

# Std Code Library(Qinhuangdao)

tingyx

Nanjing University of Science and Technology

October 3, 2023

# Contents

一切的开始	2
Codeforces/XCPC . . . . .	2
int128 . . . . .	2
数据结构	3
二维数点 . . . . .	3
可持久化线段树 . . . . .	6
区间问题	8
莫队 . . . . .	8
CDQ . . . . .	9
树上问题	13
树剖 . . . . .	13
dsu . . . . .	20
计算几何	29
二维几何: 点与向量 . . . . .	29
字符串	30
KMP . . . . .	30
杂项	32
线性基 . . . . .	32
Tarjan . . . . .	34

## 一切的开始

### Codeforces/XCPC

- 需要 C++17/C++20

```
1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second
9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll maxn = 2e5 + 10;
15 const ll mod = 998244353;
16 const ll inf32 = 1e9;
17 const ll inf64 = 1e18;
18
19
20 void solve(){
21
22 }
23
24 int main(){
25     ios;
26     //freopen("sample.txt", "r", stdin);
27     //freopen("resout.txt", "w", stdout);
28     int t = 1;
29     //cin >> t;
30     while(t--){
31         solve();
32     }
33     return 0;
34 }
35 // -----
```

### int128

- 不要使用 cin/cout, 记得关同步流

```
1  typedef __int128 i128;
2
3  i128 read()
4  {
5      i128 x = 0; bool f = 0;
6      char c = getchar();
7      while (c < '0' || c > '9')
8      {
9          if (c == '-')
10             f = 1;
11             c = getchar();
12     }
13     while (c >= '0' && c <= '9')
14     {
15         x = (x << 1) + (x << 3) + (c ^ 48);
16         c = getchar();
17     }
18     return f ? -x : x;
19 }
20
21 inline void write(i128 x)
22 {
23     if (x < 0)
24         putchar('-'), x = -x;
25     if (x > 9)
```

```

26     write(x / 10);
27     putchar(x % 10 + '0');
28 }

```

## 数据结构

### 二维数点

- 逆序对

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  const int maxn = 500010;
5  ll m;
6  ll a[maxn], b[maxn], c[maxn];
7  int lowbit(int x){return x & (-x);}
8  void add(int x, ll y){
9      for (int i = x; i <= m; i += lowbit(i)) c[i] += y;
10 }
11 ll sum(int x){
12     ll res = 0;
13     for (int i = x; i; i -= lowbit(i)) res += c[i];
14     return res;
15 }
16 int main(){
17     int n;
18     cin >> n;
19     for (int i = 1; i <= n; ++i){
20         cin >> a[i];
21         b[i] = a[i];
22     }
23     sort(b + 1, b + n + 1);
24     m = unique(b + 1, b + n + 1) - b - 1;
25     ll ans = 0;
26     for (int i = n; i; i--){
27         int k = lower_bound(b + 1, b + m + 1, a[i]) - b;
28         ans += sum(k - 1);
29         add(k, 1);
30     }
31     cout << ans;
32     return 0;
33 }

```

- 园丁的烦恼 (矩阵内点的个数)

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define pii pair<int, int>
6  #define vi vector<int>
7  #define vl vector<ll>
8  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
9  #define per(i, j, k) for(int i = (j); i >= (k); i--)
10 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
11 using namespace std;
12 typedef long long ll;
13 const ll maxn = 1e7 + 10;
14 const ll mod = 998244353;
15 const ll inf = 0x3f3f3f3f;
16
17 struct BIT{
18     int tr[maxn];
19     int lowbit(int x){return x & -x;}
20     void add(int p, int x){
21         for (; p < maxn; p += lowbit(p)) tr[p] += x;
22     }
23     ll query(int p){
24         ll sum = 0;

```

```

25         for (; p > 0; p -= lowbit(p))
26             sum += tr[p];
27         return sum;
28     }
29 }Tr;
30
31 void solve(){
32     int n, m;
33     cin >> n >> m;
34     vector<pii> pos;
35     vector<tuple<int, int, int, int>> q;
36     vector<ll> ans(m + 1);
37     rep(i, 1, n){
38         int tx, ty;
39         cin >> tx >> ty;
40         tx++, ty++;
41         pos.push_back({tx, ty});
42     }
43     sort(pos.begin(), pos.end());
44     rep(i, 1, m){
45         int x1, y1, x2, y2;
46         cin >> x1 >> y1 >> x2 >> y2;
47         x1++, y1++, x2++, y2++;
48         q.push_back({x1 - 1, y1 - 1, 1, i});
49         q.push_back({x1 - 1, y2, -1, i});
50         q.push_back({x2, y1 - 1, -1, i});
51         q.push_back({x2, y2, 1, i});
52     }
53     sort(q.begin(), q.end());
54     int cur = 0;
55     for (auto [x, y, c, id] : q){
56         while (cur < n && pos[cur].first <= x) Tr.add(pos[cur++].second, 1);
57         ans[id] += c * Tr.query(y);
58     }
59     rep(i, 1, m) cout << ans[i] << endl;
60 }
61
62 int main(){
63     ios;
64     //freopen("sample.txt", "r", stdin);
65     //freopen("resout.txt", "w", stdout);
66     int t = 1;
67     //cin >> t;
68     while(t--){
69         solve();
70     }
71     return 0;
72 }

```

- HH 的项链（区间元素种类）照常把  $x$  所在一维降掉后，发现  $y$  轴并没有明显的偏序关系。可以这样考虑，我们只计每个元素第一次在区间中出现时有贡献，设  $pre[i]$  表示位置  $i$  的元素前一次出现的位置，在整个序列中第一次出现时记为 0

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
8  #define per(i, j, k) for(int i = (j); i >= (k); i--)
9  #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
10 using namespace std;
11 typedef long long ll;
12 const ll maxn = 1e6 + 10;
13 const ll mod = 998244353;
14 const ll inf = 0x3f3f3f3f;
15
16 struct BIT{
17     ll tr[maxn];
18     int lowbit(int x){return x & -x;}
19     void add(int p, ll x){

```

```

20     for (; p < maxn; p += lowbit(p)) tr[p] += x;
21 }
22 ll query(int p){
23     ll sum = 0;
24     for (; p > 0; p -= lowbit(p))
25         sum += tr[p];
26     return sum;
27 }
28 }Tr;
29
30 ll pre[maxn], ans[maxn];
31 void solve(){
32     int n, m;
33     cin >> n;
34     vector<pll> pos;
35     vector<tuple<int, int, int, int>> q;
36     for (int i = 3; i <= n + 2; ++i){
37         int a;
38         cin >> a;
39         pos.push_back({i, pre[a] ? pre[a] : 2}), pre[a] = i;
40     }
41     sort(pos.begin(), pos.end());
42     cin >> m;
43     for (int i = 1; i <= m; ++i){
44         int l, r;
45         cin >> l >> r;
46         l += 2, r += 2;
47         q.push_back({l - 1, 1, 1, i});
48         q.push_back({l - 1, l - 1, -1, i});
49         q.push_back({r, 1, -1, i});
50         q.push_back({r, l - 1, 1, i});
51     }
52     sort(q.begin(), q.end());
53     int cur = 0;
54     for (auto [x, y, c, id] : q)
55     {
56         while (cur < n && pos[cur].first <= x)
57             Tr.add(pos[cur++].second, 1);
58         ans[id] += c * Tr.query(y);
59     }
60     for (int i = 1; i <= m; i++) cout << ans[i] << endl;
61 }
62
63 int main(){
64     ios;
65     //freopen("sample.txt", "r", stdin);
66     //freopen("resout.txt", "w", stdout);
67     int t = 1;
68     //cin >> t;
69     while(t--){
70         solve();
71     }
72     return 0;
73 }

```

● 矩阵内权值之和

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
8  #define per(i, j, k) for(int i = (j); i >= (k); i--)
9  #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
10 using namespace std;
11 typedef long long ll;
12 const ll maxn = 3e5 + 10;
13 const ll mod = 998244353;
14 const ll inf = 0x3f3f3f3f;
15

```

```

16 struct BIT{
17     ll tr[maxn];
18     int lowbit(int x){return x & -x;}
19     void add(int p, ll x){
20         for (; p < maxn; p += lowbit(p)) tr[p] += x;
21     }
22     ll query(int p){
23         ll sum = 0;
24         for (; p > 0; p -= lowbit(p))
25             sum += tr[p];
26         return sum;
27     }
28 }Tr;
29
30 void solve(){
31     int n, m;
32     cin >> n >> m;
33     vector<tuple<int, int, int>> pos;
34     vector<tuple<int, int, int, int>> q;
35     vector<ll> ans(m + 1);
36     vector<int> yy;
37     rep(i, 1, n){
38         int x, y, p;
39         cin >> x >> y >> p;
40         yy.push_back(y);
41         pos.push_back({x, y, p});
42     }
43     sort(pos.begin(), pos.end());
44     rep(i, 1, m){
45         int x1, y1, x2, y2;
46         cin >> x1 >> y1 >> x2 >> y2;
47         yy.push_back(y1 - 1), yy.push_back(y2);
48         q.push_back({x1 - 1, y1 - 1, 1, i});
49         q.push_back({x2, y1 - 1, -1, i});
50         q.push_back({x1 - 1, y2, -1, i});
51         q.push_back({x2, y2, 1, i});
52     }
53     sort(q.begin(), q.end());
54     sort(yy.begin(), yy.end());
55     yy.erase(unique(yy.begin(), yy.end()), yy.end());
56     int cur = 0;
57     for (auto [x, y, c, id] : q){
58         y = lower_bound(yy.begin(), yy.end(), y) - yy.begin() + 1;
59         while (cur < n){
60             auto [_x, _y, p] = pos[cur];
61             if (_x > x) break;
62             _y = lower_bound(yy.begin(), yy.end(), _y) - yy.begin() + 1;
63             Tr.add(_y, p), ++cur;
64         }
65         ans[id] += c * Tr.query(y);
66     }
67     for (int i = 1; i <= m; ++i) cout << ans[i] << endl;
68 }
69
70 int main(){
71     ios;
72     //freopen("sample.txt", "r", stdin);
73     //freopen("resout.txt", "w", stdout);
74     int t = 1;
75     //cin >> t;
76     while(t--){
77         solve();
78     }
79     return 0;
80 }

