点为 1。

爱丽丝现在位于顶点 1。她知道猫已经从顶点 1 跑到了树的某片叶子上，而且没有顶点被访问超过一次。叶子是没有子顶点的顶点。

每个顶点上都有一个监视器。顶点 i 上的监视器可以观察到猫是否访问了顶点 i ，以及猫去了哪个顶点 (如果顶点 i 不是叶子)。爱丽丝需要花费 a_i 秒来检查 i_{th} 监视器的数据。

爱丽丝也可以自己搜索一些叶子。搜索 i 个叶子需要 t_i 秒。请注意， i 是顶点的计数，而不是顶点的标签。

帮助爱丽丝确定要检查哪些监视器和搜索哪些树叶，以便唯一确定猫的位置，并尽可能减少所需的总时间。请注意，要检查的监视器和要搜索的树叶应该在一开始就决定好，之后不得更改。

找出最短的时间。

```

1  #include <bits/stdc++.h>
2  #include <vector>
3  #define endl '\n'
4  using namespace std;
5  typedef long double db;
6  typedef long long ll;
7
8  const ll N = 2e3 + 10;
9  const ll mod = 998244353;
10 const ll inf32 = 0x3f3f3f3f;
11 const ll inf64 = 5e18;
12
13 vector<int> G[N];
14 ll sz[N], a[N], t[N];
15 ll dp[N][N][2], g[N][N];
16 ll tmp1[N][2], tmp2[N];
17
18
19 void dfs(int u, int fa){
20     if (G[u].size() == 1 && fa){
21         sz[u] = 1;
22         dp[u][0][0] = dp[u][1][1] = 0;
23         dp[u][1][0] = a[u];
24         return;
25     }
26     dp[u][0][0] = 0;
27     g[u][0] = 0;
28     for (auto v : G[u]){
29         if (v == fa) continue;
30         dfs(v, u);
31         memset(tmp1, 0x3f, sizeof tmp1);
32         memset(tmp2, 0x3f, sizeof tmp2);
33         for (int i = sz[u]; i >= 0; --i){
34             for (int j = sz[v]; j >= 0; --j){
35                 int nxt = i + j;
36                 tmp1[nxt][0] = min(tmp1[nxt][0], dp[u][i][0] + dp[v][j][0]);
37                 tmp1[nxt][1] = min({tmp1[nxt][1], dp[u][i][0] + dp[v][j][1], dp[u][i][1] + dp[v][j][0]});
38                 tmp2[nxt] = min(tmp2[nxt], g[u][i] + min(dp[v][j][0], dp[v][j][1]));

```

```

39         }
40     }
41     sz[u] += sz[v];
42     memcpy(dp[u], tmp1, sizeof tmp1);
43     memcpy(g[u], tmp2, sizeof tmp2);
44 }
45 for (int i = 1; i <= sz[u]; ++i)
46     dp[u][i][0] = min(dp[u][i][0], g[u][i] + a[u]);
47 }
48
49 void solve(){
50     int n;
51     cin >> n;
52     for (int i = 1; i <= n; ++i) cin >> a[i];
53     for (int i = 1; i <= n; ++i) cin >> t[i];
54     for (int i = 1; i <= n; ++i) sz[i] = 0;
55     for (int i = 1; i <= n; ++i) G[i].clear();
56     for (int i = 1, u, v; i < n; ++i){
57         cin >> u >> v;
58         G[u].push_back(v);
59         G[v].push_back(u);
60     }
61     memset(dp, 0x3f, sizeof dp);
62     memset(g, 0x3f, sizeof g);
63     if (n == 1) {
64         cout << 0 << endl;
65         return;
66     }
67     ll ans = inf64;
68     dfs(1, 0);
69     for (int i = 1; i <= sz[1]; ++i){
70         ans = min(ans, t[sz[1] - i] + min(dp[1][i][0], dp[1][i][1]));
71     }
72     cout << ans << endl;
73 }
74
75 signed main(){
76     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
77     int t = 1;
78     cin >> t;
79     while(t--) solve();
80     return 0;
81 }

```

```

1  #include <bits/stdc++.h>
2  #include <vector>
3  #define endl '\n'
4  using namespace std;
5  typedef long double db;
6  typedef long long ll;
7
8  const ll N = 2e3 + 10;
9  const ll mod = 998244353;
10 const ll inf32 = 0x3f3f3f3f;
11 const ll inf64 = 5e18;
12
13 vector<int> G[N];
14 ll sz[N], a[N], t[N];
15 ll dp[N][N][2], g[N][N];
16
17 void dfs(int u, int fa){
18     if (G[u].size() == 1 && fa){
19         sz[u] = 1;
20         dp[u][0][0] = dp[u][1][1] = 0;
21         dp[u][1][0] = a[u];
22         return;
23     }
24     dp[u][0][0] = 0;
25     g[u][0] = 0;
26     for (auto v : G[u]){
27         if (v == fa) continue;
28         dfs(v, u);

```



```

29     for (int i = sz[u]; i >= 0; --i){
30         for (int j = sz[v]; j >= 0; --j){
31             int nxt = i + j;
32             dp[u][nxt][0] = min(dp[u][nxt][0], dp[u][i][0] + dp[v][j][0]);
33             dp[u][nxt][1] = min({dp[u][nxt][1], dp[u][i][0] + dp[v][j][1], dp[u][i][1] + dp[v][j][0]});
34             g[u][nxt] = min(g[u][nxt], g[u][i] + min(dp[v][j][0], dp[v][j][1]));
35         }
36     }
37     sz[u] += sz[v];
38 }
39 for (int i = 1; i <= sz[u]; ++i)
40     dp[u][i][0] = min(dp[u][i][0], g[u][i] + a[u]);
41 }
42
43 void solve(){
44     int n;
45     cin >> n;
46     for (int i = 1; i <= n; ++i) cin >> a[i];
47     for (int i = 1; i <= n; ++i) cin >> t[i];
48     for (int i = 1; i <= n; ++i) sz[i] = 0;
49     for (int i = 1; i <= n; ++i) G[i].clear();
50     for (int i = 1, u, v; i < n; ++i){
51         cin >> u >> v;
52         G[u].push_back(v);
53         G[v].push_back(u);
54     }
55     memset(dp, 0x3f, sizeof dp);
56     memset(g, 0x3f, sizeof g);
57     if (n == 1) {
58         cout << 0 << endl;
59         return;
60     }
61     ll ans = inf64;
62     dfs(1, 0);
63     for (int i = 1; i <= sz[1]; ++i){
64         ans = min(ans, t[sz[1] - i] + min(dp[1][i][0], dp[1][i][1]));
65     }
66     cout << ans << endl;
67 }
68
69 signed main(){
70     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
71     int t = 1;
72     cin >> t;
73     while(t--) solve();
74     return 0;
75 }

```

7. 树联通点集 (换根)

```

1  #include <bits/stdc++.h>
2  #include <vector>
3  #define endl '\n'
4  using namespace std;
5  typedef long double db;
6  typedef long long ll;
7
8  const ll N = 1e6 + 10;
9  const ll mod = 1e9 + 7;
10 const ll inf32 = 0x3f3f3f3f;
11 const ll inf64 = 5e18;
12
13 int n;
14 ll f[N], ans[N];
15 vector<int> G[N];
16
17 ll qmi(ll a, ll b = mod - 2){
18     ll res = 1;
19     while (b){
20         if (b & 1) res = res * a % mod;
21         a = a * a % mod;
22         b >>= 1;

```

```

23     }
24     return res;
25 }
26
27 void dfs1(int u, int fa){
28     f[u] = 1;
29     for (auto v : G[u]){
30         if (v == fa) continue;
31         dfs1(v, u);
32         f[u] = f[u] * (f[v] + 1) % mod;
33     }
34 }
35
36 void dfs2(int u, int fa){
37     if (!fa) ans[u] = f[u];
38     else if ((f[u] + 1) % mod == 0){
39         dfs1(u, u);
40         ans[u] = f[u];
41     } else ans[u] = (ans[fa] % mod * qmi(f[u] + 1) + 1) % mod * f[u] % mod;
42     for (auto v : G[u]){
43         if (v == fa) continue;
44         dfs2(v, u);
45     }
46 }
47
48 void solve(){
49     cin >> n;
50     for (int i = 1; i < n; ++i){
51         int u, v;
52         cin >> u >> v;
53         G[u].push_back(v);
54         G[v].push_back(u);
55     }
56     dfs1(1, 1);
57     dfs2(1, 0);
58     for (int i = 1; i <= n; ++i) cout << ans[i] << endl;
59 }
60
61 signed main(){
62     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
63     int t = 1;
64     //cin >> t;
65     while(t--) solve();
66     return 0;
67 }

```

8. 树划分联通块大小小于等于 k(树上背包)

另一种背包写法

```

1  #include <bits/stdc++.h>
2  #include <cstring>
3  #include <vector>
4  #define endl '\n'
5  using namespace std;
6  typedef long double db;
7  typedef long long ll;
8
9  const ll N = 2e3 + 10;
10 const ll mod = 998244353;
11 const ll inf32 = 0x3f3f3f3f;
12 const ll inf64 = 5e18;
13
14 int n, k, sz[N];
15 vector<int> G[N];
16 ll dp[N][N], ans = 0; //dp[i][j] i 所在的联通块大小为 j
17
18 void dfs(int u, int fa){
19     sz[u] = dp[u][1] = 1;
20     for (auto v : G[u]){
21         if (v == fa) continue;
22         dfs(v, u);

```

```

23     ll sum = 0;
24     for (int i = 1; i <= sz[v] && i <= k; ++i) sum = (sum + dp[v][i]) % mod;
25     for (int i = min(sz[u], k); i >= 1; --i){
26         for (int j = min(sz[v], k); j >= 1; --j){
27             int nxt = i + j;
28             if (nxt <= k){
29                 dp[u][nxt] = (dp[u][nxt] + dp[u][i] * dp[v][j]) % mod;
30             }
31         }
32         dp[u][i] = (dp[u][i] * sum) % mod;
33     }
34     sz[u] += sz[v];
35 }
36 }
37
38 void solve(){
39     cin >> n >> k;
40     for (int i = 1; i < n; ++i){
41         int u, v;
42         cin >> u >> v;
43         G[u].push_back(v);
44         G[v].push_back(u);
45     }
46     dfs(1, 0);
47     for (int i = 1; i <= k && i <= sz[1]; ++i) ans = (ans + dp[1][i]) % mod;
48     cout << ans << endl;
49 }
50
51 signed main(){
52     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
53     int t = 1;
54     //cin >> t;
55     while(t--) solve();
56     return 0;
57 }

