Contents

1 Combinatorial **2**
1.1 Markov Chains 2
1.2 Lucas 4
1.3 Grundy 6

2 Data Structures **8**
2.1 BIT 8
2.2 BIT2D 9
2.3 Dynamic Seg 11
2.4 Linear Container 12
2.5 Merge Sort Tree 16
2.6 Min Queue 17
2.7 Persistent Seg 20
2.8 Treap 21
2.9 Union Find 31

3 Geometry **33**
3.1 2D 33
3.2 3D 48
3.3 Closest Pair Points 50
3.4 Convex Hull - Sweep Line 51
3.5 Graham Scan (convex hull) 52
3.6 Min Enclosing Circle 57
3.7 Rotating Calipers 58

4 Graph **61**
4.1 2-Sat 61
4.2 Biconnected Components 65
4.3 Bipartite Matching (Hopcroft Karp) 68
4.4 Bridges/Articulation Points 69
4.5 Centroid Decomposition 71
4.6 Euler Tour 76
4.7 Max Flow (Dinic) 84
4.8 Min Cost Max Flow 89
4.9 Min-Cut Global 91
4.10 Gomory Hu 91
4.11 Heavy-Light Decomposition 94
4.12 Strongly Connected Components 120
4.13 Transitive Closure 122
4.14 Tree Isomorphism 126

5 Misc **140**
5.1 MOs 140
5.2 Ternary Search 153

6 Number Theory **154**
6.1 CRT 154
6.2 Euclid 158
6.3 Modular Inverse 159
6.4 Modular Arithmetic 160
6.5 Phi 163
6.6 Sieve 164

7 Numerical **165**
7.1 FFT 165
7.2 Fraction 167
7.3 Integration 169
7.4 linalg 171
7.5 NTT 173
7.6 Simplex 175

8 String **178**
8.1 KMP 178
8.2 Acho Corasick 180
8.3 Hash 183
8.4 Suffix Array 185
8.5 Suffix Tree 189

1 Combinatorial

1.1 Markov Chains

```
// https://www.spoj.com/problems/PCPC12F/

#include "../../../numerical/linalg/mat.cpp"

#define mat mat<double>
#define vec vec<double>

#define MAXN 112

int prox[MAXN];

int main(void)
{
  int n, m, s, l, h, t;
  while (scanf("%d %d %d %d", &n, &m, &s, &l) != EOF)
  {
    memset(prox, -1, sizeof(prox));
    for (int i = 0; i < s + l; i++)
    {
      scanf("%d %d", &h, &t);
      prox[h] = t;
    }

    int nstates = n * m + 1;
```

```
    mat Q(nstates - 1, nstates - 1); // State nstates - 1 is final state.

    for (int i = 0; i < nstates - 1; i++)
    {
      for (int roll = 1; roll <= 6; roll++)
      {
        if (i + roll >= nstates)
          Q[i][i] += 1.0 / 6;
        else
        {
          int j = i + roll;
          while (j != nstates - 1 && prox[j] != -1)
            j = prox[j];

          if (j != nstates - 1)
            Q[i][j] += 1.0 / 6;
        }
      }
    }

    mat id(nstates - 1, nstates - 1);
    for (int i = 0; i < nstates - 1; i++)
      id[i][i] = 1;

    mat N = (id - Q).inverse();

    double ans = 0;
    for (int i = 0; i < nstates - 1; i++)
      ans += N[0][i];

    printf("%.3lf\n", ans);
  }
}
```

1.2 Lucas

```
// https://www.spoj.com/problems/DCEPC13D/
#include "../../../../number_theory/factor/pollardrho.cpp"
#include "../../../../number_theory/crt/crt_system.cpp"
#include "../lucas.cpp"

vector<ll> fact[51];
ll m[51];
ll a[51];

void calc()
{
    for (int p = 2; p < 51; p++)
    {
        fact[p].resize(51);
        if (isPrime(p))
        {
            fact[p][0] = fact[p][1] = 1;
            for (int i = 2; i < 51; i++)
            {
                fact[p][i] = (fact[p][i - 1] * (i % p)) % p;
            }
        }
    }
}

int main()
{
    int t;
    scanf("%d", &t);
    calc();
    while (t--)
    {
        ll M, R, N;
        scanf("%lld%lld%lld", &N, &R, &M);
        auto factors = factorize(M);

        for (int i = 0; i < sz(factors); i++)
        {
            a[i] = chooseModP(N, R, factors[i], fact[factors[i]]);
            m[i] = factors[i];
        }

        printf("%lld\n", crt_system(a, m, sz(factors)));
    }
}
```

```cpp
//https://www.spoj.com/problems/HLP_RAMS/

#include "../../../../contest/header.hpp"

int main()
{
    int t;
    scanf("%d", &t);
    while (t--)
    {
        // Lucas consequence: C(M,N)%2 is even when exits an indice i where Mb[i] < Nb[i]
        ll n;
        scanf("%lld", &n);
        ll cnt_1 = __builtin_popcountll(n);
        ll odd = (1 << cnt_1);
        ll even = n + 1 - odd;
        printf("%lld %lld\n", even, odd);
    }
}
```

1.3 Grundy

```
/*Problem G from https://codeforces.com/gym/101128*/

#include <bits/stdc++.h>
using namespace std;

int p, k;

vector<int> v;

int mex(vector<int> a){
  for(int i = 0; i < a.size(); i++){
    while(a[i] < a.size() && a[a[i]] != a[i])
      swap(a[a[i]], a[i]);
  }
  for(int i = 0; i < a.size(); i++)
    if(a[i] != i)
      return i;

  return a.size();
}

int grundy(){
  vector<int> g(v.size());

  for(int i = v.size() - 1; i >= 0; i--){
    vector<int> rea;
    for(int j = 0; j <= min(k, (int)v.size() - 1 - i); j++){
      if(i + j < v.size() && i + j + v[i + j] <= v.size()){
        if(i + j + v[i + j] == v.size())
          rea.push_back(0);
        else
          rea.push_back(g[i + j + v[i + j]]);
      }
    }

    g[i] = mex(rea);
  }

  return g[0];
}

int main(void)
{

  scanf("%d %d", &p, &k);
  int res = 0;
  for(int i = 0; i < p; i++){
    int n;
    scanf("%d", &n);
    v.clear();
    for(int j = 0; j < n; j++){
```

```
      int x;
      scanf("%d", &x);
      v.push_back(x);
    }
    reverse(v.begin(), v.end());
    res ^= grundy();
  }

  if(res == 0)
    printf("Bob will win.\n");
  else
    printf("Alice can win.\n");
}
```

2 Data Structures

2.1 BIT

```cpp
// https://codeforces.com/contest/992/problem/E
#include "../bit.cpp"

int main(void)
{
  int n, q;
  cin >> n >> q;
  vector<int> a(n + 1);
  bit<long long> bit(n);
  for (int i = 1; i <= n; i++)
  {
    scanf("%d", &a[i]);
    bit.update(i, a[i]);
  }

  int p, x;
  for (int i = 0; i < q; i++)
  {
    scanf("%d %d", &p, &x);
    bit.update(p, x - a[p]);
    a[p] = x;

    long long sum = 0;
    while (true)
    {
      p = bit.lower_bound(sum);
      if (p > n)
        break;

      if (bit.query(p) == 2 * bit.query(p - 1))
        break;

      sum = 2 * bit.query(p);
    }

    printf("%d\n", p > n ? -1 : p);
  }
}
```

```
10 7
0 3 1 4 6 2 7 8 10 1
2 5
1 3
9 36
4 10
4 9
1 2
1 0
```

2.2 BIT2D

```
// https://codeforces.com/problemset/problem/869/E
#include "../bit2d.cpp"

#define NHASH 3
#define MAXN 3123
int tab[MAXN][MAXN][NHASH];

int main(void)
{
  int n, m, q;
  scanf("%d %d %d", &n, &m, &q);
  srand(42);

  vector<bit2d<long long>> bit;
  for (int i = 0; i < NHASH; i++)
    bit.emplace_back(n, m);

  for (int i = 0; i < q; i++)
  {
    int tp, r1, r2, c1, c2;
    scanf("%d %d %d %d %d", &tp, &r1, &c1, &r2, &c2);
    if (tp == 1)
    {
      for (int j = 0; j < NHASH; j++)
      {
        assert(bit[j].query_rect(r1, c1, r2 + 1, c2 + 1) == 0);
        tab[r1][c1][j] = rand();
        bit[j].update(r1, c1, tab[r1][c1][j]);
        bit[j].update(r1, c2 + 1, -tab[r1][c1][j]);
        bit[j].update(r2 + 1, c1, -tab[r1][c1][j]);
        bit[j].update(r2 + 1, c2 + 1, tab[r1][c1][j]);
        assert(bit[j].query_rect(r1, c1, r2 + 1, c2 + 1) == 0);
      }
    }
    else if (tp == 2)
    {
      for (int j = 0; j < NHASH; j++)
      {
        assert(bit[j].query_rect(r1, c1, r2 + 1, c2 + 1) == 0);
        bit[j].update(r1, c1, -tab[r1][c1][j]);
        bit[j].update(r1, c2 + 1, tab[r1][c1][j]);
        bit[j].update(r2 + 1, c1, tab[r1][c1][j]);
        bit[j].update(r2 + 1, c2 + 1, -tab[r1][c1][j]);
        assert(bit[j].query_rect(r1, c1, r2 + 1, c2 + 1) == 0);
      }
    }
    else
    {
      bool ok = true;
      for (int j = 0; j < NHASH; j++)
        ok = ok && (bit[j].query(r1, c1) == bit[j].query(r2, c2));
```

```
      printf("%s\n", ok ? "Yes" : "No");
    }
  }
}
```

```
5 6 5
1 2 2 4 5
1 3 3 3 3
3 4 4 1 1
2 2 2 4 5
3 1 1 4 4
```

```
2500 2500 8
1 549 1279 1263 2189
1 303 795 1888 2432
1 2227 622 2418 1161
3 771 2492 1335 1433
1 2017 2100 2408 2160
3 48 60 798 729
1 347 708 1868 792
3 1940 2080 377 1546
```

2.3 Dynamic Seg

```
#include "../dynamic_seg.cpp"

#define node node<ll>

int main(void)
{
  int s = 1e5;
  int n, a, b, c, q, tp;
  cin >> n;
  node *root = new node();
  for (int i = 0; i < n; i++)
  {
    scanf("%d", &a);
    root->update(0, 2e5, s + i, a);
  }

  cin >> q;
  while(q--)
  {
    scanf("%d", &tp);
    if (tp == 1)
    {
      scanf("%d %d", &a, &b);
      printf("%lld\n", root->get(0, 2e5, s + a - 1, s + b - 1));
    }
    else
    {
      scanf("%d", &a);
      s--;
      root->update(0, 2e5, s, a);
    }
  }
}

5
6 7 8 9 10
9
2 5
2 4
1 2 7
2 3
2 2
2 1
1 1 10
1 1 1
1 10 10
```

2.4 Linear Container

```
// https://codeforces.com/contest/1179/problem/D

#include "../line_container.cpp"

#define MAXN 512345

vector<int> graph[MAXN];
int n;

void put_edge(int a, int b)
{
  graph[a].push_back(b);
  graph[b].push_back(a);
}

ll sub_size[MAXN];
ll ans;
ll tab[MAXN];

void dfs(int a, int p)
{
  sub_size[a] = 1;
  for (int i = 0; i < graph[a].size(); i++)
  {
    if (graph[a][i] != p)
    {
      dfs(graph[a][i], a);
      sub_size[a] += sub_size[graph[a][i]];
    }
  }

  tab[a] = sub_size[a] * (n - sub_size[a]);
  for (int i = 0; i < graph[a].size(); i++)
  {
    if (graph[a][i] != p)
    {
      tab[a] = max(tab[a], tab[graph[a][i]] + (sub_size[a] - sub_size[graph[a][i]])
   * (n - sub_size[a]));
    }
  }

  line_container l;
  l.add(0, 0);
  for (int i = 0; i < graph[a].size(); i++)
  {
    // ans = max(ans , tab[graph[a][i]]);
    // for (int j = i - 1; j >= 0; j--)
    // {
    //  ans = max(ans , tab[graph[a][i]] + tab[graph[a][j]] - sub_size[graph[a][i]]
   * sub_size[graph[a][j]]);
    // }
```

```
    ans = max(ans, l.query(sub_size[graph[a][i]]) + tab[graph[a][i]]);
    l.add(-sub_size[graph[a][i]], tab[graph[a][i]]);
  }
}

int main(void)
{
  int a, b;

  scanf("%d", &n);

  for (int i = 1; i < n; i++)
  {
    scanf("%d %d", &a, &b);
    put_edge(a, b);
  }

  if (n == 2)
  {
    printf("2\n");
    return 0;
  }

  int root;
  for (int i = 1; i < n; i++)
    if (graph[i].size() > 1)
      root = i;

  dfs(root, root);

  cout << (ans) + (n) * 1ll * (n-1) / 2 << endl;
}
```

```
6
1 2
1 3
3 4
3 5
4 6
```

```
4
1 2
1 3
1 4
```

```cpp
// https://codeforces.com/contest/1083/problem/E

#include "../line_container.cpp"

struct rect {
  ll x, y, c;

  bool operator < (rect rhs) const { return x < rhs.x; }
};

#define MAXN 1123456

ll tab[MAXN];
rect r[MAXN];
int n;

int main(void)
{
  scanf("%d", &n);
  for (int i = 1; i <= n; i++)
    scanf("%lld %lld %lld", &r[i].x, &r[i].y, &r[i].c);

  sort(r + 1, r + n + 1);

  line_container l;
  l.add(0, 0);

  ll ans = 0;
  for (int i = 1; i <= n; i++)
  {
    // tab[i] = -0x3f3f3f3f3f3f3fll;
    // for (int j = 0; j < i; j++)
    // tab[i] = max(tab[i], tab[j] - r[j].x * r[i].y + r[i].x * r[i].y - r[i].c);
    //
    // With convex hull trick:
    // max k * x + m
    // k = - r[j].x
    // m = tab[j]

    tab[i] = l.query(r[i].y) + r[i].x * r[i].y - r[i].c;
    ans = max(ans, tab[i]);
    l.add(-r[i].x, tab[i]);
  }

  printf("%lld\n", ans);
}
```

```
5
732540292 225231943 59578584627893686
370353847 368653517 104069404844594138
978010227 1498336 818018890670544
16695105 875794653 6779226035907661
219075646 809015132 81930182445683568
```

```
4
6 2 4
1 6 2
2 4 3
5 3 8
```

