Standard Code Library

tingyx

njust

October 2, 2023

Contents

一切的开始

Codeforces/XCPC

- 需要 C++17/C++20

```cpp
#include <bits/stdc++.h>
#define endl '\n'
#define pll pair<ll, ll>
#define tll tuple<ll, ll, ll>
#define vi vector<int>
#define vl vector<ll>
#define x first
#define y second
#define rep(i, j, k) for(int i = (j); i <= (k); i++)
#define per(i, j ,k) for(int i = (j); i >= (k); i--)
#define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
using namespace std;
typedef long long ll;
const ll maxn = 2e5 + 10;
const ll mod = 998244353;
const ll inf32 = 1e9;
const ll inf64 = 1e18;


void solve(){

}

int main(){
    ios;
    //freopen("sample.txt", "r", stdin);
    //freopen("resout.txt", "w", stdout);
    int t = 1;
    //cin >> t;
    while(t--){
        solve();
    }
    return 0;
}
// --------------------------------------------------------------------------
```

int128

- 不要使用 cin/cout，记得关同步流

```cpp
typedef __int128 i128;

i128 read()
{
    i128 x = 0; bool f = 0;
    char c = getchar();
    while (c < '0' || c > '9')
    {
        if (c == '-')
            f = 1;
        c = getchar();
    }
    while (c >= '0' && c <= '9')
    {
        x = (x << 1) + (x << 3) + (c ^ 48);
        c = getchar();
    }
    return f ? -x : x;
}

inline void write(i128 x)
{
    if (x < 0)
        putchar('-'), x = -x;
    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    }
```

区间问题

莫队

- 区间取两个数相同概率

```cpp
#include <algorithm>
#include <cmath>
#include <cstdio>
using namespace std;
const int N = 50005;
int n, m, maxn;
int c[N];
long long sum;
int cnt[N];
long long ans1[N], ans2[N];

struct query {
  int l, r, id;

  bool operator<(const query &x) const {  // 重载 < 运算符
    if (l / maxn != x.l / maxn) return l < x.l;
    return (l / maxn) & 1 ? r < x.r : r > x.r;
  }
} a[N];

void add(int i) {
  sum += cnt[i];
  cnt[i]++;
}

void del(int i) {
  cnt[i]--;
  sum -= cnt[i];
}

long long gcd(long long a, long long b) { return b ? gcd(b, a % b) : a; }

int main() {
  scanf("%d%d", &n, &m);
  maxn = sqrt(n);
  for (int i = 1; i <= n; i++) scanf("%d", &c[i]);
  for (int i = 0; i < m; i++) scanf("%d%d", &a[i].l, &a[i].r), a[i].id = i;
  sort(a, a + m);
  for (int i = 0, l = 1, r = 0; i < m; i++) {  // 具体实现
    if (a[i].l == a[i].r) {
      ans1[a[i].id] = 0, ans2[a[i].id] = 1;
      continue;
    }
    while (l > a[i].l) add(c[--l]);
    while (r < a[i].r) add(c[++r]);
    while (l < a[i].l) del(c[l++]);
    while (r > a[i].r) del(c[r--]);
    ans1[a[i].id] = sum;
    ans2[a[i].id] = (long long)(r - l + 1) * (r - l) / 2;
  }
  for (int i = 0; i < m; i++) {
    if (ans1[i] != 0) {
      long long g = gcd(ans1[i], ans2[i]);
      ans1[i] /= g, ans2[i] /= g;
    } else
      ans2[i] = 1;
    printf("%lld/%lld\n", ans1[i], ans2[i]);
  }
  return 0;
}
```

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 <
    ↪  B.first: A.second > B.second;
9   }
10
11  void mergeSort(int l, int r, int deep, const int &cmptype) {
12      if (l == r) {
13          temp[l][deep].first = a[l];
14          temp[l][deep].second = l;
15          return;
16      }
```