```

8. 树上子链 (点权和最大)

给定一棵树 T ，树 T 上每个点都有一个权值。定义一颗树的子链的大小为：这个子链上所有结点的权值和。请在树 T 中找出一条最大的子链并输出。

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
6  using namespace std;
7  typedef long long ll;
8  const ll maxn = 2e5 + 10;
9  const ll mod = 998244353;
10 vector<ll> G[maxn];
11 ll w[maxn], dis[maxn], ans = -1e18;
12
13 void solve(){
14     int n;
15     cin >> n;
16     for (int i = 1; i <= n; ++i){
17         cin >> w[i];
18     }
19     for (int i = 1; i <= n - 1; ++i){
20         int u, v;
21         cin >> u >> v;
22         G[u].push_back(v);
23         G[v].push_back(u);
24     }
25     function<void(int, int)> dfs = [&](int u, int fa){
26         ll tmp = 0, mx1 = 0, mx2 = 0;
27         for (auto v: G[u]){
28             if (v == fa) continue;
29             dfs(v, u);
30             tmp = dis[v];

```

```

31         if (tmp >= mx1){
32             mx2 = mx1;
33             mx1 = tmp;
34         }else if (tmp >= mx2){
35             mx2 = tmp;
36         }
37     }
38     ans = max(ans, mx1 + mx2 + w[u]);
39     dis[u] = mx1 + w[u];
40 };
41 dfs(1, 0);
42 cout << ans << endl;
43 }
44
45 int main(){
46     ios;
47     int t = 1;
48     //cin >> t;
49     while(t--){
50         solve();
51     }
52     return 0;
53 }

```

9.Nim Cheater(树剖优化 dp)

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3
4  using ll = long long;
5
6  constexpr int N = 4e4 + 10;
7  constexpr int mod = 998244353;
8
9  using namespace std;
10
11 int dp[16390];
12 int f[32][16390];
13
14 void solve(){
15     int n;
16     cin >> n;
17     vector<int> a(n + 1), b(n + 1), fa(n + 1), ask(n + 1), ans(n + 1);
18     vector<vector<int>> G(n + 1);
19     int lst = 0, rt = 0;
20     for (int i = 1; i <= n; ++i) {
21         string s;
22         cin >> s;
23         if (s[0] == 'A'){
24             cin >> a[i] >> b[i];
25             if (rt == 0) rt = i, lst = i;
26             else{
27                 G[lst].push_back(i);
28                 fa[i] = lst;
29                 lst = i;
30             }
31             ask[i] = i;
32         }else{
33             lst = fa[lst];
34             ask[i] = lst;
35         }
36     }
37     vector<int> sz(n + 1), son(n + 1);
38
39     function<void(int)> dfs1 = [&](int u) {
40         sz[u] = 1;
41         for (auto v : G[u]){
42             dfs1(v);
43             sz[u] += sz[v];
44             if (sz[v] > sz[son[u]]) son[u] = v;
45         }
46     };

```

```

47     dfs1(rt);
48
49     function<void(int, int, int)> dfs2 = [&](int u, int num, int sum){
50         memcpy(dp, f[num], sizeof(dp));
51         for (int i = 0; i < 16384; ++i){
52             dp[i] = max(dp[i], f[num][i ^ a[u]] + b[u]);
53         }
54         ans[u] = sum - dp[0];
55         memcpy(f[num], dp, sizeof(f[num]));
56         for (auto v : G[u]){
57             if (v == son[u]) continue;
58             memcpy(f[num + 1], f[num], sizeof(f[num]));
59             dfs2(v, num + 1, sum + b[v]);
60         }
61         if (son[u])
62             dfs2(son[u], num, sum + b[son[u]]);
63     };
64
65     memset(f, -0x3f, sizeof(f));
66     f[0][0] = 0;
67     dfs2(rt, 0, b[rt]);
68     for (int i = 1; i <= n; ++i) cout << ans[ask[i]] << endl;
69 }
70
71 signed main(){
72     ios::sync_with_stdio(false);
73     cin.tie(nullptr);
74     int t = 1;
75     //cin >> t;
76     while(t--) solve();
77     return 0;
78 }
79

```

树状数组

1. 二维偏序

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  constexpr int MAXN = 1e7 + 5;
5  int sum[MAXN], ans[MAXN];
6  vector<pair<int, int>> vec;
7  vector<tuple<int, int, int, int>> q;
8
9  inline int lowbit(int x) { return x & (-x); }
10
11 void add(int pos, int x)
12 {
13     for (; pos < MAXN; pos += lowbit(pos))
14         sum[pos] += x;
15 }
16
17 int query_presum(int pos)
18 {
19     int ans = 0;
20     for (; pos > 0; pos -= lowbit(pos))
21         ans += sum[pos];
22     return ans;
23 }
24
25 int main()
26 {
27     ios::sync_with_stdio(0), cin.tie(0), cout.tie(0);
28     int n, m, x1, x2, y1, y2;
29     cin >> n >> m;
30     for (int i = 0; i < n; i++)
31     {
32         cin >> x1 >> y1, ++x1, ++y1;
33         vec.emplace_back(x1, y1);
34     }
35     for (int i = 0; i < m; i++)
36     {
37         cin >> x1 >> y1, ++x1, ++y1;
38         q.emplace_back(x1, y1, i);
39     }
40     sort(vec.begin(), vec.end());
41     sort(q.begin(), q.end());
42     for (int i = 0; i < m; i++)
43     {
44         int x1 = q[i].get<0>, y1 = q[i].get<1>, id = q[i].get<2>;
45         int ans = query_presum(y1);
46         for (; i < m; i++)
47         {
48             if (vec[i].first > x1)
49                 ans += sum[vec[i].second];
50             else
51                 break;
52         }
53         cout << ans << endl;
54     }
55 }

```

```

34     }
35     sort(vec.begin(), vec.end());
36     for (int i = 0; i < m; i++)
37     {
38         cin >> x1 >> y1 >> x2 >> y2;
39         ++x1, ++y1, ++x2, ++y2;
40         q.emplace_back(x1 - 1, y1 - 1, 1, i);
41         q.emplace_back(x1 - 1, y2, -1, i);
42         q.emplace_back(x2, y1 - 1, -1, i);
43         q.emplace_back(x2, y2, 1, i);
44     }
45     sort(q.begin(), q.end());
46     int cur = 0;
47     for (auto [x, y, c, id] : q)
48     {
49         while (cur < n && vec[cur].first <= x)
50             add(vec[cur].second, 1), ++cur;
51         ans[id] += c * query_presum(y);
52     }
53     for (int i = 0; i < m; i++)
54         cout << ans[i] << "\n";
55
56     return 0;
57 }

```

数位 DP

1.XOR SUM

路易斯喜欢整数。他将非负整数序列 $A = [a_1, a_2, \dots, a_k]$ 的值定义为以下公式 (\oplus 表示位排他性-或):

$$\sum_{i=1}^k \sum_{j=1}^{i-1} a_i \oplus a_j$$

他想知道有多少个不同的序列 A 满足以下条件:

- A 的长度是 k 。
- A 的值是 n 。
- $0 \leq a_i \leq m$ ($1 \leq i \leq k$).

当且仅当存在一个索引 i 时, $[a_1, \dots, a_k]$ 、 $[b_1, \dots, b_k]$ 这两个序列被认为是不同的, 即 $a_i \neq b_i$ 。告诉路易斯答案模块 $10^9 + 7$ 。

输入一行包含三个整数 n, m, k , ($0 \leq n \leq 10^{15}, 0 \leq m \leq 10^{12}, 1 \leq k \leq 18$)。

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 21;
8  const ll mod = 1e9 + 7;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11 int C[N][N];
12
13 void init(int n){
14     int i, j;
15     for (C[0][0] = i = 1; i <= n; ++i)
16         for (C[i][0] = j = 1; j <= i; ++j)
17             C[i][j] = (C[i - 1][j] + C[i - 1][j - 1]) % mod;
18 }
19
20 ll n, m, k;
21 unordered_map<ll, ll> f[N << 1][N];
22 ll dp(int now, int lim, ll sum){
23     if (sum > n) return 0;
24     if (sum + (k >> 1ll) * (k + 1 >> 1ll) * ((1ll << now + 1) - 1) < n) return 0ll;

```

```

25     if (now < 0) return sum == n;
26     if (f[now][lim].count(sum)) return f[now][lim][sum];
27     ll res = 0;
28     if (m >> now & 1){
29         for (int i = 0; i <= lim; ++i) //之前卡上界的个数
30             for (int j = 0; j <= k - lim; ++j)
31                 (res += C[lim][i] * C[k - lim][j] % mod * dp(now - 1, i, sum + (i + j) * (k - i - j) * (1ll << now)) %
↪ mod) %= mod;
32     }else{
33         for (int j = 0; j <= k - lim; ++j){
34             (res += C[lim][0] * C[k - lim][j] % mod * dp(now - 1, lim, sum + j * (k - j) * (1ll << now)) % mod) %= mod;
35         }
36     }
37     return f[now][lim][sum] = res;
38 }
39
40 void solve(){
41     cin >> n >> m >> k;
42     init(k);
43     ll ans = dp(39, k, 0);
44     cout << ans << endl;
45 }
46
47 signed main(){
48     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
49     int t = 1;
50     //cin >> t;
51     while(t--) solve();
52     return 0;
53 }

```

math

1. 区间筛法

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define int ll
4
5  using ll = long long;
6
7  constexpr int N = 1.2e6 + 10;
8  constexpr int P = 1e5 + 10;
9  constexpr int M = 108;
10 constexpr int mod = 998244353;
11
12 using namespace std;
13
14 bool vis[N];
15 int prime[P], pcnt, ans;
16
17 void init(){
18     for (int i = 2; i < N; ++i) if (!vis[i]){
19         prime[++pcnt] = i;
20         for (int j = 2; j * i < N; ++j){
21             vis[i * j] = 1;
22         }
23     }
24 }
25
26 inline int S(ll x){
27     int res = 0;
28     while(x) res += x % 10, x /= 10;
29     return res;
30 }
31
32 int val[110];
33
34 void solve(){
35     ll n;
36     ans = 0;

```

