

Std Code Library(Qinhuangdao)

tingyx

Nanjing University of Science and Technology

October 9, 2023

Contents

一切的开始	2
Codeforces/XCPC	2
int128	2
数据结构	3
二维数点	3
可持续化线段树	6
可持久化 01Trie	11
树形 DP	12
区间问题	16
莫队	16
CDQ	17
树上问题	22
树剖	22
dsu	29
计算几何	37
点线	37
多边形	40
圆	41
字符串	44
字符串哈希	44
KMP	45
Trie	47
01Trie	49
ACAM	54
manacher	58
pam	64
SA	66
杂项	69
线性基	69
Tarjan	72
位运算基础	73
虚拟源点	74
简单环	75
数位 dp	76
很多线段树, 树状数组	78
快速幂	83
lucas	83
各种背包	83
Z 函数	84

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 }

```

可持续化线段树

- 区间第 k 小

前缀和思想

```
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 #define int ll
15 const ll maxn = 2e5 + 10;
16 const ll mod = 998244353;
17 const ll inf = 0x3f3f3f3f;
18
19 struct node {
20     int ls, rs;
21     int cnt;
22 } tr[maxn << 5];
23 int idx = 0, rt[maxn << 5];
24
25 void push_up(int u) {
26     tr[u].cnt = tr[tr[u].ls].cnt + tr[tr[u].rs].cnt;
27 }
28
29 int build(int l, int r) {
30     int u = idx++;
31     if (l == r) {
32         tr[u].cnt = 0;
33         return u;
34     }
35     int mid = l + r >> 1;
36     tr[u].ls = build(l, mid);
37     tr[u].rs = build(mid + 1, r);
38     push_up(u);
39     return u;
40 }
41
42 int update(int old, int l, int r, int pos, int val) {
43     int u = idx++;
44     tr[u] = tr[old];
45     if (l == pos && r == pos) {
46         tr[u].cnt += val;
47         return u;
48     }
49     int mid = l + r >> 1;
50     if (pos <= mid) tr[u].ls = update(tr[old].ls, l, mid, pos, val);
51     else tr[u].rs = update(tr[old].rs, mid + 1, r, pos, val);
52     push_up(u);
53     return u;
54 }
55
56 int query(int l, int r, int o, int v, int kth) {
57     if (l == r) return l;
58     int mid = (l + r) >> 1;
59     int res = tr[tr[v].ls].cnt - tr[tr[o].ls].cnt;
60     if (kth <= res) return query(l, mid, tr[o].ls, tr[v].ls, kth);
61     else return query(mid + 1, r, tr[o].rs, tr[v].rs, kth - res);
62 }
63
64 int b[maxn], stb[maxn];
65 void solve() {
66     int n, m;
67     cin >> n >> m;
68     for (int i = 1; i <= n; ++i) {
69         cin >> b[i], stb[i] = b[i];
70     }
```



```

71     sort(stb + 1, stb + 1 + n);
72     int cnt = 1;
73     for (int i = 2; i <= n; ++i) {
74         if (stb[i] != stb[cnt]) stb[++cnt] = stb[i];
75     }
76     rt[0] = build(1, cnt);
77     for (int i = 1; i <= n; ++i) {
78         int p = lower_bound(stb + 1, stb + cnt + 1, b[i]) - stb;
79         rt[i] = update(rt[i - 1], 1, cnt, p, 1);
80     }
81     for (int i = 1; i <= m; ++i) {
82         int l, r, k;
83         cin >> l >> r >> k;
84         int idx = query(1, cnt, rt[l - 1], rt[r], k);
85         cout << stb[idx] << endl;
86     }
87 }
88
89 signed main() {
90     ios;
91     //freopen("sample.txt", "r", stdin);
92     //freopen("resout.txt", "w", stdout);
93     int t = 1;
94     //cin >> t;
95     while (t--) {
96         solve();
97     }
98     return 0;
99 }

```

● 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], sortb[maxn];
63 map<int, int> mp;
64 void solve(){
65     int n, m;
66     cin >> n;
67     for (int i = 1; i <= n; ++i) cin >> b[i], sortb[i] = b[i];
68     sort(sortb + 1, sortb + 1 + n);
69     int cnt = 1;
70     for (int i = 2; i <= n)
71         rt[0] = build(1, n);
72
73 }
74
75 int main(){
76     ios;
77     //freopen("sample.txt", "r", stdin);
78     //freopen("resout.txt", "w", stdout);
79     int t = 1;
80     //cin >> t;
81     while(t--){
82         solve();
83     }
84     return 0;
85 }

```

- 区间离散化，多少数字不一样

```

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 int ll
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 maxn = 3e5 + 10;
16 const ll mod = 998244353;
17 const ll inf = 0x3f3f3f3f;
18
19 struct node{

```

```

20     int ls, rs;
21     int cnt, mex;
22 }tr[maxn << 5];
23 int idx = 0, rt[maxn];
24
25 void push_up(int u){
26     tr[u].cnt = tr[tr[u].ls].cnt + tr[tr[u].rs].cnt;
27     tr[u].mex = min(tr[tr[u].ls].mex, tr[tr[u].rs].mex);
28 }
29
30 int build(int l, int r){
31     int u = idx++;
32     if (l == r){
33         tr[u].cnt = 0;
34         return u;
35     }
36     int mid = l + r >> 1;
37     tr[u].ls = build(l, mid);
38     tr[u].rs = build(mid + 1, r);
39     push_up(u);
40     return u;
41 }
42
43 int update(int old, int l, int r, int pos, int val){
44     int u = idx++;
45     tr[u] = tr[old];
46     if (l == pos && r == pos){
47         tr[u].cnt++;
48         tr[u].mex = val;
49         return u;
50     }
51     int mid = l + r >> 1;
52     if (pos <= mid) tr[u].ls = update(tr[old].ls, l, mid, pos, val);
53     else tr[u].rs = update(tr[old].rs, mid + 1, r, pos, val);
54     push_up(u);
55     return u;
56 }
57
58 int queryMex(int u, int l, int r, int pos){
59     if (l == r) return l;
60     int mid = l + r >> 1;
61     if (tr[tr[u].ls].mex < pos) return queryMex(tr[u].ls, l, mid, pos);
62     else return queryMex(tr[u].rs, mid + 1, r, pos);
63 }
64
65 int queryVal(int s, int t, int L, int R, int l, int r) {
66     if (l == L && R == r){return tr[t].cnt - tr[s].cnt;}
67     int mid = L + R >> 1;
68     if (r <= mid) return queryVal(tr[s].ls, tr[t].ls, L, mid, l, r);
69     else if (l <= mid){
70         int res = queryVal(tr[s].ls, tr[t].ls, L, mid, l, mid);
71         res += queryVal(tr[s].rs, tr[t].rs, mid + 1, R, mid + 1, r);
72         return res;
73     }
74     else return queryVal(tr[s].rs, tr[t].rs, mid + 1, R, l, r);
75 }
76
77 void solve(){
78     int n;
79     cin >> n;
80     vi a(n + 1);
81     rt[0] = build(1, n + 1);
82     for (int i = 1; i <= n; ++i){
83         cin >> a[i];
84         if (a[i] > n) a[i] = n + 1;
85         rt[i] = update(rt[i - 1], 1, n + 1, a[i], i);
86     }
87     int m, l, r;
88     cin >> m;
89     while (m--){
90         cin >> l >> r;

```

```

91     int mex = queryMex(rt[r], 1, n + 1, l);
92     int res = queryVal(rt[l - 1], rt[r], 1, n + 1, 1, mex);
93     res = r - l + 1 - res;
94     cout << res << endl;
95 }
96 }
97
98 signed main(){
99     ios;
100     //freopen("sample.txt", "r", stdin);
101     //freopen("resout.txt", "w", stdout);
102     int t = 1;
103     //cin >> t;
104     while(t--){
105         solve();
106     }
107     return 0;
108 }

```

可持久化 01Trie

- 区间 xorK 意义下的最大值

```

1  #include <algorithm>
2  #include <cstdio>
3  #include <cstring>
4  using namespace std;
5  const int maxn = 600010;
6  int n, q, a[maxn], s[maxn], l, r, x;
7  char op;
8
9  struct Trie {
10     int cnt, rt[maxn], ch[maxn * 33][2], val[maxn * 33];
11
12     void insert(int o, int lst, int v) {
13         for (int i = 28; i >= 0; i--) {
14             val[o] = val[lst] + 1; // 在原版本的基础上更新
15             if ((v & (1 << i)) == 0) {
16                 if (!ch[o][0]) ch[o][0] = ++cnt;
17                 ch[o][1] = ch[lst][1];
18                 o = ch[o][0];
19                 lst = ch[lst][0];
20             } else {
21                 if (!ch[o][1]) ch[o][1] = ++cnt;
22                 ch[o][0] = ch[lst][0];
23                 o = ch[o][1];
24                 lst = ch[lst][1];
25             }
26         }
27         val[o] = val[lst] + 1;
28         // printf("%d\n", o);
29     }
30
31     int query(int o1, int o2, int v) {
32         int ret = 0;
33         for (int i = 28; i >= 0; i--) {
34             // printf("%d %d %d\n", o1, o2, val[o1]-val[o2]);
35             int t = ((v & (1 << i)) ? 1 : 0);
36             if (val[ch[o1][!t]] - val[ch[o2][!t]])
37                 ret += (1 << i), o1 = ch[o1][!t],
38                     o2 = ch[o2][!t]; // 尽量向不同的地方跳
39             else
40                 o1 = ch[o1][t], o2 = ch[o2][t];
41         }
42         return ret;
43     }
44 } st;
45
46 int main() {
47     scanf("%d%d", &n, &q);
48     for (int i = 1; i <= n; i++) scanf("%d", a + i), s[i] = s[i - 1] ^ a[i];

```

```

49     for (int i = 1; i <= n; i++)
50         st.rt[i] = ++st.cnt, st.insert(st.rt[i], st.rt[i - 1], s[i]);
51     while (q--) {
52         scanf("%c", &op);
53         if (op == 'A') {
54             n++;
55             scanf("%d", &a + n);
56             s[n] = s[n - 1] ^ a[n];
57             st.rt[n] = ++st.cnt;
58             st.insert(st.rt[n], st.rt[n - 1], s[n]);
59         }
60         if (op == 'Q') {
61             scanf("%d%d%d", &l, &r, &x);
62             l--;
63             r--;
64             if (l == 0)
65                 printf("%d\n", max(s[n] ^ x, st.query(st.rt[r], st.rt[0], s[n] ^ x)));
66             else
67                 printf("%d\n", st.query(st.rt[r], st.rt[l - 1], s[n] ^ x));
68         }
69     }
70     return 0;
71 }

```

树形 DP

- 树的重心

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
4  using namespace std;
5  const int maxn = 2e5 + 10;
6  typedef long long ll;
7  int n;
8
9  void solve()
10 {
11     vector<vector<int>> g(n + 1);
12     for (int i = 0; i < n - 1; i++)
13     {
14         int x, y;
15         cin >> x >> y;
16         x--, y--;
17         g[x].push_back(y);
18         g[y].push_back(x);
19     }
20     vector<int> siz(n + 1);
21     int id = 1e9, Min = 1e9;
22     function<void(int, int)> dfs = [&](int x, int fa)
23     {
24         siz[x] = 1;
25         for (auto y : g[x])
26         {
27             if (y == fa)
28                 continue;
29             dfs(y, x);
30             siz[x] += siz[y];
31             int v = max(siz[x], n - siz[x]);
32             if (v <= Min)
33             {
34                 if (v < Min)
35                     Min = v, id = x;
36                 else if (x < id)
37                     id = x;
38             }
39         }
40     };
41     dfs(0, 0);
42     cout << id + 1 << " " << Min - 1 << "\n";
43 }

```

```

44
45 int main()
46 {
47     ios;
48     while (cin >> n)
49     {
50         solve();
51     }
52     return 0;
53 }

```

● 树的最大独立集

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  struct edge {
5      int v, next;
6  } e[6005];
7
8  int head[6005], n, cnt, f[6005][2], ans, is_h[6005], vis[6005];
9
10 void addedge(int u, int v) { // 建图
11     e[++cnt].v = v;
12     e[cnt].next = head[u];
13     head[u] = cnt;
14 }
15
16 void calc(int k) {
17     vis[k] = 1;
18     for (int i = head[k]; i; i = e[i].next) { // 枚举该结点的每个子结点
19         if (vis[e[i].v]) continue;
20         calc(e[i].v);
21         f[k][1] += f[e[i].v][0];
22         f[k][0] += max(f[e[i].v][0], f[e[i].v][1]); // 转移方程
23     }
24     return;
25 }
26
27 int main() {
28     scanf("%d", &n);
29     for (int i = 1; i <= n; i++) scanf("%d", &f[i][1]);
30     for (int i = 1; i < n; i++) {
31         int l, k;
32         scanf("%d%d", &l, &k);
33         is_h[l] = 1;
34         addedge(k, l);
35     }
36     for (int i = 1; i <= n; i++)
37         if (!is_h[i]) { // 从根结点开始 DFS
38             calc(i);
39             printf("%d", max(f[i][1], f[i][0]));
40             return 0;
41         }
42 }