```

## 可持久化线段树

- HH 的项链

求区间内不重复的数的个数。扫描数列建立可持久化线段树，第  $i$  个数若第一次出现，则在线段树中的位置  $i$  加 1；若不是第一次出现，将上次出现的位置减 1，在本次位置加 1。对于每个询问的区间  $[L,R]$ ，在第  $R$  个版本上的线段树只有前  $R$  个数，在线段树上查询位置  $L$ ，对经过的区间中的和进行累计即可。

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second
9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll maxn = 1e6 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17
18 struct node{
19     int ls, rs;
20     int cnt;
21 }tr[maxn << 5];
22 int idx = 0, rt[maxn];
23
24 void push_up(int u){
25     tr[u].cnt = tr[tr[u].ls].cnt + tr[tr[u].rs].cnt;
26 }
27
28 int build(int l, int r){
29     int u = idx++;
30     if (l == r){
31         tr[u].cnt = 0;
32         return u;
33     }
34     int mid = l + r >> 1;
35     tr[u].ls = build(l, mid);
36     tr[u].rs = build(mid + 1, r);
37     push_up(u);
38     return u;
39 }
40
41 int update(int old, int l, int r, int pos, int val){
42     int u = idx++;
43     tr[u] = tr[old];
44     if (l == pos && r == pos){
45         tr[u].cnt += val;
46         return u;
47     }
48     int mid = l + r >> 1;
49     if (pos <= mid) tr[u].ls = update(tr[old].ls, l, mid, pos, val);
50     else tr[u].rs = update(tr[old].rs, mid + 1, r, pos, val);
51     push_up(u);
52     return u;
53 }
54
55 int query(int l, int r, int ver, int pos){
56     if (l == r) return tr[ver].cnt;
57     int mid = l + r >> 1;
58     if (pos <= mid) return tr[tr[ver].rs].cnt + query(l, mid, tr[ver].ls, pos);
59     else return query(mid + 1, r, tr[ver].rs, pos);
60 }
61
62 int b[maxn];
63 map<int, int> mp;
64 void solve(){
65     int n, m;
66     cin >> n;
67     rt[0] = build(1, n);

```



```

68     for (int i = 1; i <= n; ++i) cin >> b[i];
69     for (int i = 1; i <= n; ++i){
70         if (mp.find(b[i]) == mp.end()){
71             mp[b[i]] = i;
72             rt[i] = update(rt[i - 1], 1, n, i, 1);
73         }else {
74             int tmp = update(rt[i - 1], 1, n, mp[b[i]], -1);
75             rt[i] = update(tmp, 1, n, i, 1);
76         }
77         mp[b[i]] = i;
78     }
79     cin >> m;
80     for (int i = 1; i <= m; ++i){
81         int l, r;
82         cin >> l >> r;
83         cout << query(1, n, rt[r], l) << endl;
84     }
85 }
86
87 int main(){
88     ios;
89     //freopen("sample.txt", "r", stdin);
90     //freopen("resout.txt", "w", stdout);
91     int t = 1;
92     //cin >> t;
93     while(t--){
94         solve();
95     }
96     return 0;
97 }

```

## 区间问题

### 莫队

- 区间取两个数相同概率

```

1  #include <algorithm>
2  #include <cmath>
3  #include <cstdio>
4  using namespace std;
5  const int N = 50005;
6  int n, m, maxn;
7  int c[N];
8  long long sum;
9  int cnt[N];
10 long long ans1[N], ans2[N];
11
12 struct query {
13     int l, r, id;
14
15     bool operator<(const query &x) const { // 重载 < 运算符
16         if (l / maxn != x.l / maxn) return l < x.l;
17         return (l / maxn) & 1 ? r < x.r : r > x.r;
18     }
19 } a[N];
20
21 void add(int i) {
22     sum += cnt[i];
23     cnt[i]++;
24 }
25
26 void del(int i) {
27     cnt[i]--;
28     sum -= cnt[i];
29 }
30
31 long long gcd(long long a, long long b) { return b ? gcd(b, a % b) : a; }
32
33 int main() {

```

```

34 scanf("%d%d", &n, &m);
35 maxn = sqrt(n);
36 for (int i = 1; i <= n; i++) scanf("%d", &c[i]);
37 for (int i = 0; i < m; i++) scanf("%d%d", &a[i].l, &a[i].r), a[i].id = i;
38 sort(a, a + m);
39 for (int i = 0, l = 1, r = 0; i < m; i++) { // 具体实现
40     if (a[i].l == a[i].r) {
41         ans1[a[i].id] = 0, ans2[a[i].id] = 1;
42         continue;
43     }
44     while (l > a[i].l) add(c[--l]);
45     while (r < a[i].r) add(c[++r]);
46     while (l < a[i].l) del(c[l++]);
47     while (r > a[i].r) del(c[r--]);
48     ans1[a[i].id] = sum;
49     ans2[a[i].id] = (long long)(r - l + 1) * (r - l) / 2;
50 }
51 for (int i = 0; i < m; i++) {
52     if (ans1[i] != 0) {
53         long long g = gcd(ans1[i], ans2[i]);
54         ans1[i] /= g, ans2[i] /= g;
55     } else
56         ans2[i] = 1;
57     printf("%lld/%lld\n", ans1[i], ans2[i]);
58 }
59 return 0;
60 }

```

## CDQ

- 逆序对

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second
9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll maxn = 2e5 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17
18 void solve(){
19     int n;
20     cin >> n;
21     vi a(n + 1), temp(n + 1);
22     ll ans = 0;
23     rep(i, 1, n) cin >> a[i];
24     function<void(int, int)> cdq = [&](int l, int r){
25         if (l == r) return;
26         int mid = l + r >> 1;
27         cdq(l, mid);
28         cdq(mid + 1, r);
29         int p1 = l, p2 = mid + 1, idx = l;
30         while (p1 <= mid && p2 <= r){
31             if (a[p1] > a[p2]) temp[idx++] = a[p1++];
32             else temp[idx++] = a[p2++], ans += p1 - l;
33         }
34         while (p1 <= mid) temp[idx++] = a[p1++];
35         while (p2 <= r) temp[idx++] = a[p2++], ans += p1 - l;
36         for (int i = l; i <= r; ++i) a[i] = temp[i];
37     };
38     cdq(1, n);
39     cout << ans << endl;

```

```

40 }
41
42 int main(){
43     ios;
44     //freopen("sample.txt", "r", stdin);
45     //freopen("resout.txt", "w", stdout);
46     int t = 1;
47     //cin >> t;
48     while(t--){
49         solve();
50     }
51     return 0;
52 }

```

- 求最长不上升子序列和最长上升子序列

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int MAXN = 100005;
4  int n, x, dp[MAXN], a[MAXN], ans;
5  pair<int, int> temp[MAXN][20]; //val, pos
6
7  bool cmp(const pair<int, int> &A, const pair<int, int> &B, const int &type) {
8      return type ? A.first != B.first ? A.first > B.first : A.second < B.second : A.first != B.first ? A.first <
9      ↪ B.first: A.second > B.second;
10 }
11
12 void mergeSort(int l, int r, int deep, const int &cmptype) {
13     if (l == r) {
14         temp[l][deep].first = a[l];
15         temp[l][deep].second = l;
16         return;
17     }
18     int mid = (l + r) >> 1;
19     mergeSort(l, mid, deep + 1, cmptype);
20     mergeSort(mid + 1, r, deep + 1, cmptype);
21     int p1 = l, p2 = mid + 1;
22     while (p1 <= mid && p2 <= r) {
23         if (cmp(temp[p1][deep + 1], temp[p2][deep + 1], cmptype)) {
24             temp[l++][deep] = temp[p1++][deep + 1];
25         } else {
26             temp[l++][deep] = temp[p2++][deep + 1];
27         }
28     }
29     while (p1 <= mid) {
30         temp[l++][deep] = temp[p1++][deep + 1];
31     }
32     while (p2 <= r) {
33         temp[l++][deep] = temp[p2++][deep + 1];
34     }
35 }
36
37 void cdqDivAlgorithm(int l, int r, int deep, const int &cmptype) {
38     if (l == r) {
39         dp[l] = max(dp[l], 1);
40         ans = max(ans, dp[l]);
41         return;
42     }
43     int mid = (l + r) >> 1;
44     cdqDivAlgorithm(l, mid, deep + 1, cmptype);
45     int p1 = l, p2 = mid + 1, premax = 0;
46     while (p1 <= mid && p2 <= r) {
47         if (cmp(temp[p1][deep + 1], temp[p2][deep + 1], cmptype)) {
48             premax = max(premax, dp[temp[p1++][deep + 1].second]);
49         } else {
50             dp[temp[p2][deep + 1].second] = max(premax + 1, dp[temp[p2][deep + 1].second]);
51             p2++;
52         }
53     }
54     while (p2 <= r) {
55         dp[temp[p2][deep + 1].second] = max(premax + 1, dp[temp[p2][deep + 1].second]);
56         p2++;
57     }
58 }

```