```

37 cin >> n;
38 if (n <= M){
39     for (int i = 1; i <= n; ++i)
40         if (n % i == S(i)) ans++;
41     cout << ans << endl;
42     return;
43 }
44 vector<pair<ll, int>> a[M + 1];
45 for (int i = 0; i <= M; ++i) val[i] = n - i;
46 for (int i = 1; 1ll * prime[i] * prime[i] <= n; ++i){
47     if (prime[i] <= M){
48         for (int j = 0; j <= M; ++j){
49             int cnt = 0;
50             while (val[j] % prime[i] == 0){
51                 val[j] /= prime[i];
52                 cnt++;
53             }
54             if (cnt) a[j].push_back({prime[i], cnt});
55         }
56     }else if (n % prime[i] <= M){
57         int j = n % prime[i], cnt = 0;
58         while (val[j] % prime[i] == 0){
59             val[j] /= prime[i];
60             cnt++;
61         }
62         if (cnt) a[j].push_back({prime[i], cnt});
63     }
64 }
65 for (int i = 0; i <= M; ++i){
66     if (val[i] > 1) a[i].push_back({val[i], 1});
67 }
68
69 using pt = vector<pair<ll, int>>::iterator;
70 auto dfs = [&](auto self, ll x, pt s, pt t, int res) -> void {
71     if (s == t) {
72         if (S(x) == res) ans += (x > res);
73         return;
74     }
75     auto [y, z] = *s;
76     ++s, self(self, x, s, t, res);
77     for (int i = 1; i <= z; ++i){
78         x *= y;
79         self(self, x, s, t, res);
80     }
81     return;
82 };
83
84 for (int i = 0; i <= M; ++i){
85     dfs(dfs, 1, a[i].begin(), a[i].end(), i);
86 }
87 cout << ans << endl;
88 }
89
90 signed main(){
91     ios::sync_with_stdio(false);
92     cin.tie(nullptr);
93     init();
94     int t = 1;
95     cin >> t;
96     while(t--) solve();
97     return 0;
98 }

```

2. 矩阵四则运算

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3
4  using ll = long long;
5
6  constexpr int N = 2e5 + 10;
7  constexpr int mod = 1e9 + 7;

```



```

8
9 using namespace std;
10
11 constexpr int SIZE = 100;
12
13 struct Matrix {
14     ll M[SIZE + 5][SIZE + 5];
15
16     void clear() {memset(M, 0, sizeof(M));}
17
18     void reset() {
19         clear();
20         for (int i = 1; i <= SIZE; i++) M[i][i] = 1;
21     }
22
23     Matrix friend operator * (const Matrix & A, const Matrix & B) {
24         Matrix C;
25         C.clear();
26         for (int i = 1; i <= SIZE; i++) {
27             for (int j = 1; j <= SIZE; j++) {
28                 for (int k = 1; k <= SIZE; k++) {
29                     C.M[i][j] = (C.M[i][j] + A.M[i][k] * B.M[k][j]) % mod;
30                 }
31             }
32         }
33         return C;
34     }
35
36     Matrix friend operator + (const Matrix & A, const Matrix & B) {
37         Matrix C;
38         C.clear();
39         for (int i = 1; i <= SIZE; i++) {
40             for (int j = 1; j <= SIZE; j++) {
41                 C.M[i][j] = (A.M[i][j] + B.M[i][j]) % mod;
42             }
43         }
44         return C;
45     }
46
47     Matrix friend operator - (const Matrix & A, const Matrix & B) {
48         Matrix C;
49         C.clear();
50         for (int i = 1; i <= SIZE; i++){
51             for (int j = 1; j <= SIZE; j++){
52                 C.M[i][j] = (A.M[i][j] - B.M[i][j] + mod) % mod;
53             }
54         }
55         return C;
56     }
57
58     Matrix friend operator ^ (const Matrix & A, ll p) {
59         Matrix C;
60         C.reset();
61         auto tmp = A;
62         while (p) {
63             if (p & 1) C = C * tmp;
64             tmp = tmp * tmp;
65             p >>= 1;
66         }
67         return C;
68     }
69 };
70
71 Matrix t3, t4, t5;
72
73 Matrix MatrixSum(const Matrix & A, ll c) {
74     if (c == 1){
75         t3 = A;
76         return A;
77     }
78     Matrix t2 = MatrixSum(A, c / 2ll);

```

```

79     if (c % 2) {
80         t4 = t3 * t3 * A;
81         t5 = t3;
82         t3 = t4;
83         return (t2 + (t2 * t5)) + t3;
84     }else{
85         t4 = t3 * t3;
86         t5 = t3;
87         t3 = t4;
88         return t2 + (t2 * t5);
89     }
90 }
91
92 Matrix f1, f2;
93
94 void solve(){
95     ll n, m, L, R;
96     cin >> n >> m >> L >> R;
97     f1.clear(), f2.clear();
98     vector<vector<ll>> e(2 * n + 1, vector<ll>(2 * n + 1, 0ll));
99     for (int i = 1; i <= n * 2; ++i){
100         for (int j = 1; j <= n * 2; ++j) {
101             e[i][j] = 0;
102         }
103     }
104     for (int i = 1; i <= m; ++i){
105         ll u, v, w;
106         cin >> u >> v >> w;
107         e[u][v] = (e[u][v] + w) % mod;
108     }
109     f1.M[1][1] = 1;
110     for (int i = 2; i <= n; ++i){
111         for (int j = 1; j < i; ++j){
112             f1.M[1][i] = (f1.M[1][i] + f1.M[1][j] * e[j][i]) % mod;
113         }
114     }
115
116     for (int i = 1; i <= n; ++i){
117         for (int j = 1; j < n + i; ++j){
118             if (j <= n) {
119                 f2.M[j][i] = (f2.M[j][i] + e[j][n + i]) % mod;
120             }else {
121                 for (int k = 1; k <= n; ++k){
122                     f2.M[k][i] = (f2.M[k][i] + f2.M[k][j - n] * e[j - n][i]) % mod;
123                 }
124             }
125         }
126     }
127     ll st = L / n; if (L % n == 0) st--;
128     ll ed = R / n; if (R % n == 0) ed--;
129     Matrix f3, f4;
130     f3.clear(), f4.clear();
131     if (!st) {
132         f3 = f1;
133     }else {
134         auto p = f2 ^ st;
135         for (int i = 1; i <= n; ++i){
136             for (int j = 1; j <= n; ++j){
137                 f3.M[1][i] = (f3.M[1][i] + f1.M[1][j] * p.M[j][i]) % mod;
138             }
139         }
140     }
141     ll ans = 0;
142     ll t1 = L % n;
143     if (!t1) t1 = n;
144     if (st == ed){
145         ll t2 = R % n;
146         if (!t2) t2 = n;
147         for (int i = t1; i <= t2; ++i){
148             ans = (ans + f3.M[1][i]) % mod;
149         }

```

```

150 }else{
151     for (int i = t1; i <= n; ++i) ans = (ans + f3.M[1][i]) % mod;
152     f4 = f3;
153     f3.clear();
154     auto p = f2 ^ ed;
155     for (int i = 1; i <= n; ++i){
156         for (int j = 1; j <= n; ++j){
157             f3.M[1][i] = (f3.M[1][i] + f1.M[1][j] * p.M[j][i]) % mod;
158         }
159     }
160     ll t2 = R % n;
161     if (!t2) t2 = n;
162     for (int i = 1; i <= t2; ++i){
163         ans = (ans + f3.M[1][i]) % mod;
164     }
165     if (st != ed - 1){
166         auto p = MatrixSum(f2, ed - st - 1);
167         f3.clear();
168         for (int i = 1; i <= n; ++i) {
169             for (int j = 1; j <= n; ++j){
170                 f3.M[1][i] = (f3.M[1][i] + f4.M[1][j] * p.M[j][i]) % mod;
171             }
172         }
173         for (int i = 1; i <= n; ++i){
174             ans = (ans + f3.M[1][i]) % mod;
175         }
176     }
177 }
178 cout << ans << endl;
179 }
180
181 signed main(){
182     ios::sync_with_stdio(false);
183     cin.tie(nullptr);
184     int t = 1;
185     cin >> t;
186     while(t--> solve());
187     return 0;
188 }

```

3. 构造异或方程组非 0 解

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3
4  using ll = long long;
5
6  constexpr int N = 2e5 + 10;
7  constexpr int mod = 998244353;
8
9  using namespace std;
10
11 array<bitset<60>, 60> matrix;
12 array<ll, 60> ans = {0};
13
14 void solve(){
15     int n, m;
16     cin >> n >> m;
17     for (int i = 1; i <= m; ++i){
18         int u, v;
19         cin >> u >> v; u--, v--;
20         matrix[u].set(v);
21         matrix[v].set(u);
22     }
23
24     for (int i = 0, j = 0; i < n; ++i){
25         for (int k = j; k < n; ++k){
26             if (matrix[k][i]){
27                 swap(matrix[j], matrix[k]);
28                 for (int l = 0; l < n; ++l){
29                     if (l != j && matrix[l][i]){
30                         matrix[l] ^= matrix[j];

```

```

31         }
32     }
33     ++j;
34     break;
35 }
36 }
37 }
38
39 for (int i = 0; i < n; ++i){
40     if (matrix[i].count() == 1){
41         cout << "No" << endl;
42         return;
43     }
44     if (!matrix[i].count()) break;
45     int a = 0;
46     while (!matrix[i][a] ++a;
47     for (int j = a + 1; j < n; ++j){
48         if (matrix[i][j]){
49             ans[a] |= 1ll << n - j - 1;
50             ans[j] |= 1ll << n - j - 1;
51         }
52     }
53 }
54
55 cout << "Yes" << endl;
56 for (int i = 0; i < n; ++i){
57     cout << ans[i] + !ans[i] << " \n"[i == n - 1];
58 }
59 }
60
61 signed main(){
62     ios::sync_with_stdio(false);
63     cin.tie(nullptr);
64     int t = 1;
65     //cin >> t;
66     while(t--) solve();
67     return 0;
68 }

```

图论

1.(有向图) 强联通分量缩点

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3
4  using ll = long long;
5
6  constexpr int N = 2e5 + 10;
7  constexpr int mod = 998244353;
8
9  using namespace std;
10
11 struct SCC{
12     int n, now, cnt;
13     vector<vector<int>> ver;
14     vector<int> dfn, low, col, id, S;
15
16     SCC(int n) : n(n), ver(n + 1), low(n + 1), id(n + 1){
17         dfn.resize(n + 1, -1);
18         col.resize(n + 1, -1);
19         now = cnt = 0;
20     }
21
22     void add(int x, int y){
23         ver[x].push_back(y);
24     }
25
26     void tarjan(int x){
27         dfn[x] = low[x] = now++;
28         S.push_back(x);