2.5 Merge Sort Tree

```
#include "../merge_sort_tree.cpp"

int main(void)
{
  int n, q, x, y, z;
  scanf("%d %d", &n, &q);
  vector<int> v(n);
  for (int i = 0; i < n; i++)
    scanf("%d", &v[i]);

  merge_sort_tree mst(v);
  for (int i = 0; i < q; i++)
  {
    scanf("%d %d %d", &x, &y, &z);
    printf("%d\n", mst.kth(x - 1, y - 1, z));
  }
}
```

2.6 Min Queue

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

using namespace std;
#define pb push_back
#define db(x) //cerr << #x << " = " << x << endl;
#define INF 0x3f3f3f3
#define fi first
#define se second
#define ll long long
#define vi vector<int>
#define vll vector<ll>
#define all(x) x.begin(), x.end()

#define MAXN 3123

ll G[MAXN*MAXN];
ll g0;
ll X, Y, Z;
ll grid[MAXN][MAXN];
ll grid_min[MAXN][MAXN];
ll N, M, A, B;

struct min_queue
{
  queue<ll> q;
  deque<ll> s;

  int size()
  {
    return (int)q.size();
  }

  void push(ll val)
  {
    while (!s.empty() && s.back() > val)
      s.pop_back();
    s.push_back(val);

    q.push(val);
  }

  void pop()
  {
    ll u = q.front();
    q.pop();

    if (!s.empty() && s.front() == u)
      s.pop_front();
  }

  ll get_min()
```

```
  {
    return s.front();
  }

};

void calc_G()
{
  G[0] = g0;
  for (int i = 1; i < MAXN*MAXN; i++)
  {
    G[i] = (G[i - 1] * X + Y) % Z;
  }
}

int main()
{
  scanf("%lld%lld%lld%lld", &N, &M, &A, &B);
  scanf("%lld%lld%lld%lld", &g0, &X, &Y, &Z);
  calc_G();
  for (int i = 1; i <= N; i++)
  {
    for (int j = 1; j <= M; j++)
    {
      grid[i][j] = G[(i - 1) * M + j - 1];
    }
  }

  // pre-calc min_grid
  for (int i = 1; i <= N; i++)
  {
    min_queue mq;
    for (int j = 1; j <= M; j++)
    {
      mq.push(grid[i][j]);
      if (mq.size() > B)
      {
        mq.pop();
      }
      ll m = mq.get_min();
      grid_min[i][j - B + 1] = m;
    }
  }

  // 1D sliding window in each column as an independent array
  ll res = 0;
  for (int j = 1; j + B - 1 <= M; j++)
  {
    min_queue mq;
    for (int i = 1; i <= N; i++)
    {
      mq.push(grid_min[i][j]);
      if (mq.size() > A)
```

```
        mq.pop();
      if (mq.size() == A)
        res += mq.get_min();
    }
  }

  printf("%lld\n", res);
}
```

2.7 Persistent Seg

```
// https://www.spoj.com/problems/MKTHNUM/
#include "../persistent_seg.cpp"

#define MAXN 112345

int a[MAXN];

int get_left(node *r)
{
  if (r && r->left)
    return r->left->val;
  return 0;
}

int query(node *r1, node *r2, int l, int r, int k)
{
  if (l == r)
    return l;

  int mid = (l + 0ll + r) / 2;

  int x = get_left(r2) - get_left(r1);
  if (k <= x)
    return query(r1 ? r1->left : r1, r2 ? r2->left : r2, l, mid, k);
  else
    return query(r1 ? r1->right : r1, r2 ? r2->right : r2, mid + 1, r, k - x);

}

int main(void)
{
  int MAXV = 1e9;
  int n, m;
  scanf("%d %d", &n, &m);
  vector<node *> roots = {new node()};
  for (int i = 0; i < n; i++)
  {
    scanf("%d", &a[i]);
    int v = roots.back()->get(0, 2*MAXV, a[i] + MAXV, a[i] + MAXV);
    roots.push_back(p_update(roots.back(), 0, 2*MAXV, a[i] + MAXV, v + 1));
  }

  for (int i = 0; i < m; i++)
  {
    int x, y, z;
    scanf("%d %d %d", &x, &y, &z);
    printf("%d\n", query(roots[x-1], roots[y], 0, 2*MAXV, z) - MAXV);
  }
}
```

2.8 Treap

```
// https://www.spoj.com/problems/GSS6/
#include "../../../contest/header.hpp"

namespace treap
{
struct node
{
  node *l = 0, *r = 0;
  int val; // Any value associated with node.
  ll tab;
  ll sum;
  ll pref;
  ll suf;

  int p;   // Node heap priority.
  int c = 1; // Node subtree size.
  node(int val) : val(val), tab(val), sum(val), pref(max(0, val)), suf(max(0, val)),
    p(rand()) {}
  void recalc();
};

int cnt(node *n) { return n ? n->c : 0; }
ll get_sum(node *n) { return n ? n->sum : 0; }
ll get_pref(node *n) { return n ? n->pref : 0; }
ll get_suf(node *n) { return n ? n->suf : 0; }
ll get_tab(node *n) { return n ? n->tab : -infll; }

void node::recalc() // To augment with extra data you should mostly add stuff to
   this function.
{
  c = cnt(l) + cnt(r) + 1;
  sum = get_sum(l) + get_sum(r) + val;
  pref = max(get_pref(l), get_sum(l) + get_pref(r) + val);
  suf = max(get_suf(r), get_sum(r) + get_suf(l) + val);
  tab = max(max(get_tab(l), get_tab(r)), get_suf(l) + val + get_pref(r));
}

// Apply function f on each tree node in order.
template <class F>
void each(node *n, F f)
{
  if (n)
  {
    each(n->l, f);
    f(n->val);
    each(n->r, f);
  }
}

// Split treap rooted at n in two treaps containing [0, k) and [k, ...)
pair<node *, node *> split(node *n, int k)
```

```
{
  if (!n)
    return {NULL, NULL};
  if (cnt(n->l) >= k) // "n->val >= k" for lower_bound(k)
  {
    auto pa = split(n->l, k);
    n->l = pa.second;
    n->recalc();
    return {pa.first, n};
  }
  else
  {
    auto pa = split(n->r, k - cnt(n->l) - 1); // and just "k"
    n->r = pa.first;
    n->recalc();
    return {n, pa.second};
  }
}

// Merge treaps l and r keeping order (l first).
node *merge(node *l, node *r)
{
  if (!l)
    return r;
  if (!r)
    return l;
  if (l->p > r->p)
  {
    l->r = merge(l->r, r);
    l->recalc();
    return l;
  }
  else
  {
    r->l = merge(l, r->l);
    r->recalc();
    return r;
  }
}

// Insert treap rooted at n into position pos of treap rooted at t.
// Also used to insert one node (e.g. root = ins(root, new node(10), 3))
node *ins(node *t, node *n, int pos)
{
  auto pa = split(t, pos);
  return merge(merge(pa.first, n), pa.second);
}

// Remove node at position pos from treap rooted at t.
node *rem(node *t, int pos)
{
  node *a, *b, *c;
  tie(a, b) = split(t, pos);
```

```
  tie(b, c) = split(b, 1);

  delete b;
  return merge(a, c);
}

// Do a query in range [l, r].
node *query(node *t, int l, int r)
{
  node *a, *b, *c;
  tie(a, b) = split(t, l);
  tie(b, c) = split(b, r - l + 1);

  printf("%lld\n", b->tab);

  return merge(merge(a, b), c);
}

// Example application: move the range [l, r) to index k.
void move(node *&t, int l, int r, int k)
{
  node *a, *b, *c;
  tie(a, b) = split(t, l);
  tie(b, c) = split(b, r - l);
  if (k <= l)
    t = merge(ins(a, b, k), c);
  else
    t = merge(a, ins(c, b, k - r));
}
} // namespace treap

int main(void)
{
  treap::node *root = nullptr;

  int n, q, x, y;
  scanf("%d", &n);
  for (int i = 0; i < n; i++)
  {
    scanf("%d", &x);
    root = treap::ins(root, new treap::node(x), i);
  }

  char tp;
  scanf("%d", &q);
  for (int i = 0; i < q; i++)
  {
    scanf(" %c", &tp);
    if (tp == 'I')
    {
      scanf("%d %d", &x, &y);
      x--;
      root = treap::ins(root, new treap::node(y), x);
```

```
    }
    else if (tp == 'D')
    {
      scanf("%d", &x);
      x--;
      root = treap::rem(root, x);
    }
    else if (tp == 'R')
    {
      scanf("%d %d", &x, &y);
      x--;
      root = treap::rem(root, x);
      root = treap::ins(root, new treap::node(y), x);
    }
    else
    {
      scanf("%d %d", &x, &y);
      x--; y--;
      root = treap::query(root, x, y);
    }
  }
}

5
3 -4 3 -1 6
10
I 6 2
Q 3 5
R 5 -4
Q 3 5
D 2
Q 1 5
I 2 -10
Q 1 6
R 2 -1
Q 1 6
```

```
// https://www.spoj.com/problems/GSS8/
#include "../../../contest/header.hpp"

typedef uint32_t ui;

ui pot[200001][11];
ui comb[11][11];

namespace treap
{
struct node
{
  node *l = 0, *r = 0;
  ui val; // Any value associated with node.
  ui tab[11];

  int p;   // Node heap priority.
  int c = 1; // Node subtree size.
  node(ui val) : val(val), p(rand()) {
    for (int i = 0; i <= 10; i++)
      tab[i] = val;
  }

  void recalc();
};

int cnt(node *n) { return n ? n->c : 0; }

void node::recalc() // To augment with extra data you should mostly add stuff to
   this function.
{
  c = cnt(l) + cnt(r) + 1;

  ui y = cnt(l) + 1;
  for (int k = 0; k <= 10; k++)
    tab[k] = val * pot[y][k];

  if (l)
    for (int k = 0; k <= 10; k++)
      tab[k] += l->tab[k];

  if (r)
    for (int k = 0; k <= 10; k++)
      for (int i = 0; i <= k; i++)
        tab[k] += r->tab[i] * pot[y][k - i] * comb[k][k - i];
}

// Apply function f on each tree node in order.
template <class F>
void each(node *n, F f)
{
  if (n)
  {
```

```
    each(n->l, f);
    f(n->val);
    each(n->r, f);
  }
}

// Split treap rooted at n in two treaps containing [0, k) and [k, ...)
pair<node *, node *> split(node *n, int k)
{
  if (!n)
    return {NULL, NULL};
  if (cnt(n->l) >= k) // "n->val >= k" for lower_bound(k)
  {
    auto pa = split(n->l, k);
    n->l = pa.second;
    n->recalc();
    return {pa.first, n};
  }
  else
  {
    auto pa = split(n->r, k - cnt(n->l) - 1); // and just "k"
    n->r = pa.first;
    n->recalc();
    return {n, pa.second};
  }
}

// Merge treaps l and r keeping order (l first).
node *merge(node *l, node *r)
{
  if (!l)
    return r;
  if (!r)
    return l;
  if (l->p > r->p)
  {
    l->r = merge(l->r, r);
    l->recalc();
    return l;
  }
  else
  {
    r->l = merge(l, r->l);
    r->recalc();
    return r;
  }
}

// Insert treap rooted at n into position pos of treap rooted at t.
// Also used to insert one node (e.g. root = ins(root, new node(10), 3))
node *ins(node *t, node *n, int pos)
{
  auto pa = split(t, pos);
```

```
  return merge(merge(pa.first, n), pa.second);
}

// Remove node at position pos from treap rooted at t.
node *rem(node *t, int pos)
{
  node *a, *b, *c;
  tie(a, b) = split(t, pos);
  tie(b, c) = split(b, 1);

  delete b;
  return merge(a, c);
}

// Do a query in range [l, r].
node *query(node *t, int l, int r, int k)
{
  node *a, *b, *c;
  tie(a, b) = split(t, l);
  tie(b, c) = split(b, r - l + 1);

  printf("%u\n", b->tab[k]);

  return merge(merge(a, b), c);
}

// Example application: move the range [l, r) to index k.
void move(node *&t, int l, int r, int k)
{
  node *a, *b, *c;
  tie(a, b) = split(t, l);
  tie(b, c) = split(b, r - l);
  if (k <= l)
    t = merge(ins(a, b, k), c);
  else
    t = merge(a, ins(c, b, k - r));
}
} // namespace treap

int main(void)
{
  for (int i = 0; i <= 10; i++)
    for (ui j = 0; j <= 200000; j++)
    {
      if (i == 0)
        pot[j][i] = 1;
      else
        pot[j][i] = pot[j][i-1] * j;
    }

  for (int i = 0; i <= 10; i++)
    for (int j = 0; j <= i; j++)
    {
```

```
      if (j == 0 || j == i)
        comb[i][j] = 1;
      else
      {
        comb[i][j] = comb[i-1][j] + comb[i-1][j-1];
      }
    }

  treap::node *root = nullptr;

  int n, q, x, y;
  ui v;
  scanf("%d", &n);
  for (int i = 0; i < n; i++)
  {
    scanf("%u", &v);
    root = treap::ins(root, new treap::node(v), i);
  }

  char tp;
  scanf("%d", &q);
  for (int i = 0; i < q; i++)
  {
    scanf(" %c", &tp);
    if (tp == 'I')
    {
      scanf("%d %u", &x, &v);
      root = treap::ins(root, new treap::node(v), x);
    }
    else if (tp == 'D')
    {
      scanf("%d", &x);
      root = treap::rem(root, x);
    }
    else if (tp == 'R')
    {
      scanf("%d %u", &x, &v);
      root = treap::rem(root, x);
      root = treap::ins(root, new treap::node(v), x);
    }
    else
    {
      int k;
      scanf("%d %d %d", &x, &y, &k);
      root = treap::query(root, x, y, k);
    }
  }
}

4
1 2 3 5
7
Q 0 2 0
```

```
I 3 4
Q 2 4 1
D 0
Q 0 3 1
R 1 2
Q 0 1 0
```

```
// http://www.spoj.com/problems/ORDERSET/
#include "../key_treap.cpp"

int main(void)
{
  int q, x;
  char tp;
  treap::node *root = NULL;
  scanf("%d", &q);
  while (q-- && scanf(" %c %d", &tp, &x))
  {
    if (tp == 'I')
      root = treap::insert(root, x);
    else if (tp == 'D')
      root = treap::remove(root, x);
    else if (tp == 'C')
      printf("%d\n", treap::count(root, x));
    else
    {
      if (treap::get_num(root) < x)
        printf("invalid\n");
      else
        printf("%d\n", treap::kth(root, x));
    }
  }
}