```

56     }
57     cdqDivAlgorithm(mid + 1, r, deep + 1, cmptype);
58 }
59
60 int main()
61 {
62     while (scanf("%d", &x) != EOF) a[++n] = x;
63     mergeSort(1, n, 0, 1);
64     cdqDivAlgorithm(1, n, 0, 1);
65     printf("%d\n", ans);
66     memset(dp, 0, sizeof(dp));
67     ans = 0;
68     mergeSort(1, n, 0, 0);
69     cdqDivAlgorithm(1, n, 0, 0);
70     printf("%d\n", ans);
71     return 0;
72 }

```

- 求地毯覆盖（最多取多少个不相互覆盖）

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int MAXN = 1000005;
4  int n, L[MAXN], R[MAXN], id[MAXN], dp[MAXN], ans;
5  int temp[MAXN];
6  void cdqDivAlgorithm(int l, int r) {
7      if (l == r) {
8          dp[id[l]] = max(1, dp[id[l]]);
9          ans = max(ans, dp[id[l]]);
10         return;
11     }
12     int mid = (l + r) >> 1;
13     cdqDivAlgorithm(l, mid);
14     int p1 = l, pl = l, p2 = mid + 1, premax = 0;
15     while (p1 <= mid && p2 <= r) {
16         if (R[id[p1]] <= L[id[p2]]) {
17             premax = max(premax, dp[id[p1++]]);
18         } else {
19             dp[id[p2]] = max(premax + 1, dp[id[p2]]);
20             ++p2;
21         }
22     }
23     while (p2 <= r) {
24         dp[id[p2]] = max(premax + 1, dp[id[p2]]);
25         ++p2;
26     }
27     cdqDivAlgorithm(mid + 1, r);
28     p1 = l, pl = l, p2 = mid + 1;
29     while (p1 <= mid && p2 <= r) {
30         if (R[id[p1]] < R[id[p2]]) {
31             temp[pl++] = id[p1++];
32         } else {
33             temp[pl++] = id[p2++];
34         }
35     }
36     while (p1 <= mid) {
37         temp[pl++] = id[p1++];
38     }
39     while (p2 <= r) {
40         temp[pl++] = id[p2++];
41     }
42     for (int i = l; i <= r; ++i) {
43         id[i] = temp[i];
44     }
45 }
46 int main()
47 {
48     scanf("%d", &n);
49     for (int i = 1; i <= n; ++i) {
50         scanf("%d %d", &L[i], &R[i]);
51         id[i] = i;
52     }

```

```

53     sort(id + 1, id + 1 + n, [](const int &A, const int &B) {
54         return L[A] < L[B];
55     });
56     cdqDivAlgorithm(1, n);
57     printf("%d\n", ans);
58     return 0;
59 }

```

### ● 动态凸包

第一行：一个整数  $N$ ，表示方案和询问的总数。接下来  $N$  行，每行开头一个单词 “Query” 或 “Project”。若单词为 Query，则后接一个整数  $T$ ，表示 Blue Mary 询问第  $T$  天的最大收益。若单词为 Project，则后接两个实数  $S, P$ ，表示该种设计方案第一天的收益  $S$ ，以及以后每天比上一天多出的收益  $P$ 。对于每一个 Query，输出一个整数，表示询问的答案，并精确到整百元  $1 \leq N \leq 100000$   $1 \leq T \leq 50000$   $0 < P < 100$ ,  $|S| \leq 10^6$

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  const int MAXN = 100005;
4  const double eps = 1e-6;
5  int m, n, id[MAXN], qid[MAXN], type[MAXN], x[MAXN], temp[MAXN], top;
6  double k[MAXN], b[MAXN], ans[MAXN];
7  char op[55];
8  inline bool cmp(const int &A, const int &B) {
9      return type[A] != type[B] ? type[A] < type[B] : type[A] ? x[A] < x[B] : k[A] < k[B];
10 }
11 inline int dcmp(double x) {
12     return x > eps ? 1 : x < -eps ? -1 : 0;
13 }
14 inline double getCross(const double &k1, const double &b1, const double &k2, const double &b2) {
15     return (b2 - b1) / (k1 - k2);
16 }
17 inline double getVal(const double &k, const double &b, const int &x)
18 {
19     return k * x + b;
20 }
21 pair<double, double> stk[MAXN];
22 void stkClear() {
23     top = 0;
24     stk[++top] = make_pair(0, 0);
25 }
26 void stkInsert(double k, double b) {
27     if (dcmp(stk[top].first - k) == 0 && dcmp(stk[top].second - b) < 0) top--;
28     if (dcmp(stk[top].first - k) == 0 && dcmp(stk[top].second - b) >= 0) return;
29     while (top >= 2 && dcmp(getCross(stk[top].first, stk[top].second, stk[top - 1].first, stk[top - 1].second) -
30 ↪ getCross(stk[top].first, stk[top].second, k, b)) > 0) top--;
31     stk[++top] = make_pair(k, b);
32 }
33 double stkQuery(int x) {
34     while (top >= 2 && dcmp(getVal(stk[top].first, stk[top].second, x) - getVal(stk[top - 1].first, stk[top -
35 ↪ 1].second, x)) < 0) --top;
36     return getVal(stk[top].first, stk[top].second, x);
37 }
38 void cdqDivAlgorithm(int l, int r) {
39     if (l == r) return;
40     int mid = (l + r) >> 1;
41     cdqDivAlgorithm(l, mid);
42     cdqDivAlgorithm(mid + 1, r);
43     stkClear();
44     for (int i = l; i <= mid && !type[id[i]]; ++i) {
45         stkInsert(k[id[i]], b[id[i]]);
46     }
47     for (int i = r; i > mid && type[id[i]]; --i) {
48         ans[qid[id[i]]] = max(ans[qid[id[i]]], stkQuery(x[id[i]]));
49     }
50     int p1 = l, p2 = l, p3 = mid + 1;
51     while (p1 <= mid && p2 <= r) {
52         if (cmp(id[p1], id[p2])) {
53             temp[p1++] = id[p1++];
54         } else {
55             temp[p1++] = id[p2++];
56         }
57     }
58     for (int i = l; i <= r; ++i) id[i] = temp[i];
59 }

```

```

55     }
56     while (p1 <= mid) {
57         temp[p1++] = id[p1++];
58     }
59     while (p2 <= r) {
60         temp[p1++] = id[p2++];
61     }
62     for (int i = l; i <= r; ++i) {
63         id[i] = temp[i];
64     }
65 }
66 int main() {
67     scanf("%d", &n);
68     for (int i = 1; i <= n; ++i) {
69         id[i] = i;
70         scanf("%s", op);
71         if (*op == 'P') {
72             type[i] = 0;
73             scanf("%lf %lf", &b[i], &k[i]);
74             b[i] -= k[i];
75         }
76         else {
77             type[i] = 1;
78             qid[i] = ++m;
79             scanf("%d", &x[i]);
80         }
81     }
82     cdqDivAlgorithm(1, n);
83     for (int i = 1; i <= m; ++i) {
84         printf("%d\n", (int)ans[i] / 100);
85     }
86     return 0;
87 }
88

```

## 树上问题

### 树剖

- 2018ICPC 青岛网络赛（多测时候用来剖的）

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second
9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll maxn = 1e5 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17
18 void solve()
19 {
20     int n, m, q, k, cnt = 0;
21     cin >> n >> m >> q;
22     vi red(n + 1);
23     vector<vector<pll>> G(n + 1);
24     vl dis(n + 1), dep(n + 1), v(n + 1);
25     vi dfn(n + 1), idx(n + 1);
26     vi son(n + 1, -1), sz(n + 1), fa(n + 1), top(n + 1);
27     function<void(int, int)> dfs1 = [&](int u, int f) {
28         son[u] = -1;
29         sz[u] = 1;

```

```

30     if(!red[u])
31         red[u] = red[f];
32     for(auto [v, w] : G[u]) {
33         if(v == f)
34             continue;
35         dep[v] = dep[u] + 1;
36         dis[v] = dis[u] + w;
37         fa[v] = u;
38         dfs1(v, u);
39         sz[u] += sz[v];
40         if(son[u] == -1 || sz[v] > sz[son[u]])
41             son[u] = v;
42     }
43 };
44 function<void(int, int)> dfs2 = [&](int u, int t) {
45     top[u] = t;
46     dfn[u] = ++cnt;
47     idx[cnt] = u;
48     if(son[u] == -1)
49         return;
50     dfs2(son[u], t);
51     for(auto [v, w] : G[u])
52         if(v != son[u] && v != fa[u])
53             dfs2(v, v);
54 };
55 auto lca = [&](int u, int v) {
56     while(top[u] != top[v]) {
57         if(dep[top[u]] > dep[top[v]])
58             u = fa[top[u]];
59         else
60             v = fa[top[v]];
61     }
62     return dep[u] > dep[v] ? v : u;
63 };
64 for(int i = 1; x; i <= m; ++i)
65     cin >> x, red[x] = x;
66 for(int i = 1; i < n; ++i) {
67     int u, v, w;
68     cin >> u >> v >> w;
69     G[u].push_back({v, w});
70     G[v].push_back({u, w});
71 }
72 dfs1(1, 0);
73 dfs2(1, 1);
74 for(int i = 1; i <= n; ++i)
75     v[i] = dis[i] - dis[red[i]];
76 while(q--) {
77     cin >> k;
78     vector<int> p(k + 1);
79     auto check = [&](ll st) {
80         vector<int> q;
81         for(int i = 1; i <= k; ++i)
82             if(v[p[i]] > st)
83                 q.push_back(p[i]);
84         if(q.size() == 0)
85             return true;
86         int mnd = n + 1, mxd = 0;
87         for(int i = 0; i < q.size(); ++i) {
88             mnd = min(mnd, dfn[q[i]]);
89             mxd = max(mxd, dfn[q[i]]);
90         }
91         int ca = lca(idx[mnd], idx[mxd]);
92         for(int i = 0; i < q.size(); ++i)
93             if(dis[q[i]] - dis[ca] > st)
94                 return false;
95         return true;
96     };
97     ll mx = 0;
98     for(int i = 1; i <= k; ++i) {
99         cin >> p[i];
100         mx = max(mx, v[p[i]]);