```cpp
        int mid = (l + r) >> 1;
        mergeSort(l, mid, deep + 1, cmptype);
        mergeSort(mid + 1, r, deep + 1, cmptype);
        int p1 = l, p2 = mid + 1;
        while (p1 <= mid && p2 <= r) {
            if (cmp(temp[p1][deep + 1], temp[p2][deep + 1], cmptype)) {
                temp[l++][deep] = temp[p1++][deep + 1];
            } else {
                temp[l++][deep] = temp[p2++][deep + 1];
            }
        }
        while (p1 <= mid) {
            temp[l++][deep] = temp[p1++][deep + 1];
        }
        while (p2 <= r) {
            temp[l++][deep] = temp[p2++][deep + 1];
        }
}

void cdqDivAlgorithm(int l, int r, int deep, const int &cmptype) {
        if (l == r) {
            dp[l] = max(dp[l], 1);
            ans = max(ans, dp[l]);
            return;
        }
        int mid = (l + r) >> 1;
        cdqDivAlgorithm(l, mid, deep + 1, cmptype);
        int p1 = l, p2 = mid + 1, premax = 0;
        while (p1 <= mid && p2 <= r) {
            if (cmp(temp[p1][deep + 1], temp[p2][deep + 1], cmptype)) {
                premax = max(premax, dp[temp[p1++][deep + 1].second]);
            } else {
                dp[temp[p2][deep + 1].second] = max(premax + 1, dp[temp[p2][deep + 1].second]);
                p2++;
            }
        }
        while (p2 <= r) {
            dp[temp[p2][deep + 1].second] = max(premax + 1, dp[temp[p2][deep + 1].second]);
            p2++;
        }
        cdqDivAlgorithm(mid + 1, r, deep + 1, cmptype);
}

int main()
{
        while (scanf("%d", &x) != EOF)a[++n] = x;
        mergeSort(1, n, 0, 1);
        cdqDivAlgorithm(1, n, 0, 1);
        printf("%d\n", ans);
        memset(dp, 0, sizeof(dp));
        ans = 0;
        mergeSort(1, n, 0, 0);
        cdqDivAlgorithm(1, n, 0, 0);
        printf("%d\n", ans);
        return 0;
}
```

- 求地毯覆盖（最多取多少个不相互覆盖）

```cpp
#include<bits/stdc++.h>
using namespace std;
const int MAXN = 1000005;
int n, L[MAXN], R[MAXN], id[MAXN], dp[MAXN], ans;
int temp[MAXN];
void cdqDivAlgorithm(int l, int r) {
        if (l == r) {
            dp[id[l]] = max(1, dp[id[l]]);
            ans = max(ans, dp[id[l]]);
            return;
        }
        int mid = (l + r) >> 1;
        cdqDivAlgorithm(l, mid);
```

```cpp
        int p1 = l, pl, p2 = mid + 1, premax = 0;
        while (p1 <= mid && p2 <= r) {
            if (R[id[p1]] <= L[id[p2]]) {
                premax = max(premax, dp[id[p1++]]);
            } else {
                dp[id[p2]] = max(premax + 1, dp[id[p2]]);
                ++p2;
            }
        }
        while (p2 <= r) {
            dp[id[p2]] = max(premax + 1, dp[id[p2]]);
            ++p2;
        }
        cdqDivAlgorithm(mid + 1, r);
        p1 = l, pl = l, p2 = mid + 1;
        while (p1 <= mid && p2 <= r) {
            if (R[id[p1]] < R[id[p2]]) {
                temp[pl++] = id[p1++];
            } else {
                temp[pl++] = id[p2++];
            }
        }
        while (p1 <= mid) {
            temp[pl++] = id[p1++];
        }
        while (p2 <= r) {
            temp[pl++] = id[p2++];
        }
        for (int i = l; i <= r; ++i) {
            id[i] = temp[i];
        }
}
int main()
{
    scanf("%d", &n);
    for (int i = 1; i <= n; ++i) {
        scanf("%d %d", &L[i], &R[i]);
        id[i] = i;
    }
    sort(id + 1, id + 1 + n, [](const int &A, const int &B) {
        return L[A] < L[B];
    });
    cdqDivAlgorithm(1, n);
    printf("%d\n", ans);
    return 0;
}
```

- 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 <= N <= 100000 1 <= T <=50000 0 < P < 100，|S| <= 10^6