```

● 树的最小支配集

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  #define N 10010
4
5  int n;
6  int e[N * 2], ne[N * 2], h[N], idx = 0;
7  int f[N][3];
8  /**
9   * f[i][0] 选 i 且 i 及 i 的子树都被覆盖了
10  * f[i][1] 不选 i 且 i 被其儿子覆盖
11  * f[i][2] 不选 i 且 i 被其父亲覆盖 (儿子可选可不选)
12  */
13 void add(int a, int b)
14 {

```

```

15     e[idx] = b, ne[idx] = h[a], h[a] = idx++;
16 }
17
18 void dfs(int u, int pre)
19 {
20     f[u][0] = 1, f[u][1] = f[u][2] = 0;
21     bool flag = true;
22     int tmp = 0x3f3f3f3f;
23     for (int i = h[u]; ~i; i = ne[i])
24     {
25         int v = e[i];
26         if (v == pre)
27             continue;
28         dfs(v, u);
29         f[u][2] += min(f[v][1], f[v][0]);
30         f[u][0] += min(min(f[v][0], f[v][1]), f[v][2]);
31         if (f[v][0] <= f[v][1])
32         {
33             flag = false;
34             f[u][1] += f[v][0];
35         }
36         else
37         {
38             f[u][1] += f[v][1];
39             tmp = min(tmp, f[v][0] - f[v][1]);
40         }
41     }
42     if (flag)
43         f[u][1] += tmp;
44 }
45
46 int main()
47 {
48     memset(f, 0x3f, sizeof f);
49     memset(h, -1, sizeof h);
50     scanf("%d", &n);
51     for (int i = 1; i < n; i++)
52     {
53         int a, b;
54         scanf("%d%d", &a, &b);
55         add(a, b), add(b, a);
56     }
57     dfs(1, -1);
58     int ans = min(f[1][0], f[1][1]);
59     cout << ans << endl;
60     return 0;
61 }

```

● 树的最小覆盖点

```

1  #include <bits/stdc++.h>
2  #define endl '\n'
3  #define mset(s, _) memset(s, _, sizeof(s))
4  #define rep(i, l, r) for (int i = l; i <= r; ++i)
5  #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
6  using namespace std;
7  const int N = 4e3 + 10, mod = 1e9 + 7;
8  int n, m;
9  int h[N], nex[N], v[N], idx;
10 void add(int a, int b) {
11     v[idx] = b; nex[idx] = h[a]; h[a] = idx ++ ;
12 }
13
14 int f[N][5], st[N];
15 void init() {
16     mset(h, -1); mset(f, 0); mset(st, 0); idx = 0;
17 }
18
19 void dp(int u) {
20     bool fg = 0;
21     for (int i = h[u]; ~i; i = nex[i]) {
22         int j = v[i];

```

```

23         fg = 1;
24         dp(j);
25         f[u][0] += f[j][1];
26         f[u][1] += min(f[j][0], f[j][1]);
27     }
28     f[u][1] += 1;
29     if(!fg) {
30         f[u][0] = 0; f[u][1] = 1;
31     }
32 }
33
34 int main() {
35     while(cin >> n) {
36         init();
37         rep(i, 1, n) {
38             int a, num, b; char t;
39             cin >> a >> t >> t >> num >> t;
40             rep(j, 1, num) {
41                 cin >> b; add(a, b); st[b] = 1;
42             }
43         }
44         int root = 0;
45         while(st[root]) root ++ ;
46         dp(root);
47         cout << min(f[root][1], f[root][0]) << endl;
48     }
49
50     return 0;
51 }

```

● 树上背包

最多切 q 条边，剩下多少东西

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int N = 105;
4  int dp[N][N]; //dp[i][j] 以 i 为根的子树保留 j 个分支可以得到的最大的苹果数量
5  int h[N], e[N << 1], nx[N << 1], w[N << 1];
6  int cnt = 1;
7  void add(int a, int b, int v)
8  {
9      e[cnt] = b;
10     w[cnt] = v;
11     nx[cnt] = h[a];
12     h[a] = cnt++;
13 }
14 int n, q;
15 void dfs(int u, int f)
16 {
17     for (int i = h[u]; i; i = nx[i])
18     {
19         int v = e[i];
20         if (v != f)
21         {
22             dfs(v, u);
23             for (int j = q; j >= 1; j--)
24             {
25                 for (int k = 0; k <= j - 1; k++)
26                 {
27                     dp[u][j] = max(dp[u][j], dp[u][k] + w[i] + dp[v][j - k - 1]);
28                 }
29             }
30         }
31     }
32 }
33 int main()
34 {
35
36     cin >> n >> q;
37     int a, b, v;
38     for (int i = 0; i < n - 1; i++)

```



```

39     {
40         cin >> a >> b >> v;
41         add(a, b, v);
42         add(b, a, v);
43     }
44     dfs(1, -1);
45     cout << dp[1][q];
46 }

```

- 树的直径 (带点权)

```

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 }

```

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)) {

```

```

23     temp[l++][deep] = temp[p1++][deep + 1];
24 } else {
25     temp[l++][deep] = temp[p2++][deep + 1];
26 }
27 }
28 while (p1 <= mid) {
29     temp[l++][deep] = temp[p1++][deep + 1];
30 }
31 while (p2 <= r) {
32     temp[l++][deep] = temp[p2++][deep + 1];
33 }
34 }
35
36 void cdqDivAlgorithm(int l, int r, int deep, const int &cmptype) {
37     if (l == r) {
38         dp[l] = max(dp[l], 1);
39         ans = max(ans, dp[l]);
40         return;
41     }
42     int mid = (l + r) >> 1;
43     cdqDivAlgorithm(l, mid, deep + 1, cmptype);
44     int p1 = l, p2 = mid + 1, premax = 0;
45     while (p1 <= mid && p2 <= r) {
46         if (cmp(temp[p1][deep + 1], temp[p2][deep + 1], cmptype)) {
47             premax = max(premax, dp[temp[p1++][deep + 1].second]);
48         } else {
49             dp[temp[p2][deep + 1].second] = max(premax + 1, dp[temp[p2][deep + 1].second]);
50             p2++;
51         }
52     }
53     while (p2 <= r) {
54         dp[temp[p2][deep + 1].second] = max(premax + 1, dp[temp[p2][deep + 1].second]);
55         p2++;
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, 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] : 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) -
↪ getCross(stk[top].first, stk[top].second, k, b)) > 0) top--;
30     stk[++top] = make_pair(k, b);
31 }
32 double stkQuery(int x) {
33     while (top >= 2 && dcmp(getVal(stk[top].first, stk[top].second, x) - getVal(stk[top - 1].first, stk[top -
↪ 1].second, x)) < 0) --top;
34     return getVal(stk[top].first, stk[top].second, x);
35 }
36 void cdqDivAlgorithm(int l, int r) {
37     if (l == r) return;
38     int mid = (l + r) >> 1;
39     cdqDivAlgorithm(l, mid);
40     cdqDivAlgorithm(mid + 1, r);
41     stkClear();
42     for (int i = l; i <= mid && !type[id[i]]; ++i) {
43         stkInsert(k[id[i]], b[id[i]]);
44     }
45     for (int i = r; i > mid && type[id[i]]; --i) {
46         ans[qid[id[i]]] = max(ans[qid[id[i]]], stkQuery(x[id[i]]));
47     }
48     int p1 = l, p1 = l, p2 = mid + 1;
49     while (p1 <= mid && p2 <= r) {
50         if (cmp(id[p1], id[p2])) {
51             temp[p1++] = id[p1++];
52         } else {
53             temp[p1++] = id[p2++];
54         }
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, mxl = 0;
87             for(int i = 0; i < q.size(); ++i) {
88                 mnd = min(mnd, dfn[q[i]]);
89                 mxl = max(mxl, dfn[q[i]]);
90             }
91             int ca = lca(idl[mnd], idl[mxl]);
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], skp;
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.builde(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

```

```

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

点线

```

1  #include <bits/stdc++.h>
2  using namespace std;
3
4  typedef double db;
5  const db EPS = 1e-9;
6
7  inline int sign(db a) {
8      return a < -EPS ? -1 : a > EPS;
9  }
10
11 inline int cmp(db a, db b) {

```

```

12     return sign(a - b);
13 }
14
15 struct P {
16     db x, y;
17     P() {}
18     P(db _x, db _y) : x(_x), y(_y) {}
19     //重构加减乘除
20     P operator+(P p) { return {x + p.x, y + p.y}; }
21     P operator-(P p) { return {x - p.x, y - p.y}; }
22     P operator*(db d) { return {x * d, y * d}; }
23     P operator/(db d) { return {x / d, y / d}; }
24
25     bool operator<(P p) const {
26         int c = cmp(x, p.x);
27         if (c)
28             return c == -1;
29         return cmp(y, p.y) == -1;
30     }
31
32     bool operator==(P o) const { return cmp(x, o.x) == 0 && cmp(y, o.y) == 0; }
33
34     db dot(P p) { return x * p.x + y * p.y; } //点积
35     db det(P p) { return x * p.y - y * p.x; } //叉积
36
37     db distTo(P p) { return (*this - p).abs(); }
38     db alpha() { return atan2(y, x); }
39     void read() { cin >> x >> y; }
40     void write() { cout << "(" << x << "," << y << ")" << endl; }
41     db abs() { return sqrt(abs2()); }
42     db abs2() { return x * x + y * y; }
43     P rot90() { return P(-y, x); }
44     P unit() { return *this / abs(); }
45     int quad() const { return sign(y) == 1 || (sign(y) == 0 && sign(x) >= 0); }
46     P rot(db an) {
47         return {x * cos(an) - y * sin(an), x * sin(an) + y * cos(an)};
48     }
49 };
50
51 #define cross(p1, p2, p3) \
52     ((p2.x - p1.x) * (p3.y - p1.y) - (p3.x - p1.x) * (p2.y - p1.y))
53 #define crossOp(p1, p2, p3) sign(cross(p1, p2, p3))
54
55 // 直线 p1p2, q1q2 是否恰有一个交点
56 bool chkLL(P p1, P p2, P q1, P q2) {
57     db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
58     return sign(a1 + a2) != 0;
59 }
60
61 // 求直线 p1p2, q1q2 的交点
62 P isLL(P p1, P p2, P q1, P q2) {
63     db a1 = cross(q1, q2, p1), a2 = -cross(q1, q2, p2);
64     return (p1 * a2 + p2 * a1) / (a1 + a2);
65 }
66
67 // 判断区间 [l1, r1], [l2, r2] 是否相交
68 bool intersect(db l1, db r1, db l2, db r2) {
69     if (l1 > r1)
70         swap(l1, r1);
71     if (l2 > r2)
72         swap(l2, r2);
73     return !(cmp(r1, l2) == -1 || cmp(r2, l1) == -1);
74 }
75
76 // 线段 p1p2, q1q2 相交
77 bool isSS(P p1, P p2, P q1, P q2) {
78     return intersect(p1.x, p2.x, q1.x, q2.x) &&
79         intersect(p1.y, p2.y, q1.y, q2.y) &&
80         crossOp(p1, p2, q1) * crossOp(p1, p2, q2) <= 0 &&
81         crossOp(q1, q2, p1) * crossOp(q1, q2, p2) <= 0;
82 }

```

```

83 // 线段 p1p2, q1q2 严格相交
84 bool isSS_strict(P p1, P p2, P q1, P q2) {
85     return crossOp(p1, p2, q1) * crossOp(p1, p2, q2) < 0 &&
86         crossOp(q1, q2, p1) * crossOp(q1, q2, p2) < 0;
87 }
88
89 // m 在 a 和 b 之间
90 bool isMiddle(db a, db m, db b) {
91     /*if (a > b) swap(a, b);
92     return cmp(a, m) <= 0 && cmp(m, b) <= 0;*/
93     return sign(a - m) == 0 || sign(b - m) == 0 || (a < m != b < m);
94 }
95
96 bool isMiddle(P a, P m, P b) {
97     return isMiddle(a.x, m.x, b.x) && isMiddle(a.y, m.y, b.y);
98 }
99
100 // 点 p 在线段 p1p2 上
101 bool onSeg(P p1, P p2, P q) {
102     return crossOp(p1, p2, q) == 0 && isMiddle(p1, q, p2);
103 }
104 // q1q2 和 p1p2 的交点 在 p1p2 上?
105 // 点 p 严格在 p1p2 上
106 bool onSeg_strict(P p1, P p2, P q) {
107     return crossOp(p1, p2, q) == 0 &&
108         sign((q - p1).dot(p1 - p2)) * sign((q - p2).dot(p1 - p2)) < 0;
109 }
110
111 // 求 q 到 直线 p1p2 的投影 (垂足) 注意 : p1 != p2
112 P proj(P p1, P p2, P q) {
113     P dir = p2 - p1;
114     return p1 + dir * (dir.dot(q - p1) / dir.abs2());
115 }
116
117 // 求 q 以 直线 p1p2 为轴的反射
118 P reflect(P p1, P p2, P q) {
119     return proj(p1, p2, q) * 2 - q;
120 }
121
122 // 求 q 到 线段 p1p2 的最小距离
123 db nearest(P p1, P p2, P q) {
124     if (p1 == p2)
125         return p1.distTo(q);
126     P h = proj(p1, p2, q);
127     if (isMiddle(p1, h, p2))
128         return q.distTo(h);
129     return min(p1.distTo(q), p2.distTo(q));
130 }
131
132 // 求 线段 p1p2 与 线段 q1q2 的距离
133 db disSS(P p1, P p2, P q1, P q2) {
134     if (isSS(p1, p2, q1, q2))
135         return 0;
136     return min(min(nearest(p1, p2, q1), nearest(p1, p2, q2)),
137         min(nearest(q1, q2, p1), nearest(q1, q2, p2)));
138 }
139
140 // 极角排序
141 sort(p, p + n, [&](P a, P b) {
142     int qa = a.quad(), qb = b.quad();
143     if (qa != qb)
144         return qa < qb;
145     else
146         return sign(a.det(b)) > 0;
147 });
148
149