```

```

101     }
102     ll l = 0, r = mx;
103     while(l < r) {
104         ll mid = (l + r) >> 1;
105         if(check(mid))
106             r = mid;
107         else
108             l = mid + 1;
109     }
110     cout << l << endl;
111 }
112 }
113
114 int main()
115 {
116     ios;
117     // freopen("sample.txt", "r", stdin);
118     // freopen("resout.txt", "w", stdout);
119     int t = 1;
120     cin >> t;
121     while(t--) {
122         solve();
123     }
124     return 0;
125 }
126

```

#### ● 树上操作

1. 节点  $x$  加上  $a$
2. 节点  $x$  的子树中所有点的点权加  $a$
3. 询问某个点  $x$  到根节点

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second
9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll maxn = 2e5 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17 const int N = 1e5 + 10, M = N * 2;
18
19 int n, m;
20 // w 为节点权值
21 int h[N], w[N], e[M], ne[M], idx;
22 // id[x] 为节点 x 的新编号, nw[x] 是新编号为 x 的节点的权值
23 int id[N], nw[N], cnt;
24 // dep 为深度, sz 为子树大小, top[x] 是 x 所在重链的头结点,
25 // fa[x] 为 x 父亲, son[x] 为 x 的重儿子
26 int dep[N], sz[N], top[N], fa[N], son[N];
27 struct Tree {
28     int l, r;
29     ll sum, add;
30 } tr[N << 2];
31
32 void add(int a, int b) {
33     e[idx] = b, ne[idx] = h[a], h[a] = idx++;
34 }
35
36 // 第一次 dfs, 求节点深度、父亲、子树大小和重儿子
37 void dfs1(int u, int from, int depth) {
38     dep[u] = depth, fa[u] = from, sz[u] = 1;

```



```

39     for (int i = h[u]; ~i; i = ne[i]) {
40         int v = e[i];
41         if (v == from) continue;
42         dfs1(v, u, depth + 1);
43         sz[u] += sz[v];
44         if (sz[son[u]] < sz[v]) son[u] = v;
45     }
46 }
47
48 // 第二次 dfs, t 为 u 重链头结点
49 void dfs2(int u, int t) {
50     id[u] = ++cnt, nw[cnt] = w[u], top[u] = t;
51     // 到叶子了, 直接返回
52     if (!son[u]) return;
53     // 先遍历重儿子
54     dfs2(son[u], t);
55     // 遍历轻儿子
56     for (int i = h[u]; ~i; i = ne[i]) {
57         int v = e[i];
58         if (v == fa[u] || v == son[u]) continue;
59         dfs2(v, v);
60     }
61 }
62
63 void pushup(int u) {
64     tr[u].sum = tr[u << 1].sum + tr[u << 1 | 1].sum;
65 }
66
67 void pushdown(int u) {
68     auto &root = tr[u], &left = tr[u << 1], &right = tr[u << 1 | 1];
69     if (root.add) {
70         left.sum += root.add * (left.r - left.l + 1);
71         left.add += root.add;
72         right.sum += root.add * (right.r - right.l + 1);
73         right.add += root.add;
74         root.add = 0;
75     }
76 }
77
78 void build(int u, int l, int r) {
79     tr[u] = {l, r, nw[l], 0};
80     if (l == r) return;
81     int mid = l + r >> 1;
82     build(u << 1, l, mid), build(u << 1 | 1, mid + 1, r);
83     pushup(u);
84 }
85
86 void update(int u, int l, int r, ll k) {
87     if (l <= tr[u].l && tr[u].r <= r) {
88         tr[u].add += k;
89         tr[u].sum += k * (tr[u].r - tr[u].l + 1);
90         return;
91     }
92     pushdown(u);
93     int mid = tr[u].l + tr[u].r >> 1;
94     if (l <= mid) update(u << 1, l, r, k);
95     if (r > mid) update(u << 1 | 1, l, r, k);
96     pushup(u);
97 }
98
99 ll query(int u, int l, int r) {
100     if (l <= tr[u].l && tr[u].r <= r) return tr[u].sum;
101     pushdown(u);
102     int mid = tr[u].l + tr[u].r >> 1;
103     ll res = 0;
104     if (l <= mid) res += query(u << 1, l, r);
105     if (r > mid) res += query(u << 1 | 1, l, r);
106     return res;
107 }
108
109 void update_path(int u, int v, ll k) {

```

```

110 while (top[u] != top[v]) {
111     if (dep[top[u]] < dep[top[v]]) swap(u, v);
112     // u 的重链头更深, 并且 u 重链头在 dfs 序里下标更小, 直接更新 u 重链头到 u 这段区间
113     update(1, id[top[u]], id[u], k);
114     // u 跳到重链头上面
115     u = fa[top[u]];
116 }
117 if (dep[u] < dep[v]) swap(u, v);
118 update(1, id[v], id[u], k);
119 }
120
121 ll query_path(int u, int v) {
122     ll res = 0;
123     while (top[u] != top[v]) {
124         if (dep[top[u]] < dep[top[v]]) swap(u, v);
125         res += query(1, id[top[u]], id[u]);
126         u = fa[top[u]];
127     }
128     if (dep[u] < dep[v]) swap(u, v);
129     res += query(1, id[v], id[u]);
130     return res;
131 }
132
133 void update_tree(int u, ll k) {
134     update(1, id[u], id[u] + sz[u] - 1, k);
135 }
136
137 ll query_tree(int u) {
138     return query(1, id[u], id[u] + sz[u] - 1);
139 }
140
141 void solve() {
142     int n, q;
143     memset(h, -1, sizeof h);
144     cin >> n >> q;
145     int cnt = 0;
146     for (int i = 1; i <= n; ++i) cin >> w[i];
147     for (int i = 1; i <= n - 1; ++i) {
148         int u, v;
149         cin >> u >> v;
150         add(u, v);
151         add(v, u);
152     }
153     dfs1(1, 1, 0);
154     dfs2(1, 1);
155     build(1, 1, n);
156     while (q--) {
157         int t, u;
158         ll k;
159         cin >> t >> u;
160         if (t == 1) {
161             cin >> k;
162             update_path(u, u, k);
163         } else if (t == 2) {
164             cin >> k;
165             update_tree(u, k);
166         } else cout << query_path(1, u) << endl;
167     }
168 }
169
170 int main() {
171     ios;
172     //freopen("sample.txt", "r", stdin);
173     //freopen("resout.txt", "w", stdout);
174     int t = 1;
175     //cin >> t;
176     while (t--) {
177         solve();
178     }
179     return 0;
180 }

```

- 树上路径

1. 将以  $u$  为根的子树内节点 (包括  $u$ ) 的权值加  $val$
2. 将  $(u, v)$  路径上的节点权值加  $val$
3. 询问  $(u, v)$  路径上节点的权值两两相乘的和

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define int ll
8  #define x first
9  #define y second
10 #define rep(i, j, k) for(int i = (j); i <= (k); i++)
11 #define per(i, j, k) for(int i = (j); i >= (k); i--)
12 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
13 using namespace std;
14 typedef long long ll;
15 const ll mod = 1e9 + 7;
16 const ll inf = 0x3f3f3f3f;
17 const int N = 1e5 + 10, M = N * 2;
18
19 int n, m;
20 int h[N], a[N], e[M], ne[M], idx;
21 int id[N], cnt, rnk[N];
22 int dep[N], sz[N], top[N], fa[N], son[N];
23 ll inv2;
24
25 void add(int u, int v) {
26     e[idx] = v, ne[idx] = h[u], h[u] = idx++;
27 }
28 ll qmi(ll x, ll k) {
29     ll res = 1;
30     while (k) {
31         if (k & 1) res = res * x % mod;
32         x = x * x % mod;
33         k >>= 1;
34     }
35     return res;
36 }
37 struct Segment {
38     struct Node {
39         int l, r;
40         ll sum, psum, add;
41     } tr[N * 4];
42     void pushup(int u) {
43         tr[u].sum = (tr[u << 1].sum + tr[u << 1 | 1].sum) % mod;
44         tr[u].psum = (tr[u << 1].psum + tr[u << 1 | 1].psum) % mod;
45         return;
46     }
47     void pushdown(Node& u, Node& l, Node& r) {
48         if (u.add) {
49             ll x = u.add;
50             l.psum = (l.psum + 2 * l.sum * x % mod + (ll)x * x % mod * (l.r - l.l + 1) % mod) % mod;
51             r.psum = (r.psum + 2 * r.sum * x % mod + (ll)x * x % mod * (r.r - r.l + 1) % mod) % mod;
52             l.sum = (l.sum + (ll)x * (l.r - l.l + 1) % mod) % mod;
53             r.sum = (r.sum + (ll)x * (r.r - r.l + 1) % mod) % mod;
54             l.add = (l.add + x) % mod;
55             r.add = (r.add + x) % mod;
56             u.add = 0;
57         }
58         return;
59     }
60     void pushdown(int u) {
61         pushdown(tr[u], tr[u << 1], tr[u << 1 | 1]);
62     }
63     void build(int u, int l, int r) {
64         tr[u] = {l, r};
65         if (l == r) {