8
I -1
I -1
I 2
C 0
K 2
D -1
K 1
K 2
```

2.9 Union Find

```
// https://codeforces.com/gym/101556
#include "../union_find.cpp"
#include "../../../geometry/2d/2d.cpp"

#define point point<double>

ostream &operator<<(ostream &os, point p)
{
  return os << "(" << p.x << ", " << p.y << ")";
}

int main(void)
{
  int n, m, a, b;
  scanf("%d %d", &n, &m);
  vector<point> p(n);
  vector<vector<pii>> graph(n);
  for (int i = 0; i < n; i++)
    scanf("%lf %lf", &p[i].x, &p[i].y);

  for (int i = 0; i < m; i++)
  {
    scanf("%d %d", &a, &b);
    graph[a].push_back({b, i});
    graph[b].push_back({a, i});
  }

  union_find dsu(m);
  vector<double> diff(n);
  double ans = 0;
  for (int i = 0; i < n; i++)
  {
    if (graph[i].size() == 2)
    {
      ans += acos(-1) - abs((p[graph[i][0].first] - p[i]).angle(p[graph[i][1].first]
    - p[i]));
      diff[i] = 0;
      dsu.join(graph[i][0].second, graph[i][1].second);
    }
    else
    {
      vector<pair<double, int>> vals;
      for (int j = 1; j < 4; j++)
      {
        int k, l;
        if (3 + j == 6)
          k = 1, l = 2;
        else if (4 + j == 6)
          k = 1, l = 3;
        else
          k = 2, l = 3;
```

```
      double d1 = acos(-1) - abs((p[graph[i][0].first] - p[i]).angle(p[graph[i][j
  ].first] - p[i]));
      double d2 = acos(-1) - abs((p[graph[i][k].first] - p[i]).angle(p[graph[i][l
  ].first] - p[i]));

      vals.push_back({d1 + d2, j});
    }

    sort(vals.begin(), vals.end());
    diff[i] = vals[1].first - vals[0].first;
    ans += vals[0].first;

    for (int j = vals[0].second; j < 4; j = 4)
    {
      int k, l;
      if (3 + j == 6)
        k = 1, l = 2;
      else if (4 + j == 6)
        k = 1, l = 3;
      else
        k = 2, l = 3;

      dsu.join(graph[i][0].second, graph[i][j].second);
      dsu.join(graph[i][k].second, graph[i][l].second);
    }
  }
}

vector<pair<double, int>> pi;
for (int i = 0; i < n; i++)
  pi.push_back({diff[i], i});
sort(pi.begin(), pi.end());

for (int i = 0; i < n; i++)
{
  a = pi[i].second;
  for (int j = 0; j < sz(graph[a]); j++)
    for (int k = j + 1; k < sz(graph[a]); k++)
    {
      if (dsu.find(graph[a][j].second) != dsu.find(graph[a][k].second))
      {
        ans += diff[a];
        dsu.join(graph[a][j].second, graph[a][k].second);
      }
    }
}

printf("%.20lf\n", ans);
}
```

3 Geometry

3.1 2D

```
// https://codeforces.com/contest/438/problem/C
#include "../2d.cpp"

#define point point<ll>
#define segment segment<ll>
#define MAXN 212

const ll mod = 1e9 + 7;

point p[MAXN];
int n;

bool been[MAXN][MAXN];
ll tab[MAXN][MAXN];

ll polyside = 0;

ll pd(int l, int r)
{
  if (been[l][r])
    return tab[l][r];

  been[l][r] = true;

  bool ok = true;
  for (int k = 0; k < n; k++)
  {
    if (l != k && l != (k + 1) % n && r != k && r != (k + 1) % n)
    {
      segment s1 = segment(p[l], p[r]);
      segment s2 = segment(p[k], p[(k + 1) % n]);

      if (s2.intersect(s1).size())
      {
        ok = false;
      }
    }
    if (l != k && r != k)
    {
      segment s1 = segment(p[l], p[r]);
      segment s2 = segment(p[k], p[k]);

      if (s2.intersect(s1).size())
      {
        ok = false;
      }
    }
  }
```

```
  if (!ok)
  {
    return tab[l][r] = 0;
  }

  if (r - l <= 1)
    return tab[l][r] = 1;

  ll retv = 0;
  for (int i = l + 1; i < r; i++)
    if ((polyside * p[r].cross(p[l], p[(l + 1) % n]) <= 0 || polyside * p[l].cross(p
   [(l + 1) % n], p[i]) >= 0) &&
      (polyside * p[(r - 1 + n) % n].cross(p[r], p[l]) <= 0 || polyside * p[(r - 1 +
    n) % n].cross(p[r], p[i]) >= 0))
    {
      if (polyside * p[r].cross(p[l], p[i]) >= 0)
        retv = (retv + pd(l, i) * pd(i, r)) % mod;
    }

  return tab[l][r] = retv;
}

int main(void)
{
  cin >> n;
  for (int i = 0; i < n; i++)
    cin >> p[i].x >> p[i].y;

  point p0(0, 0);
  ll sum = 0;
  for (int i = 0; i < n; i++)
    sum += p0.cross(p[i], p[(i + 1) % n]);

  polyside = (sum > 0) ? 1 : -1;

  cout << pd(0, n - 1) << endl;
}
```

```
10
10 -10
6 -2
8 6
6 5
1 4
-2 6
-3 8
-6 -2
5 -9
9 -10



40
5 11
```

```
19 17
23 5
34 10
40 22
34 25
39 37
13 40
-7 39
-38 39
-3 16
1 15
-3 15
-34 14
-33 2
-27 0
-27 -3
-18 -3
-37 -11
-37 -13
-22 -5
-20 -9
-24 -29
-5 0
-10 -12
-16 -28
-15 -34
-10 -26
-11 -22
31 -36
-9 -12
-5 -13
38 -37
15 -16
6 0
0 7
18 -6
30 -17
40 -25
30 -15
```

```
5
0 0
1 0
1 1
0 1
-2 -1
```

```
4
0 0
1 0
```

```
0 1
-1 0

4
0 0
0 1
1 1
1 0
```

```
// https://codeforces.com/problemset/problem/681/E
#include "../2d.cpp"

#define point point<double>
#define segment segment<double>
#define circle circle<double>
#define MAXN 112345

#define M_PI 3.14159265358979323846

circle c[MAXN];

int main(void)
{
  point p0;
  double v, t;
  int n;
  cin >> p0.x >> p0.y >> v >> t >> n;
  for (int i = 0; i < n; i++)
  {
    scanf("%lf %lf %lf", &c[i].center.x, &c[i].center.y, &c[i].r);
    c[i].center = c[i].center - p0;
    if (c[i].center.dist() < c[i].r + EPS)
    {
      printf("1\n");
      return 0;
    }
  }

  p0 = p0 - p0;
  circle safe = {p0, v * t};
  vector<pair<double, double>> arcs;
  for (int i = 0; i < n; i++)
  {
    if (c[i].center.dist() < c[i].r + safe.r - EPS)
    {
      pair<point, point> sf;
      sf = c[i].tangents(p0);
      if (sf.first.dist() > safe.r + EPS)
      {
        if (c[i].intersect(safe, sf) <= 0)
        {
          debug(safe.center, safe.r);
          debug(sf);
          debug(c[i].center, c[i].r);
          assert(false);
        }
      }

      pair<double, double> tmp = {sf.first.angle(), sf.second.angle()};

      if (tmp.first > tmp.second)
        swap(tmp.first, tmp.second);
```

```
      if (tmp.second - tmp.first < M_PI)
        arcs.push_back(tmp);
      else
      {
        arcs.push_back({-M_PI, tmp.first});
        arcs.push_back({tmp.second, M_PI});
      }
    }
  }

  double ans = 0;
  sort(arcs.begin(), arcs.end());
  for (int i = 0; i < arcs.size(); i++)
  {
    double l = arcs[i].first;
    double r = arcs[i].second;
    while (i + 1 < arcs.size() && arcs[i + 1].first <= r)
      r = max(r, arcs[++i].second);

    ans += r - l;
  }

  printf("%.15lf\n", ans / (2 * M_PI));
}
```

```
0 0 56484 14541
1
660338024 488408755 953
```

```
0 0 1 0
1
1 0 1
```

```
0 0 1 1
3
1 1 1
-1 -1 1
-2 2 1
```

```cpp
//Testado com o geogebra
//Valores usados/Saidas esperadas: (in )

#include "../2d.cpp"

#define point point<double>
#define circle circle<double>

int main(){
  int n = 5;
  while(n--){
    point c1, c2;
    double r1, r2;
    cin >> c1.x >> c1.y >> r1;
    cin >> c2.x >> c2.y >> r2;

    circle cir1, cir2;

    cir1.center = c1;
    cir1.r = r1;

    cir2.center = c2;
    cir2.r = r2;

    pair<point, point> pi1, pi2, pe1, pe2;

    cir1.outter_tangents(cir2, pe1, pe2);
    cir1.inner_tangents(cir2, pi1, pi2);

    printf("(%.2f %.2f)(%.2f %.2f)(%.2f %.2f)(%.2f %.2f)\n", pi1.first.x, pi1.first.
   y, pi1.second.x, pi1.second.y,
    pi2.first.x, pi2.first.y, pi2.second.x, pi2.second.y);

    printf("(%.2f %.2f)(%.2f %.2f)(%.2f %.2f)(%.2f %.2f)\n", pe1.first.x, pe1.first.
   y, pe1.second.x, pe1.second.y,
    pe2.first.x, pe2.first.y, pe2.second.x, pe2.second.y);

    cout << endl;
  }
}
```

```
-2 2 1.5
-2 6 2

-3 -2 2
2 -2 1

0 0 1
2 2 1

0 0 2
0 -1 1
```

```
0 0 1
2 0 1

(-2.73 3.31)(-1.27 3.31)(-1.03 4.25)(-2.97 4.25)
(-3.49 1.81)(-0.51 1.81)(-3.98 5.75)(-0.02 5.75)

(-1.80 -0.40)(-1.80 -3.60)(1.40 -2.80)(1.40 -1.20)
(-2.60 -0.04)(-2.60 -3.96)(2.20 -1.02)(2.20 -2.98)

(-0.00 1.00)(1.00 -0.00)(2.00 1.00)(1.00 2.00)
(-0.71 0.71)(0.71 -0.71)(1.29 2.71)(2.71 1.29)

(nan nan)(-nan -nan)(nan nan)(-nan -nan)
(0.00 -2.00)(-2.00 -2.00)(0.00 -2.00)(-2.00 -2.00)

(1.00 0.00)(1.00 -1.00)(1.00 0.00)(1.00 -1.00)
(0.00 1.00)(0.00 -1.00)(2.00 1.00)(2.00 -1.00)
```

```
// https://open.kattis.com/problems/segmentintersection
#include "../2d.cpp"

#define point point<double>
#define segment segment<double>

int main(void)
{
  segment s1, s2;
  int n;
  scanf("%d", &n);
  while(n--)
  {
    scanf("%lf %lf %lf %lf %lf %lf %lf %lf", &s1.pi.x, &s1.pi.y, &s1.pf.x, &s1.pf.y,
    &s2.pi.x, &s2.pi.y, &s2.pf.x, &s2.pf.y);
    auto v = s1.intersect(s2);
    if (v.size() == 0)
      printf("none\n");
    else if (v.size() == 1)
      printf("%.2lf %.2lf\n", (abs(v[0].x) < 0.005) ? 0.0 : v[0].x, (abs(v[0].y) <
   0.005) ? 0.0 : v[0].y);
    else
    {
      printf("%.2lf %.2lf %.2lf %.2lf\n",  (abs(v[0].x) < 0.005) ? 0.0 : v[0].x, (
   abs(v[0].y) < 0.005) ? 0.0 : v[0].y, (abs(v[1].x) < 0.005) ? 0.0 : v[1].x, (abs(v
   [1].y) < 0.005) ? 0.0 : v[1].y);
    }
  }
}
```

```
5
-10 0 10 0 0 -10 0 10
-10 0 10 0 -5 0 5 0
1 1 1 1 1 1 2 1
1 1 1 1 2 1 2 1
1871 5789 216 -517 189 -518 3851 1895
```

```cpp
// https://codeforces.com/gym/101554/problem/H

#include "../2d.cpp"

#define point point<ll>

struct segment_id : public segment<ll>
{
  int id;

  bool operator<(const segment_id &rhs) const
  {
    if (pi.x <= rhs.pi.x && rhs.pi.x <= pf.x)
      return pi.cross(pf, rhs.pi) > 0;
    else if (pi.x <= rhs.pf.x && rhs.pf.x <= pf.x)
      return pi.cross(pf, rhs.pf) > 0;
    else
      return rhs.pi.cross(rhs.pf, pi) < 0;
  }
};

struct event
{
  ll x, y;
  int id, tp;

  bool operator<(const event &rhs) const
  {
    return make_tuple(x, tp, -y) < make_tuple(rhs.x, rhs.tp, -rhs.y);
  }
};

int main(void)
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int n;
  cin >> n;
  vector<segment_id> s(n);
  for (int i = 0; i < n; i++)
  {
    cin >> s[i].pi.x >> s[i].pi.y >> s[i].pf.x >> s[i].pf.y;
    s[i].id = i;
    if (s[i].pf < s[i].pi)
      swap(s[i].pi, s[i].pf);
  }

  ll x_ball;
  int ball_first_seg;
  cin >> x_ball;

  vector<int> prox_seg(n);
```

```
vector<event> e;
e.push_back({x_ball, 10000000ll, -1, 1});

for (int i = 0; i < n; i++)
{
  e.push_back({s[i].pi.x, s[i].pi.y, i, 0});
  e.push_back({s[i].pf.x, s[i].pf.y, i, 2});
}

sort(e.begin(), e.end());
set<segment_id> cur;
for (int i = 0; i < sz(e); i++)
{
  if (e[i].tp == 0)
  {
    if (s[e[i].id].pi.y < s[e[i].id].pf.y)
    {
      auto it = cur.lower_bound(s[e[i].id]);
      if (it != cur.begin())
        prox_seg[e[i].id] = (--it)->id;
      else
        prox_seg[e[i].id] = -1;
    }

    cur.insert(s[e[i].id]);
  }
  else if (e[i].tp == 2)
  {
    cur.erase(s[e[i].id]);

    if (s[e[i].id].pf.y < s[e[i].id].pi.y)
    {
      auto it = cur.lower_bound(s[e[i].id]);
      if (it != cur.begin())
        prox_seg[e[i].id] = (--it)->id;
      else
        prox_seg[e[i].id] = -1;
    }
  }
  else
  {
    auto it = cur.rbegin();
    if (it != cur.rend())
      ball_first_seg = it->id;
    else
      ball_first_seg = -1;
  }
}