```


多边形

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  #define rep(i, a, n) for (int i = a; i < n; i++)
4  typedef double db;
5  const db EPS = 1e-9;
6
7  //求多边形面积
8  db area(vector<P> ps) {
9      db ret = 0;
10     rep(i, 0, ps.size()) ret += ps[i].det(ps[(i + 1) % ps.size()]);
11     return ret / 2;
12 }
13 //点包含
14 int contain(vector<P> ps, P p) { // 2:inside,1:on_seg,0:outside
15     int n = ps.size(), ret = 0;
16     rep(i, 0, n) {
17         P u = ps[i], v = ps[(i + 1) % n];
18         if (onSeg(u, v, p))
19             return 1;
20         if (cmp(u.y, v.y) <= 0)
21             swap(u, v);
22         if (cmp(p.y, u.y) > 0 || cmp(p.y, v.y) <= 0)
23             continue;
24         ret ^= crossOp(p, u, v) > 0;
25     }
26     return ret * 2;
27 }
28 //严格凸包
29 vector<P> convexHull(vector<P> ps) {
30     int n = ps.size();
31     if (n <= 1)
32         return ps;
33     sort(ps.begin(), ps.end());
34     vector<P> qs(n * 2);
35     int k = 0;
36     for (int i = 0; i < n; qs[k++] = ps[i++])
37         while (k > 1 && crossOp(qs[k - 2], qs[k - 1], ps[i]) <= 0)
38             --k;
39     for (int i = n - 2, t = k; i >= 0; qs[k++] = ps[i--])
40         while (k > t && crossOp(qs[k - 2], qs[k - 1], ps[i]) <= 0)
41             --k;
42     qs.resize(k - 1);
43     return qs;
44 }
45
46 //不严格凸包
47 vector<P> convexHullNonStrict(vector<P> ps) {
48     // caution: need to unique the Ps first
49     int n = ps.size();
50     if (n <= 1)
51         return ps;
52     sort(ps.begin(), ps.end());
53     vector<P> qs(n * 2);
54     int k = 0;
55     for (int i = 0; i < n; qs[k++] = ps[i++])
56         while (k > 1 && crossOp(qs[k - 2], qs[k - 1], ps[i]) < 0)
57             --k;
58     for (int i = n - 2, t = k; i >= 0; qs[k++] = ps[i--])
59         while (k > t && crossOp(qs[k - 2], qs[k - 1], ps[i]) < 0)
60             --k;
61     qs.resize(k - 1);
62     return qs;
63 }
64 //旋转卡壳
65 db convexDiameter(vector<P> ps) {
66     int n = ps.size();
67     if (n <= 1)
68         return 0;
69     int is = 0, js = 0;
```

```

70     rep(k, 1, n) is = ps[k] < ps[is] ? k : is, js = ps[js] < ps[k] ? k : js;
71     int i = is, j = js;
72     db ret = ps[i].distTo(ps[j]);
73     do {
74         if ((ps[(i + 1) % n] - ps[i]).det(ps[(j + 1) % n] - ps[j]) >= 0)
75             (++j) %= n;
76         else
77             (++i) %= n;
78         ret = max(ret, ps[i].distTo(ps[j]));
79     } while (i != is || j != js);
80     return ret;
81 }
82
83 //切多边形
84 vector<P> convexCut(const vector<P>& ps, P q1, P q2) {
85     vector<P> qs;
86     int n = ps.size();
87     rep(i, 0, n) {
88         P p1 = ps[i], p2 = ps[(i + 1) % n];
89         int d1 = crossOp(q1, q2, p1), d2 = crossOp(q1, q2, p2);
90         if (d1 >= 0)
91             qs.push_back(p1);
92         if (d1 * d2 < 0)
93             qs.push_back(isLL(p1, p2, q1, q2));
94     }
95     return qs;
96 }

```

圆

```

1  #define rep(i, a, n) for (int i = a; i < n; i++)
2  const double PI = acos(-1.0);
3
4  //判断两个圆的关系
5  int type(P o1, db r1, P o2, db r2) {
6      db d = o1.distTo(o2);
7      if (cmp(d, r1 + r2) == 1)
8          return 4;
9      if (cmp(d, r1 + r2) == 0)
10         return 3;
11     if (cmp(d, abs(r1 - r2)) == 1)
12         return 2;
13     if (cmp(d, abs(r1 - r2)) == 0)
14         return 1;
15     return 0;
16 }
17 //圆和线相交
18 vector<P> isCL(P o, db r, P p1, P p2) {
19     if (cmp(abs((o - p1).det(p2 - p1) / p1.distTo(p2)), r) > 0)
20         return {};
21     db x = (p1 - o).dot(p2 - p1), y = (p2 - p1).abs2(),
22         d = x * x - y * ((p1 - o).abs2() - r * r);
23     d = max(d, (db)0.0);
24     P m = p1 - (p2 - p1) * (x / y), dr = (p2 - p1) * (sqrt(d) / y);
25     return {m - dr, m + dr}; // along dir: p1->p2
26 }
27
28 //两个圆相交的交点
29 vector<P> isCC(P o1,
30     db r1,
31     P o2,
32     db r2) { // need to check whether two circles are the same
33     db d = o1.distTo(o2);
34     if (cmp(d, r1 + r2) == 1)
35         return {};
36     if (cmp(d, abs(r1 - r2)) == -1)
37         return {};
38     d = min(d, r1 + r2);
39     db y = (r1 * r1 + d * d - r2 * r2) / (2 * d), x = sqrt(r1 * r1 - y * y);
40     P dr = (o2 - o1).unit();
41     P q1 = o1 + dr * y, q2 = dr.rot90() * x;

```

```

42     return {q1 - q2, q1 + q2}; // along circle 1
43 }
44
45 //求切线, 默认求外公切线, 求内公切线的话, r2 改成负数, 求点到圆的切线, r2 改成 0
46 // extanCC, intanCC : -r2, tanCP : r2 = 0
47 vector<pair<P, P>> tanCC(P o1, db r1, P o2, db r2) {
48     P d = o2 - o1;
49     db dr = r1 - r2, d2 = d.abs2(), h2 = d2 - dr * dr;
50     if (sign(d2) == 0 || sign(h2) < 0)
51         return {};
52     h2 = max((db)0.0, h2);
53     vector<pair<P, P>> ret;
54     for (db sign : {-1, 1}) {
55         P v = (d * dr + d.rot90() * sqrt(h2) * sign) / d2;
56         ret.push_back({o1 + v * r1, o2 + v * r2});
57     }
58     if (sign(h2) == 0)
59         ret.pop_back();
60     return ret;
61 }
62
63 db rad(P p1, P p2) {
64     return atan2l(p1.det(p2), p1.dot(p2));
65 }
66 //圆和三角形的面积交
67 db areaCT(db r, P p1, P p2) {
68     vector<P> is = isCL(P(0, 0), r, p1, p2);
69     if (is.empty())
70         return r * r * rad(p1, p2) / 2;
71     bool b1 = cmp(p1.abs2(), r * r) == 1, b2 = cmp(p2.abs2(), r * r) == 1;
72     if (b1 && b2) {
73         P md = (is[0] + is[1]) / 2;
74         if (sign((p1 - md).dot(p2 - md)) <= 0)
75             return r * r * (rad(p1, is[0]) + rad(is[1], p2)) / 2 +
76                 is[0].det(is[1]) / 2;
77         else
78             return r * r * rad(p1, p2) / 2;
79     }
80     if (b1)
81         return (r * r * rad(p1, is[0]) + is[0].det(p2)) / 2;
82     if (b2)
83         return (p1.det(is[1]) + r * r * rad(is[1], p2)) / 2;
84     return p1.det(p2) / 2;
85 }
86
87 //内心
88 P inCenter(P A, P B, P C) {
89     double a = (B - C).abs(), b = (C - A).abs(), c = (A - B).abs();
90     return (A * a + B * b + C * c) / (a + b + c);
91 }
92 //外心
93 P circumCenter(P a, P b, P c) {
94     P bb = b - a, cc = c - a;
95     double db = bb.abs2(), dc = cc.abs2(), d = 2 * bb.det(cc);
96     return a - P(bb.y * dc - cc.y * db, cc.x * db - bb.x * dc) / d;
97 }
98 //垂心
99 P orthoCenter(P a, P b, P c) {
100     P ba = b - a, ca = c - a, bc = b - c;
101     double Y = ba.y * ca.y * bc.y, A = ca.x * ba.y - ba.x * ca.y,
102         x0 = (Y + ca.x * ba.y * b.x - ba.x * ca.y * c.x) / A,
103         y0 = -ba.x * (x0 - c.x) / ba.y + ca.y;
104     return {x0, y0};
105 }
106
107 //最小圆覆盖, 随机增量法
108 pair<P, db> min_circle(vector<P> ps) {
109     random_device rd;
110     mt19937 rng(rd());
111     shuffle(ps.begin(), ps.end(), rng);
112     // random_shuffle(ps.begin(), ps.end());

```

```

113     int n = ps.size();
114     P o = ps[0];
115     db r = 0;
116     rep(i, 1, n) if (o.distTo(ps[i]) > r + EPS) {
117         o = ps[i], r = 0;
118         rep(j, 0, i) if (o.distTo(ps[j]) > r + EPS) {
119             o = (ps[i] + ps[j]) / 2;
120             r = o.distTo(ps[i]);
121             rep(k, 0, j) if (o.distTo(ps[k]) > r + EPS) {
122                 o = circumCenter(ps[i], ps[j], ps[k]);
123                 r = o.distTo(ps[i]);
124             }
125         }
126     }
127     return {o, r};
128 }
129
130 db intergal(db x, db y, db r, db L, db R) {
131     return r * r * (R - L) + x * r * (sinl(R) - sinl(L)) +
132         y * r * (-cosl(R) + cosl(L));
133 }
134
135 db calc_area_circle(P c, db r, db L, db R) {
136     return intergal(c.x, c.y, r, L, R) / 2;
137 }
138
139 db norm(db x) {
140     while (x < 0)
141         x += 2 * PI;
142     while (x > 2 * PI)
143         x -= 2 * PI;
144     return x;
145 }
146
147
148
149 //圖面積并
150 P cs[N];
151 db rs[N];
152
153 void work() {
154     vector<int> cand = {};
155     rep(i, 0, n) {
156         bool ok = 1;
157         rep(j, 0, n) if (i != j) {
158             if (rs[j] > rs[i] + EPS &&
159                 rs[i] + cs[i].distTo(cs[j]) <= rs[j] + EPS) {
160                 ok = 0;
161                 break;
162             }
163             if (cs[i] == cs[j] && cmp(rs[i], rs[j]) == 0 && j < i) {
164                 ok = 0;
165                 break;
166             }
167         }
168         if (ok)
169             cand.push_back(i);
170     }
171
172     rep(i, 0, cand.size()) cs[i] = cs[cand[i]], rs[i] = rs[cand[i]];
173     n = cand.size();
174
175     db area = 0;
176
177     // work
178     rep(i, 0, n) {
179         vector<pair<db, int>> ev = {{0, 0}, {2 * PI, 0}};
180
181         int cur = 0;
182
183         rep(j, 0, n) if (j != i) {

```

```

184         auto ret = isCC(cs[i], rs[i], cs[j], rs[j]);
185         if (!ret.empty()) {
186             db l = (ret[0] - cs[i]).alpha();
187             db r = (ret[1] - cs[i]).alpha();
188             l = norm(l);
189             r = norm(r);
190             ev.push_back({l, 1});
191             ev.push_back({r, -1});
192             if (l > r)
193                 ++cur;
194         }
195     }
196
197     sort(ev.begin(), ev.end());
198     rep(j, 0, ev.size() - 1) {
199         cur += ev[j].se;
200         if (cur == 0) {
201             area += calc_area_circle(cs[i], rs[i], ev[j].fi, ev[j + 1].fi);
202         }
203     }
204 }
205 }

```

字符串

字符串哈希

- 取双模

```

1  #include<bits/stdc++.h>
2  #include<unordered_map>
3  #define debug cout << "debug--- "
4  #define debug_ cout << "\n---debug---\n"
5  #define oper(a) operator<(const a& ee)const
6  #define forr(a,b,c) for(int a=b;a<=c;a++)
7  #define mem(a,b) memset(a,b,sizeof a)
8  #define cinios (ios::sync_with_stdio(false),cin.tie(0),cout.tie(0))
9  #define all(a) a.begin(),a.end()
10 #define sz(a) (int)a.size()
11 #define endl "\n"
12 #define ul (u << 1)
13 #define ur (u << 1 | 1)
14 using namespace std;
15
16 typedef unsigned long long ull;
17 typedef long long ll;
18 typedef pair<ll, int> PII;
19
20 const int N = 1e5 + 10, M = 2e6 + 10, mod = 1e9 + 7;
21 int INF = 0x3f3f3f3f; ll LNF = 0x3f3f3f3f3f3f3f3f;
22 int n, m, B = 10, ki;
23
24 const int mod1 = 1e9 + 9;
25
26 ll p1[N], P1 = 131, p2[N], P2 = 13331;
27 ll h[N], h2[N];
28 //乘法开 ll, mod 取 int
29
30 int get1(int l, int r) {
31     return (h[r] - (h[l - 1] * p1[r - l + 1]) % mod + mod) % mod;
32 }
33 int get2(int l, int r) {
34     return (h2[r] - (h2[l - 1] * p2[r - l + 1]) % mod1 + mod1) % mod1;
35 }
36
37 char str[N];
38
39 void solve() {
40     cin >> n >> m;
41     cin >> str + 1;

```