```

```

66         tr[u].sum = a[rnk[l]];
67         tr[u].psum = (ll)a[rnk[l]] * a[rnk[l]] % mod;
68         return;
69     }
70     int mid = (l + r) >> 1;
71     build(u << 1, l, mid);
72     build(u << 1 | 1, mid + 1, r);
73     pushup(u);
74     return;
75 }
76 void update(int u, int l, int r, ll x) {
77     if (l <= tr[u].l && tr[u].r <= r) {
78         tr[u].psum = (tr[u].psum + 2 * tr[u].sum * x % mod + (ll)x * x % mod * (tr[u].r - tr[u].l + 1) % mod) % mod;
79         tr[u].sum = (tr[u].sum + (ll)(tr[u].r - tr[u].l + 1) * x % mod) % mod;
80         tr[u].add = (tr[u].add + x) % mod;
81         return;
82     }
83     pushdown(u);
84     int mid = (tr[u].l + tr[u].r) >> 1;
85     if (l <= mid) update(u << 1, l, r, x);
86     if (mid < r) update(u << 1 | 1, l, r, x);
87     pushup(u);
88     return;
89 }
90 ll query_sum(int u, int l, int r) {
91     if (l <= tr[u].l && tr[u].r <= r) return tr[u].sum;
92     pushdown(u);
93     int mid = (tr[u].l + tr[u].r) >> 1;
94     ll res = 0;
95     if (l <= mid) res = (res + query_sum(u << 1, l, r)) % mod;
96     if (mid < r) res = (res + query_sum(u << 1 | 1, l, r)) % mod;
97     return res;
98 }
99 ll query_psum(int u, int l, int r) {
100     if (l <= tr[u].l && tr[u].r <= r) return tr[u].psum;
101     pushdown(u);
102     int mid = (tr[u].l + tr[u].r) >> 1;
103     ll res = 0;
104     if (l <= mid) res = (res + query_psum(u << 1, l, r)) % mod;
105     if (mid < r) res = (res + query_psum(u << 1 | 1, l, r)) % mod;
106     return res;
107 }
108 } Tr;
109
110 //Tree
111 void dfs1(int u, int from, int depth) {
112     dep[u] = depth, fa[u] = from, sz[u] = 1;
113     for (int i = h[u]; ~i; i = ne[i]) {
114         int v = e[i];
115         if (v == from) continue;
116         dfs1(v, u, depth + 1);
117         sz[u] += sz[v];
118         if (sz[son[u]] < sz[v]) son[u] = v;
119     }
120 }
121 void dfs2(int u, int t) {
122     id[u] = ++cnt, top[u] = t;
123     rnk[cnt] = u;
124     if (!son[u]) return;
125     dfs2(son[u], t);
126     for (int i = h[u]; ~i; i = ne[i]) {
127         int v = e[i];
128         if (v == fa[u] || v == son[u]) continue;
129         dfs2(v, v);
130     }
131 }
132 void update_path(int u, int v, ll k) { //更新路径
133     while (top[u] != top[v]) {
134         if (dep[top[u]] < dep[top[v]]) swap(u, v);
135         Tr.update(1, id[top[u]], id[u], k);
136         u = fa[top[u]];

```

```

137     }
138     if (dep[u] < dep[v]) swap(u, v);
139     Tr.update(1, id[v], id[u], k);
140 }
141 ll query_path(int u, int v) {
142     ll res_sum = 0, res_psum = 0;
143     while (top[u] != top[v]) {
144         if (dep[top[u]] < dep[top[v]]) swap(u, v);
145         res_sum = (res_sum + Tr.query_sum(1, id[top[u]], id[u])) % mod;
146         res_psum = (res_psum + Tr.query_psum(1, id[top[u]], id[u])) % mod;
147         u = fa[top[u]];
148     }
149     if (dep[u] < dep[v]) swap(u, v);
150     res_sum = (res_sum + Tr.query_sum(1, id[v], id[u])) % mod;
151     res_psum = (res_psum + Tr.query_psum(1, id[v], id[u])) % mod;
152     return (res_sum * res_sum % mod - res_psum + mod) % mod * inv2 % mod;
153 }
154 //Tree
155
156 void solve() {
157     inv2 = qmi(2, mod - 2);
158     cin >> n >> m;
159     for (int i = 1; i <= n; ++i) cin >> a[i];
160     memset(h, -1, sizeof h);
161     for (int i = 1; i <= n - 1; ++i) {
162         int u, v;
163         cin >> u >> v;
164         add(u, v);
165         add(v, u);
166     }
167     dfs1(1, 0, 1);
168     dfs2(1, 1);
169     Tr.build(1, 1, n);
170
171     while (m--) {
172         int op;
173         cin >> op;
174         ll u, v, k;
175         if (op == 1) {
176             cin >> u >> k;
177             Tr.update(1, id[u], id[u] + sz[u] - 1, k);
178         } else if (op == 2) {
179             cin >> u >> v >> k;
180             update_path(u, v, k);
181         } else {
182             cin >> u >> v;
183             cout << query_path(u, v) << endl;
184         }
185     }
186 }
187
188 signed main() {
189     ios;
190     //freopen("sample.txt", "r", stdin);
191     //freopen("resout.txt", "w", stdout);
192     int t = 1;
193     //cin >> t;
194     while (t--) {
195         solve();
196     }
197     return 0;
198 }

```

## dsu

- 树上数颜色

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  const int N = 2e5 + 5;

```

```

5
6  int n;
7
8  // g[u]: 存储与 u 相邻的结点
9  vector<int> g[N];
10
11 // sz: 子树大小
12 // big: 重儿子
13 // col: 结点颜色
14 // L[u]: 结点 u 的 DFS 序
15 // R[u]: 结点 u 子树中结点的 DFS 序的最大值
16 // Node[i]: DFS 序为 i 的结点
17 // ans: 存答案
18 // cnt[i]: 颜色为 i 的结点个数
19 // totColor: 目前出现过的颜色个数
20 int sz[N], big[N], col[N], L[N], R[N], Node[N], totdfn;
21 int ans[N], cnt[N], totColor;
22
23 void add(int u) {
24     if (cnt[col[u]] == 0) ++totColor;
25     cnt[col[u]]++;
26 }
27
28 void del(int u) {
29     cnt[col[u]]--;
30     if (cnt[col[u]] == 0) --totColor;
31 }
32
33 int getAns() { return totColor; }
34
35 void dfs0(int u, int p) {
36     L[u] = ++totdfn;
37     Node[totdfn] = u;
38     sz[u] = 1;
39     for (int v : g[u])
40         if (v != p) {
41             dfs0(v, u);
42             sz[u] += sz[v];
43             if (!big[u] || sz[big[u]] < sz[v]) big[u] = v;
44         }
45     R[u] = totdfn;
46 }
47
48 void dfs1(int u, int p, bool keep) {
49     // 计算轻儿子的答案
50     for (int v : g[u])
51         if (v != p && v != big[u]) {
52             dfs1(v, u, false);
53         }
54     // 计算重儿子答案并保留计算过程中的数据 (用于继承)
55     if (big[u]) {
56         dfs1(big[u], u, true);
57     }
58     for (int v : g[u])
59         if (v != p && v != big[u]) {
60             // 子树结点的 DFS 序构成一段连续区间, 可以直接遍历
61             for (int i = L[v]; i <= R[v]; i++) {
62                 add(Node[i]);
63             }
64         }
65     add(u);
66     ans[u] = getAns();
67     if (keep == false) {
68         for (int i = L[u]; i <= R[u]; i++) {
69             del(Node[i]);
70         }
71     }
72 }
73
74 int main() {
75     scanf("%d", &n);

```

```

76     for (int i = 1; i <= n; i++) scanf("%d", &col[i]);
77     for (int i = 1; i < n; i++) {
78         int u, v;
79         scanf("%d%d", &u, &v);
80         g[u].push_back(v);
81         g[v].push_back(u);
82     }
83     dfs0(1, 0);
84     dfs1(1, 0, false);
85     for (int i = 1; i <= n; i++) printf("%d%c", ans[i], " \n"[i == n]);
86     return 0;
87 }

```

- 子树权值不大于 k 的数量

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second
9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll N = 1e6 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17 int tr[N];
18 int h[N], to[2 * N], ne[2 * N], cnt;
19 int sz[N], dep[N], fa[N], son[N];
20 int top[N], dfn, L[N], R[N], idx[N], skip;
21 int a[N], sum, ans[N];
22 int n, m;
23
24 vector<pll> q[N];
25 int lowbit(int x) {return x & -x;}
26 void add(int p, int k) {for (int i = p; i < N; i += lowbit(i)) tr[i] += k;}
27 int query(int p) {int res = 0; for (int i = p; i; i -= lowbit(i)) res += tr[i]; return res;}
28
29 void addedge(int u, int v) {
30     to[++cnt] = v;
31     ne[cnt] = h[u];
32     h[u] = cnt;
33 }
34
35 void dfs1(int u, int f) {
36     sz[u] = 1;
37     dep[u] = dep[f] + 1;
38     fa[u] = f;
39     for (int i = h[u]; i; i = ne[i]) {
40         int v = to[i];
41         if (v == f) continue;
42         dfs1(v, u);
43         sz[u] += sz[v];
44         if (!son[u] || sz[son[u]] < sz[v]) son[u] = v;
45     }
46 }
47
48 void dfs2(int u, int t) {
49     L[u] = ++dfn;
50     idx[dfn] = u;
51     top[u] = t;
52     if (son[u]) dfs2(son[u], t);
53     for (int i = h[u]; i; i = ne[i]) {
54         int v = to[i];
55         if (v != fa[u] && v != son[u])
56             dfs2(v, v);
57     }