```cpp
#include<bits/stdc++.h>
using namespace std;
const int MAXN = 100005;
const double eps = 1e-6;
int m, n, id[MAXN], qid[MAXN], type[MAXN], x[MAXN], temp[MAXN], top;
double k[MAXN], b[MAXN], ans[MAXN];
char op[55];
inline bool cmp(const int &A, const int &B) {
    return type[A] != type[B] ? type[A] < type[B] : type[A] ? x[A] < x[B] : k[A] < k[B];
}
inline int dcmp(double x) {
    return x > eps ? 1 : x < -eps ? -1 : 0;
}
inline double getCross(const double &k1, const double &b1, const double &k2, const double &b2) {
    return (b2 - b1) / (k1 - k2);
}
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, pl = l, p2 = mid + 1;
49        while (p1 <= mid && p2 <= r) {
50            if (cmp(id[p1], id[p2])) {
51                temp[pl++] = id[p1++];
52            } else {
53                temp[pl++] = id[p2++];
54            }
55        }
56        while (p1 <= mid) {
57            temp[pl++] = id[p1++];
58        }
59        while (p2 <= r) {
60            temp[pl++] = 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）

```cpp
1   #include <bits/stdc++.h>
2   #define endl '\n'
3   #define pll pair<ll, ll>
4   #define tll tuple<ll, ll, ll>
5   #define vi vector<int>
6   #define vl vector<ll>
7   #define x first
8   #define y second
9   #define rep(i, j, k) for(int i = (j); i <= (k); i++)
10  #define per(i, j, k) for(int i = (j); i >= (k); i--)
11  #define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
12  using namespace std;
13  typedef long long ll;
14  const ll maxn = 1e5 + 10;
15  const ll mod = 998244353;
16  const ll inf = 0x3f3f3f3f;
17
18  void solve()
19  {
20      int n, m, q, k, cnt = 0;
21      cin >> n >> m >> q;
22      vi red(n + 1);
23      vector<vector<pll>> G(n + 1);
24      vl dis(n + 1), dep(n + 1), v(n + 1);
25      vi dfn(n + 1), idx(n + 1);
26      vi son(n + 1, -1), sz(n + 1), fa(n + 1), top(n + 1);
27      function<void(int, int)> dfs1 = [&](int u, int f) {
28          son[u] = -1;
29          sz[u] = 1;
30          if(!red[u])
31              red[u] = red[f];
32          for(auto [v, w] : G[u]) {
33              if(v == f)
34                  continue;
35              dep[v] = dep[u] + 1;
36              dis[v] = dis[u] + w;
37              fa[v] = u;
38              dfs1(v, u);
39              sz[u] += sz[v];
40              if(son[u] == -1 || sz[v] > sz[son[u]])
41                  son[u] = v;
42          }
43      };
44      function<void(int, int)> dfs2 = [&](int u, int t) {
45          top[u] = t;
46          dfn[u] = ++cnt;
47          idx[cnt] = u;
48          if(son[u] == -1)
49              return;
50          dfs2(son[u], t);
51          for(auto [v, w] : G[u])
52              if(v != son[u] && v != fa[u])
53                  dfs2(v, v);
54      };
55      auto lca = [&](int u, int v) {
56          while(top[u] != top[v]) {
57              if(dep[top[u]] > dep[top[v]])
58                  u = fa[top[u]];
59              else
60                  v = fa[top[v]];
61          }
```

```
62          return dep[u] > dep[v] ? v : u;
63      };
64      for(int i = 1, x; i <= m; ++i)
65          cin >> x, red[x] = x;
66      for(int i = 1; i < n; ++i) {
67          int u, v, w;
68          cin >> u >> v >> w;
69          G[u].push_back({v, w});
70          G[v].push_back({u, w});
71      }
72      dfs1(1, 0);
73      dfs2(1, 1);
74      for(int i = 1; i <= n; ++i)
75          v[i] = dis[i] - dis[red[i]];
76      while(q--) {
77          cin >> k;
78          vector<int> p(k + 1);
79          auto check = [&](ll st) {
80              vector<int> q;
81              for(int i = 1; i <= k; ++i)
82                  if(v[p[i]] > st)
83                      q.push_back(p[i]);
84              if(q.size() == 0)
85                  return true;
86              int mnd = n + 1, mxd = 0;
87              for(int i = 0; i < q.size(); ++i) {
88                  mnd = min(mnd, dfn[q[i]]);
89                  mxd = max(mxd, dfn[q[i]]);
90              }
91              int ca = lca(idx[mnd], idx[mxd]);
92              for(int i = 0; i < q.size(); ++i)
93                  if(dis[q[i]] - dis[ca] > st)
94                      return false;
95              return true;
96          };
97          ll mx = 0;
98          for(int i = 1; i <= k; ++i) {
99              cin >> p[i];
100             mx = max(mx, v[p[i]]);
101         }
102         ll l = 0, r = mx;
103         while(l < r) {
104             ll mid = (l + r) >> 1;
105             if(check(mid))
106                 r = mid;
107             else
108                 l = mid + 1;
109         }
110         cout << l << endl;
111     }
112 }
113
114 int main()
115 {
116     ios;
117     // freopen("sample.txt", "r", stdin);
118     // freopen("resout.txt", "w", stdout);
119     int t = 1;
120     cin >> t;
121     while(t--) {
122         solve();
123     }
124     return 0;
125 }
126
```