```

42
43 p1[0] = p2[0] = 1;
44
45 for (int i = 1; i <= n; i++) {
46     p1[i] = (p1[i - 1] * P1) % mod;
47     p2[i] = (p2[i - 1] * P2) % mod1;
48
49     h[i] = ((h[i - 1] * P1) % mod + str[i] - '0' + 1) % mod;
50     h2[i] = ((h2[i - 1] * P2) % mod1 + str[i] - '0' + 1) % mod1;
51 }
52
53 while (m--)
54 {
55     int l1, r1, l2, r2;
56     cin >> l1 >> r1 >> l2 >> r2;
57     if (get1(l1, r1) == get1(l2, r2) && get2(l1, r1) == get2(l2, r2)) cout << "Yes\n";
58     else cout << "No\n";
59 }
60 }
61
62 signed main() {
63     cinios;
64     int T = 1;
65     for (int t = 1; t <= T; t++) {
66         solve();
67     }
68     return 0;
69 }

```

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

```

Trie

- trie & topo

可自定字符间大小关系，求多少个串可以成为字典序最小的串

```

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 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 = 3e4 + 10, maxm = (3e4 + 10) * 26;
13 const ll mod = 998244353;
14 const ll inf = 0x3f3f3f3f;
15
16 int tr[maxn][26], idx = 0;
17 bool vis[maxn];
18 vector<int> edge[26];
19 int in[26];
20
21 void insert(string s)
22 {
23     int x = 0;
24     for (auto op : s)
25     {
26         auto c = op - 'a';
27         if (!tr[x][c])
28             tr[x][c] = ++idx;
29         x = tr[x][c];
30     }
31     vis[x] = 1;
32 }
33
34 bool query(string s)
35 {
36     auto topo = [&]() {
37         queue<int> q;
38         int cnt = 0;
39         for (int i = 0; i < 26; ++i) if (!in[i]) q.push(i);
40         while(!q.empty()) {
41             auto op = q.front();
42             q.pop(), cnt++;
43             for (auto v : edge[op]) {
44                 if (!--in[v]) q.push(v);
45             }
46         }
47         return cnt == 26;
48     };
49
50     int x = 0;
51     for (int i = 0; i < s.size(); ++i) {
52         auto c = s[i] - 'a';
53         for (int j = 0; j < 26; ++j) {
54             if (j == c || !tr[x][j]) continue;
55             edge[c].push_back(j);
56             in[j]++;
57         }
58         x = tr[x][c];
59         if (vis[x] && i != s.size() - 1) return false;
60     }
61     return topo();
62 }
63
64 void solve()
65 {
66     int n;
67     cin >> n;
68     vector<string> v(n + 1);
69     for (int i = 1; i <= n; ++i)
70     {
71         cin >> v[i];
72         insert(v[i]);
73     }
74     vector<string> res;
75     for (int op = 1; op <= n; ++op)
76     {
77         for (int i = 0; i < 26; ++i)
78             in[i] = 0, edge[i].clear();
79         if (query(v[op]))
80             res.push_back(v[op]);

```

```

81     }
82     cout << res.size() << endl;
83     for (auto s : res)
84         cout << s << endl;
85 }
86
87 int main()
88 {
89     ios;
90     // freopen("sample.txt", "r", stdin);
91     // freopen("resout.txt", "w", stdout);
92     int t = 1;
93     //cin >> t;
94     while (t--)
95     {
96         solve();
97     }
98     return 0;
99 }

```

01Trie

- 两数最大异或和

```

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 = 2e5 + 10;
15 const ll maxm = maxn * 32;
16 const ll mod = 998244353;
17 const ll inf = 0x3f3f3f3f;
18
19 int tr[maxm][2], idx, n;
20
21 void insert(int x){
22     int p = 0;
23     for (int i = 31; i >= 0; --i){
24         int c = x >> i & 1;
25         if (!tr[p][c]) tr[p][c] = ++idx;
26         p = tr[p][c];
27     }
28 }
29
30 int query(int x){
31     int res = 0, p = 0;
32     for (int i = 31; i >= 0; --i){
33         int c = x >> i & 1;
34         if (tr[p][c ^ 1]){
35             p = tr[p][c ^ 1];
36             res += 1 << i;
37         }else{
38             p = tr[p][c];
39         }
40     }
41     return res;
42 }
43
44 void solve(){
45     cin >> n;
46     int ans = 0;
47     for (int i = 0; i < n; ++i){
48         int x; cin >> x;

```

```

48         ans = max(ans, query(x));
49         insert(x);
50     }
51     cout << ans << endl;
52 }
53
54 int main(){
55     ios;
56     //freopen("sample.txt", "r", stdin);
57     //freopen("resout.txt", "w", stdout);
58     int t = 1;
59     //cin >> t;
60     while(t--){
61         solve();
62     }
63     return 0;
64 }

```

● 区间异或最大值

```

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 maxm = maxn * 21;
16 const ll mod = 998244353;
17 const ll inf = 0x3f3f3f3f;
18
19 int a[maxn], s[maxn];
20 int tr[maxm][2], tot;
21
22 void insert(int x)
23 {
24     int p = 0;
25     for (int i = 20; i >= 0; --i)
26     {
27         int c = x >> i & 1;
28         if (!tr[p][c])
29             tr[p][c] = ++tot;
30         p = tr[p][c];
31     }
32 }
33
34 int query(int x)
35 {
36     int p = 0, res = 0;
37     for (int i = 20; i >= 0; --i)
38     {
39         int c = x >> i & 1;
40         if (tr[p][!c])
41         {
42             p = tr[p][!c];
43             res += 1 << i;
44         }
45         else
46             p = tr[p][c];
47     }
48     return res;
49 }
50
51 map<int, int> mp;
52

```

```

53 void solve()
54 {
55     int n, l, r;
56     cin >> n;
57     insert(0);
58     mp[0] = 0;
59     int ans = -1;
60     for (int i = 1; i <= n; ++i){
61         cin >> a[i];
62         s[i] = s[i - 1] ^ a[i];
63         insert(s[i]);
64         int tmp = query(s[i]);
65         if (tmp > ans){
66             ans = tmp;
67             r = i;
68             l = mp[tmp ^ s[i]] + 1;
69         }
70         mp[s[i]] = i;
71     }
72     cout << ans << " " << l << " " << r << endl;
73 }
74
75 int main()
76 {
77     ios;
78     // freopen("sample.txt", "r", stdin);
79     // freopen("resout.txt", "w", stdout);
80     int t = 1;
81     // cin >> t;
82     while (t--)
83     {
84         solve();
85     }
86     return 0;
87 }

```

● Border1

给一个长度为 n 的仅包含小写字母的字符串 S ，一个正整数 k ，求一个最长的字符串 T ，满足：1. T 为 S 的前缀 2. T 为 S 的后缀 3. T 在 S 中至少出现 k 次

```

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 vi G[maxn];
19 int sz[maxn];
20
21 vector<int> prefix_function(string s)
22 {
23     G[0].push_back(1);
24     int n = (int) s.length();
25     vector<int> pi(n);
26     for(int i = 2; i < n; i++) {
27         pi[i] = pi[i - 1];
28         while(pi[i] && s[i] != s[pi[i] + 1])
29             pi[i] = pi[pi[i]];
30         pi[i] += (s[i] == s[pi[i] + 1]);

```

```

31     G[pi[i]].push_back(i);
32 }
33 return pi;
34 }
35
36 void dfs(int u)
37 {
38     int sum = 1;
39     for (auto v : G[u]){
40         dfs(v);
41         sum += sz[v];
42     }
43     sz[u] = sum;
44 }
45
46 void solve()
47 {
48     int n, k;
49     cin >> n >> k;
50     string s;
51     cin >> s;
52     s = " " + s;
53     auto nxt = prefix_function(s);
54     dfs(0);
55     int u = n;
56     while (u && sz[u] < k) u = nxt[u];
57     if (!u) cout << -1 << endl;
58     else cout << s.substr(1, u) << endl;
59 }
60
61 int main()
62 {
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 }

```

● Border2

给一个长度为 n 的仅包含小写字母的字符串 S ，有 Q 次操作：1. 修改操作：1 ch 表示向字符串末尾添加一个字符 ch 2. 查询操作：2 k ，求一个最长的字符串 T 满足： T 为 S 的前缀， T 为 S 的后缀，且 T 在 S 中至少出现次

```

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 = 5e5 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17
18 vi G[maxn];
19 int tr[maxn], dfn[maxn], low[maxn], f[maxn][21], tot = 0;
20
21 int lowbit(int x)
22 {
23     return x & (-x);

```

```

24 }
25
26 vector<pair<int, int>> qry(maxn);
27
28 void add(int x, int val)
29 {
30     for(int i = x; i <= tot; i += lowbit(i))
31         tr[i] += val;
32 }
33 int query(int x)
34 {
35     int res = 0;
36     for(int i = x; i; i -= lowbit(i))
37         res += tr[i];
38     return res;
39 }
40
41 vector<int> prefix_function(string s)
42 {
43     G[0].push_back(1);
44     int n = (int) s.length();
45     vector<int> pi(n);
46     for(int i = 2; i < n; i++) {
47         pi[i] = pi[i - 1];
48         while(pi[i] && s[i] != s[pi[i] + 1])
49             pi[i] = pi[pi[i]];
50         pi[i] += (s[i] == s[pi[i] + 1]);
51         G[pi[i]].push_back(i);
52     }
53     return pi;
54 }
55
56 void dfs(int u)
57 {
58     dfn[u] = ++tot;
59     for(auto v : G[u]) {
60         f[v][0] = u;
61         for(int i = 1; i <= 20; ++i)
62             f[v][i] = f[f[v][i - 1]][i - 1];
63         dfs(v);
64     }
65     low[u] = tot;
66 }
67
68 void solve()
69 {
70     int n, q;
71     string s;
72     cin >> n >> q >> s;
73     s = " " + s;
74     for(int i = 1; i <= q; ++i) {
75         cin >> qry[i].x;
76         if(qry[i].x == 2)
77             cin >> qry[i].y;
78         else {
79             char ch;
80             cin >> ch;
81             qry[i].y = ch;
82             s += ch;
83         }
84     }
85     auto nxt = prefix_function(s);
86     dfs(0);
87     for(int i = 1; i <= n; ++i)
88         add(dfn[i], 1);
89     for(int i = 1; i <= q; ++i) {
90         if(qry[i].x == 1)
91             add(dfn[+n], 1);
92         else {
93             int cur = n;
94             for(int j = 20; j >= 0; --j) {

```

```

95         int k = qry[i].y;
96         int p = f[cur][j];
97         if(query(low[p]) - query(dfn[p] - 1) < k)
98             cur = p;
99     }
100     int ans = -1;
101     if(f[cur][0])
102         ans = f[cur][0];
103     cout << ans << endl;
104 }
105 }
106 }
107
108 int main()
109 {
110     ios;
111     // freopen("sample.txt", "r", stdin);
112     // freopen("resout.txt", "w", stdout);
113     int t = 1;
114     //cin >> t;
115     while(t--) {
116         solve();
117     }
118     return 0;
119 }

```

ACAM

- AC 自动机模板

```

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 int tr[maxn][26], cnt = 0;
19 int sz[maxn], id[maxn], fail[maxn];
20 vi G[maxn];
21 string s;
22
23 void insert(int x)
24 {
25     int p = 0;
26     for(int i = 0; i < s.size(); ++i) {
27         int c = s[i] - 'a';
28         if(!tr[p][c])
29             tr[p][c] = ++cnt;
30         p = tr[p][c];
31     }
32     id[x] = p;
33 }
34
35 void build()
36 {
37     queue<int> q;
38     for(int i = 0; i < 26; ++i)
39         if(tr[0][i])
40             q.push(tr[0][i]);
41     while(q.size()) {

```

```

42     int u = q.front();
43     q.pop();
44     for(int i = 0; i < 26; ++i) {
45         int &v = tr[u][i];
46         if(v) {
47             fail[v] = tr[fail[u]][i];
48             q.push(tr[u][i]);
49         }
50         else
51             v = tr[fail[u]][i];
52     }
53 }
54 }
55
56 void dfs(int u)
57 {
58     for(auto v : G[u]) {
59         dfs(v);
60         sz[u] += sz[v];
61     }
62 }
63
64 void solve()
65 {
66     int n;
67     cin >> n;
68     for(int i = 1; i <= n; ++i) {
69         cin >> s;
70         insert(i);
71     }
72     build();
73     cin >> s;
74     int p = 0;
75     for(auto c : s) {
76         p = tr[p][c - 'a'];
77         sz[p]++;
78     }
79     for(int i = 1; i <= cnt; ++i)
80         G[fail[i]].push_back(i);
81     dfs(0);
82     for(int i = 1; i <= n; ++i)
83         cout << sz[id[i]] << endl;
84 }
85
86 int main()
87 {
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 }