```

```

58     R[u] = dfn;
59 }
60
61 void get(int u, int op){
62     for (int i = L[u]; i <= R[u]; ++i){
63         if (idx[i] == skp){i = R[idx[i]]; continue;}
64         add(a[idx[i]], op);
65     }
66     if (op == -1) return;
67     for (auto x : q[u]) ans[x.second] = query(x.first);
68 }
69
70 void dsu(int u){
71     for (int i = h[u]; i; i = ne[i]){
72         int v = to[i];
73         if (v == fa[u] || v == son[u]) continue;
74         dsu(v);
75     }
76     if (son[u]) {dsu(son[u]), skp = son[u];}
77     get(u, 1);
78     if (u == top[u]){
79         skp = 0;
80         get(u, -1);
81     }
82 }
83 void solve() {
84     cin >> n;
85     rep(i, 1, n) cin >> a[i];
86     int u, v;
87     for (int i = 1; i <= n - 1; ++i){
88         cin >> u >> v;
89         addedge(u, v);
90         addedge(v, u);
91     }
92     cin >> m;
93     int x, k;
94     rep(i, 1, m){
95         cin >> x >> k;
96         q[x].push_back({k, i});
97     }
98     dfs1(1, 0);
99     dfs2(1, 1);
100    dsu(1);
101    for (int i = 1; i <= m; ++i) cout << ans[i] << endl;
102 }
103
104 int main() {
105     ios;
106     //freopen("sample.txt", "r", stdin);
107     //freopen("resout.txt", "w", stdout);
108     int t = 1;
109     //cin >> t;
110     while (t--) {
111         solve();
112     }
113     return 0;
114 }

```

#### ● 子树查询类问题

现在将会问你  $m$  个问题。对于每个问题，它将会给你三个参数  $x, l, r$  表示询问以  $x$  为根的子树中，节点深度在该子树中不小于  $l$  且不大于  $r$  的所有节点。你需要告诉智乃酱三个信息，所有符合条件节点的最小值，最大值，以及它们的和。

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>
7  #define x first
8  #define y second

```



```

9  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10 #define per(i, j, k) for(int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 const ll maxn = 1e5 + 10;
15 const ll mod = 998244353;
16 const ll inf32 = 1e9;
17 const ll inf64 = 2e18;
18
19 int tot, h[maxn], len[maxn], L[maxn], R[maxn], fa[maxn], son[maxn], dfn, n, m, x, l, r, u, v;
20 ll val[maxn];
21
22 struct node {
23     ll Sum, Max, Min;
24 } ans[maxn];
25
26 struct qnode {
27     int id;
28     int l, r;
29     qnode(int _id = 0, int _l = 0, int _r = 0) {id = _id, l = _l, r = _r;}
30 };
31
32 struct edges {
33     int to, next;
34 } e[2 * maxn];
35 vector<qnode> lis[maxn];
36
37 struct tnode
38 {
39     ll Sum, Max, Min;
40     int l, r;
41 };
42 tnode operator + (const tnode &a, const tnode &b)
43 {
44     tnode c;
45     c.l = a.l;
46     c.r = b.r;
47     c.Sum = a.Sum + b.Sum;
48     c.Max = max(a.Max, b.Max);
49     c.Min = min(a.Min, b.Min);
50     return c;
51 }
52
53 struct Segment_Tree
54 {
55     tnode t[4 * maxn];
56     int mp[maxn];
57     void update (int root)
58     {
59         int ch = root << 1;
60         t[root] = t[ch] + t[ch + 1];
61     }
62     void buildt(int root, int l, int r)
63     {
64         t[root].l = l;
65         t[root].r = r;
66         if (l != r)
67         {
68             int mid = (l + r) >> 1;
69             int ch = root << 1;
70             buildt(ch, l, mid);
71             buildt(ch + 1, mid + 1, r);
72             update(root);
73         }
74         else
75         {
76             mp[l] = root;
77             t[root].Max = -inf64;
78             t[root].Min = inf64;
79             t[root].Sum = 0;

```

```

80     }
81 }
82 void change(int pos, long long delta, long long nmax, long long nmin)
83 {
84     int root = mp[pos];
85     t[root].Sum += delta;
86     t[root].Max = max(t[root].Max, nmax);
87     t[root].Min = min(t[root].Min, nmin);
88     while (root >= 1) update(root);
89 }
90 tnode getdata(int pos)
91 {
92     return t[mp[pos]];
93 }
94 tnode getseg(int root, int l, int r)
95 {
96     if (t[root].l == l && t[root].r == r)
97     {
98         return t[root];
99     }
100     int mid = (t[root].l + t[root].r) >> 1;
101     int ch = root << 1;
102     if (r <= mid) return getseg(ch, l, r);
103     else if (l > mid) return getseg(ch + 1, l, r);
104     else return getseg(ch, l, mid) + getseg(ch + 1, mid + 1, r);
105 }
106 };
107 Segment_Tree ST;
108
109 void add_edge(int u, int to)
110 {
111     e[++tot].to = to;
112     e[tot].next = h[u];
113     h[u] = tot;
114     return;
115 }
116
117
118 void dfs1(int x, int father)
119 {
120     fa[x] = father;
121     for (int i = h[x]; i; i = e[i].next)
122     {
123         if (e[i].to != father)
124         {
125             dfs1(e[i].to, x);
126             if (!son[x] || len[son[x]] < len[e[i].to]) son[x] = e[i].to;
127         }
128     }
129     len[x] = len[son[x]] + 1;
130     return;
131 }
132
133 void dfs2(int x)
134 {
135     L[x] = ++dfn;
136     R[x] = L[x] + len[x] - 1;
137     if (son[x]) dfs2(son[x]);
138     for (int i = h[x]; i; i = e[i].next)
139     {
140         if (e[i].to != fa[x] && e[i].to != son[x])
141         {
142             dfs2(e[i].to);
143         }
144     }
145     return;
146 }
147
148 void dfs(int x)
149 {
150     if (son[x])

```

```

151     {
152         dfs(son[x]);
153     }
154     for (int i = h[x]; i; i = e[i].next)
155     {
156         if (e[i].to != fa[x] && e[i].to != son[x])
157         {
158             dfs(e[i].to);
159             for (int j = L[e[i].to], k = 1; j <= R[e[i].to]; ++j, ++k)
160             {
161                 tnode temp = ST.getdata(j);
162                 ST.change(L[x] + k, temp.Sum, temp.Max, temp.Min);
163             }
164         }
165     }
166     ST.change(L[x], val[x], val[x], val[x]);
167     for (auto &i : lis[x])
168     {
169         tnode temp = ST.getseg(1, L[x] + i.l, L[x] + i.r);
170         ans[i.id].Sum = temp.Sum;
171         ans[i.id].Max = temp.Max;
172         ans[i.id].Min = temp.Min;
173     }
174     return;
175 }
176
177 void solve() {
178     cin >> n;
179     for (int i = 1; i <= n; ++i) cin >> val[i];
180     for (int i = 1; i <= n - 1; ++i) {
181         cin >> u >> v;
182         add_edge(u, v);
183         add_edge(v, u);
184     }
185     dfs1(1, 0);
186     dfs2(1);
187     ST.built(1, 1, n);
188     cin >> m;
189     for (int i = 1; i <= m; ++i) {
190         cin >> x >> l >> r;
191         lis[x].push_back(qnode(i, l, r));
192     }
193     dfs(1);
194     for (int i = 1; i <= m; ++i)
195     {
196         cout << ans[i].Min << " " << ans[i].Max << " " << ans[i].Sum << endl;
197     }
198 }
199
200 int main() {
201     ios;
202     //freopen("sample.txt", "r", stdin);
203     //freopen("resout.txt", "w", stdout);
204     int t = 1;
205     //cin >> t;
206     while (t--) {
207         solve();
208     }
209     return 0;
210 }
211

```

### ● 小Q与树

$$u \sum v \sum \min(a[u], a[v]) * \text{dis}(u, v)$$

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define vi vector<int>
6  #define vl vector<ll>

```

```

7  #define int ll
8  #define rep(i, j, k) for(int i = (j); i <= (k); i++)
9  #define per(i, j, k) for(int i = (j); i >= (k); i--)
10 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
11 using namespace std;
12 typedef long long ll;
13 const ll maxn = 2e5 + 10;
14 const ll mod = 998244353;
15 const ll inf = 0x3f3f3f3f;
16
17 int n, h[maxn], to[maxn << 1], nxt[maxn << 1], cnt = 0;
18 int sz[maxn], son[maxn], dep[maxn], L[maxn], R[maxn], f[maxn], idx[maxn], top[maxn], dfn = 0;
19 ll sum[maxn], ans = 0;
20 struct node {
21     int x, id;
22 } a[maxn];
23
24 void add(int u, int v) {
25     to[++cnt] = v;
26     nxt[cnt] = h[u];
27     h[u] = cnt;
28 }
29
30 void dfs1(int u, int fa) {
31     f[u] = fa;
32     dep[u] = dep[fa] + 1;
33     sz[u] = 1;
34     for (int i = h[u]; i; i = nxt[i]) {
35         int v = to[i];
36         if (v == fa) continue;
37         dfs1(v, u);
38         sz[u] += sz[v];
39         if (sz[v] > sz[son[u]])
40             son[u] = v;
41     }
42 }
43
44 void dfs2(int u, int t) {
45     top[u] = t;
46     L[u] = ++dfn;
47     idx[dfn] = u;
48     if (son[u]) dfs2(son[u], t);
49     for (int i = h[u]; i; i = nxt[i]) {
50         int v = to[i];
51         if (v == f[u] || v == son[u]) continue;
52         dfs2(v, v);
53     }
54     R[u] = dfn;
55 }
56
57 struct Segment {
58     struct Node {
59         int l, r;
60         int sum, add;
61     } tr[maxn * 4];
62     void pushup(int u) {
63         tr[u].sum = tr[u << 1].sum + tr[u << 1 | 1].sum;
64     }
65     void pushdown(int u) {
66         if (tr[u].add) {
67             int x = tr[u].add;
68             tr[u << 1].sum += (tr[u << 1].r - tr[u << 1].l + 1) * x;
69             tr[u << 1 | 1].sum += (tr[u << 1 | 1].r - tr[u << 1 | 1].l + 1) * x;
70             tr[u << 1].add += x;
71             tr[u << 1 | 1].add += x;
72             tr[u].add = 0;
73         }
74         return;
75     }
76     void build(int u, int l, int r) {
77         tr[u] = {l, r};