```

```

29     for (auto y : ver[x]) {
30         if (dfn[y] == -1) {
31             tarjan(y);
32             low[x] = min(low[x], low[y]);
33         } else if (col[y] == -1) {
34             low[x] = min(low[x], dfn[y]);
35         }
36     }
37     if (dfn[x] == low[x]) {
38         int pre;
39         cnt++;
40         id[cnt] = x;
41         do {
42             pre = S.back();
43             col[pre] = cnt;
44             S.pop_back();
45         } while (pre != x);
46     }
47 }
48
49 auto work () {
50     for (int i = 1; i <= n; ++i) {
51         if (dfn[i] == -1) tarjan(i);
52     }
53
54     vector<int> siz(cnt + 1);
55     vector<vector<int>> adj(cnt + 1);
56
57     for (int i = 1; i <= n; ++i) {
58         siz[col[i]]++;
59         for (auto j : ver[i]) {
60             int x = col[i], y = col[j];
61             if (x != y) {
62                 adj[x].push_back(y);
63             }
64         }
65     }
66
67     //return tuple{cnt, adj, col, sz};
68     return tuple{cnt, col, siz, id};
69 }
70 };
71
72 void solve() {
73     int n, m;
74     cin >> n >> m;
75     SCC scc(n);
76     vector<pair<int, int>> e(n + 1);
77     for (int i = 1; i <= m; ++i) {
78         int u, v;
79         cin >> u >> v;
80         e[i] = {u, v};
81         scc.add(u, v);
82     }
83
84     auto [cnt, col, sz, id] = scc.work();
85
86     vector<int> in(n + 1);
87     for (int i = 1; i <= m; ++i) {
88         auto [u, v] = e[i];
89         if (col[u] != col[v]) {
90             in[col[v]]++;
91         }
92     }
93     vector<int> ans;
94     // for (int i = 1; i <= cnt; ++i) {
95     //     cout << id[i] << " \n"[i == cnt];
96     // }
97     for (int i = 1; i <= cnt; ++i) {
98         if (in[i] == 0) ans.push_back(id[i]);
99     }

```

```

100     sort(ans.begin(), ans.end());
101     cout << ans.size() << endl;
102     for (auto x : ans) {
103         cout << x << " ";
104     }
105     cout << endl;
106 }
107
108 signed main(){
109     ios::sync_with_stdio(false);
110     cin.tie(nullptr);
111     int t = 1;
112     //cin >> t;
113     while(t--) solve();
114     return 0;
115 }

```

Segment Tree

1. 线段树 I

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define int ll
4  using namespace std;
5  typedef long double db;
6  typedef long long ll;
7
8  const ll N = 4e5 + 10;
9  const ll mod = 998244353;
10 const ll inf32 = 0x3f3f3f3f;
11 const ll inf64 = 5e18;
12
13 ll a[N], sum[N << 2], tag[N << 2];
14
15 void push_up(int x) {sum[x] = sum[x << 1] + sum[x << 1 | 1];}
16
17 void build(int u, int l, int r){
18     tag[u] = 0;
19     if (l == r) {sum[u] = a[l]; return;}
20     int mid = (l + r) >> 1;
21     build(u << 1, l, mid);
22     build(u << 1 | 1, mid + 1, r);
23     push_up(u);
24 }
25
26 inline void f(int u, int l, int r, int k){
27     tag[u] += k;
28     sum[u] += k * (r - l + 1);
29 }
30
31 inline void push_down(int u, int l, int r){
32     int mid = (l + r) >> 1;
33     f(u << 1, l, mid, tag[u]);
34     f(u << 1 | 1, mid + 1, r, tag[u]);
35     tag[u] = 0;
36 }
37
38 inline void update(int u, int nl, int nr, int l, int r, int k){
39     if (nl <= l && r <= nr) {f(u, l, r, k); return;}
40     push_down(u, l, r);
41     int mid = (l + r) >> 1;
42     if (nl <= mid) update(u << 1, nl, nr, l, mid, k);
43     if (nr > mid) update(u << 1 | 1, nl, nr, mid + 1, r, k);
44     push_up(u);
45 }
46
47 inline int query(int u, int qx, int qy, int l, int r){
48     ll res = 0;
49     if (qx <= l && r <= qy) return sum[u];
50     int mid = l + r >> 1;

```

```

51     push_down(u, l, r);
52     if (qx <= mid) res += query(u << 1, qx, qy, l, mid);
53     if (qy > mid) res += query(u << 1 | 1, qx, qy, mid + 1, r);
54     return res;
55 }
56
57 void solve(){
58     int n, m;
59     cin >> n >> m;
60     for (int i = 1; i <= n; ++i) cin >> a[i];
61     build(1, 1, n);
62     while(m--){
63         int op, x, y, k;
64         cin >> op >> x >> y;
65         if (op == 1) {
66             cin >> k;
67             update(1, x, y, 1, n, k);
68         } else {
69             cout << query(1, x, y, 1, n) << endl;
70         }
71     }
72 }
73
74 signed main(){
75     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
76     int t = 1;
77     //cin >> t;
78     while(t--){
79         solve();
80     }
81     return 0;
82 }

```

2. 线段树 II

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 4e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 ll a[N], sum[N << 2], taga[N << 2], tagm[N << 2], p;
13
14 void push_up(int x) {sum[x] = (sum[x << 1] + sum[x << 1 | 1]) % p;}
15
16 void build(int u, int l, int r){
17     taga[u] = 0;
18     tagm[u] = 1;
19     if (l == r) {sum[u] = a[l]; return;}
20     int mid = (l + r) >> 1;
21     build(u << 1, l, mid);
22     build(u << 1 | 1, mid + 1, r);
23     push_up(u);
24 }
25
26 inline void push_down(int u, int l, int r){
27     int m = l + r >> 1;
28     ll &tm = tagm[u], &ta = taga[u];
29     if (tm != 1){
30         taga[u << 1] = (taga[u << 1] * tm) % p;
31         taga[u << 1 | 1] = (taga[u << 1 | 1] * tm) % p;
32         tagm[u << 1] = (tagm[u << 1] * tm) % p;
33         tagm[u << 1 | 1] = (tagm[u << 1 | 1] * tm) % p;
34         sum[u << 1] = (sum[u << 1] * tm) % p;
35         sum[u << 1 | 1] = (sum[u << 1 | 1] * tm) % p;
36         tm = 1;
37     }

```

```

38     if (ta){
39         taga[u << 1] = (taga[u << 1] + ta) % p;
40         taga[u << 1 | 1] = (taga[u << 1 | 1] + ta) % p;
41         sum[u << 1] = (sum[u << 1] + ta * (m - l + 1)) % p;
42         sum[u << 1 | 1] = (sum[u << 1 | 1] + ta * (r - m)) % p;
43         ta = 0;
44     }
45 }
46
47 inline void update1(int u, int nl, int nr, int l, int r, int k){
48     if (nl <= l && r <= nr) {
49         tagm[u] = (tagm[u] * k) % p;
50         taga[u] = (taga[u] * k) % p;
51         sum[u] = (sum[u] * k) % p;
52         return;
53     }
54     int mid = (l + r) >> 1;
55     push_down(u, l, r);
56     if (nl <= mid) update1(u << 1, nl, nr, l, mid, k);
57     if (nr > mid) update1(u << 1 | 1, nl, nr, mid + 1, r, k);
58     push_up(u);
59 }
60
61 inline void update2(int u, int nl, int nr, int l, int r, int k){
62     if (nl <= l && r <= nr) {
63         taga[u] = (taga[u] + k) % p;
64         sum[u] = (sum[u] + k * (r - l + 1)) % p;
65         return;
66     }
67     int mid = (l + r) >> 1;
68     push_down(u, l, r);
69     if (nl <= mid) update2(u << 1, nl, nr, l, mid, k);
70     if (nr > mid) update2(u << 1 | 1, nl, nr, mid + 1, r, k);
71     push_up(u);
72 }
73
74 inline int query(int u, int qx, int qy, int l, int r){
75     ll res = 0;
76     if (qx <= l && r <= qy) return sum[u];
77     int mid = l + r >> 1;
78     push_down(u, l, r);
79     if (qx <= mid) res = (res + query(u << 1, qx, qy, l, mid)) % p;
80     if (qy > mid) res = (res + query(u << 1 | 1, qx, qy, mid + 1, r)) % p;
81     return res;
82 }
83
84 void solve(){
85     int n, m;
86     cin >> n >> m >> p;
87     for (int i = 1; i <= n; ++i) cin >> a[i];
88     build(1, 1, n);
89     while(m--){
90         int op, x, y, k;
91         cin >> op >> x >> y;
92         if (op == 1) {
93             cin >> k;
94             update1(1, x, y, 1, n, k);
95         } else if (op == 2){
96             cin >> k;
97             update2(1, x, y, 1, n, k);
98         } else{
99             cout << query(1, x, y, 1, n) << endl;
100         }
101     }
102 }
103
104 signed main(){
105     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
106     int t = 1;
107     //cin >> t;
108     while(t--){