// debug(ball_first_seg);
// debug(prox_seg);

ll ans = x_ball;
```

```
  for (int i = ball_first_seg; i != -1; i = prox_seg[i])
  {
    ans = (s[i].pi.y < s[i].pf.y ? s[i].pi.x : s[i].pf.x);
  }

  cout << ans << endl;
}
```

```
// https://www.spoj.com/problems/ELASTIC/

#include<bits/stdc++.h>
#include "../2d.cpp"

using namespace std;

#define point point<double>
#define circle circle<double>
#define segment segment<double>
#define line line<double>

vector<vector<int> > adj;
vector<vector<double> > weight;
bool vis[3123];

priority_queue<pair<double, int> > pq;

void process(int x){
  vis[x] = 1;
  for(int i = 0; i < (int)adj[x].size(); i++){
    int nx = adj[x][i];
    double w = weight[x][i];

    if(vis[nx] == 0){
      pq.push({-w, nx});
    }
  }
}

double angle(point a, point o, point b){
  return abs((a - o).angle(b - o));
}

int main(){

  int n;

  scanf("%d", &n);
  while(1){
    if(n == -1)
      break;

    vector<circle> c(n);

    for(int i = 0; i < n; i++)
      scanf("%lf %lf %lf", &c[i].center.x, &c[i].center.y, &c[i].r);

    adj.clear();
    adj.resize(n);
    weight.clear();
    weight.resize(n);
```

```
for(int i = 0; i < n; i++)
  for(int j = i + 1; j < n; j++){
    pair<point, point> pi, pj;
    c[i].outter_tangents(c[j], pi, pj);
    double val = 0;
    double angi = abs(angle(pi.first, c[i].center, pi.second));
    double angj = abs(angle(pj.first, c[j].center, pj.second));

    //printf("%lf %lf\n", angi*360/(2*acos(-1)), angj*360/(2*acos(-1)));

    if(c[i].r < c[j].r){
      val += c[i].r*min(angi, 2*acos(-1) - angi);
      val += c[j].r*max(angj, 2*acos(-1) - angj);
    }
    else{
      val += c[i].r*max(angi, 2*acos(-1) - angi);
      val += c[j].r*min(angj, 2*acos(-1) - angj);
    }

    //printf("vala: %lf\n", val);

    val += (pi.first - pj.first).dist() + (pi.second - pj.second).dist();

    assert(!isnan(val));

    adj[i].push_back(j);
    weight[i].push_back(val);

    adj[j].push_back(i);
    weight[j].push_back(val);
  }

memset(vis, 0, sizeof(vis));

process(0);

double res = 0;

while(!pq.empty()){
  int at = pq.top().second;
  double w = -pq.top().first;
  pq.pop();

  if(vis[at] == 0){
    vis[at] = 1;
    res += w;
    process(at);
  }
}

printf("%.3lf\n", res);

scanf("%d", &n);
```

```
  }
  return 0;
}
```

3.2 3D

```
#include "../3d.cpp"

#define RADIUS 6378

map<string, int> str2id;
map<int, string> id2str;
vector<double> vet_lat, vet_longi;

double to_rad(double a)
{
    double res = a * M_PI / 180.0;
    while (res < 0)
    {
        res += 2 * M_PI;
    }
    return res;
}

int main()
{
    string city;
    int id = 0;
    while (cin >> city, city != "#")
    {
        double lat, longi;
        cin >> lat >> longi;
        vet_lat.push_back(lat);
        vet_longi.push_back(longi);
        str2id[city] = id;
        id2str[id] = city;
        ++id;
    }

    string u, v;
    while (cin >> u >> v, u != "#")
    {
        cout << u << " - " << v << endl;
        if (str2id.count(u) == 0 || str2id.count(v) == 0)
            cout << "Unknown" << endl;
        else
        {
            double res = spherical_distance(to_rad(vet_longi[str2id[u]]),
                                            to_rad(vet_lat[str2id[u]]),
                                            to_rad(vet_longi[str2id[v]]),
                                            to_rad(vet_lat[str2id[v]]), RADIUS);
            cout << int(round(res)) << " km" << endl;
        }
    }
}
```

```cpp
// https://open.kattis.com/problems/airlinehub
#include "../3d.cpp"

double to_rad(double a)
{
    double res = a * M_PI / 180.0;
    while (res < 0)
    {
        res += 2 * M_PI;
    }
    return res;
}

int main()
{
    ll n;
    while (scanf("%lld", &n) != EOF)
    {
        vector<pair<double, double>> vet_points(n);
        for (int i = 0; i < n; i++)
        {
            scanf("%lf%lf", &vet_points[i].first, &vet_points[i].second);
        }

        double best_v = 1e9;
        int best_e;
        for (int i = 0; i < n; i++)
        {
            double max_dist = 0;
            for (int j = 0; j < n; j++)
            {
                max_dist = max(max_dist, spherical_distance(to_rad(vet_points[i].
   second), to_rad(vet_points[i].first),
                                                            to_rad(vet_points[j].
   second), to_rad(vet_points[j].first), 1));
            }

            if (max_dist <= best_v)
            {
                best_v = max_dist;
                best_e = i;
            }
        }

        printf("%.2lf %.2lf\n", vet_points[best_e].first, vet_points[best_e].second)
   ;
    }
}
```

3.3 Closest Pair Points

```
// https://www.spoj.com/problems/CLOPPAIR/

#include "../closest_pair.cpp"

point p[MAXN];

int main(void)
{
  int n;
  scanf("%d", &n);
  for (int i = 0; i < n; p[i].id = i, i++)
    scanf("%lld %lld", &p[i].x, &p[i].y);

  closest_pair::closest_pair(p, n);
  printf("%d %d %.6lf\n", closest_pair::idx[0], closest_pair::idx[1], sqrt(
   closest_pair::ans));
}
```

3.4 Convex Hull - Sweep Line

```
// https://codeforces.com/gym/101128/my

#include "../convex_hull.cpp"

int main(void)
{
  int n, q;
  scanf("%d", &n);
  vector<point> p(n);
  for (int i = 0; i < n; i++)
    scanf("%lld %lld", &p[i].x, &p[i].y);

  vector<point> upper, lower;
  convex_hull(p, upper, lower);

  scanf("%d", &q);
  int ans = 0;
  while (q--)
  {
    point o;
    scanf("%lld %lld", &o.x, &o.y);
    if (point_in_ch(o, upper, lower))
      ans++;
  }

  printf("%d\n", ans);
}
```

3.5 Graham Scan (convex hull)

```
/*Problem: https://icpcarchive.ecs.baylor.edu/index.php?option=com_onlinejudge&
   Itemid=8&page=show_problem&problem=2559*/

#include<bits/stdc++.h>

typedef long long ll;

using namespace std;

template<typename T>
struct point
{
  typedef point<T> P;
  T x, y;

  explicit point(T x = 0, T y = 0) : x(x), y(y) {}
  //Double version: bool operator<(P p) const { return  fabs(x - p.x) < EPS ? y < p.
   y : x < p.x; }
  bool operator<(P p) const { /*return tie(x, y) < tie(p.x, p.y);*/ return y != p.y
   ? y > p.y : x < p.x; }
  //Double version: bool operator==(P p) const { return fabs(x - p.x) < EPS && fabs(
   y - p.y) < EPS; }
  bool operator==(P p) const { return tie(x, y) == tie(p.x, p.y); }
  P operator+(P p) const { return P(x + p.x, y + p.y); }
  P operator-(P p) const { return P(x - p.x, y - p.y); }
  T dot(P p) const { return x * p.x + y * p.y; }
  T cross(P p) const { return x * p.y - y * p.x; }
  T cross(P a, P b) const { return (a - *this).cross(b - *this); }
};

template<typename T>
bool cmp(point<T> a, point<T> b){
  if(a.cross(b) != 0)
    return a.cross(b) > 0;
  return a.x*a.x + a.y*a.y < b.x*b.x + b.y*b.y;
}

template<typename T>
vector<point<T> > CH(vector<point<T> > points){
  point<T> pivot = points[0];
  for(auto p : points)
    pivot = min(pivot, p);

  for(int i = 0; i < (int) points.size(); i++)
    points[i] = points[i] - pivot;

  sort(points.begin(), points.end(), cmp<ll>);

  for(int i = 0; i < (int) points.size(); i++)
    points[i] = points[i] + pivot;
```

```
  points.push_back(points[0]);

  vector<point<T> > ch;

  for(auto p : points){
    //Trocar segunda comparacao pra <= para discartar pontos do meio de arestas no
   ch
    //Double: trocar segunda comparaÃğÃčo por < EPS (descarta pontos em arestas)
    while(ch.size() > 1 && !(p == ch[ch.size() - 2]) && ch[ch.size() - 2].cross(ch[
   ch.size() - 1] , p) <= 0)
      ch.pop_back();
    ch.push_back(p);
  }

  ch.pop_back();

  return ch;
}

int main(){
  int t;
  scanf("%d", &t);

  for(int cse = 1; cse <= t; cse++){
    int x, n;
    scanf("%d %d", &x, &n);
    vector<point<ll> > p(n);

    for(int i = 0; i < n; i++){
      scanf("%lld %lld", &p[i].x, &p[i].y);
    }

    vector<point<ll> > ch = CH(p);

    printf("%d %d\n", x, (int)ch.size());

    vector<point<ll> > res;

    res.push_back(ch[0]);
    for(int i = ch.size() - 1; i > 0; i--)
      res.push_back(ch[i]);

    for(int i = 0; i < (int)ch.size(); i++)
      printf("%lld %lld\n", res[i].x, res[i].y);
  }

}
```

```cpp
/*Problem: https://codeforces.com/group/3qadGzUdR4/contest/101706/problem/G*/
/*Group: https://codeforces.com/group/3qadGzUdR4/members*/

#include<bits/stdc++.h>

typedef long long ll;

using namespace std;

template<typename T>
struct point
{
  typedef point<T> P;
  T x, y;
  int label;

  explicit point(T x = 0, T y = 0, int label = -1) : x(x), y(y), label(label) {}
  //Double version: bool operator<(P p) const { return  fabs(x - p.x) < EPS ? y < p.
   y : x < p.x; }
  bool operator<(P p) const { return tie(x, y) < tie(p.x, p.y); }
  //Double version: bool operator==(P p) const { return fabs(x - p.x) < EPS && fabs(
   y - p.y) < EPS; }
  bool operator==(P p) const { return tie(x, y) == tie(p.x, p.y) && label == p.label
   ; }
  P operator+(P p) const { return P(x + p.x, y + p.y, label); }
  T dist2() const { return x*x + y*y; }
  P operator-(P p) const { return P(x - p.x, y - p.y, label); }
  T dot(P p) const { return x * p.x + y * p.y; }
  T cross(P p) const { return x * p.y - y * p.x; }
  T cross(P a, P b) const { return (a - *this).cross(b - *this); }
  long double dist() const { return sqrt((long double)dist2()); }
};

template<typename T>
bool cmp(point<T> a, point<T> b){
  if(a.cross(b) != 0)
    return a.cross(b) > 0;
  return a.dist2() < b.dist2();
}

template<typename T>
vector<point<T> > CH(vector<point<T> > points){
  point<T> pivot = points[0];
  for(auto p : points)
    pivot = min(pivot, p);

  for(int i = 0; i < (int) points.size(); i++)
    points[i] = points[i] - pivot;

  sort(points.begin(), points.end(), cmp<ll>);

  for(int i = 0; i < (int) points.size(); i++)
    points[i] = points[i] + pivot;
```

```
  points.push_back(points[0]);

  vector<point<T> > ch;

  for(auto p : points){
    //Trocar segunda comparacao pra <= para discartar pontos do meio de arestas no
   ch
    //Double: trocar segunda comparaÃğÃčo por < EPS (descarta pontos em arestas)
    while(ch.size() > 1 && !(p == ch[ch.size() - 2]) && ch[ch.size() - 2].cross(ch[
   ch.size() - 1] , p) <= 0)
      ch.pop_back();
    ch.push_back(p);
  }

  ch.pop_back();

  return ch;
}
template<typename T>
T areaPol2(vector<point<T> > pol){
  T area = 0;
  for(int i = 0; i < (int)pol.size() - 1; i++)
    area += pol[i].cross(pol[i+1]);

  area += pol[pol.size() - 1].cross(pol[0]);
  return area;
}

int main(){
  int n;
  scanf("%d", &n);
  vector<point<ll> > p(n);

  for(int i = 0; i < n; i++){
    scanf("%lld %lld", &p[i].x, &p[i].y);
    p[i].label = i + 1;
  }

  vector<point<ll> > ch = CH(p);

  printf("%d\n", (int)ch.size());

  for(int i = 0; i < (int)ch.size(); i++)
    printf("%d%c", ch[i].label, " \n"[i == ch.size() - 1]);

  long double diam = 0;
  for(int i = 0; i < (int)ch.size() - 1; i++)
    diam += (ch[i] - ch[i+1]).dist();
  diam += (ch[0] - ch[ch.size() - 1]).dist();

  printf("%.12Lf\n", diam);
```

```
  //Importante imprimir assim, double com uma casa da problema (exemplo: pontos
   (0,0);(1,1);(0,1)
  printf("%lld", areaPol2(ch)/2);
  if(areaPol2(ch)%2)
    printf(".5\n");
  else
    printf(".0\n");


}
```

3.6 Min Enclosing Circle

```
// https://codeforces.com/gym/102299

#include "../randomized.cpp"
// #include "../ternary_search.cpp"

int main(void)
{
  int n;
  scanf("%d", &n);
  vector<point> p(n);
  for (int i = 0; i < n; i++)
    scanf("%lf %lf", &p[i].x, &p[i].y);

  if (n == 0)
    printf("0 0 0\n");
  else
  {
    circle ans = min_enclosing_circle(p/*, -1e4, 1e4, 1e-7*/);
    printf(" %.15lf %.15lf %.15lf\n", ans.center.x, ans.center.y, ans.r * ans.r / 2)
   ;
  }
}

5
0 0
0 1
1 0
1 1
2 2
2
0 0
4 0
0
```

3.7 Rotating Calipers

```
// https://open.kattis.com/problems/fenceortho
// IMPORTANT: need to change convex hull code to pass (no collinear points).