```

```

78     if (l == r) return;
79     int mid = (l + r) >> 1;
80     build(u << 1, l, mid);
81     build(u << 1 | 1, mid + 1, r);
82     return;
83 }
84 void modify(int u, int l, int r, int x) {
85     if (l <= tr[u].l && tr[u].r <= r) {
86         tr[u].add += x;
87         tr[u].sum += (tr[u].r - tr[u].l + 1) * x;
88         return;
89     }
90     pushdown(u);
91     int mid = (tr[u].l + tr[u].r) >> 1;
92     if (l <= mid) modify(u << 1, l, r, x);
93     if (mid < r) modify(u << 1 | 1, l, r, x);
94     pushup(u);
95     return;
96 }
97 int query(int u, int l, int r) {
98     if (l <= tr[u].l && tr[u].r <= r) return tr[u].sum;
99     pushdown(u);
100    int mid = (tr[u].l + tr[u].r) >> 1;
101    int res = 0;
102    if (l <= mid) res += query(u << 1, l, r);
103    if (mid < r) res += query(u << 1 | 1, l, r);
104    return res;
105 }
106 } Tr;
107
108 void tree_add(int x, int y) {
109     while (top[x] != top[y]) {
110         if (dep[top[x]] < dep[top[y]]) swap(x, y);
111         Tr.modify(1, L[top[x]], L[x], 1);
112         x = f[top[x]];
113     }
114     if (dep[x] < dep[y]) swap(x, y);
115     Tr.modify(1, L[y], L[x], 1);
116     return;
117 }
118
119 int tree_sum(int x, int y) {
120     int res = 0;
121     while (top[x] != top[y]) {
122         if (dep[top[x]] < dep[top[y]]) swap(x, y);
123         res += Tr.query(1, L[top[x]], L[x]);
124         x = f[top[x]];
125     }
126     if (dep[x] < dep[y]) swap(x, y);
127     res += Tr.query(1, L[y], L[x]);
128     return res;
129 }
130
131 void solve() {
132     cin >> n;
133     rep(i, 1, n) {
134         cin >> a[i].x;
135         a[i].id = i;
136     }
137     sort(a + 1, a + n + 1, [&](node p, node q) {return p.x > q.x;});
138     rep(i, 1, n - 1) {
139         int u, v;
140         cin >> u >> v;
141         add(u, v);
142         add(v, u);
143     }
144     dfs1(1, 0);
145     dfs2(1, 1);
146     Tr.build(1, 1, n);
147     rep(i, 1, n) sum[i] = sum[i - 1] + dep[a[i].id];
148     rep(i, 1, n){

```

```

149         int x = a[i].id;
150         ans = (ans + (ll)dep[x] * (i - 1) * a[i].x % mod + sum[i - 1] * a[i].x % mod - 2ll * a[i].x * tree_sum(1, x) %
↪ mod + mod) % mod;
151         tree_add(1, x);
152     }
153     cout << ans * 2ll % mod << endl;
154 }
155 signed main() {
156     ios;
157     //freopen("sample.txt", "r", stdin);
158     //freopen("resout.txt", "w", stdout);
159     int t = 1;
160     //cin >> t;
161     while (t--) {
162         solve();
163     }
164     return 0;
165 }

```

## 计算几何

### 二维几何：点与向量

```

1  #define y1 yy1
2  #define nxt(i) ((i + 1) % s.size())
3  typedef double LD;
4  const LD PI = 3.14159265358979323846;
5  const LD eps = 1E-10;
6  int sgn(LD x) { return fabs(x) < eps ? 0 : (x > 0 ? 1 : -1); }
7  struct L;
8  struct P;
9  typedef P V;
10 struct P {
11     LD x, y;
12     explicit P(LD x = 0, LD y = 0): x(x), y(y) {}
13     explicit P(const L& l);
14 };
15 struct L {
16     P s, t;
17     L() {}
18     L(P s, P t): s(s), t(t) {}
19 };
20
21 P operator + (const P& a, const P& b) { return P(a.x + b.x, a.y + b.y); }
22 P operator - (const P& a, const P& b) { return P(a.x - b.x, a.y - b.y); }
23 P operator * (const P& a, LD k) { return P(a.x * k, a.y * k); }
24 P operator / (const P& a, LD k) { return P(a.x / k, a.y / k); }
25 inline bool operator < (const P& a, const P& b) {
26     return sgn(a.x - b.x) < 0 || (sgn(a.x - b.x) == 0 && sgn(a.y - b.y) < 0);
27 }
28 bool operator == (const P& a, const P& b) { return !sgn(a.x - b.x) && !sgn(a.y - b.y); }
29 P::P(const L& l) { *this = l.t - l.s; }
30 ostream &operator << (ostream &os, const P &p) {
31     return (os << "(" << p.x << ", " << p.y << ")");
32 }
33 istream &operator >> (istream &is, P &p) {
34     return (is >> p.x >> p.y);
35 }
36
37 LD dist(const P& p) { return sqrt(p.x * p.x + p.y * p.y); }
38 LD dot(const V& a, const V& b) { return a.x * b.x + a.y * b.y; }
39 LD det(const V& a, const V& b) { return a.x * b.y - a.y * b.x; }
40 LD cross(const P& s, const P& t, const P& o = P()) { return det(s - o, t - o); }
41 // -----

```

# 字符串

## KMP

- KMP 模板

```
1  #include <bits/stdc++.h>
2
3  using namespace std;
4
5  const int N = 1e6 + 10;
6
7  vector<int> prefix_function(string s)
8  {
9      int n = (int)s.length();
10     vector<int> pi(n);
11     for (int i = 2; i < n; i++)
12     {
13         pi[i] = pi[i - 1];
14         while (pi[i] && s[i] != s[pi[i] + 1])
15             pi[i] = pi[pi[i]];
16         pi[i] += (s[i] == s[pi[i] + 1]);
17     }
18     return pi;
19 }
20
21 int main(void)
22 {
23     ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
24     string s1, s2;
25     cin >> s1 >> s2;
26     s1 = " " + s1;
27     s2 = " " + s2;
28     auto nxt = prefix_function(s2);
29     for (int i = 1, j = 0; i < s1.size(); i++)
30     {
31         while (j && s1[i] != s2[j + 1])
32             j = nxt[j];
33         if (s1[i] == s2[j + 1])
34             j++;
35         if (j == s2.size() - 1)
36         {
37             cout << i - j + 1 << "\n";
38             j = nxt[j];
39         }
40     }
41     for (int i = 1; i < s2.size(); i++)
42         cout << nxt[i] << " ";
43
44     return 0;
45 }
```

- carpet(二维 KMP) 有一个  $n*m$  的地毯,  $a_{ij}$  表示地毯每格的元素,  $b_{ij}$  表示地毯每格的价格, 要求选取一块价格最大值最小的地毯, 并且这块地毯无限铺开之后, 原地毯是其子矩阵

```
1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define pll pair<ll, ll>
4  #define tll tuple<ll, ll, ll>
5  #define x first
6  #define y second
7  #define int ll
8  #define rep(i, j, k) for (int i = (j); i <= (k); i++)
9  #define per(i, j, k) for (int i = (j); i >= (k); i--)
10 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
11 using namespace std;
12 typedef long long ll;
13 const ll maxn = 1e6 + 10;
14 const ll mod = 998244353;
15 const ll inf = 0x3f3f3f3f;
16
```

```

17 vector<int> prefix_function(string s)
18 {
19     int n = (int)s.length();
20     vector<int> pi(n);
21     for (int i = 2; i < n; i++)
22     {
23         pi[i] = pi[i - 1];
24         while (pi[i] && s[i] != s[pi[i] + 1])
25             pi[i] = pi[pi[i]];
26         pi[i] += (s[i] == s[pi[i] + 1]);
27     }
28     return pi;
29 }
30
31 int get_length(vector<string> s)
32 {
33     int len = s[1].size() - 1;
34     int ret = len;
35     vector<int> cnt(len + 1);
36     for (int i = 1; i < s.size(); ++i)
37     {
38         string tmp = s[i];
39         auto nxt = prefix_function(tmp);
40         int j = len;
41         while (j)
42         {
43             cnt[len - nxt[j]]++;
44             j = nxt[j];
45         }
46     }
47     for (int i = 1; i <= len; ++i)
48         if (cnt[i] == s.size() - 1)
49         {
50             ret = i;
51             break;
52         }
53     return ret;
54 }
55
56 void solve()
57 {
58     int n, m;
59     cin >> n >> m;
60     vector<string> s1(n + 1);
61     for (int i = 1; i <= n; ++i)
62         cin >> s1[i], s1[i] = " " + s1[i];
63     vector<string> s2(m + 1);
64     for (int i = 1; i <= m; ++i)
65     {
66         string tmp = " ";
67         for (int j = 1; j <= n; ++j)
68             tmp += s1[j][i];
69         s2[i] = tmp;
70     }
71     vector<vector<int>> a(n + 1, vector<int>(m + 1, 0));
72     for (int i = 1; i <= n; ++i)
73         for (int j = 1; j <= m; ++j)
74             cin >> a[i][j];
75     int p = get_length(s1), q = get_length(s2);
76     ll ans = 1e9;
77     deque<int> dq;
78     auto b = a;
79     for (int i = 1; i <= n; ++i){
80         while (dq.size()) dq.pop_back();
81         for (int j = 1; j <= m; ++j){
82             while (dq.size() && j - dq.front() + 1 > p) dq.pop_front();
83             while (dq.size() && a[i][dq.back()] <= a[i][j]) dq.pop_back();
84             dq.push_back(j);
85             b[i][j] = a[i][dq.front()];
86         }
87     }