```

● 单词 (出现了多少次)

```

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  int tr[maxn][26], cnt = 0;
19  int sz[maxn], id[maxn], fail[maxn], ed[maxn], ans[maxn];
20  vi G[maxn];
21  string s;
22
23  void insert(int x)
24  {
25      int p = 0;
26      for (int i = 0; i < s.size(); ++i) {
27          int c = s[i] - 'a';
28          if (!tr[p][c])
29              tr[p][c] = ++cnt;
30          p = tr[p][c];
31          sz[p]++;
32      }
33      id[x] = p;
34      ed[p] = s.size();
35  }
36
37  void build()
38  {
39      queue<int> q;
40      for (int i = 0; i < 26; ++i)
41          if (tr[0][i])
42              q.push(tr[0][i]);
43      while (q.size()) {
44          int u = q.front();
45          q.pop();
46          for (int i = 0; i < 26; ++i) {
47              int &v = tr[u][i];
48              if (v) {
49                  fail[v] = tr[fail[u]][i];
50                  q.push(tr[u][i]);
51              }
52              else
53                  v = tr[fail[u]][i];
54          }
55      }
56  }
57
58  void dfs(int u)
59  {
60      for(auto v : G[u]) {
61          dfs(v);
62          sz[u] += sz[v];
63      }
64  }
65
66  void solve() {
67      int n;
68      cin >> n;
69      for (int i = 1; i <= n; ++i) {
70          cin >> s;
71          insert(i);
72      }
73      build();
74      for (int i = 1; i <= cnt; ++i)
75          G[fail[i]].push_back(i);
76      dfs(0);
77      for (int i = 1; i <= n; ++i)
78          cout << sz[id[i]] << endl;
79  }
80
81  int main() {
82      ios;
83      //freopen("sample.txt", "r", stdin);
84      //freopen("resout.txt", "w", stdout);

```

```

85     int t = 1;
86     //cin >> t;
87     while (t--) {
88         solve();
89     }
90     return 0;
91 }

```

- 文本生成器

长度为 n 的串中，出现任一给字符串的个数的方案书

```

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 = 4e6 + 10;
15 const ll mod = 10007;
16 const ll inf = 0x3f3f3f3f;
17
18 int tr[maxn][26], tot;
19 int fail[maxn];
20 vi G[maxn];
21 int ok[maxn];
22 int f[10010][6010];
23
24 void insert(string s)
25 {
26     int p = 0;
27     for (int i = 0; i < s.size(); ++i)
28     {
29         int c = s[i] - 'A';
30         if (!tr[p][c])
31             tr[p][c] = ++tot;
32         p = tr[p][c];
33     }
34     ok[p] = 1;
35 }
36
37 void build()
38 {
39     queue<int> q;
40     for (int i = 0; i < 26; ++i)
41         if (tr[0][i])
42             q.push(tr[0][i]);
43     while (!q.empty())
44     {
45         auto u = q.front();
46         q.pop();
47         for (int i = 0; i < 26; ++i){
48             auto &v = tr[u][i];
49             if (v){
50                 fail[v] = tr[fail[u]][i];
51                 q.push(v);
52             }else
53                 v = tr[fail[u]][i];
54         }
55     }
56 }
57
58 void dfs(int u){
59     for (auto v : G[u]){
60         if (ok[u]) ok[v] = 1;

```

```

61     dfs(v);
62 }
63 }
64
65 void solve()
66 {
67     int n, m;
68     cin >> n >> m;
69     for (int i = 1; i <= n; ++i){
70         string s;
71         cin >> s;
72         insert(s);
73     }
74     build();
75     for (int i = 1; i <= tot; ++i) G[fail[i]].push_back(i);
76     dfs(0);
77     f[0][0] = 1;
78     for (int i = 0; i <= m; ++i){
79         for (int j = 0; j <= tot; ++j){
80             for (int k = 0; k < 26; ++k){
81                 if (!ok[tr[j][k]]) f[i + 1][tr[j][k]] = (f[i + 1][tr[j][k]] + f[i][j]) % mod;
82             }
83         }
84     }
85     ll ans = 0;
86     for (int i = 0; i <= tot; ++i)
87         if (!ok[i]) ans += f[m][i];
88     ll sum = 1;
89     for (int i = 1; i <= m; ++i)
90         sum = sum * 26 % mod;
91     cout << ((sum - ans) % mod + mod) % mod;
92 }
93
94 int main()
95 {
96     ios;
97     // freopen("sample.txt", "r", stdin);
98     // freopen("resout.txt", "w", stdout);
99     int t = 1;
100    // cin >> t;
101    while (t--)
102    {
103        solve();
104    }
105    return 0;
106 }

```

manacher

- 查回文

给出 l, r 求 l, r 区间内满足点对下字符串为回文串的方案点对数

位于左半个区间的回文中心，在延伸的过程中只可能被区间的左边界截断，位于右半个区间的，只可能被区间的右边界截断。对于每个 i ，我给区间 $[i - r[i] + 1, i]$ 这些位置加一，然后我要求的东西就转化成了区间 $[l, +\infty)$ 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 iinf 0x3f3f3f3f
10 #define linf (1ll << 60)
11 #define rep(i, j, k) for (int i = (j); i <= (k); i++)
12 #define per(i, j, k) for (int i = (j); i >= (k); i--)
13 #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
14 using namespace std;
15 typedef long long ll;

```

```

16  const ll maxn = 4e5 + 10;
17  const ll mod = 998244353;
18  const ll inf = 0x3f3f3f3f;
19
20  struct Manacher
21  {
22      int r[maxn], p[maxn], n;
23      void clear() { memset(r, 0, sizeof r), memset(p, 0, sizeof p); }
24      void calc(string s, int N)
25      {
26          n = N;
27          int i, j, mx(0), center;
28          r[0] = -2;
29          for (i = 1; i <= N; i++)
30              r[2 * i] = s[i];
31          for (i = 1; i <= N; i++)
32              r[2 * i - 1] = -1;
33          r[2 * N + 1] = -1;
34          for (i = 1; i <= 2 * N + 1; i++)
35          {
36              if (mx >= i)
37                  p[i] = min(p[2 * center - i], mx - i + 1);
38              else
39                  p[i] = 1;
40              while (r[i - p[i]] == r[i + p[i]])
41                  p[i]++;
42              if (i + p[i] - 1 > mx)
43              {
44                  mx = i + p[i] - 1;
45                  center = i;
46              }
47          }
48      }
49  } mnc;
50  ll n, id[maxn], l[maxn], r[maxn], ans[maxn], q;
51  struct SegmentTree
52  {
53      ll mn[maxn << 2], mx[maxn << 2], sum[maxn << 2], add[maxn << 2], set[maxn << 2], L[maxn << 2], R[maxn << 2];
54      void maketag_set(ll o, ll v)
55      {
56          add[o] = 0;
57          set[o] = v;
58          mx[o] = mn[o] = v;
59          sum[o] = (R[o] - L[o] + 1) * v;
60      }
61      void maketag_add(ll o, ll v)
62      {
63          add[o] += v;
64          mx[o] += v, mn[o] += v;
65          sum[o] += (R[o] - L[o] + 1) * v;
66      }
67      void pushdown(ll o)
68      {
69          if (L[o] == R[o])
70              return;
71          if (~set[o])
72          {
73              maketag_set(o << 1, set[o]);
74              maketag_set(o << 1 | 1, set[o]);
75              set[o] = -1;
76          }
77          if (add[o])
78          {
79              maketag_add(o << 1, add[o]);
80              maketag_add(o << 1 | 1, add[o]);
81              add[o] = 0;
82          }
83      }
84      void pushup(ll o)
85      {
86          mx[o] = max(mx[o << 1], mx[o << 1 | 1]);

```

```

87     mn[o] = min(mn[o << 1], mn[o << 1 | 1]);
88     sum[o] = sum[o << 1] + sum[o << 1 | 1];
89 }
90 void build(ll o, ll l, ll r, ll *array = NULL)
91 {
92     ll mid(l + r >> 1);
93     L[o] = l, R[o] = r;
94     add[o] = 0;
95     set[o] = -1;
96     if (l == r)
97     {
98         if (array)
99             mn[o] = mx[o] = sum[o] = array[l];
100        else
101            mn[o] = mx[o] = sum[o] = 0;
102        return;
103    }
104    build(o << 1, l, mid, array);
105    build(o << 1 | 1, mid + 1, r, array);
106    pushup(o);
107 }
108 void Set(ll o, ll l, ll r, ll v)
109 {
110     ll mid(L[o] + R[o] >> 1);
111     if (l <= L[o] and r >= R[o])
112     {
113         maketag_set(o, v);
114         return;
115     }
116     pushdown(o);
117     if (l <= mid)
118         Set(o << 1, l, r, v);
119     if (r > mid)
120         Set(o << 1 | 1, l, r, v);
121     pushup(o);
122 }
123 void Add(ll o, ll l, ll r, ll v)
124 {
125     ll mid(L[o] + R[o] >> 1);
126     if (l <= L[o] and r >= R[o])
127     {
128         maketag_add(o, v);
129         return;
130     }
131     pushdown(o);
132     if (l <= mid)
133         Add(o << 1, l, r, v);
134     if (r > mid)
135         Add(o << 1 | 1, l, r, v);
136     pushup(o);
137 }
138 ll Sum(ll o, ll l, ll r)
139 {
140     pushdown(o);
141     ll mid(L[o] + R[o] >> 1), ans(0);
142     if (l <= L[o] and r >= R[o])
143         return sum[o];
144     if (l <= mid)
145         ans += Sum(o << 1, l, r);
146     if (r > mid)
147         ans += Sum(o << 1 | 1, l, r);
148     return ans;
149 }
150 ll Min(ll o, ll l, ll r)
151 {
152     ll mid(L[o] + R[o] >> 1), ans(linf);
153     if (l <= L[o] and r >= R[o])
154         return mn[o];
155     pushdown(o);
156     if (l <= mid)
157         ans = min(ans, Min(o << 1, l, r));

```

```

158     if (r > mid)
159         ans = min(ans, Min(o << 1 | 1, l, r));
160     return ans;
161 }
162 ll Max(ll o, ll l, ll r)
163 {
164     ll mid(L[o] + R[o] >> 1), ans(-1inf);
165     if (l <= L[o] and r >= R[o])
166         return mx[o];
167     pushdown(o);
168     if (l <= mid)
169         ans = max(ans, Max(o << 1, l, r));
170     if (r > mid)
171         ans = max(ans, Max(o << 1 | 1, l, r));
172     return ans;
173 }
174 } segtree;
175
176 void solve()
177 {
178     cin >> n >> q;
179     string s;
180     cin >> s;
181     s = " " + s;
182     mnc.calc(s, n);
183     for (int i = 1; i <= q; ++i){
184         cin >> l[i] >> r[i];
185         l[i] = 2 * l[i] - 1;
186         r[i] = 2 * r[i] + 1;
187         id[i] = i;
188     }
189     sort(id + 1, id + q + 1, [&](ll a, ll b){return l[a] + r[a] < l[b] + r[b];});
190     segtree.build(1, 1, 2 * n + 1);
191     int j = 1;
192     for (int i = 1; i <= q; ++i){
193         while(j <= (l[id[i]] + r[id[i]] >> 1)){
194             segtree.Add(1, j - mnc.p[j] + 1, j, 1);
195             ++j;
196         }
197         ans[id[i]] += segtree.Sum(1, l[id[i]], 2 * n);
198     }
199     segtree.build(1, 1, 2 * n + 1);
200     j = 2 * n + 1;
201     for (int i = q; i >= 1; --i){
202         while (j > (l[id[i]] + r[id[i]] >> 1)){
203             segtree.Add(1, j, j + mnc.p[j] - 1, 1);
204             --j;
205         }
206         ans[id[i]] += segtree.Sum(1, 1, r[id[i]]);
207     }
208     for (int i = 1; i <= q; ++i){
209         ans[i] -= (r[i] + 1 >> 1) - (l[i] >> 1);
210         cout << ans[i] / 2 << endl;
211     }
212 }
213
214 int main()
215 {
216     ios;
217     // freopen("sample.txt", "r", stdin);
218     // freopen("resout.txt", "w", stdout);
219     int t = 1;
220     //cin >> t;
221     while (t--)
222     {
223         solve();
224     }
225     return 0;
226 }

```

● 拉拉队排练

按照女生的个数降序排序之后，前 K 个和谐小群体的女生个数的乘积是多少。由于答案可能很大，只要你告诉她，答案除以 19930726 的余数是多少就行了

```
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 inf = 0x3f3f3f3f;
14 const int N = 2e6 + 10;
15 const int mod = 19930726;
16 string s;
17 int d[N];
18 int mp[N];
19
20 int qmi(int a, int n)
21 {
22     int res = 1;
23     a %= mod;
24     while (n)
25     {
26         if (n & 1) res = res * a % mod;
27         a = a * a % mod;
28         n >>= 1;
29     }
30     return res;
31 }
32
33 void manacher(int n)
34 {
35     d[1] = 1;
36
37     for (int i = 2, l, r = 1; i <= n; i++)
38     {
39         if (r >= i) d[i] = min(r - i + 1, d[r - i + 1]); // 在加速盒子内
40         while (s[i - d[i]] == s[i + d[i]]) d[i]++; // 盒外暴力
41         if (i + d[i] - 1 > r) r = i + d[i] - 1, l = i - d[i] + 1; // 更新加速盒子（根据有边界）
42         mp[d[i] * 2 - 1]++;
43     }
44 }
45
46 signed main()
47 {
48     ios;
49     int n, k;
50     cin >> n >> k >> s;
51     s = '@' + s + '.';
52     manacher(n);
53
54     if (n % 2 == 0) n--;
55
56     int ans = 1, sum = 0;
57     for (int i = n; i > 0; i -= 2)
58     {
59         sum += mp[i];
60         if (k < sum)
61         {
62             ans = ans * qmi(i, k) % mod;
63             k -= sum;
64             break;
65         }
66         else
67         {
68             ans = ans * qmi(i, sum) % mod;
```