#include "../convex_polygon_bounding_box.cpp"

int main(void)
{
  int n;
  while (cin >> n && n)
  {
    vector<point<ll>> p(n);
    for (int i = 0; i < n; i++)
      cin >> p[i].x >> p[i].y;

    auto ch = CH(p);
    double ans = min_bounding_box_perimeter(ch);

    printf("%.20lf\n", ans);
  }
}
```

```cpp
// https://open.kattis.com/problems/roberthood
// IMPORTANT: need to change convex hull code to pass (no collinear points).

#include "../convex_polygon_diameter.cpp"

int main(void)
{
  int n;
  cin >> n;
  vector<point<ll>> p(n);
  for (int i = 0; i < n; i++)
    cin >> p[i].x >> p[i].y;

  auto ch = CH(p);

  double ans = convex_polygon_diameter(ch);;

  printf("%.20lf\n", ans);
}
```

```
// https://onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=
   show_problem&problem=3552
// IMPORTANT: need to change convex hull code to pass (no collinear points).

#include "../convex_polygon_width.cpp"

int main(void)
{
  int n, t = 1;
  while (cin >> n && n)
  {
    vector<point<ll>> p(n);
    for (int i = 0; i < n; i++)
      cin >> p[i].x >> p[i].y;

    auto ch = CH(p);
    double ans = convex_polygon_width(ch);

    printf("Case %d: %.2lf\n", t++, ans);
  }
}
```

4 Graph

4.1 2-Sat

```cpp
// https://codeforces.com/gym/101968/problem/J
#include "../2sat.cpp"

int main(void)
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int T;
  cin >> T;
  while (T--)
  {
    int n, m, a, b, c;
    cin >> n >> m;
    two_sat st(n);
    for (int i = 0; i < m; i++)
    {
      cin >> a >> b >> c;
      a--, b--;
      st.either(a, b);
      if (c)
        st.at_most_one({a, b});
    }

    cout << (st.solve() ? "YES" : "NO") << endl;
  }
}
```

```
// https://www.spoj.com/problems/TORNJEVI/
#include "../2sat.cpp"

#define MAXN 112

char grid[MAXN][MAXN];
int num[MAXN][MAXN];

enum dirs { down, up, right, left };
int dx[4] = {1, -1, 0, 0};
int dy[4] = {0, 0, 1, -1};
int ans[2][2] = {{4,3}, {1,2}};

int go(int i, int j, int dir)
{
  do
  {
    i += dx[dir];
    j += dy[dir];
  } while (grid[i][j] != '#' && grid[i][j] != 'T');

  return (grid[i][j] == 'T') ? num[i][j] : -1;
}

int var(int x, int dir)
{
  int tmp = 2*x+dir/2;
  return (dir&1) ? (~tmp) : tmp;
}

int main(void)
{
  int n, m;
  scanf("%d %d", &n, &m);
  memset(grid, '#', sizeof(grid));

  int tw = 0;
  for (int i = 1; i <= n; i++)
    for (int j = 1; j <= m; j++)
      scanf(" %c", &grid[i][j]);

  for (int i = 1; i <= n; i++)
    for (int j = 1; j <= m; j++)
      if (grid[i][j] == 'T')
        num[i][j] = tw++;

  two_sat st(2*tw);

  for (int i = 1; i <= n; i++)
    for (int j = 1; j <= m; j++)
      if (grid[i][j] == 'T')
        for (int k = 0; k < 4; k++)
        {
```

```
        int y = go(i, j, k);
        if (y != -1)
          st.set_true(~var(num[i][j], k));
      }

  for (int i = 1; i <= n; i++)
    for (int j = 1; j <= m; j++)
      if (grid[i][j] == 'n')
      {
        int l = go(i, j, dirs::left);
        int r = go(i, j, dirs::right);
        int d = go(i, j, dirs::down);
        int u = go(i, j, dirs::up);

        if ((l == -1 && r == -1) || (l != -1 && r != -1))
        {
          if (u != -1)
            st.set_true(var(u, down));
          else
            st.set_true(var(d, up));
        }
        else if ((d == -1 && u == -1) || (d != -1 && u != -1))
        {
          if (l != -1)
            st.set_true(var(l, dirs::right));
          else
            st.set_true(var(r, dirs::left));
        }
        else
        {
          if (l != -1)
          {
            if (u != -1)
              st.either(var(l, dirs::right), var(u, down));
            else
              st.either(var(l, dirs::right), var(d, up));
          }
          else
          {
            if (u != -1)
              st.either(var(r, dirs::left), var(u, down));
            else
              st.either(var(r, dirs::left), var(d, up));
          }
        }
      }

  st.solve();
  for (int i = 1; i <= n; i++, printf("\n"))
    for (int j = 1; j <= m; j++)
      if (grid[i][j] == 'T')
      {
        printf("%d", ans[st.values[var(num[i][j], dirs::down)]][st.values[var(num[i
```

```
   ][j], dirs::right)]]);
      }
      else
      {
        printf("%c", grid[i][j]);
      }
}
```

4.2 Biconnected Components

```cpp
// https://codeforces.com/gym/101492/problem/G

#include "../biconnected_components.cpp"

#define MAXN 51234
vector<int> graph[MAXN];

int main(void)
{
  int n, m, a, b;
  scanf("%d %d", &n, &m);
  for (int i = 0; i < m; i++)
  {
    scanf("%d %d", &a, &b);
    graph[a].push_back(b);
    graph[b].push_back(a);
  }

  int ans = 0;
  auto rdm = apb(1, n, graph, [&](vector<pii> v){
    set<int> s;
    for (int i = 0; i < sz(v); i++)
    {
      s.insert(v[i].first);
      s.insert(v[i].second);
    }

    ans = max(ans, sz(s));
  });

  cout << ans << endl;
}
```

```cpp
// https://www.urionlinejudge.com.br/judge/pt/problems/view/2199

#include "../biconnected_components.cpp"

#define MAXN 51234
vector<int> graph[MAXN];

int main(void)
{
  int m, n, a, b, T = 0;
  while(scanf("%d", &m) && m != 0)
  {
    n = 0;
    for (int i = 0; i <= m + 1; i++)
      graph[i].clear();

    for (int i = 0; i < m; i++)
    {
      scanf("%d %d", &a, &b);
      graph[a].push_back(b);
      graph[b].push_back(a);
      n = max(a, n);
      n = max(b, n);
    }

    vector<vector<int>> bcc;
    apb sol(1, n, graph, [&](vector<pii> v){
      set<int> s;
      for (int i = 0; i < sz(v); i++)
      {
        s.insert(v[i].first);
        s.insert(v[i].second);
      }

      bcc.push_back(vector<int>(s.begin(), s.end()));
    });

    printf("Case %d: ", ++T);
    if (bcc.size() == 1)
      printf("2 %lld\n", n*1ll*(n-1)/2);
    else
    {
      int ans1 = 0;
      long long ans2 = 1;
      for (int i = 0; i < sz(bcc); i++)
      {
        int cnt = 0;
        for (int v : bcc[i])
          if (sol.art[v])
            cnt++;
        if (cnt == 1)
        {
          ans1++;
```

```
            ans2 *= (sz(bcc[i]) - 1);
          }
        }

        printf("%d %lld\n", ans1, ans2);
      }
    }
}
```

```
9
1 3
4 1
3 5
1 2
2 6
1 5
6 3
1 6
3 2
6
1 2
1 3
2 4
2 5
3 6
3 7
0
```

4.3 Bipartite Matching (Hopcroft Karp)

```
// http://www.spoj.com/problems/MATCHING/

#include "../hopcroft_karp.cpp"
#include <bits/stdc++.h>
using namespace std;

int main(void)
{
  int a, b, c, m, l, r;
  scanf("%d %d %d", &l, &r, &m);
  hopcroft::init(l, r);
  for (int i = 0; i < m; i++)
  {
    scanf("%d %d", &a, &b);
    hopcroft::graph[a].push_back(b + l);
  }

  cout << hopcroft::hopcroft() << endl;
}
```

4.4 Bridges/Articulation Points

```
// https://www.spoj.com/problems/SUBMERGE/
#include "../bridges_art_points.cpp"

#define MAXN 1123

vector<int> graph[MAXN];

int main(void)
{
  int n, m, a, b, T;
    scanf("%d", &T);
  for (int t = 1; t <= T; t++)
  {
        scanf("%d %d", &n, &m);
    for (int i = 0; i<= n; i++)
      graph[i].clear();
    for (int i = 0; i < m; i++)
    {
      scanf("%d %d", &a, &b);
      graph[a].push_back(b);
      graph[b].push_back(a);
    }

    apb rdm(1, n, graph);
        for (int i = 0; i < rdm.bridges.size(); i++)
      rdm.bridges[i] = {min(rdm.bridges[i].first, graph[rdm.bridges[i].first][rdm.
   bridges[i].second]), max(rdm.bridges[i].first, graph[rdm.bridges[i].first][rdm.
   bridges[i].second])};

    sort(rdm.bridges.begin(), rdm.bridges.end());

    printf("Caso #%d\n", t);
    if (rdm.bridges.size())
    {
      printf("%d\n", (int) rdm.bridges.size());
      for (int i = 0; i < rdm.bridges.size(); i++)
        printf("%d %d\n", rdm.bridges[i].first, rdm.bridges[i].second);
    }
    else
      printf("Sin bloqueos\n");
  }
}
```

```
// https://www.spoj.com/problems/SUBMERGE/
#include "../bridges_art_points.cpp"

#define MAXN 112345

vector<int> graph[MAXN];

int main(void)
{
  int n, m, a, b;
  while (scanf("%d %d", &n, &m) && n)
  {
    for (int i = 0; i<= n; i++)
      graph[i].clear();
    for (int i = 0; i < m; i++)
    {
      scanf("%d %d", &a, &b);
      graph[a].push_back(b);
      graph[b].push_back(a);
    }

    apb rdm(1, n, graph);
    printf("%d\n", (int) count(rdm.art.begin(), rdm.art.end(), true));
  }
}
```

```
3 3
1 2
2 3
1 3
6 8
1 3
6 1
6 3
4 1
6 4
5 2
3 2
3 5
0 0
```

4.5 Centroid Decomposition

```
// https://codeforces.com/contest/342/problem/E

#include "../centroid.cpp"
#include "../../lca/lca.cpp"

/*
  BIT: element update, range sum query and sum lower_bound in O(log(N)).
  Represents an array of elements in range [1, N].
*/

template <class T>
struct bit
{
  int n, LOGN;
  vector<T> val;
  bit(int _n = 0) : n(_n), LOGN(log2(n + 1)), val(_n + 1, 0) {}

  // val[pos] += x
  void update(int pos, T x)
  {
    for (int i = pos; i <= n; i += -i & i)
      val[i] += x;
  }

  // sum of range [1, pos]
  T query(int pos)
  {
    T retv = 0;
    for (int i = pos; i > 0; i -= -i & i)
      retv += val[i];
    return retv;
  }

  // min pos such that sum of [1, pos] >= sum, or n + 1 if none exists.
  int lower_bound(T x)
  {
    T sum = 0;
    int pos = 0;

    for (int i = LOGN; i >= 0; i--)
      if (pos + (1 << i) <= n && sum + val[pos + (1 << i)] < x)
        sum += val[pos += (1 << i)];

    return pos + 1; // pos will have position of largest value less than x.
  }
};

int par[MAXN];
bit<int> ans[MAXN]; // Using BIT as a multiset for reasons.

void dfs(int a, int p, int c, int l, int &mh)
```

```
{
  mh = max(mh, l);
  par[a] = c;

  for (int i = 0; i < sz(graph[a]); i++)
    if (graph[a][i] != p && !block[graph[a][i]])
      dfs(graph[a][i], a, c, l + 1, mh);
}

void process(int a, int sz)
{
  int mh = 1;
  for (int i = 0; i < sz(graph[a]); i++)
    if (!block[graph[a][i]])
      dfs(graph[a][i], a, a, 1, mh);

  ans[a] = bit<int>(mh + 1);
}

int cnt_ans[MAXN];
bool color[MAXN];

void toggle(int a, lca_preprocess &lca)
{
  color[a] = !color[a];
  for (int v = a; v != 0; v = par[v])
    if (color[a])
    {
      cnt_ans[v]++;
      ans[v].update(lca.dist(a, v) + 1, 1);
    }
    else
    {
      cnt_ans[v]--;
      ans[v].update(lca.dist(a, v) + 1, -1);
    }
}

int get_ans(int a, lca_preprocess &lca)
{
  int retv = inf;
  for (int v = a; v != 0; v = par[v])
    if (cnt_ans[v])
      retv = min(retv, ans[v].lower_bound(1) - 1 + lca.dist(a, v));

  return retv;
}

int main(void)
{
  int n, m, a, b;
  scanf("%d %d", &n, &m);
  for (int i = 0; i + 1 < n; i++)
```

```
  {
    scanf("%d %d", &a, &b);
    put_edge(a, b);
  }

  decomp(1, n);

  lca_preprocess lca(1, n, graph);

  toggle(1, lca);
  for (int i = 0; i < m; i++)
  {
    scanf("%d %d", &a, &b);
    if (a == 1)
      toggle(b, lca);
    else
    {
      int x = get_ans(b, lca);
      printf("%d\n", (x == inf ? -1 : x));
    }
  }
}
```