```



```

88     for (int j = 1; j <= m; ++j){
89         while (dq.size()) dq.pop_back();
90         for (int i = 1; i <= n; ++i){
91             while (dq.size() && i - dq.front() + 1 > q) dq.pop_front();
92             while (dq.size() && b[dq.back()][j] <= b[i][j]) dq.pop_back();
93             dq.push_back(i);
94             if (i >= q && j >= p)
95                 ans = min(ans, 1ll * b[dq.front()][j]);
96         }
97     }
98     ans = ans * (p + 1) * (q + 1);
99     cout << ans << endl;
100 }
101
102 signed main()
103 {
104     ios;
105     // freopen("sample.txt", "r", stdin);
106     // freopen("resout.txt", "w", stdout);
107     int t = 1;
108     // cin >> t;
109     while (t--)
110     {
111         solve();
112     }
113     return 0;
114 }
115

```

## 杂项

### 线性基

- 线性基模板 (总异或最大值)

```

1  ll p[64];
2  void insert(ll x){
3      for (int i = 63; ~i; --i){
4          if (!(x >> i)) continue;
5          if (!p[i]){
6              p[i] = x;
7              break;
8          }
9          x ^= p[i];
10     }
11 }

```

- 区间线性基 (区间异或最大值, 强制在线)

```

1  #include<bits/stdc++.h>
2  #define M 500009
3  using namespace std;
4  int read() {
5      int f = 1, re = 0; char ch;
6      for (ch = getchar(); !isdigit(ch) && ch != '-'; ch = getchar());
7      if (ch == '-') {f = -1, ch = getchar();}
8      for (; isdigit(ch); ch = getchar()) re = (re << 3) + (re << 1) + ch - '0';
9      return re * f;
10 }
11 int pos[M][32], p[M][32], t, n, m, lastans;
12 void insert(int val, int num, int po) {
13     for (int i = 30; i >= 0; i--) {
14         if (val & (1ll << i)) {
15             if (!p[num][i]) {p[num][i] = val, pos[num][i] = po; return;}
16             else if (pos[num][i] < po) {
17                 swap(val, p[num][i]);
18                 swap(po, pos[num][i]);
19             } val ^= p[num][i];
20         }
21     } return;

```

```

22 }
23 int query(int l, int r) {
24     int ans = 0;
25     for (int i = 30; i >= 0; i--)
26         if (pos[r][i] >= l && (p[r][i]^ans) > ans) ans ^= p[r][i];
27     return ans;
28 }
29 signed main() {
30     t = read();
31     while (t--) {
32         n = read(), m = read(); lastans = 0;
33         memset(p, 0, sizeof(p));
34         memset(pos, 0, sizeof(pos));
35         for (int i = 1; i <= n; i++) {
36             int x = read();
37             for (int j = 0; j <= 30; j++)
38                 p[i][j] = p[i - 1][j], pos[i][j] = pos[i - 1][j];
39             insert(x, i, i);
40         }
41         for (int i = 1; i <= m; i++) {
42             int opt = read();
43             if (opt) {
44                 int x = read()^lastans; n++;
45                 for (int j = 0; j <= 30; j++)
46                     p[n][j] = p[n - 1][j], pos[n][j] = pos[n - 1][j];
47                 insert(x, n, n);
48             }
49             else {
50                 int l = (read()^lastans) % n + 1;
51                 int r = (read()^lastans) % n + 1;
52                 if (l > r) swap(l, r);
53                 printf("%d\n", lastans = query(l, r));
54             }
55         }
56     } return 0;
57 }

```

- 区间问题 (异或和, 区间内是否存在异或和为 x)

```

1  #include <bits/stdc++.h>
2  #define ll long long
3  using namespace std;
4  constexpr ll maxn = 4e5 + 5;
5  int pos[65];
6  ll p[65], t, n, m;
7  bool ans[maxn];
8  void insert(ll val, int P)
9  {
10     for (int i = 59; i >= 0; i--)
11     {
12         if (val & (1ll << i))
13         {
14             if (!p[i])
15             {
16                 p[i] = val, pos[i] = P;
17                 return;
18             }
19             else if (pos[i] < P)
20             {
21                 swap(val, p[i]);
22                 swap(P, pos[i]);
23             }
24             val ^= p[i];
25         }
26     }
27     return;
28 }
29 bool query(int l, ll val)
30 {
31     for (int i = 59; i >= 0; i--)
32     {
33         if (val & (1ll << i))

```

```

34     {
35         if (!p[i])
36             return false;
37         if (pos[i] < l)
38             return false;
39         val ^= p[i];
40     }
41 }
42 return true;
43 }
44 signed main()
45 {
46     ios::sync_with_stdio(false);
47     cin.tie(nullptr);
48     cin >> n >> m;
49     vector<ll> a(n + 1);
50     vector<tuple<int, int, ll, int>> q(m);
51     for (int i = 1; i <= n; i++)
52         cin >> a[i];
53     for (int i = 0; i < m; i++)
54     {
55         auto &[l, r, val, id] = q[i];
56         cin >> l >> r >> val, id = i;
57     }
58     sort(q.begin(), q.end(), [&](auto x, auto y)
59     {
60         auto &[l1, r1, val1, id1] = x;
61         auto &[l2, r2, val2, id2] = y;
62         return (r1==r2)?(l1<l2):(r1<r2); });
63     int R = 0;
64     for (int i = 0; i < m; i++)
65     {
66         auto &[l, r, val, id] = q[i];
67         while (R < r)
68             insert(a[R + 1], R + 1, R++);
69         ans[id] = query(l, val);
70     }
71     for (int i = 0; i < m; i++)
72     {
73         cout << (ans[i] ? "Yes\n" : "No\n");
74     }
75
76     return 0;
77 }
78

```

## Tarjan

- 缩点

```

1 //Tarjan 缩点 (删去一个点, 有多少点对不能互通)
2 #include <bits/stdc++.h>
3 #define endl '\n'
4 #define pll pair<ll, ll>
5 #define tll tuple<ll, ll, ll>
6 #define x first
7 #define y second
8 #define int ll
9 #define rep(i, j, k) for (int i = (j); i <= (k); i++)
10 #define per(i, j, k) for (int i = (j); i >= (k); i--)
11 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12 using namespace std;
13 typedef long long ll;
14 typedef __int128 i128;
15 const ll maxn = 1e6 + 10;
16 const ll mod = 998244353;
17 const ll inf = 0x3f3f3f3f;
18
19 ll n, m;
20 ll head[maxn], nxt[maxn], to[maxn], tot = 1;
21 ll dfn[maxn], low[maxn];

```

```

22  bool vis[maxn];
23  ll cnt;
24  ll deg[maxn];
25  ll ans[maxn];
26  ll sz[maxn];
27
28  void addedge(int u, int v)
29  {
30      nxt[++tot] = head[u];
31      to[head[u] = tot] = v;
32      nxt[++tot] = head[v];
33      to[head[v] = tot] = u;
34  }
35
36  void tarjan(int u, int lst)
37  {
38      dfn[u] = low[u] = ++cnt;
39      ll sum = 0;
40      sz[u] = 1;
41      for (int i = head[u]; i; i = nxt[i])
42      {
43          if (i != (lst ^ 1))
44          {
45              int v = to[i];
46              if (!dfn[v])
47              {
48                  tarjan(v, i);
49                  sz[u] += sz[v];
50                  low[u] = min(low[u], low[v]);
51                  if (low[v] >= dfn[u])
52                  {
53                      // 找到新的双连通分量
54                      ans[u] += 1ll * sz[v] * (n - sz[v]);
55                      sum += sz[v];
56                      ++deg[u];
57                      if (deg[u] > 1 || u != 1)
58                          vis[u] = 1;
59                  }
60              }
61              else
62                  low[u] = min(low[u], dfn[v]);
63          }
64      }
65      if (vis[u])
66      {
67          ans[u] += 1ll * (n - (sum + 1)) * (sum + 1) + n - 1;
68      } else
69          ans[u] = 2 * (n - 1);
70  }
71  void solve()
72  {
73      cin >> n >> m;
74      for (int i = 1; i <= m; ++i)
75      {
76          int u, v;
77          cin >> u >> v;
78          addedge(u, v);
79      }
80      tarjan(1, -1);
81      for (int i = 1; i <= n; ++i)
82      {
83          if (vis[i])
84          {
85              cout << ans[i] << endl;
86          }
87          else
88          {
89              cout << 2ll * (n - 1) << endl;
90          }
91      }
92  }

```

```
93
94 signed main()
95 {
96     ios;
97     //freopen("sample.txt", "r", stdin);
98     //freopen("res.txt", "w", stdout);
99     int t = 1;
100     // cin >> t;
101     while (t--)
102     {
103         solve();
104     }
105
106     return 0;
107 }
```