- 树上操作

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

```cpp
#include <bits/stdc++.h>
#define endl '\n'
#define pll pair<ll, ll>
#define tll tuple<ll, ll, ll>
#define vi vector<int>
#define vl vector<ll>
#define x first
#define y second
#define rep(i, j, k) for(int i = (j); i <= (k); i++)
#define per(i, j ,k) for(int i = (j); i >= (k); i--)
#define ios ios::sync_with_stdio(false), cin.tie(0), cout.tie(0)
using namespace std;
typedef long long ll;
const ll maxn = 2e5 + 10;
const ll mod = 998244353;
const ll inf = 0x3f3f3f3f;
const int N = 1e5 + 10, M = N * 2;

int n, m;
// w 为节点权值
int h[N], w[N], e[M], ne[M], idx;
// id[x] 为节点 x 的新编号, nw[x] 是新编号为 x 的节点的权值
int id[N], nw[N], cnt;
// dep 为深度, sz 为子树大小, top[x] 是 x 所在重链的头结点,
// fa[x] 为 x 父亲, son[x] 为 x 的重儿子
int dep[N], sz[N], top[N], fa[N], son[N];
struct Tree {
    int l, r;
    ll sum, add;
} tr[N << 2];

void add(int a, int b) {
    e[idx] = b, ne[idx] = h[a], h[a] = idx++;
}

// 第一次 dfs, 求节点深度、父亲、子树大小和重儿子
void dfs1(int u, int from, int depth) {
    dep[u] = depth, fa[u] = from, sz[u] = 1;
    for (int i = h[u]; ~i; i = ne[i]) {
        int v = e[i];
        if (v == from) continue;
        dfs1(v, u, depth + 1);
        sz[u] += sz[v];
        if (sz[son[u]] < sz[v]) son[u] = v;
    }
}

// 第二次 dfs, t 为 u 重链头结点
void dfs2(int u, int t) {
    id[u] = ++cnt, nw[cnt] = w[u], top[u] = t;
    // 到叶子了, 直接返回
    if (!son[u]) return;
    // 先遍历重儿子
    dfs2(son[u], t);
    // 遍历轻儿子
    for (int i = h[u]; ~i; i = ne[i]) {
        int v = e[i];
        if (v == fa[u] || v == son[u]) continue;
        dfs2(v, v);
    }
}

void pushup(int u) {
    tr[u].sum = tr[u << 1].sum + tr[u << 1 | 1].sum;
}

void pushdown(int u) {
    auto &root = tr[u], &left = tr[u << 1], &right = tr[u << 1 | 1];
    if (root.add) {
        left.sum += root.add * (left.r - left.l + 1);
        left.add += root.add;
```

```cpp
            right.sum += root.add * (right.r - right.l + 1);
            right.add += root.add;
            root.add = 0;
        }
    }

    void build(int u, int l, int r) {
        tr[u] = {l, r, nw[l], 0};
        if (l == r) return;
        int mid = l + r >> 1;
        build(u << 1, l, mid), build(u << 1 | 1, mid + 1, r);
        pushup(u);
    }

    void update(int u, int l, int r, ll k) {
        if (l <= tr[u].l && tr[u].r <= r) {
            tr[u].add += k;
            tr[u].sum += k * (tr[u].r - tr[u].l + 1);
            return;
        }
        pushdown(u);
        int mid = tr[u].l + tr[u].r >> 1;
        if (l <= mid) update(u << 1, l, r, k);
        if (r > mid) update(u << 1 | 1, l, r, k);
        pushup(u);
    }

    ll query(int u, int l, int r) {
        if (l <= tr[u].l && tr[u].r <= r) return tr[u].sum;
        pushdown(u);
        int mid = tr[u].l + tr[u].r >> 1;
        ll res = 0;
        if (l <= mid) res += query(u << 1, l, r);
        if (r > mid) res += query(u << 1 | 1, l, r);
        return res;
    }

    void update_path(int u, int v, ll k) {
        while (top[u] != top[v]) {
            if (dep[top[u]] < dep[top[v]]) swap(u, v);
            // 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
            update(1, id[top[u]], id[u], k);
            // u 跳到重链头上面
            u = fa[top[u]];
        }
        if (dep[u] < dep[v]) swap(u, v);
        update(1, id[v], id[u], k);
    }

    ll query_path(int u, int v) {
        ll res = 0;
        while (top[u] != top[v]) {
            if (dep[top[u]] < dep[top[v]]) swap(u, v);
            res += query(1, id[top[u]], id[u]);
            u = fa[top[u]];
        }
        if (dep[u] < dep[v]) swap(u, v);
        res += query(1, id[v], id[u]);
        return res;
    }

    void update_tree(int u, ll k) {
        update(1, id[u], id[u] + sz[u] - 1, k);
    }

    ll query_tree(int u) {
        return query(1, id[u], id[u] + sz[u] - 1);
    }

    void solve() {
        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++;
```

```cpp
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  }
```