```
// https://www.codechef.com/problems/PRIMEDST

#include "../../../numerical/fft/fft.cpp"
#include "../../../number_theory/sieve/sieve.cpp"
#include "../centroid.cpp"

ll ans[MAXN];

void dfs(int a, int p, int l, vector<ll> &nh)
{
  while (sz(nh) <= l)
    nh.push_back(0);

  nh[l]++;
  for (int i = 0; i < sz(graph[a]); i++)
    if (graph[a][i] != p && !block[graph[a][i]])
      dfs(graph[a][i], a, l + 1, nh);
}

void process(int a, int sz)
{
  vector<ll> tot(1);
  tot[0] = 1;

  for (int i = 0; i < sz(graph[a]); i++)
    if (!block[graph[a][i]])
    {
      vector<ll> nh;
      dfs(graph[a][i], a, 1, nh);

      for (int j = 0; j < sz(nh); j++)
      {
        while (sz(tot) <= j)
          tot.push_back(0);

        tot[j] += nh[j];
      }

      auto intra_paths = multiply(nh, nh);
      for (int j = 0; j < sz(intra_paths); j++)
        if (intra_paths[j] != 0)
          ans[j] -= intra_paths[j];
    }

  auto total_paths = multiply(tot, tot);
  total_paths[0]--;
  for (int j = 0; j < sz(total_paths); j++)
    if (total_paths[j] != 0)
      ans[j] += total_paths[j];
}

int main(void)
{
```

```
  sieve::init();
  cin.sync_with_stdio(0);
  cin.tie(0);

  int n, a, b;
  cin >> n;
  for (int i = 0; i + 1 < n; i++)
  {
    cin >> a >> b;
    put_edge(a, b);
  }

  decomp(1, n);

  ll total = 0;
  for (int i = 0; i < n; i++)
    if (i > 1 && sieve::lp[i] == i)
      total += ans[i] / 2;

  if (n == 1)
  {
    cout << total << endl;
  }
  else
  {
    printf("%.20lf\n", total / (n * 1ll * (n - 1) / 2.0));
  }
}
```

4.6 Euler Tour

```
//https://codeforces.com/contest/547/problem/D
#include "../../../contest/header.hpp"

namespace euler
{

#define MAXM 212345
#define MAXN 212345

struct edge
{
    int u, v, id;
};

struct vertice
{
    set<int> outs;            // edges indexes
    int in_degree = 0; // not used with undirected graphs
};

int n, m;
edge edges[MAXM];
vertice vertices[2 * MAXN];
set<int>::iterator its[2 * MAXN];
bool used_edge[MAXM];

void init()
{
    for (int i = 0; i < n; i++)
    {
        its[i] = vertices[i].outs.begin();
    }
}

vi euler_tour(int src)
{
    vi ret_edges;
    vector<pii> s = {{src, -1}};
    while (!s.empty())
    {
        int x = s.back().first;
        int e = s.back().second;
        auto &it = its[x], end = vertices[x].outs.end();

        while (it != end && used_edge[*it])
            ++it;

        if (it == end)
        {
            ret_edges.push_back(e);
            s.pop_back();
```

```
        }
        else
        {
            auto edge = edges[*it];
            int v = edge.u == x ? edge.v : edge.u;
            s.push_back({v, *it});
            used_edge[*it] = true;
        }
    }

  reverse(all(ret_edges));
  return ret_edges;

}

} // namespace euler


int main()
{
    scanf("%d", &euler::m);
    euler::n = 2 * MAXM;
    for (int i = 0; i < euler::m; i++)
    {
        int x, y;
        scanf("%d%d", &x, &y);
        y += MAXM;
        euler::edges[i] = {x, y, i};
        euler::vertices[x].outs.insert(i);
        euler::vertices[y].outs.insert(i);
    }

    queue<int> odds;
    for (int i = 0; i < 2 * MAXM; i++)
    {
        if (!euler::vertices[i].outs.empty())
            odds.push(i);
    }

    stack<pii> removed_edges;

    while (!odds.empty())
    {
        int u = odds.front();
        odds.pop();
        if ((int)euler::vertices[u].outs.size() & 1)
        {
            int e = *euler::vertices[u].outs.begin();
            removed_edges.push({e, u});
            int x = euler::edges[e].u;
            int y = euler::edges[e].v;
            euler::vertices[x].outs.erase(e);
            euler::vertices[y].outs.erase(e);
```

```
            if ((int)euler::vertices[y].outs.size() & 1)
            {
                odds.push(y);
            }

            if ((int)euler::vertices[x].outs.size() & 1)
            {
                odds.push(x);
            }
        }
    }

    vector<char> color(euler::m);
    vi balance(2 * MAXM);
    euler::init();

    for (int i = 0; i < 2 * MAXM; i++)
    {
        if (!euler::vertices[i].outs.empty() && !euler::used_edge[*euler::vertices[i
    ].outs.begin()])
        {
            auto res = euler::euler_tour(i);
            char c = 'r';
            for (int j : res)
            {
                if (j == -1) continue;
                color[j] = c;
                if (c == 'r')
                {
                    balance[euler::edges[j].u]++;
                    balance[euler::edges[j].v]++;
                    c = 'b';
                }
                else
                {
                    balance[euler::edges[j].u]--;
                    balance[euler::edges[j].v]--;
                    c = 'r';
                }
            }
        }
    }

    while(!removed_edges.empty())
    {
        int e, u, v;
        tie(e, u) = removed_edges.top();
        v = euler::edges[e].u == u ? euler::edges[e].v:euler::edges[e].u;
        removed_edges.pop();

        if (balance[v] > 0)
        {
```

```
            color[e] = 'b';
            balance[u]--;
            balance[v]--;
        }
        else
        {
            color[e] = 'r';
            balance[u]++;
            balance[v]++;
        }
    }

    for(char c : color)
        printf("%c", c);
    printf("\n");

}
```

```cpp
// https://codeforces.com/contest/723/problem/E
#include "../eulertour.cpp"

int vis[MAXM];

int dfs(int u)
{
    int res = 0;
    for (int e : euler::vertices[u].outs)
    {
        if (!vis[e])
        {
            vis[e] = 1;
            int v = u == euler::edges[e].u ? euler::edges[e].v : euler::edges[e].u;
            res += dfs(v) + 1;
        }
    }
    return res;
}

int main()
{
    int t;
    scanf("%d", &t);
    while (t--)
    {
        scanf("%d%d", &euler::n, &euler::m);
        for (int i = 0; i < euler::m; i++)
        {
            int u, v;
            scanf("%d%d", &u, &v);
            --u, --v;
            euler::edges[i].u = u;
            euler::edges[i].v = v;
            euler::edges[i].id = i;

            euler::vertices[u].outs.push_back(i);
            euler::vertices[v].outs.push_back(i);
        }

        vi odds;
        int evens = 0;
        for (int i = 0; i < euler::n; i++)
        {
            if (sz(euler::vertices[i].outs) & 1)
                odds.push_back(i);
            else
                ++evens;
        }

        set<pii> created_edges;
        for (int i = 0; i < sz(odds); i += 2)
        {
```

```
            int u = odds[i];
            int v = odds[i + 1];
            euler::edges[i + euler::m].u = u;
            euler::edges[i + euler::m].v = v;
            euler::edges[i + euler::m].id = i + euler::m;
            euler::vertices[u].outs.push_back(i + euler::m);
            euler::vertices[v].outs.push_back(i + euler::m);

            created_edges.insert({min(u, v), max(u, v)});
        }

        euler::init();
        printf("%d\n", evens);
        for (int i = 0; i < euler::n; i++)
        {
            if (!euler::vertices[i].outs.empty() &&
                !euler::used_edge[euler::vertices[i].outs.back()])
            {
                int n_edges = dfs(i);
                auto tour = euler::euler_tour(n_edges, i);
                for (int j = 0; j < sz(tour) - 1; ++j)
                {
                    int u = tour[j];
                    int v = tour[j + 1];
                    auto it = created_edges.find({min(u, v), max(u, v)});
                    if (it == created_edges.end())
                    {
                        printf("%d %d\n", u + 1, v + 1);
                    }
                    else
                        created_edges.erase(it);
                }
            }
        }

        for (int i = 0; i < euler::n; i++)
        {
            euler::vertices[i].outs.clear();
        }

        for (int i = 0; i < euler::m + sz(odds); i++)
        {
            euler::used_edge[i] = 0;
            vis[i] = 0;
        }
    }

}
```

```
// https://www.spoj.com/problems/WORDS1/

#include "../eulertour.cpp"

char aux[1123];

void reset()
{
    for (int i = 0; i < euler::n; i++)
    {
        euler::vertices[i].outs.clear();
        euler::vertices[i].in_degree = 0;
    }
    for (int i = 0; i < euler::m; i++)
    {
        euler::used_edge[i] = false;
    }

}

int find_src()
{
  int src = 0;
  for (int i = 0; i < euler::n; i++)
  {
    if ((int)sz(euler::vertices[i].outs) > euler::vertices[i].in_degree)
    {
      src = i;
      break;
    }
    if (euler::vertices[i].in_degree)
      src = i;
  }

  return src;
}

bool check_condition()
{
  int c = 0;
  for (int i = 0; i < euler::n; ++i)
      c += abs(euler::vertices[i].in_degree - sz(euler::vertices[i].outs));
  return c <= 2;
}

int main()
{
    int t;
    scanf("%d", &t);
    euler::n = 26;
    while (t--)
    {
        scanf("%d", &euler::m);
```

```
        for (int i = 0; i < euler::m; i++)
        {
            scanf("%s", aux);
            string str = aux;
            int u = str[0] - 'a';
            int v = str.back() - 'a';
            euler::edges[i] = {u, v, i};
            euler::vertices[u].outs.push_back(i);
            euler::vertices[v].in_degree++;
        }

        euler::init();

        if (!check_condition() || euler::euler_tour(euler::m, find_src()).empty())
        {
            printf("The door cannot be opened.\n");
        }
        else
        {
            printf("Ordering is possible.\n");
        }

        reset();
    }

}
```

4.7 Max Flow (Dinic)

```
// http://www.spoj.com/problems/FASTFLOW/

#include "../dinic.cpp"

int main(void)
{
    int a, b, c, n, m;
    cin >> n >> m;

    dinic::init(n, 1, n);
    for (int i = 0; i < m; i++)
    {
        cin >> a >> b >> c;
        dinic::put_edge_undirected(a, b, c);
    }

    cout << dinic::max_flow() << endl;
}
```

```
4 6
1 2 3
2 3 4
3 1 2
2 2 5
3 4 3
4 3 3
```

```
// https://codeforces.com/gym/102007/attachments Problem I.

#include "../dinic.cpp"

#define MAXN 212345

int p[MAXN];
vector<int> graph[MAXN];
vector<int> cost[MAXN];

int sh[12];
int sh_cap[12];
long long dist[12][MAXN];

typedef pair<long long, int> pli;

void dijkstra(int n, int a, long long d[])
{
    memset(d, 0x3f, sizeof(long long) * (n + 1));

    priority_queue<pli> q;
    q.push(pli(0, a));
    d[a] = 0;
    while (!q.empty())
    {
        a = q.top().second;
        long long tmp = q.top().first;
        q.pop();
        if (-tmp != d[a])
            continue;

        for (int i = 0; i < (int)graph[a].size(); i++)
        {
            if (d[graph[a][i]] > d[a] + cost[a][i])
            {
                d[graph[a][i]] = d[a] + cost[a][i];
                q.push(pli(-d[graph[a][i]], graph[a][i]));
            }
        }
    }
}

long long foo(int n, int s, long long mid)
{
    dinic::init(n + s + 1, 0, n + s + 1);

    for (int i = 1; i <= n; i++)
        dinic::put_edge(0, i, p[i]);

    for (int i = 1; i <= n; i++)
        for (int j = 0; j < s; j++)
            if (dist[j][i] <= mid)
                dinic::put_edge(i, n + j + 1, inf);
```

```
    for (int i = 0; i < s; i++)
        dinic::put_edge(n + i + 1, n + s + 1, sh_cap[i]);

    return dinic::max_flow();
}

int main(void)
{
    int n, m, s, u, v, w;
    cin >> n >> m >> s;
    long long total_pop = 0;
    for (int i = 1; i <= n; i++)
    {
        scanf("%d", &p[i]);
        total_pop += p[i];
    }

    for (int i = 0; i < m; i++)
    {
        scanf("%d %d %d", &u, &v, &w);
        graph[u].push_back(v);
        cost[u].push_back(w);
        graph[v].push_back(u);
        cost[v].push_back(w);
    }

    for (int i = 0; i < s; i++)
    {
        scanf("%d %d", &sh[i], &sh_cap[i]);
        dijkstra(n, sh[i], dist[i]);
    }

    long long bot = 0, top = 1e16;
    while (bot < top)
    {
        long long mid = (bot + top) / 2;

        if (foo(n, s, mid) == total_pop)
        {
            top = mid;
        }
        else
        {
            bot = mid + 1;
        }
    }

    cout << bot << endl;
}
```

```
7 8 3
0 1 1 1 1 0 2
```

```
1 2 1
2 3 1
3 1 1
4 6 5
4 3 1
6 7 10
7 5 3
5 6 3
6 5
1 1
2 1
```

```cpp
// http://www.spoj.com/problems/MATCHING/

#include "../dinic.cpp"

int main(void)
{
    int a, b, l, r, m;
    scanf("%d %d %d", &l, &r, &m);

    dinic::init(l + r + 2, l + r, l + r + 1);

    for (int i = 0; i < m; i++)
    {
        scanf("%d %d", &a, &b);
        a--, b--;
        dinic::put_edge(a, b + l, 1);
    }

    for (int i = 0; i < l; i++)
        dinic::put_edge(l + r, i, 1);
    for (int i = l; i < l + r; i++)
        dinic::put_edge(i, l + r + 1, 1);

    cout << dinic::max_flow() << endl;
}
```

```
5 4 6
5 2
1 2
4 3
3 1
2 2
4 4
```

4.8 Min Cost Max Flow

```
// https://open.kattis.com/problems/mincostmaxflow

#include "../min_cost_max_flow.cpp"
#include <bits/stdc++.h>
using namespace std;

int main(void)
{
  int n, m, s, t, a, b, c, w;
  scanf("%d %d %d %d", &n, &m, &s, &t);

  mcmf::init(n, s, t);
  for (int i = 0; i < m; i++)
  {
    scanf("%d %d %d %d", &a, &b, &c, &w);
    mcmf::put_edge(a, b, c, w);
  }

  pll ans = mcmf::mincost_maxflow();
  cout << ans.first << " " << ans.second << endl;
}

2 1 1 0
0 1 1000 100

4 4 0 3
0 1 4 10
1 2 2 10
0 2 4 30
2 3 4 10
```

```cpp
// https://www.spoj.com/problems/SPHIWAY/

#include "../min_cost_max_flow.cpp"
#include <bits/stdc++.h>
using namespace std;

int main(void)
{
  int n, m, s, t, a, b, c;
  scanf("%d %d %d %d", &n, &m, &s, &t);

  mcmf::init(n, 0, t);
  for (int i = 0; i < m; i++)
  {
    scanf("%d %d %d", &a, &b, &c);
    mcmf::put_edge(a, b, 1, c);
    mcmf::put_edge(b, a, 1, c);
  }

  mcmf::put_edge(0, s, 2, 0);

  pll ans = mcmf::mincost_maxflow();
  if (ans.first == 2)
    printf("%lld\n", ans.second);
  else
    printf("-1\n");
}
```