```

69         k -= sum;
70     }
71 }
72 }
73 if (k > 0) cout << -1 << endl;
74 else cout << ans << endl;
75
76 return 0;
77 }

```

● 最长双回文串

```

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 #define int ll
15 const ll maxn = 1e6 + 10;
16 const ll mod = 998244353;
17 const ll inf = 0x3f3f3f3f;
18
19 char s[maxn];
20 int p[maxn], L[maxn], R[maxn];
21 int n;
22 void Manacher(string t)
23 {
24     s[0] = '@', s[1] = '#';
25     int cnt = 1;
26     for (auto x : t)
27     {
28         s[++cnt] = x;
29         s[++cnt] = '#';
30     }
31     n = cnt;
32     for (int i = 1, mid = 0, r = 0; i <= n; i++)
33     {
34         if (i <= r) p[i] = min(p[2 * mid - i], r - i + 1);
35         while (s[i - p[i]] == s[i + p[i]]) p[i]++;
36         if (i + p[i] > r) r = i + p[i] - 1, mid = i;
37         int l = i - p[i] - 1;
38         int rr = i - p[i] + 1;
39         R[rr] = max(R[rr], p[i] - 1); // 以 rr 为回文串右端点的最长回文串
40         L[l] = max(L[l], p[i] - 1); // 以 l 为回文串左端点的最长回文串
41     }
42 }
43
44 void solve(){
45     string t;
46     cin >> t;
47     n = t.length();
48     Manacher(t);
49     ll ans = 0;
50     for (int i = 3; i <= n; i += 2) R[i] = max(R[i], R[i - 2] - 2);
51     for (int i = n - 1; i >= 1; i -= 2) L[i] = max(L[i], L[i + 2] - 2);
52     for (int i = 1; i <= n; i += 2) if (R[i] && L[i]) ans = max(ans, 1ll * (L[i] + R[i]));
53     cout << ans << endl;
54 }
55
56 signed main(){
57     ios;
58     //freopen("sample.txt", "r", stdin);
59     //freopen("resout.txt", "w", stdout);
60     int t = 1;

```



```

61     //cin >> t;
62     while(t--){
63         solve();
64     }
65     return 0;
66 }

```

pam

- 本质不同回文字串个数

一个串的本质不同回文子串个数等于回文树的状态数 (排除奇根和偶根两个状态)

- 回文子串出现次数

(最大 (回文字串出现的次数 * 回文子串的长度))

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  const int maxn = 300000 + 5;
4  typedef long long ll;
5  namespace pam {
6  int sz, tot, last;
7  int cnt[maxn], ch[maxn][26], len[maxn], fail[maxn];
8  char s[maxn];
9
10 int node(int l) { // 建立一个新节点, 长度为 l
11     sz++;
12     memset(ch[sz], 0, sizeof(ch[sz]));
13     len[sz] = l;
14     fail[sz] = cnt[sz] = 0;
15     return sz;
16 }
17
18 void clear() { // 初始化
19     sz = -1;
20     last = 0;
21     s[tot = 0] = '$';
22     node(0);
23     node(-1);
24     fail[0] = 1;
25 }
26
27 int getfail(int x) { // 找后缀回文
28     while (s[tot - len[x] - 1] != s[tot]) x = fail[x];
29     return x;
30 }
31
32 void insert(char c) { // 建树
33     s[++tot] = c;
34     int now = getfail(last);
35     if (!ch[now][c - 'a']) {
36         int x = node(len[now] + 2);
37         fail[x] = ch[getfail(fail[now])][c - 'a'];
38         ch[now][c - 'a'] = x;
39     }
40     last = ch[now][c - 'a'];
41     cnt[last]++;
42 }
43
44 ll solve() {
45     ll ans = 0;
46     for (int i = sz; i >= 0; i--) {
47         cnt[fail[i]] += cnt[i];
48     }
49     for (int i = 1; i <= sz; i++) { // 更新答案
50         ans = max(ans, 1ll * len[i] * cnt[i]);
51     }
52     return ans;
53 }
54 } // namespace pam

```

```

55
56 char s[maxn];
57
58 int main() {
59     int n;
60     cin >> n;
61     pam::clear();
62     scanf("%s", s + 1);
63     for (int i = 1; s[i]; i++) {
64         pam::insert(s[i]);
65     }
66     printf("%lld\n", pam::solve());
67     return 0;
68 }

```

● 例题 (Colourful String)

子字符串不同颜色的数量的和

```

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 = 1e6 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17
18
19 int n;
20 string s;
21
22 struct PAM
23 {
24     int last, idx;
25     vector<array<int, 26>> tr;
26     vector<int> fail, len, cnt, mask;
27
28     PAM(): last(0), idx(0) {
29         fail.resize(n + 2), len.resize(n + 2), tr.resize(n + 2), mask.resize(n + 2), cnt.resize(n + 2);
30     }
31     int newnode(int l) { //新增一个结点, 长度为 l
32         len[idx] = l;
33         tr[idx].fill(0);
34         return idx++;
35     }
36     void init() {
37         idx = last = 0;
38         newnode(0), newnode(-1); //偶根长度为 0, 奇根长度为-1
39         s[0] = -1, fail[0] = 1; //偶根的失配边指向奇根, 奇根的失配边指向偶根
40     }
41     int get_fail(int p, int i) {
42         while (s[i - len[p] - 1] != s[i]) p = fail[p];
43         return p;
44     }
45     void insert(int i) {
46         int u = s[i] - 'a', p = get_fail(last, i);
47         if (!tr[p][u]) {
48             int now = newnode(len[p] + 2);
49             mask[now] = mask[p] | (1 << u);
50             fail[now] = tr[get_fail(fail[p], i)][u];
51             tr[p][u] = now;
52         }
53         last = tr[p][u];

```

```

54     cnt[last]++;
55 }
56 };
57
58
59 void solve() {
60     cin >> s;
61     n = s.length();
62     s = " " + s;
63     PAM pam;
64     pam.init();
65     for (int i = 1; i <= n; ++i) pam.insert(i);
66     for (int i = pam.idx - 1; ~i; --i) pam.cnt[pam.fail[i]] += pam.cnt[i];
67     ll res = 0;
68     for (int i = 2; i < pam.idx; ++i) res += pam.cnt[i] * __builtin_popcount(pam.mask[i]);
69     cout << res << endl;
70 }
71
72 int main() {
73     ios;
74     //freopen("sample.txt", "r", stdin);
75     //freopen("resout.txt", "w", stdout);
76     int t = 1;
77     //cin >> t;
78     while (t--) {
79         solve();
80     }
81     return 0;
82 }

```

SA

- $n \log^2 n$ (倍增法)

```

1  #include <algorithm>
2  #include <cstdio>
3  #include <cstring>
4  #include <iostream>
5
6  using namespace std;
7
8  const int N = 1000010;
9
10 char s[N];
11 int n, w, sa[N], rk[N << 1], oldrk[N << 1];
12
13 // 为了防止访问 rk[i+w] 导致数组越界，开两倍数组。
14 // 当然也可以在访问前判断是否越界，但直接开两倍数组方便一些。
15
16 int main() {
17     int i, p;
18
19     scanf("%s", s + 1);
20     n = strlen(s + 1);
21     for (i = 1; i <= n; ++i) sa[i] = i, rk[i] = s[i];
22
23     for (w = 1; w < n; w <= 1) {
24         sort(sa + 1, sa + n + 1, [](int x, int y) {
25             return rk[x] == rk[y] ? rk[x + w] < rk[y + w] : rk[x] < rk[y];
26         }); // 这里用到了 lambda
27         memcpy(oldrk, rk, sizeof(rk));
28         // 由于计算 rk 的时候原来的 rk 会被覆盖，要先复制一份
29         for (p = 0, i = 1; i <= n; ++i) {
30             if (oldrk[sa[i]] == oldrk[sa[i - 1]] &&
31                 oldrk[sa[i] + w] == oldrk[sa[i - 1] + w]) {
32                 rk[sa[i]] = p;
33             } else {
34                 rk[sa[i]] = ++p;
35             } // 若两个子串相同，它们对应的 rk 也需要相同，所以要去重
36         }
37     }
38 }

```

```

38     for (i = 1; i <= n; ++i) printf("%d ", sa[i]);
39
40     return 0;
41 }
42
43 • SA-IS
44
45 // 后缀类型
46 #define L_TYPE 0
47 #define S_TYPE 1
48
49 // 判断一个字符是否为 LMS 字符
50 inline bool is_lms_char(int *type, int x) {
51     return x > 0 && type[x] == S_TYPE && type[x - 1] == L_TYPE;
52 }
53
54 // 判断两个 LMS 子串是否相同
55 inline bool equal_substring(int *S, int x, int y, int *type) {
56     do {
57         if (S[x] != S[y])
58             return false;
59         x++, y++;
60     } while (!is_lms_char(type, x) && !is_lms_char(type, y));
61
62     return S[x] == S[y];
63 }
64
65 // 诱导排序 (从 * 型诱导到 L 型、从 L 型诱导到 S 型)
66 // 调用之前应将 * 型按要求放入 SA 中
67 inline void induced_sort(int *S, int *SA, int *type, int *bucket, int *lbucket,
68     int *sbucket, int n, int SIGMA) {
69     for (int i = 0; i <= n; i++)
70         if (SA[i] > 0 && type[SA[i] - 1] == L_TYPE)
71             SA[lbucket[S[SA[i] - 1]]++] = SA[i] - 1;
72     for (int i = 1; i <= SIGMA; i++) // Reset S-type bucket
73         sbucket[i] = bucket[i] - 1;
74     for (int i = n; i >= 0; i--)
75         if (SA[i] > 0 && type[SA[i] - 1] == S_TYPE)
76             SA[sbucket[S[SA[i] - 1]]--] = SA[i] - 1;
77 }
78
79 // SA-IS 主体
80 // S 是输入字符串, length 是字符串的长度, SIGMA 是字符集的大小
81 static int *SAIS(int *S, int length, int SIGMA) {
82     int n = length - 1;
83     int *type = new int[n + 1]; // 后缀类型
84     int *position = new int[n + 1]; // 记录 LMS 子串的起始位置
85     int *name = new int[n + 1]; // 记录每个 LMS 子串的新名称
86     int *SA = new int[n + 1]; // SA 数组
87     int *bucket = new int[SIGMA + 1]; // 每个字符的桶
88     int *lbucket = new int[SIGMA + 1]; // 每个字符的 L 型桶的起始位置
89     int *sbucket = new int[SIGMA + 1]; // 每个字符的 S 型桶的起始位置
90
91     // 初始化每个桶
92     memset(bucket, 0, sizeof(int) * (SIGMA + 1));
93     for (int i = 0; i <= n; i++)
94         bucket[S[i]]++;
95     lbucket[0] = sbucket[0] = 0;
96     for (int i = 1; i <= SIGMA; i++) {
97         bucket[i] += bucket[i - 1];
98         lbucket[i] = bucket[i - 1];
99         sbucket[i] = bucket[i] - 1;
100     }
101
102     // 确定后缀类型 (利用引理 2.1)
103     type[n] = S_TYPE;
104     for (int i = n - 1; i >= 0; i--) {
105         if (S[i] < S[i + 1])
106             type[i] = S_TYPE;
107         else if (S[i] > S[i + 1])
108             type[i] = L_TYPE;
109     }

```

```

65         else
66             type[i] = type[i + 1];
67     }
68
69     // 寻找每个 LMS 子串
70     int cnt = 0;
71     for (int i = 1; i <= n; i++)
72         if (type[i] == S_TYPE && type[i - 1] == L_TYPE)
73             position[cnt++] = i;
74
75     // 对 LMS 子串进行排序
76     fill(SA, SA + n + 1, -1);
77     for (int i = 0; i < cnt; i++)
78         SA[sbucket[S[position[i]]]--] = position[i];
79     induced_sort(S, SA, type, bucket, lbucket, sbucket, n, SIGMA);
80
81     // 为每个 LMS 子串命名
82     fill(name, name + n + 1, -1);
83     int lastx = -1, namecnt = 1; // 上一次处理的 LMS 子串与名称的计数
84     bool flag = false; // 这里顺便记录是否有重复的字符
85     for (int i = 1; i <= n; i++) {
86         int x = SA[i];
87
88         if (is_lms_char(type, x)) {
89             if (lastx >= 0 && !equal_substring(S, x, lastx, type))
90                 namecnt++;
91             // 因为只有相同的 LMS 子串才会有同样的名称
92             if (lastx >= 0 && namecnt == name[lastx])
93                 flag = true;
94
95             name[x] = namecnt;
96             lastx = x;
97         }
98     } // for
99     name[n] = 0;
100
101     // 生成 S1
102     int *S1 = new int[cnt];
103     int pos = 0;
104     for (int i = 0; i <= n; i++)
105         if (name[i] >= 0)
106             S1[pos++] = name[i];
107
108     int *SA1;
109     if (!flag) {
110         // 直接计算 SA1
111         SA1 = new int[cnt + 1];
112
113         for (int i = 0; i < cnt; i++)
114             SA1[S1[i]] = i;
115     } else
116         SA1 = SAIS(S1, cnt, namecnt); // 递归计算 SA1
117
118     // 从 SA1 诱导到 SA
119     lbucket[0] = sbucket[0] = 0;
120     for (int i = 1; i <= SIGMA; i++) {
121         lbucket[i] = bucket[i - 1];
122         sbucket[i] = bucket[i] - 1;
123     }
124     fill(SA, SA + n + 1, -1);
125     for (int i = cnt - 1; i >= 0; i--) // 这里是逆序扫描 SA1, 因为 SA 中 S 型桶是倒序的
126         SA[sbucket[S[position[SA1[i]]]--] = position[SA1[i]];
127     induced_sort(S, SA, type, bucket, lbucket, sbucket, n, SIGMA);
128
129     // 后缀数组计算完毕
130     return SA;
131 }