计算几何

二维几何：点与向量

```
1   #define y1 yy1
2   #define nxt(i) ((i + 1) % s.size())
3   typedef double LD;
4   const LD PI = 3.14159265358979323846;
5   const LD eps = 1E-10;
6   int sgn(LD x) { return fabs(x) < eps ? 0 : (x > 0 ? 1 : -1); }
7   struct L;
8   struct P;
9   typedef P V;
10  struct P {
11      LD x, y;
12      explicit P(LD x = 0, LD y = 0): x(x), y(y) {}
13      explicit P(const L& l);
14  };
```

```cpp
struct L {
    P s, t;
    L() {}
    L(P s, P t): s(s), t(t) {}
};

P operator + (const P& a, const P& b) { return P(a.x + b.x, a.y + b.y); }
P operator - (const P& a, const P& b) { return P(a.x - b.x, a.y - b.y); }
P operator * (const P& a, LD k) { return P(a.x * k, a.y * k); }
P operator / (const P& a, LD k) { return P(a.x / k, a.y / k); }
inline bool operator < (const P& a, const P& b) {
    return sgn(a.x - b.x) < 0 || (sgn(a.x - b.x) == 0 && sgn(a.y - b.y) < 0);
}
bool operator == (const P& a, const P& b) { return !sgn(a.x - b.x) && !sgn(a.y - b.y); }
P::P(const L& l) { *this = l.t - l.s; }
ostream &operator << (ostream &os, const P &p) {
    return (os << "(" << p.x << "," << p.y << ")");
}
istream &operator >> (istream &is, P &p) {
    return (is >> p.x >> p.y);
}

LD dist(const P& p) { return sqrt(p.x * p.x + p.y * p.y); }
LD dot(const V& a, const V& b) { return a.x * b.x + a.y * b.y; }
LD det(const V& a, const V& b) { return a.x * b.y - a.y * b.x; }
LD cross(const P& s, const P& t, const P& o = P()) { return det(s - o, t - o); }
// -----------------------------------------
```