```



```

109     solve();
110 }
111 return 0;
112 }

```

3. 扫描线

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 struct line{
13     int x1, x2, h, tag;
14 }li[N];
15 bool cmp (line a, line b) {return a.h < b.h;}
16
17 struct Tree{
18     int l, r, len, cnt;
19 }tr[N * 8];
20
21 int d[N], n;
22
23 void build(int p, int l, int r){
24     tr[p] = {l, r};
25     if (l == r) return;
26     int mid = l + r >> 1;
27     build(p << 1, l, mid);
28     build(p << 1 | 1, mid + 1, r);
29 }
30
31 void push_up(int p){
32     int l = tr[p].l, r = tr[p].r;
33     if (tr[p].cnt) tr[p].len = d[r + 1] - d[l];
34     else tr[p].len = tr[p << 1].len + tr[p << 1 | 1].len;
35 }
36
37 void update(int p, int l, int r, int k){
38     if (l <= tr[p].l && tr[p].r <= r) {
39         tr[p].cnt += k;
40         push_up(p);
41         return;
42     }
43     int mid = tr[p].l + tr[p].r >> 1;
44     if (l <= mid) update(p << 1, l, r, k);
45     if (r > mid) update(p << 1 | 1, l, r, k);
46     push_up(p);
47 }
48
49 void solve(){
50     cin >> n;
51     for (int i = 1; i <= n; ++i){
52         int a, b, c, e;
53         cin >> a >> b >> c >> e;
54         li[i] = {a, c, b, 1};
55         li[i + n] = {a, c, e, -1};
56         d[i] = a, d[i + n] = c;
57     }
58     n <<= 1;
59     sort(li + 1, li + n + 1, cmp);
60     sort(d + 1, d + n + 1);
61     int len = unique(d + 1, d + n + 1) - d - 1;
62     build(1, 1, len - 1); //只需要 lens - 1 个区间位置即可
63     ll ans = 0;
64     for (int i = 1; i < n; ++i){
65         int x1 = lower_bound(d + 1, d + len + 1, li[i].x1) - d;

```

```

66         int x2 = lower_bound(d + 1, d + len + 1, li[i].x2) - d;
67         update(1, x1, x2 - 1, li[i].tag);
68         ans += 1ll * (li[i + 1].h - li[i].h) * tr[1].len;
69     }
70     cout << ans << endl;
71 }
72
73 signed main(){
74     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
75     int t = 1;
76     //cin >> t;
77     while(t--) solve();
78     return 0;
79 }

```

4. 窗口的星星 (另一个版本的扫描线)

对于每组数据:

第一行 3 个整数 n, W, H 表示有 n 颗星星, 窗口宽为 W , 高为 H 。

接下来 n 行, 每行三个整数 x_i, y_i, l_i 表示星星的坐标在 (x_i, y_i) , 亮度为 l_i 。

输出 T 个整数, 表示每组数据中窗口星星亮度总和的最大值。

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long double db;
5  typedef long long ll;
6
7  const ll N = 2e5 + 10;
8  const ll mod = 998244353;
9  const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;
11
12 ll n, w, h, d[N];
13 struct line{
14     ll x1, x2, h, val;
15     bool operator < (const line& a) const{
16         return (h != a.h) ? h < a.h : val > a.val;
17     }
18 }L[N];
19
20 struct SegmentTree{
21     ll l, r, mx, add;
22 }T[N << 2];
23
24 void push_up(int p){
25     T[p].mx = max(T[p << 1].mx, T[p << 1 | 1].mx);
26 }
27
28 void build(int p, int l, int r){
29     T[p].l = l, T[p].r = r;
30     T[p].mx = T[p].add = 0;
31     if (l == r) return;
32     int mid = l + r >> 1;
33     build(p << 1, l, mid);
34     build(p << 1 | 1, mid + 1, r);
35     push_up(p);
36 }
37
38 void push_down(int p){
39     if (T[p].add){
40         T[p << 1].add += T[p].add;
41         T[p << 1 | 1].add += T[p].add;
42         T[p << 1].mx += T[p].add;
43         T[p << 1 | 1].mx += T[p].add;
44         T[p].add = 0;
45     }
46 }
47

```

```

48 void update(int p, int ql, int qr, ll val){
49     int l = T[p].l, r = T[p].r;
50     if (ql <= l && r <= qr){
51         T[p].add += val;
52         T[p].mx += val;
53         return;
54     }
55     push_down(p);
56     int mid = l + r >> 1;
57     if (ql <= mid) update(p << 1, ql, qr, val);
58     if (qr > mid) update(p << 1 | 1, ql, qr, val);
59     push_up(p);
60 }
61
62 void solve(){
63     cin >> n >> w >> h;
64     for (int i = 1; i <= n; ++i){
65         ll x, y, l;
66         cin >> x >> y >> l;
67         d[(i << 1) - 1] = y;
68         d[(i << 1)] = y + h - 1;
69         L[(i << 1) - 1] = (line){y, y + h - 1, x, l};
70         L[(i << 1)] = (line){y, y + h - 1, x + w - 1, -l};
71     }
72     n <= 1;
73     sort(d, d + n + 1);
74     sort(L + 1, L + n + 1);
75     ll cnt = unique(d, d + n + 1) - d - 1;
76     for (int i = 1; i <= n; ++i){
77         ll x1 = lower_bound(d + 1, d + cnt + 1, L[i].x1) - d;
78         ll x2 = lower_bound(d + 1, d + cnt + 1, L[i].x2) - d;
79         L[i].x1 = x1, L[i].x2 = x2;
80     }
81     build(1, 1, cnt);
82     ll ans = 0;
83     for (int i = 1; i <= n; ++i){
84         update(1, L[i].x1, L[i].x2, L[i].val);
85         ans = max(ans, T[1].mx);
86     }
87     cout << ans << endl;
88 }
89
90 signed main(){
91     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
92     int t = 1;
93     cin >> t;
94     while(t--){ solve(); }
95     return 0;
96 }

```

5. 二分 + 线段树

输入数据的第一行为两个整数 n 和 m , n 表示序列的长度, m 表示局部排序的次数。

第二行为 n 个整数, 表示 1 到 n 的一个排列。

接下来输入 m 行, 每一行有三个整数 op, l, r , op 为 0 代表升序排序, op 为 1 代表降序排序, l, r 表示排序的区间。

最后输入一个整数 q , 表示排序完之后询问的位置

输出数据仅有一行, 一个整数, 表示按照顺序将全部的部分排序结束后第 q 位置上的数字。

```

1 #include <bits/stdc++.h>
2 #define endl '\n'
3 using namespace std;
4 typedef long double db;
5 typedef long long ll;
6
7 const ll N = 1e5 + 10;
8 const ll mod = 998244353;
9 const ll inf32 = 0x3f3f3f3f;
10 const ll inf64 = 5e18;

```

```

11
12 struct Sort
13 { // 记录这 m 次排序操作
14     int op, l, r;
15 } q[N];
16 struct Node
17 {
18     // 线段树
19     int l, r; // sum 记录 01 序列中 1 的个数
20     int sum, lazy; // lazy 为懒标记: 1 代表将此段全部变为 1, -1 代表将此段全部变为 0
21 } tr[N * 4];
22 int n, m, k, a[N];
23 void pushup(int u)
24 {
25     tr[u].sum = tr[u << 1].sum + tr[u << 1 | 1].sum;
26 }
27 void pushdown(int u)
28 {
29     if (tr[u].lazy)
30     {
31         tr[u << 1].lazy = tr[u << 1 | 1].lazy = tr[u].lazy;
32         if (tr[u].lazy == 1)
33         {
34             tr[u << 1].sum = tr[u << 1].r - tr[u << 1].l + 1;
35             tr[u << 1 | 1].sum = tr[u << 1 | 1].r - tr[u << 1 | 1].l + 1;
36         }
37         else
38             tr[u << 1].sum = tr[u << 1 | 1].sum = 0;
39         tr[u].lazy = 0;
40     }
41 }
42 void build(int u, int l, int r, int x)
43 {
44     if (l == r)
45         tr[u] = {l, r, a[l] >= x, 0}; // 序列中大于等于 x 的数变为 1, 小于 x 的数变为 0
46     else
47     {
48         tr[u] = {l, r};
49         int mid = l + r >> 1;
50         build(u << 1, l, mid, x), build(u << 1 | 1, mid + 1, r, x);
51         pushup(u);
52     }
53 }
54 int query(int u, int l, int r) // 查询 [l, r] 中 1 的个数
55 {
56     if (l <= tr[u].l && tr[u].r <= r)
57         return tr[u].sum;
58     pushdown(u);
59     int mid = tr[u].l + tr[u].r >> 1;
60     int sum = 0;
61     if (mid >= l)
62         sum = query(u << 1, l, r);
63     if (mid < r)
64         sum += query(u << 1 | 1, l, r);
65     return sum;
66 }
67 void update(int u, int l, int r, int c) // 将 [l, r] 区间中的数全部变为 c
68 {
69     if (l <= tr[u].l && tr[u].r <= r)
70     {
71         tr[u].sum = c * (tr[u].r - tr[u].l + 1);
72         tr[u].lazy = c ? 1 : -1;
73     }
74     else
75     {
76         pushdown(u);
77         int mid = tr[u].l + tr[u].r >> 1;
78         if (mid >= l)
79             update(u << 1, l, r, c);
80         if (mid < r)
81             update(u << 1 | 1, l, r, c);
82         pushup(u);

```

```

82     }
83 }
84 bool queryPoint(int u, int x) // 查询 x 位置上的数是否为 1
85 {
86     if (tr[u].l == tr[u].r)
87         return tr[u].sum;
88     pushdown(u);
89     int mid = tr[u].l + tr[u].r >> 1;
90     if (x <= mid)
91         return queryPoint(u << 1, x);
92     else
93         return queryPoint(u << 1 | 1, x);
94 }
95 bool check(int mid) // 检查此答案值是否合法
96 {
97     build(1, 1, n, mid); // 用此答案值建树
98     for (int i = 1; i <= m; i++)
99     {
100         int op = q[i].op, l = q[i].l, r = q[i].r; // 对 [l,r] 区间进行排序
101         int cnt = query(1, l, r); // 查询 [l,r] 中 1 的个数
102         if (cnt == 0 || cnt == r - l + 1)
103             continue; // 如果区间中的数全部相同, 那么不需要进行排序
104         if (op)
105         {
106             update(1, l, cnt + l - 1, 1);
107             update(1, cnt + l, r, 0);
108         }
109         else
110         {
111             update(1, l, r - cnt, 0);
112             update(1, r - cnt + 1, r, 1);
113         }
114     }
115     return queryPoint(1, k); // 所有操作完成后查看 k 位置上的数是否为 1
116 }
117
118 void solve()
119 {
120     cin >> n >> m;
121     for (int i = 1; i <= n; ++i)
122         cin >> a[i];
123     for (int i = 1; i <= m; ++i)
124     {
125         cin >> q[i].op >> q[i].l >> q[i].r;
126     }
127     cin >> k;
128     int l = 1, r = n;
129     while (l < r)
130     {
131         int mid = l + r + 1 >> 1;
132         if (check(mid))
133             l = mid;
134         else
135             r = mid - 1;
136     }
137     cout << l << endl;
138 }
139
140 signed main()
141 {
142     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
143     int t = 1;
144     // cin >> t;
145     while (t--)
146         solve();
147     return 0;
148 }