```

- 从字符串首尾取字符最小化字典序

暴力做法就是每次最坏 $O(n)$ 地判断当前应该取首还是尾（即比较取首得到的字符串与取尾得到的反串的大小），只需优化这一判断过程

即可。由于需要在原串后缀与反串后缀构成的集合内比较大小，可以将反串拼接在原串后，并在中间加上一个没出现过的字符（如 #，代码中可以直接使用空字符），求后缀数组，即可 $O(1)$ 完成这一判断。

```
1  #include <cctype>
2  #include <cstdio>
3  #include <cstring>
4  #include <iostream>
5
6  using namespace std;
7
8  const int N = 1000010;
9
10 char s[N];
11 int n, sa[N], id[N], oldrk[N * 2], rk[N * 2], px[N], cnt[N];
12
13 bool cmp(int x, int y, int w) {
14     return oldrk[x] == oldrk[y] && oldrk[x + w] == oldrk[y + w];
15 }
16
17 int main() {
18     int i, w, m = 200, p, l = 1, r, tot = 0;
19
20     cin >> n;
21     r = n;
22
23     for (i = 1; i <= n; ++i)
24         while (!isalpha(s[i] = getchar()))
25             ;
26     for (i = 1; i <= n; ++i)
27         rk[i] = rk[2 * n + 2 - i] = s[i]; // 拼接正反两个字符串，中间空出一个字符
28
29     n = 2 * n + 1;
30     // 求后缀数组
31     for (i = 1; i <= n; ++i) ++cnt[rk[i]];
32     for (i = 1; i <= m; ++i) cnt[i] += cnt[i - 1];
33     for (i = n; i >= 1; --i) sa[cnt[rk[i]]--] = i;
34
35     for (w = 1; w < n; w *= 2, m = p) { // m=p 就是优化计数排序值域
36         for (p = 0, i = n; i > n - w; --i) id[++p] = i;
37         for (i = 1; i <= n; ++i)
38             if (sa[i] > w) id[++p] = sa[i] - w;
39         memset(cnt, 0, sizeof(cnt));
40         for (i = 1; i <= n; ++i) ++cnt[px[i] = rk[id[i]]];
41         for (i = 1; i <= m; ++i) cnt[i] += cnt[i - 1];
42         for (i = n; i >= 1; --i) sa[cnt[px[i]]--] = id[i];
43         memcpy(oldrk, rk, sizeof(rk));
44         for (p = 0, i = 1; i <= n; ++i)
45             rk[sa[i]] = cmp(sa[i], sa[i - 1], w) ? p : ++p;
46     }
47     // 利用后缀数组  $O(1)$  进行判断
48     while (l <= r) {
49         printf("%c", rk[l] < rk[n + 1 - r] ? s[l++] : s[r--]);
50         if ((++tot) % 80 == 0) puts(""); // 回车
51     }
52
53     return 0;
54 }
```

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 }

```

```

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 }

```

位运算基础

```

1  去掉最后一位
2  x >> 1
3  在最后一位加个 0
4  x << 1
5  在最后一位加个 1
6  (x << 1) | 1
7  把最后一位变成 1
8  x | 1
9  把最后一位变成 0
10 (x | 1) & 1
11 最后一位取反
12 x ^ 1
13 把右数第 k 位变成 1
14 x | (1 << (k & 1))
15 把右数第 k 位变成 0
16 x & (~(1 << (k & 1)))
17 右数第 k 位取反
18 x ^ (1 << (k & 1))
19 取末 k 位
20 x & ((1 << k) & 1)
21 取右数第 k 位
22 (x >> (k & 1)) & 1
23 把末 k 位变成 1
24 x | ((1 << k) & 1)

```

```

25 把右边连续的 1 变成 0
26 x & (x + 1)
27 把右边第一个 0 变成 1
28 x | (x + 1)
29 取右边连续的 1
30 (x ^ (x + 1)) >> 1
31 去掉右起第一个 1 的左边
32 x & (-x)

```

虚拟源点

- 843div2D

给定 n 个点，每个点的权值为 a_i 。两个位置 i, j 存在一个长度为 1 的边当且仅当 $\gcd(a_i, a_j) > 1$ 。求 S 到 T 的最短路

```

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 = 6e5 + 10;
15 const ll mod = 998244353;
16 const ll inf = 0x3f3f3f3f;
17
18 int prime[maxn], cnt = 0;
19 bool vis[maxn];
20 int minp[maxn];
21 int idx[maxn];
22
23 void init(int n) {
24     for (int i = 2; i <= n; ++i) {
25         if (vis[i] == false) {
26             prime[++cnt] = i;
27             minp[i] = i;
28             idx[i] = cnt;
29         }
30         for (int j = 1; j <= cnt && i * prime[j] <= n; ++j) {
31             minp[i * prime[j]] = prime[j];
32             vis[i * prime[j]] = 1;
33             if (i % prime[j] == 0) break;
34         }
35     }
36 }
37
38 int a[maxn];
39 set<int> e[maxn];
40 ll dis[maxn];
41 priority_queue<pair<int, int>> q;
42 int vs[maxn], pre[maxn];
43
44 void dij(int s, int exn, int n) {
45     rep(i, 1, exn) dis[i] = 1e18;
46     dis[s] = 0;
47     q.push({0, s});
48     while (!q.empty()) {
49         pair<int, int> cur = q.top();
50         q.pop();
51         if (vs[cur.y]) continue;
52         int u = cur.y;
53         vs[u] = 1;
54         for (auto v : e[u]) {
55             int w = 1;
56             if (v > n) w = 0;

```

```

57         if (dis[v] > dis[u] + w) {
58             dis[v] = dis[u] + w;
59             pre[v] = u;
60             if (!vs[v]) q.push({ -dis[v], v});
61         }
62     }
63 }
64 }
65
66 void solve() {
67     int n;
68     cin >> n;
69     for (int i = 1; i <= n; ++i) cin >> a[i];
70     int s, t;
71     cin >> s >> t;
72     int exn = n;
73     for (int i = 1; i <= n; ++i) {
74         int tmp = a[i];
75         while (tmp > 1) {
76             int tar = idx[minp[tmp]];
77             exn = max(exn, n + tar);
78             e[n + tar].insert(i);
79             e[i].insert(n + tar);
80             tmp /= minp[tmp];
81         }
82     }
83     dij(s, exn, n);
84     if (dis[t] == 1e18) cout << -1 << endl;
85     else {
86         vector<int> ans;
87         int tmp = t;
88         while (tmp != s && tmp != 0) {
89             if (tmp <= n) ans.push_back(tmp);
90             tmp = pre[tmp];
91         }
92         ans.push_back(s);
93         reverse(ans.begin(), ans.end());
94         cout << ans.size() << endl;
95         for (auto it : ans) cout << it << " ";
96         cout << endl;
97     }
98 }
99 int main() {
100     ios;
101     init(3e5 + 10);
102     //freopen("sample.txt", "r", stdin);
103     //freopen("resout.txt", "w", stdout);
104     int t = 1;
105     //cin >> t;
106     while (t--) {
107         solve();
108     }
109     return 0;
110 }

```

简单环

```

1  #include <bits/stdc++.h>
2  #define ll long long
3  using namespace std;
4  const int MOD = 998244353;
5  const int maxn = 25;
6  ll ans[maxn], dp[1 << 20][maxn], num[1 << 20];
7  int n, m, k;
8  bool vis[maxn][maxn];
9  ll ksm(ll a, ll b)
10 {
11     if (b == 0)
12         return 1;
13     if (b == 1)
14         return a % MOD;

```

```

15     ll mid = ksm(a, b >> 1);
16     if (b & 1)
17         return mid * mid % MOD * a % MOD;
18     else
19         return mid * mid % MOD;
20 }
21 void Init()
22 {
23     for (int st = 0; st < (1 << n); st++)
24     {
25         int cur = st, cnt = 0;
26         while (cur)
27         {
28             if (cur & 1)
29                 cnt++;
30             cur >>= 1;
31         }
32         num[st] = cnt;
33     }
34 }
35 int main()
36 {
37     scanf("%d%d%d", &n, &m, &k);
38     Init();
39     for (int i = 1; i <= m; i++)
40     {
41         int u, v;
42         scanf("%d%d", &u, &v);
43         vis[u][v] = true, vis[v][u] = true;
44     }
45     for (int i = 1; i <= n; i++)
46         dp[1 << (i - 1)][i] = 1;
47     for (int st = 1; st < (1 << n); st++)
48     {
49         int lowbit = st & (-st), s = 0;
50         while (lowbit)
51             s++, lowbit >>= 1;
52         for (int j = 1; j <= n; j++)
53             if (dp[st][j])
54             {
55                 if (vis[j][s] && num[st] > 2)
56                     ans[num[st] % k] = (ans[num[st] % k] + dp[st][j]) % MOD;
57                 for (int k = s + 1; k <= n; k++)
58                     if ((st & (1 << (k - 1))) == 0 && vis[j][k])
59                     {
60                         int p = st | (1 << (k - 1));
61                         dp[p][k] = (dp[p][k] + dp[st][j]) % MOD;
62                     }
63             }
64     }
65     ll invtwo = ksm(2, MOD - 2);
66     for (int i = 1; i <= k; i++)
67         cout << ans[i - 1] * invtwo % MOD << endl;
68     return 0;
69 }

```

数位 dp

```

1  /*
2   * 第一行，一个整数  $T$ ，代表数据组数对于每组数据，
3   * 有三个数字  $l, r, n$ 
4   * 接下来  $n$  行，每行一个数字  $x$ ，接下来一个数  $len$  表示数字  $x$  在数字串中连续出现的次数不能大于  $len$ 
5   * 对于每组数据
6   * 输出一个整数，表示  $l, r$  中满足约束的数字个数。(对 20020219 取模)
7   */
8  #include <bits/stdc++.h>
9  #define endl '\n'
10 #define pll pair<ll, ll>
11 #define tll tuple<ll, ll, ll>
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 maxn = 20;
16 const ll mod = 20020219;
17 ll len[maxn];
18 ll a[maxn];
19 ll f[maxn][maxn][maxn];
20 ll l, r, n;
21
22 void solve()
23 {
24     // flag 表示是否能直接返回值, 也就是说前 pos-1 位与原数是否不同
25     // 相同则这一位收到限制需要继续递归求解
26     // 不同则不受限制, 如果之前算过了可以直接返回
27     function<ll(ll, ll, ll, bool)> dp = [&](ll pos, ll x, ll cnt, bool flag)
28     {
29         if (pos == 0)
30             return 1ll;
31         if (flag && f[pos][x][cnt])
32             return f[pos][x][cnt];
33         int up = flag ? 9 : a[pos];
34         ll ans = 0;
35         for (int i = 0; i <= up; ++i)
36         {
37             if (i == x)
38             {
39                 if (cnt + 1 > len[i])
40                     continue;
41                 ans = (ans + dp(pos - 1, i, cnt + 1, flag || (i < up))) % mod;
42             }
43             else
44             {
45                 ans = (ans + dp(pos - 1, i, 1, flag || (i < up))) % mod;
46             }
47         }
48         if (flag)
49             f[pos][x][cnt] = ans % mod;
50         return ans % mod;
51     };
52     function<ll(ll)> clac = [&](ll x)
53     {
54         int id = 0;
55         while (x)
56         {
57             a[++id] = x % 10;
58             x /= 10;
59         }
60         return dp(id, 0, 0, 0);
61     };
62
63     cin >> l >> r >> n;
64     memset(len, 0x3f, sizeof(len));
65     memset(f, 0, sizeof(f));
66     for (int i = 1; i <= n; i++)
67     {
68         ll x, cnt;
69         cin >> x >> cnt;
70         len[x] = min(len[x], cnt);
71     }
72     cout << (clac(r) - clac(l - 1) + mod) % mod << "\n";
73 }
74
75 int main()
76 {
77     ios;
78     int t = 1;
79     cin >> t;
80     while (t--)
81     {
82         solve();
83     }
84     return 0;

```

85 }

很多线段树，树状数组