字符串

KMP

- KMP 模板

```cpp
#include <bits/stdc++.h>

using namespace std;

const int N = 1e6 + 10;

vector<int> prefix_function(string s)
{
    int n = (int)s.length();
    vector<int> pi(n);
    for (int i = 2; i < n; i++)
    {
        pi[i] = pi[i - 1];
        while (pi[i] && s[i] != s[pi[i] + 1])
            pi[i] = pi[pi[i]];
        pi[i] += (s[i] == s[pi[i] + 1]);
    }
    return pi;
}

int main(void)
{
    ios::sync_with_stdio(false), cin.tie(0), cout.tie(0);
    string s1, s2;
    cin >> s1 >> s2;
    s1 = " " + s1;
    s2 = " " + s2;
    auto nxt = prefix_function(s2);
    for (int i = 1, j = 0; i < s1.size(); i++)
    {
        while (j && s1[i] != s2[j + 1])
            j = nxt[j];
        if (s1[i] == s2[j + 1])
            j++;
        if (j == s2.size() - 1)
        {
```

```cpp
            cout << i - j + 1 << "\n";
            j = nxt[j];
        }
    }
    for (int i = 1; i < s2.size(); i++)
        cout << nxt[i] << " ";

    return 0;
}
```

- carpet(二维 KMP) 有一个 n*m 的地毯，aij 表示地毯每格的元素，bij 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

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

vector<int> prefix_function(string s)
{
    int n = (int)s.length();
    vector<int> pi(n);
    for (int i = 2; i < n; i++)
    {
        pi[i] = pi[i - 1];
        while (pi[i] && s[i] != s[pi[i] + 1])
            pi[i] = pi[pi[i]];
        pi[i] += (s[i] == s[pi[i] + 1]);
    }
    return pi;
}

int get_length(vector<string> s)
{
    int len = s[1].size() - 1;
    int ret = len;
    vector<int> cnt(len + 1);
    for (int i = 1; i < s.size(); ++i)
    {
        string tmp = s[i];
        auto nxt = prefix_function(tmp);
        int j = len;
        while (j)
        {
            cnt[len - nxt[j]]++;
            j = nxt[j];
        }
    }
    for (int i = 1; i <= len; ++i)
        if (cnt[i] == s.size() - 1)
        {
            ret = i;
            break;
        }
    return ret;
}

void solve()
{
    int n, m;
```

```cpp
    cin >> n >> m;
    vector<string> s1(n + 1);
    for (int i = 1; i <= n; ++i)
        cin >> s1[i], s1[i] = " " + s1[i];
    vector<string> s2(m + 1);
    for (int i = 1; i <= m; ++i)
    {
        string tmp = " ";
        for (int j = 1; j <= n; ++j)
            tmp += s1[j][i];
        s2[i] = tmp;
    }
    vector<vector<int>> a(n + 1, vector<int>(m + 1, 0));
    for (int i = 1; i <= n; ++i)
        for (int j = 1; j <= m; ++j)
            cin >> a[i][j];
    int p = get_length(s1), q = get_length(s2);
    ll ans = 1e9;
    deque<int> dq;
    auto b = a;
    for (int i = 1; i <= n; ++i){
        while (dq.size()) dq.pop_back();
        for (int j = 1; j <= m; ++j){
            while (dq.size() && j - dq.front() + 1 > p) dq.pop_front();
            while (dq.size() && a[i][dq.back()] <= a[i][j]) dq.pop_back();
            dq.push_back(j);
            b[i][j] = a[i][dq.front()];
        }
    }
    for (int j = 1; j <= m; ++j){
        while (dq.size()) dq.pop_back();
        for (int i = 1; i <= n; ++i){
            while (dq.size() && i - dq.front() + 1 > q) dq.pop_front();
            while (dq.size() && b[dq.back()][j] <= b[i][j]) dq.pop_back();
            dq.push_back(i);
            if (i >= q && j >= p)
                ans = min(ans, 1ll * b[dq.front()][j]);
        }
    }
    ans = ans * (p + 1) * (q + 1);
    cout << ans << endl;
}

signed main()
{
    ios;
    // freopen("sample.txt", "r", stdin);
    // freopen("resout.txt", "w", stdout);
    int t = 1;
    // cin >> t;
    while (t--)
    {
        solve();
    }
    return 0;
}

```

杂项

STL

- copy

```cpp
template <class InputIterator, class OutputIterator>
  OutputIterator copy (InputIterator first, InputIterator last, OutputIterator result);
```