```

6. 线段树合并 (HDU2024 暑期多校 1003)

```

1 #include <bits/stdc++.h>
2 #include <vector>

```

```

3  #define endl '\n'
4  using namespace std;
5  typedef long double db;
6  typedef long long ll;
7  typedef unsigned long long ull;
8
9  const ll N = 5e5 + 10;
10 const ll mod = 998244353;
11 const ll inf32 = 0x3f3f3f3f;
12 const ll inf64 = 5e18;
13
14 vector<int> G[N];
15 int p[N], rk[N];
16
17 int rt[N], ls[N * 60], rs[N * 60], cnt;
18 int c[N * 60];
19 ull t[N * 60];
20 ull cur = 0;
21 ull ans[N], S1[N], S2[N], a[N];
22
23
24 void push_up(int p){
25     t[p] = t[ls[p]] + t[rs[p]];
26     c[p] = c[ls[p]] + c[rs[p]];
27 }
28
29 void upd(int l, int r, int &p, int x, ull v) {
30     if (!p)
31         p = ++cnt;
32     if (l == r) {
33         c[p] = 1;
34         t[p] = v;
35         return;
36     }
37     int m = (l + r) >> 1;
38     if (x <= m)
39         upd(l, m, ls[p], x, v);
40     else
41         upd(m + 1, r, rs[p], x, v);
42     push_up(p);
43 }
44
45 int merge(int x, int y, ull prev1, ull prev2) {
46     if (!x || !y) {
47         cur += prev1 * t[y];
48         cur += prev2 * t[x];
49         return x + y;
50     }
51     int z = ++cnt;
52     ls[z] = merge(ls[x], ls[y], prev1, prev2);
53     rs[z] = merge(rs[x], rs[y], prev1 + c[ls[x]], prev2 + c[ls[y]]);
54     push_up(z);
55     return z;
56 }
57
58 void dfs(int u, int fa){
59     upd(0, N, rt[u], rk[u], a[u] * a[u]);
60     S1[u] = a[u];
61     S2[u] = a[u] * a[u];
62     for (auto v : G[u]){
63         if (v == fa) continue;
64         dfs(v, u);
65         S1[u] += S1[v];
66         S2[u] += S2[v];
67         ans[u] += ans[v];
68         cur = 0;
69         rt[u] = merge(rt[u], rt[v], 0, 0);
70         ans[u] += cur;
71     }
72 }
73

```

```

74 void solve(){
75     int n;
76     cin >> n;
77     for (int i = 1; i < n; ++i){
78         int u, v;
79         cin >> u >> v;
80         --u, --v;
81         G[u].push_back(v);
82         G[v].push_back(u);
83     }
84     for (int i = 0; i < n; ++i) cin >> a[i];
85     for (int i = 0; i < n; ++i) p[i] = i;
86     sort(p, p + n, [&](int i, int j){return a[i] < a[j];});
87     for (int i = 0; i < n; ++i) rk[p[i]] = i;
88     dfs(0, -1);
89     for (int i = 0; i < n; ++i){
90         ans[i] *= 2;
91         ans[i] += S2[i];
92         ans[i] -= S1[i] * S1[i];
93     }
94     ull res = 0;
95     for (int i = 0; i < n; ++i) res ^= ans[i];
96     cout << res << endl;
97 }
98
99 signed main(){
100     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
101     int t = 1;
102     //cin >> t;
103     while(t--) solve();
104     return 0;
105 }

```

7. 动态开点线段树

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  const int N = 100010;
5  #define int long long
6
7  struct node {
8      int l, r;
9      int add, sum;
10 } tr[N << 1];
11 // 正常线段树, 这里不开 4 倍大小会 RE
12
13 int n, m, idx, root;
14 int a[N];
15
16 void pushup(int p) {
17     tr[p].sum = tr[tr[p].l].sum + tr[tr[p].r].sum;
18 }
19
20 void pushdown(int p, int l, int r) {
21     if(tr[p].add) {
22         int mid = l + r >> 1;
23         tr[tr[p].l].sum += (mid - l + 1) * tr[p].add, tr[tr[p].l].add += tr[p].add;
24         tr[tr[p].r].sum += (r - mid) * tr[p].add, tr[tr[p].r].add += tr[p].add;
25         tr[p].add = 0;
26     }
27 }
28
29 void build(int &p, int l, int r) {
30     if(!p) p = ++idx;
31     if(l == r) { tr[p].sum = a[l]; return; }
32     int mid = l + r >> 1;
33     build(tr[p].l, l, mid), build(tr[p].r, mid + 1, r);
34     pushup(p);
35 }
36
37 void modify(int &p, int l, int r, int ql, int qr, int k) {

```

```

38     if(!p) p = ++idx;
39     if(l >= ql && r <= qr) {
40         tr[p].sum += (r - l + 1) * k;
41         tr[p].add += k;
42         return ;
43     }
44     pushdown(p, l, r);
45     int mid = l + r >> 1;
46     if(ql <= mid) modify(tr[p].l, l, mid, ql, qr, k);
47     if(qr > mid) modify(tr[p].r, mid + 1, r, ql, qr, k);
48     pushup(p);
49 }
50
51 int query(int p, int l, int r, int ql, int qr) {
52     if(l >= ql && r <= qr) { return tr[p].sum; }
53     int mid = l + r >> 1;
54     pushdown(p, l, r);
55     int v = 0;
56     if(ql <= mid) v = query(tr[p].l, l, mid, ql, qr);
57     if(qr > mid) v += query(tr[p].r, mid + 1, r, ql, qr);
58     return v;
59 }
60
61 signed main() {
62     cin >> n >> m;
63     for (int i = 1; i <= n; i++) cin >> a[i];
64
65     build(root, 1, n);
66
67     int op, x, y, k;
68     while(m--) {
69         cin >> op >> x >> y;
70         if(op == 1) {
71             cin >> k;
72             modify(root, 1, n, x, y, k);
73         } else {
74             cout << query(root, 1, n, x, y) << endl;
75         }
76     }
77     return 0;
78 }

```

8. 线段树 (单点修改、区间最值)

```

1  #include <bits/stdc++.h>
2  #include <vector>
3  #define endl '\n'
4  using namespace std;
5  typedef long long ll;
6
7  const int N = 2e5 + 10;
8  const int mod = 998244353;
9  const int inf32 = 0x3f3f3f3f;
10 const ll inf64 = 4e18;
11
12 struct Node
13 {
14     int l, r;
15     ll v1, v2;
16 } tr[N * 4];
17
18 ll a[N], b[N];
19
20 void pushup(int u)
21 {
22     tr[u].v1 = max(tr[u << 1].v1, tr[u << 1 | 1].v1);
23     tr[u].v2 = min(tr[u << 1].v2, tr[u << 1 | 1].v2);
24 }
25
26 void build(int u, int l, int r)
27 {
28     tr[u] = {l, r};

```



```

29     if (l == r) {
30         tr[u].v1 = tr[u].v2 = b[l];
31         return;
32     }
33     int mid = l + r >> 1;
34     build(u << 1, l, mid), build(u << 1 | 1, mid + 1, r);
35     pushup(u);
36 }
37
38 ll query1(int u, int l, int r)
39 {
40     if (tr[u].l >= l && tr[u].r <= r) return tr[u].v1;
41     int mid = tr[u].l + tr[u].r >> 1;
42     ll v = -inf64;
43     if (l <= mid) v = query1(u << 1, l, r);
44     if (r > mid) v = max(v, query1(u << 1 | 1, l, r));
45     return v;
46 }
47
48 ll query2(int u, int l, int r)
49 {
50     if (tr[u].l >= l && tr[u].r <= r) return tr[u].v2;
51     int mid = tr[u].l + tr[u].r >> 1;
52     ll v = inf64;
53     if (l <= mid) v = query2(u << 1, l, r);
54     if (r > mid) v = min(v, query2(u << 1 | 1, l, r));
55     return v;
56 }
57
58 void modify(int u, int x, ll v)
59 {
60     if (tr[u].l == x && tr[u].r == x) tr[u].v1 += v, tr[u].v2 += v;
61     else
62     {
63         int mid = tr[u].l + tr[u].r >> 1;
64         if (x <= mid) modify(u << 1, x, v);
65         else modify(u << 1 | 1, x, v);
66         pushup(u);
67     }
68 }
69
70
71 void solve(){
72     int n;
73     cin >> n;
74     for (int i = 1; i <= n; ++i)
75         cin >> a[i];
76     for (int i = 2; i <= n; ++i) {
77         b[i] = a[i] - a[i - 1];
78     }
79     build(1, 1, n);
80     int q;
81     cin >> q;
82     while (q--){
83         int op, l, r;
84         ll x;
85         cin >> op >> l >> r;
86         if (op == 1) {
87             cin >> x;
88             modify(1, l, x);
89             modify(1, r + 1, -x);
90         }else if (op == 2){
91             if (l == r){
92                 cout << 1 << endl;
93             }else{
94                 ll mx = query1(1, l + 1, r);
95                 ll mn = query2(1, l + 1, r);
96                 if (mx == 0 && mn == 0){
97                     cout << 1 << endl;
98                 }else{
99                     cout << 0 << endl;

```