```
5 7 1 5
1 2 3
1 4 8
2 3 5
2 4 4
3 5 5
4 3 8
4 5 3
```

4.9 Min-Cut Global

```
#include "../min_cut.cpp"

int main(void)
{
  int T, n, m, a, b, c;
  scanf("%d", &T);
  while (T--)
  {
    scanf("%d %d", &n, &m);
    vector<vector<int>> graph(n);
    for (int i = 0; i < n; i++)
      graph[i] = vector<int>(n, 0);
    for (int i = 0; i < m; i++)
    {
      scanf("%d %d %d", &a, &b, &c);
      a--;
      b--;
      graph[a][b] = c;
      graph[b][a] = c;
    }

    printf("%d\n", mincut(n, graph).first);
  }
}
```

```
3
5 6
1 2 1
1 3 1
2 3 1
3 4 10
3 5 1
4 5 1
4 4
1 2 10
2 3 5
3 4 20
4 1 50
3 2
1 2 1
2 3 2
```

4.10 Gomory Hu

```
// https://codeforces.com/contest/343/problem/E

#include "../gomory_hu.cpp"

int ans;
vector<int> ord;
vector<set<pii>> graph;
```

```
tuple<int,int,int> dfs(int a, int p)
{
  tuple<int,int,int> retv = {inf, 0, 0};
  for (auto &e : graph[a])
    if (e.first != p)
    {
      retv = min(retv, dfs(e.first, a));
      retv = min(retv, {e.second, e.first, a});
    }

  return retv;
}

void solve(int a)
{
  int u, v, w;
  tie(w, u, v) = dfs(a, a);
  if (w == inf)
    ord.push_back(a);
  else
  {
    graph[u].erase({v, w});
    graph[v].erase({u, w});
    solve(u);
    solve(v);
    ans += w;
  }
}

int main(void)
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int n, m, a, b, c;
  cin >> n >> m;
  graph.assign(n, set<pii>());

  gomory_hu gh;
  for (int i = 0; i < m; i++)
  {
    cin >> a >> b >> c;
    gh.add_edge(a - 1, b - 1, c);
  }

  auto tree_edges = gh.solve(n);

  for (int i = 1; i < n; i++)
  {
    graph[tree_edges[i].second].insert({i, (int)tree_edges[i].first});
    graph[i].insert({tree_edges[i].second, (int)tree_edges[i].first});
  }
```

```
  solve(0);

  cout << ans << endl;
  for (int i = 0; i < n; i++)
    cout << ord[i] + 1 << " ";
  cout << endl;
}
```

```
6 11
1 2 10
1 6 8
2 3 4
2 5 2
2 6 3
3 4 5
3 5 4
3 6 2
4 5 7
4 6 2
5 6 3
```

```
// https://www.urionlinejudge.com.br/judge/pt/problems/view/2082
#include "../gomory_hu.cpp"

int main(void)
{
  int T, n, m, a, b, c;
  scanf("%d", &T);
  while (T--)
  {
    scanf("%d %d", &n, &m);
    gomory_hu gh;

    vector<vector<int>> graph(n);
    for (int i = 0; i < n; i++)
      graph[i] = vector<int>(n, 0);

    for (int i = 0; i < m; i++)
    {
      scanf("%d %d %d", &a, &b, &c);
      a--;
      b--;
      graph[a][b] = c;
      graph[b][a] = c;
    }

    for (int i = 0; i < n; i++)
      for (int j = i + 1; j < n; j++)
        if (graph[i][j] > 0)
          gh.add_edge(i, j, graph[i][j]);

    ll ans = inf;
    auto edges = gh.solve(n);
    for (int i = 1; i < n; i++)
      ans = min(ans, edges[i].first);

    printf("%lld\n", ans);
  }
}
```

4.11 Heavy-Light Decomposition

```
/*
 * https://codeforces.com/gym/102299/problem/G
 */

#include <bits/stdc++.h>
using namespace std;

#define MAXN 112345
#define inf 0x3f3f3f3f

#define ll long long
#define pb push_back
```

```
typedef vector<ll> vll;
typedef vector<int> vi;

#define MAX 100010
#define MAXLOG 19

vector<vi> adj;

int subsize[MAXN], parent[MAXN];
int chainNo = 0, chainHead[MAXN], chainPos[MAXN], chainInd[MAXN], chainSize[MAXN];
set<pair<int, int> > sets[MAXN];
int p[MAXN];
int h[MAXN];

bool hasBoss[MAXN];

void hld(int cur){
  if(chainHead[chainNo] == -1)
    chainHead[chainNo] = cur;

  chainInd[cur] = chainNo;
  chainPos[cur] = chainSize[chainNo];
  chainSize[chainNo]++;

  int ind  = -1, mai = -1;
  for(int i = 0; i < (int)adj[cur].size(); i++){
    if(adj[cur][i] != parent[cur] && subsize[adj[cur][i]] > mai){
      mai = subsize[adj[cur][i]];
      ind = i;
    }
  }

  if(ind >= 0)
    hld(adj[cur][ind]);

  for(int i = 0; i < (int)adj[cur].size(); i++)
    if(adj[cur][i] != parent[cur] && i != ind){
      chainNo++;
      hld(adj[cur][i]);
    }
}

int dfs0(int pos, int prev = -1){
  int res = 1;
  if(prev != -1){
    h[pos] = h[prev] + 1;
  }
  for(int i = 0; i < (int)adj[pos].size(); i++){
    int nx = adj[pos][i];
    if(nx != prev){
      res += dfs0(nx, pos);
      parent[nx] = pos;
```

```
    }
  }
  return subsize[pos] = res;
}

int query_up(int u){
  int uchain = chainInd[u];

  if(sets[uchain].empty() || (*(sets[uchain].begin())).first > chainPos[u]){
    u = chainHead[uchain];
    u = parent[u];
    if(u == -1)
      return -1;
    return query_up(u);
  }

  set<pair<int, int> > :: iterator it = sets[uchain].upper_bound({chainPos[u],
   9999999});

  it--;

  return (*it).second;
}

void update(int u){
  int uchain = chainInd[u];
  sets[uchain].insert({chainPos[u], u});
}

struct query{
  char type;

  int par;

  query(){}

  query(char t, int p){
    type = t;
    par = p;
  }
};

vector<query> queries;

int up[MAXN][MAXLOG];
int upmin[MAXN][MAXLOG];

int main(void)
{
  memset(chainHead, -1, sizeof(chainHead));
  memset(parent, -1, sizeof(parent));

  int n, m;
```

```
  scanf("%d %d", &n, &m);

  adj.resize(n);

  for(int i = 0; i < n; i++)
    scanf("%d", &p[i]);

  for(int i = 0; i < m; i++){
    char c;
    scanf(" %c", &c);

    if(c == '+'){
      int a, b;
      scanf("%d %d", &a, &b);
      a--;
      b--;
      adj[a].pb(b);
      adj[b].pb(a);
      hasBoss[b] = 1;
      queries.pb(query('+', b));
    }

    else{
      int a;
      scanf("%d", &a);
      a--;
      queries.pb(query('?', a));
    }
  }

  int fBoss = -1;
  for(int i = 0; i < n; i++){
    if(hasBoss[i] == 0){
      if(fBoss == -1)
        fBoss = i;
      else{
        adj[i].pb(fBoss);
        adj[fBoss].pb(i);
        queries.pb(query('+', i));
      }
    }
  }

  dfs0(fBoss);
  hld(fBoss);

  for (int i = 0; i < n; i++)
  {
    up[i][0] = parent[i];
    upmin[i][0] = p[i];
  }
```

```
  up[fBoss][0] = fBoss;

  for (int j = 1; j < MAXLOG; j++)
    for (int i = 0; i < n; i++)
    {
      up[i][j] = up[up[i][j-1]][j-1];
      upmin[i][j] = min(upmin[i][j-1], upmin[up[i][j-1]][j-1]);
    }


  vector<int> res;
  for(int i = queries.size() - 1; i >= 0; i--){
    if(queries[i].type == '+'){
      update(queries[i].par);
    }

    else{
      int pos = queries[i].par;
      int zeroParent = query_up(pos);
      if (zeroParent == -1)
        zeroParent = fBoss;

      int height = h[zeroParent];

      int ans = p[zeroParent];
      int cur = pos;
      for (int j = MAXLOG - 1; j >= 0; j--)
      {
        if (h[up[cur][j]] >= height)
        {
          ans = min(ans, upmin[cur][j]);
          cur = up[cur][j];
        }
      }
      res.push_back(ans);
    }
  }

  for (int i = (int) res.size() - 1; i >= 0; i--)
    printf("%d\n", res[i]);
}
```

```
/*
 * Problema de encontrar aresta com menor valor em arvore (incluindo updates)
 * https://www.spoj.com/problems/QTREE/
 * */

#include<bits/stdc++.h>

using namespace std;

#define ll long long
#define pb push_back

typedef vector<ll> vll;
typedef vector<int> vi;


#define MAXN 100010
#define INF 2000000000
#define MAXLOG 20

//Vetores LCA
int h[MAXN];
int par[MAXN][MAXLOG];

//Vetor que guarda valores das segs
class segtree{
public:
  vector<int> tree;
  int spam;

  segtree(vector<int> &a){
    tree.resize(4*a.size() + 5);
    spam = a.size() - 1;
    this->build(1, 0, spam, a);
  }

  void build(int node, int left, int right, vector<int> &a){
    if(left == right){
      tree[node] = a[left];
      return;
    }

    int mid = (left + right)/2;

    build(2*node, left, mid, a);
    build(2*node + 1, mid + 1, right, a);

    tree[node] = max(tree[2*node], tree[2*node + 1]);
  }

  void update(int pos, int val, int node = 1, int left = -1, int right = -1){
    if(left == -1){
      left = 0;
```

```
      right = spam;
    }

    if(left == right){
      tree[node] = val;
      return;
    }

    int mid = (left + right)/2;

    if(pos <= mid)
      update(pos, val, 2*node, left, mid);

    else
      update(pos, val, 2*node + 1, mid + 1, right);

    tree[node] = max(tree[2*node], tree[2*node + 1]);
  }

  int query(int posL, int posR, int node = 1, int left = -1, int right = -1){

    if(left == -1){
      left = 0;
      right = spam;
    }

    if(posL > right || posR < left)
      return -INF;

    if(posL <= left && posR >= right)
      return tree[node];

    int mid = (left + right)/2;

    return max(query(posL, posR, 2*node, left, mid), query(posL, posR, 2*node + 1,
   mid + 1, right));
  }

};

vector<segtree> segs;

//Vetor que guarda a arvore
vector<vi> adj;

int subsize[MAXN], parent[MAXN], chainPos[MAXN], chainInd[MAXN];
//ComeÃğar com zero
int chainNo = 0, chainSize[MAXN];
//ComeÃğar com -1
int chainHead[MAXN];

void hld(int cur){
  //Seta variaveis do segmento
```

```
  if(chainHead[chainNo] == -1)
    chainHead[chainNo] = cur;

  chainInd[cur] = chainNo;
  chainPos[cur] = chainSize[chainNo];
  chainSize[chainNo]++;

  //Encontra maior subarvore
  int ind = -1, mai = -1;
  for(int i = 0; i < (int)adj[cur].size(); i++){
    if(adj[cur][i] != parent[cur] && subsize[adj[cur][i]] > mai){
      mai = subsize[adj[cur][i]];
      ind  = i;
    }
  }

  //Continua segmento atual
  if(ind >= 0)
    hld(adj[cur][ind]);

  //Gera novos segmentos
  for(int i = 0; i < (int)adj[cur].size(); i++)
    if(adj[cur][i] != parent[cur] && i != ind){
      chainNo++;
      hld(adj[cur][i]);
    }
}

//garantir que v eh pai de u!! (Por ex com LCA)
int query_up(int u, int v){
  int uchain = chainInd[u], vchain = chainInd[v];
  int ans = -INF;
  while(1){
    //Query termina no segmento atual
    if(uchain == vchain){
      ans = max(ans, segs[uchain].query(chainPos[v], chainPos[u]));
      break;
    }

    ans = max(ans, segs[uchain].query(0, chainPos[u]));

    //Query sobe para proximo segmento
    u = chainHead[uchain];
    u = parent[u];
    uchain = chainInd[u];
  }

  return ans;
}

int dfs0(int pos, int prev = -1){
  int res  = 1;
  for(int i = 0; i < (int)adj[pos].size(); i++){
```

```
    int nx  = adj[pos][i];
    if(nx != prev){
      res += dfs0(nx, pos);
      parent[nx] = pos;
    }
  }
  return subsize[pos] = res;
}

void dfs(int v,int p = -1){
  par[v][0] = p;
  if(p + 1)
    h[v] = h[p] + 1;
  for(int i = 1;i < MAXLOG;i ++)
    if(par[v][i-1] + 1)
      par[v][i] = par[par[v][i-1]][i-1];
  for(auto u : adj[v])  if(p - u)
    dfs(u,v);
}

int LCA(int v,int u){
  if(h[v] < h[u])
    swap(v,u);
  for(int i = MAXLOG - 1;i >= 0;i --)
    if(par[v][i] + 1 and h[par[v][i]] >= h[u])
      v = par[v][i];

  if(v == u)
    return v;
  for(int i = MAXLOG - 1;i >= 0;i --)
    if(par[v][i] - par[u][i])
      v = par[v][i], u = par[u][i];
  return par[v][0];
}

int main()
{
  int t;

  scanf("%d", &t);

  while(t--){
    int n;
    scanf("%d", &n);
    adj.clear();
    adj.resize(n);

    vector<pair<int, int> > in(n);
    vector<int> w(n), ed(n);

    for(int i = 0; i < n - 1; i++){
      int a, b, c;
      scanf("%d %d %d", &a, &b, &c);
```

```
    a--;
    b--;
    adj[a].pb(b);
    adj[b].pb(a);
    in[i] = {a, b};
    w[i] = c;
  }