```
1 //线段树单点修改，区间查询
2 void build(int u, int l, int r)
3 {
4     tr[u] = {l, r};
5     if (l == r) return;
6     int mid = (l + r) >> 1;
7     build(u << 1, l, mid);
8     build(u << 1 | 1, mid + 1, r);
9 }
10 //modify(l, position, valuse)
11 void modify(int u, int x, int v){
12     if (tr[u].l == tr[u].r)
13     {
14         tr[u].v = v;
15         return;
16     }
17     int mid = (tr[u].l + tr[u].r) >> 1;
18     if (x <= mid)
19     {
20         modify(u << 1, x, v);
21     }
22     else
23     {
24         modify(u << 1 | 1, x, v);
25     }
26     tr[u].v = max(tr[u << 1].v, tr[u << 1 | 1].v);
27 }
28 //query(l, l, r)
29 int query(int u, int l, int r)
30 {
31     if (l <= tr[u].l && r >= tr[u].r)
32     {
33         return tr[u].v;
34     }
35
36     int mid = (tr[u].l + tr[u].r) >> 1;
37     int a = 0;
38     int b = 0;
39     if (l <= mid)
40     {
41         a = query(u << 1, l, r);
42     }
43     if (r > mid)
44     {
45         b = query(u << 1 | 1, l, r);
46     }
47     return max(a, b);
48 }
49
50 //线段树区间加，区间查询
51 ll a[maxn], w[maxn * 4], lazyTag[maxn * 4];
52 void pushup(int u) { w[u] = w[u << 1] + w[u << 1 | 1]; } // 上推
53 void makeTag(int u, int len, ll x)
54 { // 下放 lazytag
55     lazyTag[u] += x;
56     w[u] += len * x;
57 }
58 void pushdown(int u, int L, int R)
59 { // 下放 lazytag 的索引
60     int M = (L + R) >> 1;
61     makeTag(u << 1, M - L + 1, lazyTag[u]);
62     makeTag(u << 1 | 1, R - M, lazyTag[u]);
63     lazyTag[u] = 0;
64 }
65 void build(int u, int L, int R)
66 { // 递归建树
67     if (L == R)
```

```

68     {
69         w[u] = a[L];
70         return;
71     }
72     int M = (L + R) >> 1;
73     build(u << 1, L, M);
74     build(u << 1 | 1, M + 1, R);
75     pushup(u);
76 }
77 bool inRange(int L, int R, int l, int r) { return (l <= L) && (R <= r); } // 判断 [L,R] 是否被 [l,r] 包含
78 bool outofRange(int L, int R, int l, int r) { return (L > r) || (R < l); } // 判断 [L,R] 是否与 [l,r] 完全无交集
79 ll query(int u, int L, int R, int l, int r)
80 { // 区间查询
81     if (inRange(L, R, l, r))
82         return w[u];
83     else if (!outofRange(L, R, l, r))
84     {
85         int M = (L + R) >> 1;
86         pushdown(u, L, R);
87         return query(u << 1, L, M, l, r) + query(u << 1 | 1, M + 1, R, l, r);
88     }
89     else
90         return 0;
91 }
92 void update(int u, int L, int R, int l, int r, ll x)
93 { // 区间修改
94     if (inRange(L, R, l, r))
95         makeTag(u, R - L + 1, x);
96     else if (!outofRange(L, R, l, r))
97     {
98         int M = (L + R) >> 1;
99         pushdown(u, L, R);
100         update(u << 1, L, M, l, r, x);
101         update(u << 1 | 1, M + 1, R, l, r, x);
102         pushup(u);
103     }
104 }
105 int main()
106 {
107     int n, m;
108     cin >> n >> m;
109     for (int i = 1; i <= n; ++i)
110     {
111         cin >> a[i];
112     }
113     build(1, 1, n);
114     for (int i = 1; i <= m; ++i)
115     {
116         int op, x, y;
117         ll k;
118         cin >> op;
119         if (op == 1)
120         {
121             cin >> x >> y >> k;
122             update(1, 1, n, x, y, k);
123         }
124         else
125         {
126             cin >> x >> y;
127             cout << query(1, 1, n, x, y) << endl;
128         }
129     }
130     return 0;
131 }
132 //线段树区间乘法, 区间加法, 区间查询
133 ll n, m, p;
134 int ls(int u) { return u << 1; }
135 int rs(int u) { return u << 1 | 1; }
136 int mid(int l, int r) { return (l + r) >> 1; }
137 struct Node
138 {

```



```

139     ll ad;
140     ll mu = 1;
141 } tag[maxn * 4];
142 ll a[maxn], w[maxn * 4];
143 void pushup(int u) { w[u] = (w[ls(u)] + w[rs(u)]) % p; }
144 void build(int u, int L, int R)
145 {
146     if (L == R)
147     {
148         w[u] = a[L] % p;
149         return;
150     }
151     int M = mid(L, R);
152     build(ls(u), L, M);
153     build(rs(u), M + 1, R);
154     pushup(u);
155 }
156 bool inRange(int L, int R, int l, int r) { return (l <= L) && (R <= r); } // 判断 [L,R] 是否被 [l,r] 包含
157 bool outofRange(int L, int R, int l, int r) { return (L > r) || (R < l); } // 判断 [L,R] 是否与 [l,r] 完全无交集
158 void pushdown(int u, int L, int R)
159 {
160     int M = mid(L, R);
161     if (tag[u].mu != 1)
162     {
163         tag[ls(u)].ad = tag[ls(u)].ad * tag[u].mu % p;
164         tag[rs(u)].ad = tag[rs(u)].ad * tag[u].mu % p;
165         tag[ls(u)].mu = tag[ls(u)].mu * tag[u].mu % p;
166         tag[rs(u)].mu = tag[rs(u)].mu * tag[u].mu % p;
167         w[ls(u)] = w[ls(u)] * tag[u].mu % p;
168         w[rs(u)] = w[rs(u)] * tag[u].mu % p;
169         tag[u].mu = 1;
170     }
171     if (tag[u].ad)
172     {
173         w[ls(u)] = (w[ls(u)] + (M - L + 1) * tag[u].ad) % p;
174         tag[ls(u)].ad = (tag[ls(u)].ad + tag[u].ad) % p;
175         w[rs(u)] = (w[rs(u)] + (R - M) * tag[u].ad) % p;
176         tag[rs(u)].ad = (tag[rs(u)].ad + tag[u].ad) % p;
177         tag[u].ad = 0;
178     }
179 }
180 ll query(int u, int L, int R, int l, int r)
181 {
182     if (inRange(L, R, l, r))
183         return w[u];
184     else if (!outofRange(L, R, l, r))
185     {
186         int M = mid(L, R);
187         pushdown(u, L, R);
188         return (query(ls(u), L, M, l, r) + query(rs(u), M + 1, R, l, r)) % p;
189     }
190     else
191         return 0;
192 }
193 void update1(int u, int L, int R, int l, int r, ll k)
194 {
195     if (inRange(L, R, l, r))
196     {
197         tag[u].mu = tag[u].mu * k % p;
198         tag[u].ad = tag[u].ad * k % p;
199         w[u] = w[u] * k % p;
200         return;
201     }
202     else if (!outofRange(L, R, l, r))
203     {
204         int M = mid(L, R);
205         pushdown(u, L, R);
206         update1(ls(u), L, M, l, r, k);
207         update1(rs(u), M + 1, R, l, r, k);
208         pushup(u);
209     }

```

```

210 }
211 void update2(int u, int L, int R, int l, int r, ll k)
212 {
213     if (inRange(L, R, l, r))
214     {
215         tag[u].ad = (tag[u].ad + k) % p;
216         w[u] = (w[u] + (R - L + 1) * k) % p;
217         return;
218     }
219     else if (!outofRange(L, R, l, r))
220     {
221         int M = mid(L, R);
222         pushdown(u, L, R);
223         update2(ls(u), L, M, l, r, k);
224         update2(rs(u), M + 1, R, l, r, k);
225         pushup(u);
226     }
227 }
228 int main()
229 {
230     cin >> n >> m >> p;
231     for (int i = 1; i <= n; ++i)
232         cin >> a[i];
233     build(1, 1, n);
234     for (int i = 1; i <= m; ++i)
235     {
236         int op, x, y;
237         ll k;
238         cin >> op >> x >> y;
239         if (op == 1)
240         {
241             cin >> k;
242             update1(1, 1, n, x, y, k);
243         }
244         else if (op == 2)
245         {
246             cin >> k;
247             update2(1, 1, n, x, y, k);
248         }
249         else
250         {
251             cout << query(1, 1, n, x, y) % p << endl;
252         }
253     }
254     return 0;
255 }
256 //树状数组单点修改区间查询
257 #include <bits/stdc++.h>
258 using namespace std;
259 int tree[500010];
260 int n, m;
261 int lowbit(int x) { return x & -x; }
262 void add(int x, int k)
263 {
264     while (x <= n)
265     {
266         tree[x] += k;
267         x += lowbit(x);
268     }
269 }
270 int sum(int x)
271 {
272     int ans = 0;
273     while (x != 0)
274     {
275         ans += tree[x];
276         x -= lowbit(x);
277     }
278     return ans;
279 }
280 int main()

```

```

281 {
282     cin >> n;
283     for (int i = 1; i <= n; i++)
284     {
285         int a;
286         cin >> a;
287         add(i, a);
288     }
289     cin >> m;
290     for (int i = 1; i <= m; i++)
291     {
292         int u, v, w;
293         cin >> u >> v >> w;
294         if (u == 1)
295             add(v, w);
296         else if (u == 2)
297             cout << sum(w) - sum(v - 1) << endl;
298     }
299     return 0;
300 }
301 //树状数组区间修改单点查询
302 #include <bits/stdc++.h>
303 using namespace std;
304 int tree[500010], y[500010];
305 int n, m;
306 int lowbit(int x) { return x & -x; }
307 void add(int x, int k)
308 {
309     while (x <= n)
310     {
311         tree[x] += k;
312         x += lowbit(x);
313     }
314 }
315 int sum(int x)
316 {
317     int ans = 0;
318     while (x != 0)
319     {
320         ans += tree[x];
321         x -= lowbit(x);
322     }
323     return ans;
324 }
325 int main()
326 {
327     cin >> n >> m;
328     for (int i = 1; i <= n; i++)
329         cin >> y[i];
330     for (int i = 1; i <= m; i++)
331     {
332         int u, a, b, c, v;
333         cin >> u;
334         if (u == 1)
335         {
336             cin >> a >> b >> c;
337             add(a, c);
338             add(b + 1, -c);
339         }
340         else if (u == 2)
341         {
342             cin >> v;
343             cout << y[v] + sum(v) << endl;
344         }
345     }
346     return 0;
347 }
348 //树状数组区间修改区间查询
349 #include <bits/stdc++.h>
350 using namespace std;
351 #define int long long

```

```

352 #define MAXN (int)(1e6 + 5)
353
354 int n,m;
355 int a[MAXN];
356 struct BIT{
357     int bit1[MAXN],bit2[MAXN];
358     int lowbit(int x){return x & (-x);}
359     void add(int i, int v){
360         int k = v * i;
361         while(i <= n){
362             bit1[i] += v, bit2[i] += k; // 维护的重点部分
363             i += lowbit(i);
364         }
365     }
366     int sum(int *b, int i){
367         int res = 0;
368         while(i >= 1){
369             res += b[i];
370             i -= lowbit(i);
371         }
372         return res;
373     }
374     int qry(int l, int r){
375         return sum(bit1, r) * (r + 1) - sum(bit1, l - 1) * l - (sum(bit2, r) - sum(bit2, l - 1));
376     }
377 }bt;
378
379 signed main(){
380     cin >> n >> m;
381     for(int i = 1; i <= n; i++)
382         cin >> a[i], bt.add(i, a[i] - a[i - 1]);
383     while(m--){
384         int op, l, r, x;
385         cin >> op >> l >> r;
386         if(op == 1){cin >> x; bt.add(l, x), bt.add(r + 1, -x);}
387         if(op == 2){printf("%lld\n", bt.qry(l, r));}
388     }
389     return 0;
390 }

```

快速幂

```

1 int qpow(long long a, int b) {
2     int ans = 1;
3     a = (a % p + p) % p;
4     for (; b; b >>= 1) {
5         if (b & 1) ans = (a * ans) % p;
6         a = (a * a) % p;
7     }
8     return ans;
9 }

```

lucas

```

1 long long Lucas(long long n, long long m, long long p) {
2     if (m == 0) return 1;
3     return (C(n % p, m % p, p) * Lucas(n / p, m / p, p)) % p;
4 }

```

各种背包

```

1 //01
2 for (int i = 1; i <= n; i++)
3     for (int l = W; l >= w[i]; l--) f[l] = max(f[l], f[l - w[i]] + v[i]);
4 //完全
5 for (int i = 1; i <= n; i++)
6     for (int l = w[i]; l <= W; l++)
7         if (f[l - w[i]] + v[i] > f[l]) f[l] = f[l - w[i]] + v[i];
8 //分组

```

```

9  for (int k = 1; k <= ts; k++)          // 循环每一组
10  for (int i = m; i >= 0; i--)          // 循环背包容量
11  for (int j = 1; j <= cnt[k]; j++)    // 循环该组的每一个物品
12  if (i >= w[t[k][j]])                // 背包容量充足
13  dp[i] = max(dp[i], dp[i - w[t[k][j]]] + c[t[k][j]]); // 像 0-1 背包一样状态转移

```

Z 函数

```

1  vector<int> z_function_trivial(string s) {
2      int n = (int)s.length();
3      vector<int> z(n);
4      for (int i = 1; i < n; ++i)
5          while (i + z[i] < n && s[z[i]] == s[i + z[i]]) ++z[i];
6      return z;
7  }

```