```

100     }
101 }
102 }else if (op == 3){
103     if (l == r){
104         cout << 1 << endl;
105     }else{
106         ll mn = query2(1, l + 1, r);
107         cout << (mn > 0) << endl;
108     }
109 }else if (op == 4){
110     if (l == r){
111         cout << 1 << endl;
112     }else{
113         ll mx = query1(1, l + 1, r);
114         cout << (mx < 0) << endl;
115     }
116 }else {
117     int ql = l + 1, qr = r + 1, ok = 0;
118     while (ql < qr){
119         int mid = (ql + qr) >> 1;
120         ll mn = query2(1, l + 1, mid);
121         if (mn > 0) {
122             ok = mid;
123             ql = mid + 1;
124         }else{
125             qr = mid;
126         }
127     }
128     if (!ok) {
129         cout << 0 << endl;
130     }else{
131         if (ok >= r) {
132             cout << 0 << endl;
133         }else{
134             ll mx = query1(1, ok + 1, r);
135             if (mx < 0) {
136                 cout << 1 << endl;
137             }else{
138                 cout << 0 << endl;
139             }
140         }
141     }
142 }
143 }
144 }
145
146 signed main(){
147     ios::sync_with_stdio(false);
148     cin.tie(nullptr);
149     int t = 1;
150     //cin >> t;
151     while(t--) solve();
152     return 0;
153 }

```

9. 单点修改, 区间最大字段和

```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  #define int long long
5
6  const int maxn = 1001000;
7  int n, m;
8  int ans;
9
10 struct tree{
11     int lmax; // 当前区间最大前缀和
12     int rmax; // 当前区间最大后缀和
13     int maxx; // 当前区间最大子段和
14     int sum; // 当前区间的和
15 }t[4 * maxn];

```

```

16
17 void push_up(int rt){
18     t[rt].sum = t[rt << 1].sum + t[rt << 1 | 1].sum;
19     // 当前区间的和: 左子树的和 + 右子树的和
20     t[rt].rmax = max(t[rt << 1 | 1].rmax, t[rt << 1 | 1].sum + t[rt << 1].rmax);
21     // 当前区间的最大后缀和: 右子树的最大后缀和 or 右子树的和 + 左子树的最大后缀和
22     t[rt].lmax = max(t[rt << 1].lmax, t[rt << 1].sum + t[rt << 1 | 1].lmax);
23     // 当前区间的最大前缀和: 左子树的最大前缀和 or 左子树的和 + 右子树的最大前缀和
24     t[rt].maxx = max(t[rt << 1].rmax + t[rt << 1 | 1].lmax, max(t[rt << 1].maxx, t[rt << 1 | 1].maxx));
25     // 当前区间的最大子段和: 左子树的最大子段和 or 右子树的最大子段和 or 左子树的最大后缀和 + 右子树的最大前缀和
26 }
27
28 void build(int rt, int l, int r){
29     if(l == r){
30         cin >> t[rt].maxx;
31         t[rt].lmax = t[rt].rmax = t[rt].sum = t[rt].maxx;
32         return;
33     }
34     int mid = l + r >> 1;
35     build(rt << 1, l, mid);
36     build(rt << 1 | 1, mid + 1, r);
37     push_up(rt);
38 }
39
40 void update(int rt, int l, int r, int x, int y){
41     if(l == r){
42         t[rt].lmax = t[rt].rmax = t[rt].maxx = t[rt].sum = y;
43         return;
44     }
45     int mid = l + r >> 1;
46     if(mid >= x) update(rt << 1, l, mid, x, y);
47     else update(rt << 1 | 1, mid + 1, r, x, y);
48     push_up(rt);
49 }
50
51 tree query(int rt, int l, int r, int x, int y){
52     if(l >= x && r <= y) return t[rt]; // 区间完全覆盖, 直接返回该节点
53     int mid = l + r >> 1;
54     if(y <= mid) return query(rt << 1, l, mid, x, y); // 只在左区间, 直接查询左区间
55     else if(x > mid) return query(rt << 1 | 1, mid + 1, r, x, y); // 只在右区间, 直接查询右区间
56     else{
57         tree res_l = query(rt << 1, l, mid, x, y);
58         tree res_r = query(rt << 1 | 1, mid + 1, r, x, y);
59         tree res;
60         // res_l 记录左覆盖区间, res_r 记录右覆盖区间, 合并后得到 res
61         // 用 push_up 同样的方式更新 res
62         res.sum = res_l.sum + res_r.sum;
63         res.lmax = max(res_l.sum + res_r.lmax, res_l.lmax);
64         res.rmax = max(res_r.rmax, res_r.sum + res_l.rmax);
65         res.maxx = max(max(res_l.maxx, res_r.maxx), res_l.rmax + res_r.lmax);
66         return res;
67     }
68 }
69
70 signed main(){
71     cin >> n >> m;
72     build(1, 1, n);
73     int opt, x, y;
74     while(m--){
75         cin >> opt >> x >> y;
76         if(opt == 1){
77             if(x > y) swap(x, y);
78             tree ans = query(1, 1, n, x, y);
79             cout << ans.maxx << '\n';
80         }
81         else update(1, 1, n, x, y);
82     }
83     return 0;
84 }

```

10. 半群 (一次函数嵌套)

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define int ll
4  using namespace std;
5  typedef long long ll;
6
7  const int N = 5e5 + 10;
8  const int mod = 998244353;
9  const int inf32 = 0x3f3f3f3f;
10 const ll inf64 = 4e18;
11
12 int n, m, K[N], b[N];
13
14 struct node{
15     int l, r;
16     ll mul, sum;
17 }t[N << 2];
18
19 node unite(node x, node y){
20     node ans;
21     ans.l = x.l, ans.r = y.r;
22     ans.mul = x.mul * y.mul % mod;
23     ans.sum = (x.sum * y.mul % mod + y.sum) % mod;
24     return ans;
25 }
26
27 void push_up(int u){
28     t[u] = unite(t[u << 1], t[u << 1 | 1]);
29 }
30
31 void build(int u, int l, int r){
32     if (l == r){
33         t[u].l = t[u].r = l;
34         t[u].sum = b[l];
35         t[u].mul = K[l];
36         return;
37     }
38     int mid = l + r >> 1;
39     build(u << 1, l, mid);
40     build(u << 1 | 1, mid + 1, r);
41     push_up(u);
42 }
43
44 void modify(int u, int x, int K, int B){
45     if (t[u].l == t[u].r)
46     {
47         t[u].sum = B;
48         t[u].mul = K;
49         return;
50     }
51     int mid = t[u].l + t[u].r >> 1;
52     if (x <= mid) modify(u << 1, x, K, B);
53     else modify(u << 1 | 1, x, K, B);
54     push_up(u);
55 }
56
57 node query(int u, int l, int r){
58     if (t[u].l == l && t[u].r == r) return t[u];
59     int mid = t[u].l + t[u].r >> 1;
60     if (r <= mid) return query(u << 1, l, r);
61     else if (l > mid) return query(u << 1 | 1, l, r);
62     else return unite(query(u << 1, l, mid), query(u << 1 | 1, mid + 1, r));
63 }
64
65 void solve(){
66     cin >> n >> m;
67     for (int i = 1; i <= n; ++i) cin >> K[i] >> b[i];
68     build(1, 1, n);
69     for (int i = 1; i <= m; ++i){
70         int op;
71         cin >> op;

```

```

72     if (op == 0){
73         int p, c, d;
74         cin >> p >> c >> d;
75         p++;
76         modify(1, p, c, d);
77     }else{
78         int l, r, x;
79         cin >> l >> r >> x;
80         l++;
81         auto ans = query(1, l, r);
82         cout << (ans.mul * x % mod + ans.sum) % mod << endl;
83     }
84 }
85 }
86
87 signed main(){
88     ios::sync_with_stdio(false);
89     cin.tie(nullptr);
90     int t = 1;
91     //cin >> t;
92     while(t--) solve();
93     return 0;
94 }

```

11. 树上半群修改, 路径查询

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  using namespace std;
4  typedef long long ll;
5
6  const int N = 2e5 + 10;
7  const int mod = 998244353;
8  const int inf32 = 0x3f3f3f3f;
9  const ll inf64 = 4e18;
10
11 struct Info {
12     int a1, b1, a2, b2;
13 };
14
15 Info operator + (const Info &l, const Info &r) {
16     Info ret;
17     //信息合并
18     ret.a1 = (1LL * l.a1 * r.a1 % mod); //向下的
19     ret.b1 = (1LL * r.a1 * l.b1 % mod + r.b1) % mod;
20     ret.a2 = (1LL * l.a2 * r.a2 % mod); //向上的
21     ret.b2 = (1LL * l.a2 * r.b2 % mod + l.b2) % mod;
22     return ret;
23 }
24
25 struct node {
26     int a, b;
27 };
28
29 node operator + (const node &l, const node &r) {
30     node ret;
31     //信息合并
32     ret.a = (1LL * l.a * r.a % mod);
33     ret.b = (1LL * r.a * l.b % mod + r.b) % mod;
34     return ret;
35 }
36
37 Info t[N << 2];
38 int a[N], b[N], sz[N], top[N], dep[N], in[N], dfn[N], f[N];
39
40 struct Segment{
41     int n;
42
43     void push_up(int u) {
44         t[u] = t[u << 1] + t[u << 1 | 1];
45     }
46 }

```

```

47 void build(int u, int l, int r){
48     if (l == r){
49         t[u].a1 = t[u].a2 = a[dfn[l]];
50         t[u].b1 = t[u].b2 = b[dfn[l]];
51         return;
52     }
53     int mid = l + r >> 1;
54     build(u << 1, l, mid);
55     build(u << 1 | 1, mid + 1, r);
56     push_up(u);
57 }
58
59 void modify(int u, int l, int r, int p, const Info & v){
60     if (l == r){
61         t[u] = v;
62         return;
63     }
64     int mid = l + r >> 1;
65     if (p <= mid) modify(u << 1, l, mid, p, v);
66     else modify(u << 1 | 1, mid + 1, r, p, v);
67     push_up(u);
68 }
69
70 void modify(int u, int l, int r, int ql, int qr, const Info & v){
71     if (l == ql && r == qr){
72         t[u] = v;
73         return;
74     }
75     int mid = l + r >> 1;
76     if (qr <= mid) modify(u << 1, l, mid, ql, qr, v);
77     else if (ql > mid) modify(u << 1 | 1, mid + 1, r, ql, qr, v);
78     else {
79         modify(u << 1, l, mid, ql, mid, v);
80         modify(u << 1 | 1, mid + 1, r, mid + 1, qr, v);
81     }
82     push_up(u);
83 }
84
85 Info query(int rt, int l, int r, int ql, int qr) {
86     if (l == ql && r == qr) {
87         return t[rt];
88     }
89     int mid = l + r >> 1;
90     if (qr <= mid) return query(rt << 1, l, mid, ql, qr);
91     else if (ql > mid) return query(rt << 1 | 1, mid + 1, r, ql, qr);
92     else {
93         return query(rt << 1, l, mid, ql, mid) + query(rt << 1 | 1, mid + 1, r, mid + 1, qr);
94     }
95 }
96
97 Info query(int ql, int qr) {
98     if (ql > qr){
99         return Info {1, 0, 1, 0};
100     }
101     return query(1, 1, n, ql, qr);
102 }
103 }tree;
104
105 void solve(){
106     int n, q;
107     cin >> n >> q;
108     for (int i = 1; i <= n; ++i){
109         cin >> a[i] >> b[i];
110     }
111     vector<vector<int>> G(n + 1);
112     for (int i = 1; i < n; ++i){
113         int u, v;
114         cin >> u >> v;
115         u++, v++;
116         G[u].push_back(v);
117         G[v].push_back(u);

```

```

118 }
119 auto dfs1 = [&](auto self, int u, int fa) -> void {
120     if (fa) G[u].erase(find(G[u].begin(), G[u].end(), fa));
121     sz[u]++;
122     for (auto &v : G[u]){
123         dep[v] = dep[u] + 1;
124         f[v] = u;
125         self(self, v, u);
126         sz[u] += sz[v];
127         if (sz[v] > sz[G[u][0]]){
128             swap(G[u][0], v);
129         }
130     }
131 };
132
133 int tot = 0 ;
134 auto dfs2 = [&](auto self, int u, int fa) -> void {
135     in[u] = ++tot;
136     dfn[in[u]] = u;
137     for (auto &v : G[u]){
138         top[v] = (v == G[u][0] ? top[u] : v);
139         self(self, v, u);
140     }
141 };
142
143 dep[1] = 0;
144 dfs1(dfs1, 1, 0);
145 top[1] = 0;
146 dfs2(dfs2, 1, 0);
147 tree.n = n;
148 tree.build(1, 1, n);
149
150 // 向上的
151 auto path1 = [&](auto self, int x, int y) -> node{
152     node ans = {1, 0};
153     while(top[x] != top[y]){
154         if (dep[top[x]] < dep[top[y]]) std::swap(x, y);
155         Info res = tree.query(in[top[x]], in[x]);
156         ans = (ans + node{res.a2, res.b2});
157         x = f[top[x]];
158     }
159     if (dep[x] > dep[y]) std::swap(x, y);
160     Info res = tree.query(in[x] + 1, in[y]);
161     ans = (ans + node{res.a2, res.b2});
162     return ans;
163 };
164
165 auto path2 = [&](auto self, int x, int y) -> node{
166     node ans = {1, 0};
167     while(top[x] != top[y]){
168         if (dep[top[x]] < dep[top[y]]) std::swap(x, y);
169         Info res = tree.query(in[top[x]], in[x]);
170         ans = (node{res.a1, res.b1} + ans);
171         x = f[top[x]];
172     }
173     if (dep[x] > dep[y]) std::swap(x, y);
174     Info res = tree.query(in[x], in[y]);
175     ans = (node{res.a1, res.b1} + ans);
176     return ans;
177 };
178
179 auto lca = [&](auto self, int u, int v) -> int{
180     while (top[u] != top[v]) {
181         if (dep[top[u]] > dep[top[v]]) {
182             u = f[top[u]];
183         } else {
184             v = f[top[v]];
185         }
186     }
187     return dep[u] < dep[v] ? u : v;
188 };

```

```

189
190 while(q--){
191     int op, l, r, x;
192     cin >> op >> l >> r >> x;
193     l++;
194     if (op == 0){
195         tree.modify(1, 1, n, in[l], Info{r, x, r, x});
196     }else{
197         r++;
198         int lc = lca(lca, l, r);
199         node ansL = path1(path1, l, lc);
200         node ansR = path2(path2, lc, r);
201         int ans = (1ll * ansL.a * x + ansL.b) % mod;
202         ans = (1ll * ansR.a * ans + ansR.b) % mod;
203         cout << ans << endl;
204     }
205 }
206 }
207
208 signed main(){
209     ios::sync_with_stdio(false);
210     cin.tie(nullptr);
211     int t = 1;
212     //cin >> t;
213     while(t--) solve();
214     return 0;
215 }

```

12. 动态开点，权值线段树

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3
4  using ll = long long;
5
6  constexpr int N = 1e5 + 10;
7  constexpr int M = 1e7 + 10;
8  constexpr int mod = 998244353;
9
10 using namespace std;
11
12 struct SegmentTree{
13     int ls, rs;
14     int num;
15 }tr[M];
16
17 int n, q, cnt, np1[N], np2[N], f[N];
18
19 ll m, k, a[N];
20
21 void update(int u, ll l, ll r, ll p, int sum) {
22     tr[u].num = sum;
23     if (l == r) return;
24     ll mid = l + r >> 1;
25     if (p <= mid){
26         if (!tr[u].ls){
27             tr[u].ls = ++cnt;
28         }
29         update(tr[u].ls, l, mid, p, sum);
30     }else{
31         if (!tr[u].rs){
32             tr[u].rs = ++cnt;
33         }
34         update(tr[u].rs, mid + 1, r, p, sum);
35     }
36 }
37
38 int query(int u, ll l, ll r, ll ql, ll qr) {
39     if (ql <= l && r <= qr) return tr[u].num;
40     ll mid = l + r >> 1;
41     int ans = n + 2;
42     if (ql <= mid) {

```



```

43     ans = min(ans, query(tr[u].ls, l, mid, ql, qr));
44 }
45 if (qr > mid) {
46     ans = min(ans, query(tr[u].rs, mid + 1, r, ql, qr));
47 }
48 return ans;
49 }
50
51 void solve(){
52     cin >> n >> m >> k;
53     for (int i = 1; i <= n; ++i) cin >> a[i];
54     cnt = 1, tr[0].num = tr[1].num = n + 1;
55     for (int i = n; i >= 1; --i){
56         np1[i] = query(1, 1ll, m, a[i], min(m, a[i] + k));
57         np2[i] = query(1, 1ll, m, max(1ll, a[i] - k), a[i]);
58         update(1, 1ll, m, a[i], i);
59     }
60     cin >> q;
61     while (q--){
62         int l, r;
63         cin >> l >> r;
64         for (int i = l; i <= r; ++i) f[i] = 0;
65         int ans = 0;
66         for (int i = r; i >= l; --i){
67             f[i] = 1;
68             if (np1[i] <= r) f[i] = max(f[i], f[np1[i]] + 1);
69             if (np2[i] <= r) f[i] = max(f[i], f[np2[i]] + 1);
70             ans = max(ans, f[i]);
71         }
72         cout << (r - l + 1 - ans) << endl;
73     }
74     for (int i = 0; i <= cnt; ++i) tr[i].num = tr[i].ls = tr[i].rs = 0;
75 }
76
77 signed main(){
78     ios::sync_with_stdio(false);
79     cin.tie(nullptr);
80     int t = 1;
81     cin >> t;
82     while(t--){ solve();
83         return 0;
84     }
85 }

```

串串

1. 本质不同子序列数量

```

1 vector<int> dp(n + 1);
2 map<char, int> vis;
3 for (int i = pl + 1; i < pr; ++i){
4     if(!vis.count(s[i])){
5         vis[s[i]] = 1;
6         dp[i] = (dp[i - 1] * 2 + 1) % P;
7     }else{
8         dp[i] = (((dp[i - 1] * 2) % P - dp[vis[s[i]] - 1] + P) % P + P) % P;
9     }
10    vis[s[i]] = i;
11 }

```

2. 子序列自动机

```

1 #include <bits/stdc++.h>
2 #define endl '\n'
3 #define pll pair<i64, i64>
4 #define tll tuple<i64, i64, i64>
5 #define all(a) a.begin() + 1, a.end()
6 using namespace std;
7 using i64 = long long;
8 using db = long double;
9 const i64 N = 1e5 + 10;
10 const i64 mod = 998244353;

```

```

11  const i64 inf32 = 1e9;
12  const i64 inf64 = 5e18;
13
14  int typ, n, q, m;
15  int root[N];
16  struct PersistentTree{
17      int ls[N << 5], rs[N << 5], nxt[N << 5], tot = 0;
18      //单点修改
19      inline void update(int & rt, int old, int l, int r, int p, int k){
20          rt = ++tot;
21          ls[rt] = ls[old], rs[rt] = rs[old], nxt[rt] = nxt[old];
22          if (l == r){
23              nxt[rt] = k;
24              return;
25          }
26          int mid = l + r >> 1;
27          if (p <= mid) update(ls[rt], ls[old], l, mid, p, k);
28          else update(rs[rt], rs[old], mid + 1, r, p, k);
29      }
30      //单点查询
31      inline int query(int rt, int l, int r, int p){
32          if (l == r) return nxt[rt];
33          int mid = l + r >> 1;
34          if (p <= mid) return query(ls[rt], l, mid, p);
35          else return query(rs[rt], mid + 1, r, p);
36      }
37      //建树
38      inline void build(vector<int> a){
39          int n = a.size() - 1;
40          for (int i = n; i >= 1; --i){
41              update(root[i], root[i + 1], 1, m, a[i], i + 1);
42          }
43      }
44  }T;
45
46  void solve(){
47      cin >> typ >> n >> q >> m;
48      vector<int> a(n + 1);
49      for (int i = 1; i <= n; ++i) cin >> a[i];
50      T.build(a);
51      while (q--){
52          int k;
53          cin >> k;
54          bool ok = true;
55          int rt = 1;
56          while (k--){
57              int p;
58              cin >> p;
59              int to = T.query(root[rt], 1, m, p);
60              if (!to) ok = false;
61              if (to) rt = to;
62          }
63          if (ok) cout << "Yes" << endl;
64          else cout << "No" << endl;
65      }
66  }
67
68  int main(){
69      ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
70      int t = 1;
71      //cin >> t;
72      while(t--) solve();
73      return 0;
74  }

```