  //Inicializa estrutura de dados
  memset(chainHead, -1, sizeof(chainHead));
  memset(par, -1, sizeof(par));

  dfs(0);
  dfs0(0);
  hld(0);

  for(int i = 0; i < n - 1; i++){
    if(parent[in[i].first] == in[i].second)
      ed[i] = in[i].first;
    else
      ed[i] = in[i].second;
  }

  vector<vector<int> > hldSegs(chainNo + 1);

  for(int i = 0; i <= chainNo; i++){
    hldSegs[i].resize(chainSize[i]);
  }

  for(int i = 0; i < n - 1; i++){
    int j = ed[i];
    hldSegs[chainInd[j]][chainPos[j]] = w[i];
  }

  segs.clear();
  for(int i = 0; i <= chainNo; i++){
    segs.push_back(segtree(hldSegs[i]));
  }

  char s[10];

  do{
    scanf(" %s", s);

    if(strcmp(s, "QUERY") == 0){
      int a, b;
      scanf("%d %d", &a, &b);
      a--;b--;
      int u = LCA(a, b);
      int val = segs[chainInd[u]].query(chainPos[u], chainPos[u]);
      segs[chainInd[u]].update(chainPos[u], -INF);
      printf("%d\n", max(query_up(a, u), query_up(b, u)));
```

```
        segs[chainInd[u]].update(chainPos[u], val);
      }

      else if(strcmp(s, "CHANGE") == 0){
        int i, ti;
        scanf("%d %d", &i, &ti);
        i--;
        int vrt = ed[i];

        segs[chainInd[vrt]].update(chainPos[vrt], ti);
      }
      else
        break;
    }while(1);

  }

  //Inicializar estruturas usadas
}
```

```
// https://www.spoj.com/problems/QTREE/
#include "../hld_dadalto.cpp"
#include "../../../data_structures/segment_tree/seg.cpp"

struct edge
{
  int a, b, c;
};

int main(void)
{
  int T;
  scanf("%d", &T);
  while (T--)
  {
    int n;
    scanf("%d", &n);
    vector<vector<int>> graph(n + 1);
    vector<edge> e(n - 1);
    for (int i = 0; i < n - 1; i++)
    {
      scanf("%d %d %d", &e[i].a, &e[i].b, &e[i].c);
      graph[e[i].a].push_back(e[i].b);
      graph[e[i].b].push_back(e[i].a);
    }

    heavy_light<segtree, false> hld(1, n, graph);
    for (int i = 0; i < n - 1; i++)
    {
      if (hld.h[e[i].a] > hld.h[e[i].b])
        hld.update(e[i].a, e[i].c);
      else
        hld.update(e[i].b, e[i].c);
    }

    char s[10];
    while (scanf("%s", s) && s[0] != 'D')
    {
      int x, y;
      scanf("%d %d", &x, &y);
      if (s[0] == 'Q')
        printf("%d\n", hld.query_path(x, y, 0, [](int a, int b) { return max(a, b);
   }));
      else
      {
        int i = x - 1;
        e[i].c = y;

        if (hld.h[e[i].a] > hld.h[e[i].b])
          hld.update(e[i].a, e[i].c);
        else
          hld.update(e[i].b, e[i].c);
      }
```

```
    }
  }
}
```

```
// https://www.spoj.com/problems/GSS7/
#include "../../../contest/header.hpp"

template <class DS, bool VALUES_IN_VERTICES> // DS for data structure. Values in
   vertices, true or false.
struct heavy_light
{
  vector<int> p, heavy, h; // parent, heavy child of vertex, height of vertex.
  vector<int> num;     // number of vertex (in an order where paths are contiguos
   intervals).
  vector<int> root;    // root of heavy path of a given vertex.
  DS ds;

  template <class G>
  heavy_light(int a, int n, const G &graph) : p(n + 1), heavy(n + 1, -1), h(n + 1),
   num(n + 1), root(n + 1), ds(n + 1)
  {
    p[a] = a;
    h[a] = 0;
    dfs(graph, a);
    for (int i = 0, id = 0; i <= n; ++i)
      if (heavy[p[i]] != i) // parent of the root is itself, so this works.
        for (int j = i; j != -1; j = heavy[j])
        {
          root[j] = i;
          num[j] = id++;
        }
  }

  template <class G>
  int dfs(const G &graph, int a)
  {
    int size = 1, max_subtree = 0;
    for (int u : graph[a])
      if (u != p[a])
      {
        p[u] = a;
        h[u] = h[a] + 1;
        int subtree = dfs(graph, u);
        if (subtree > max_subtree)
          heavy[a] = u, max_subtree = subtree;
        size += subtree;
      }
    return size;
  }

  template <class BO> // BO for binary_operation
  void process_path(int u, int v, BO op)
  {
    for (; root[u] != root[v]; v = p[root[v]])
    {
      if (h[root[u]] > h[root[v]])
        swap(u, v);
```

```
    op(num[root[v]], num[v]);
  }
  if (h[u] > h[v])
    swap(u, v);
  op(num[u] + (VALUES_IN_VERTICES ? 0 : 1), num[v]);
}

template <class T>
void update(int v, const T &value)
{
  ds.update(num[v], value);
}

template <class T>
T query(int v)
{
  return ds.get(num[v], num[v]);
}

template <class T>
void update_path(int u, int v, const T &value)
{
  process_path(u, v, [this, &value](int l, int r) { ds.update(l, r, value); });
}

template <class T, class F>
T query_path(int u, int v, T res /* initial value */, F join /* join value with
 query result */)
{
  process_path(u, v, [this, &res, &join](int l, int r) { res = join(res, ds.get(l,
  r)); });
  return res;
}

int lca(int u, int v)
{
  for (; root[u] != root[v]; v = p[root[v]])
  {
    if (h[root[u]] > h[root[v]])
      swap(u, v);
  }

  if (h[u] > h[v])
    swap(u, v);

  return u;
}

// Given that u is on the path from v to the root and u != v,
// returns the child of u which is on the path u->v.
int prev(int u, int v)
{
  int retv = v;
```

```
    for (; root[u] != root[v]; v = p[root[v]])
    {
      retv = root[v];
    }

    if (u == v)
      return retv;

    return heavy[u];
  }
};

#define left(i) ((i) << 1)
#define right(i) (((i) << 1) + 1)

struct csum
{
  int pref, gen, suf, tot;
};

csum join(csum l, csum r)
{
  return {max(l.pref, l.tot + r.pref), max(max(l.gen, r.gen), l.suf + r.pref), max(r
   .suf, r.tot + l.suf), l.tot + r.tot};
}

struct segtree
{
  vector<csum> val;
  vector<int> delta;
  int n;

  segtree(int n) : val(4 * (n + 1), {0, 0, 0, 0}), delta(4 * (n + 1), -inf), n(n)
  {
  }

  void prop(int id, int l, int r)
  {
    if (delta[id] != -inf)
    {
      if (l != r)
      {
        delta[left(id)] = delta[id];
        delta[right(id)] = delta[id];
      }

      val[id].suf = val[id].pref = val[id].gen = max(0, (r - l + 1) * delta[id]);
      val[id].tot = (r - l + 1) * delta[id];
      delta[id] = -inf;
    }
  }

  void update(int id, int l, int r, int a, int b, int x)
```

```
{
  if (a == l && b == r)
  {
    delta[id] = x;
    prop(id, l, r);
  }
  else
  {
    prop(id, l, r);
    int mid = (l + r) / 2;
    if (b <= mid)
    {
      update(left(id), l, mid, a, b, x);
      prop(right(id), mid + 1, r);
    }
    else if (a > mid)
    {
      update(right(id), mid + 1, r, a, b, x);
      prop(left(id), l, mid);
    }
    else
    {
      update(left(id), l, mid, a, mid, x);
      update(right(id), mid + 1, r, mid + 1, b, x);
    }

    val[id] = join(val[left(id)], val[right(id)]);
  }
}

// Get the minimum value in range [a, b].
csum get(int id, int l, int r, int a, int b)
{
  prop(id, l, r);
  if (a == l && b == r)
    return val[id];
  else
  {
    int mid = (l + r) / 2;
    if (b <= mid)
      return get(left(id), l, mid, a, b);
    else if (a > mid)
      return get(right(id), mid + 1, r, a, b);
    else
      return join(get(left(id), l, mid, a, mid), get(right(id), mid + 1, r, mid +
 1, b));
  }
}

void update(int a, int b, int x)
{
  return update(1, 0, n - 1, a, b, x);
}
```

```
  csum get(int a, int b)
  {
    return get(1, 0, n - 1, a, b);
  }
};

int main(void)
{
  int n, q, a, b, c;
  scanf("%d", &n);
  vector<vector<int>> graph(n + 1);
  vector<int> v(n + 1, -1);
  for (int i = 1; i <= n; i++)
    scanf("%d", &v[i]);

  for (int i = 0; i < n - 1; i++)
  {
    scanf("%d %d", &a, &b);
    graph[a].push_back(b);
    graph[b].push_back(a);
  }

  heavy_light<segtree, true> hld(1, n, graph);
  for (int i = 1; i <= n; i++)
    hld.update_path(i, i, v[i]);

  scanf("%d", &q);
  for (int i = 0; i < q; i++)
  {
    int tp;
    scanf("%d", &tp);
    if (tp == 2)
    {
      scanf("%d %d %d", &a, &b, &c);
      hld.update_path(a, b, c);
    }
    else
    {
      scanf("%d %d", &a, &b);
      if (hld.h[a] > hld.h[b])
        swap(a, b);

      if (a == b)
      {
        printf("%d\n", hld.query_path(a, b, (csum) {0,0,0,0}, [](csum prev, csum cur
   ) { return join(cur, prev); }).gen);
      }
      else
      {
        csum l = {0,0,0,0};
        csum r = {0,0,0,0};
        int lca = hld.lca(a, b);
```

```
        int prev = hld.prev(lca, b);
        l = hld.query_path(a, lca, l, [](csum prev, csum cur) { return join(cur,
   prev); });
        r = hld.query_path(prev, b, r, [](csum prev, csum cur) { return join(cur,
   prev); });

        swap(l.pref, l.suf);
        printf("%d\n", join(l, r).gen);
      }
    }
  }
}

5
-3 -2 1 2 3
1 2
2 3
1 4
4 5
3
1 2 5
2 3 4 2
1 2 5
```

```cpp
// https://www.spoj.com/problems/QTREE3/
#include "../hld_dadalto.cpp"

struct data_s
{
  set<pii> s;

  data_s(int n) {}

  int get(int a, int b)
  {
    auto it = s.lower_bound(pii(a, -1));
    if (it != s.end() && it->first <= b)
    {
      return it->second;
    }
    else
      return -1;
  }

  void update(int a, int x)
  {
    if (x > 0)
      s.insert(pii(a, x));
    else
      s.erase(pii(a, -x));
  }
};

struct edge
{
  int a, b, c;
};

int main(void)
{
  int n, q, a, b;
  scanf("%d %d", &n, &q);
  vector<vector<int>> graph(n + 1);
  vector<int> v(n + 1, -1);
  for (int i = 0; i < n - 1; i++)
  {
    scanf("%d %d", &a, &b);
    graph[a].push_back(b);
    graph[b].push_back(a);
  }

  heavy_light<data_s, true> hld(1, n, graph);

  for (int i = 0; i < q; i++)
  {
    int x, y;
    scanf("%d %d", &x, &y);
```

```
    if (x == 1)
      printf("%d\n", hld.query_path(1, y, -1, [](int a, int b) { return (b == -1) ?
   a : b; }));
    else
    {
      v[y] *= -1;
      hld.update(y, v[y] * y);
    }
  }
}

9 8
1 2
1 3
2 4
2 9
5 9
7 9
8 9
6 8
1 3
0 8
1 6
1 7
0 2
1 9
0 2
1 9
```

```
// https://www.spoj.com/problems/QTREE6/
#include "../hld_dadalto.cpp"

int cur_color = 0;

struct data_s
{
  set<pii> s[2];

  data_s(int n) {}

  int get(int a, int b)
  {
    auto it = s[cur_color ^ 1].upper_bound(pii(b, inf));
    if (it != s[cur_color ^ 1].begin() && (--it)->first >= a)
    {
      it = s[cur_color].lower_bound(pii(it->first, -1));
      if (it != s[cur_color].end() && it->first <= b)
        return -it->second;
      else
        return 0;
    }
    else
    {
      it = s[cur_color].lower_bound(pii(a, -1));
      if (it != s[cur_color].end() && it->first <= b)
        return it->second;
      else
        assert(false); // Assuming interval is not empty.
    }
  }

  void update(int a, int x)
  {
    if (x > 0)
    {
      s[0].insert(pii(a, x));
      s[1].erase(pii(a, x));
    }
    else
    {
      s[0].erase(pii(a, -x));
      s[1].insert(pii(a, -x));
    }
  }
};

#define left(i) ((i) << 1)
#define right(i) (((i) << 1) + 1)

struct segtree
{
  vector<int> val;
```

```
  int n;

  segtree(int n) : val(4 * (n + 1), 0), n(n) {}

  // Sum x in all elements in range [a, b].
  void update(int id, int l, int r, int a, int b, int x)
  {
    if (a == l && b == r)
    {
      val[id] += x;
    }
    else
    {
      int mid = (l + r) / 2;
      if (b <= mid)
        update(left(id), l, mid, a, b, x);
      else if (a > mid)
        update(right(id), mid + 1, r, a, b, x);
      else
      {
        update(left(id), l, mid, a, mid, x);
        update(right(id), mid + 1, r, mid + 1, b, x);
      }
    }
  }

  int get(int id, int l, int r, int a)
  {
    if (l == r)
      return val[id];
    else
    {
      int mid = (l + r) / 2;
      if (a <= mid)
        return get(left(id), l, mid, a) + val[id];
      else if (a > mid)
        return get(right(id), mid + 1, r, a) + val[id];
    }
  }

  int get(int a, int b)
  {
    assert(a == b);
    return get(1, 0, n - 1, a);
  }

  void update(int a, int b, int x)
  {
    update(1, 0, n - 1, a, b, x);
  }
};

int main(void)
```

```
{
  int n, q, a, b;
  scanf("%d", &n);
  vector<vector<int>> graph(n + 1);
  vector<int> v(n + 1, 1);
  for (int i = 0; i < n - 1; i++)
  {
    scanf("%d %d", &a, &b);
    graph[a].push_back(b);
    graph[b].push_back(a);
  }

  heavy_light<data_s, true> hld(1, n, graph);
  for (int i = 1; i <= n; i++)
    hld.update(i, i);

  heavy_light<segtree, true> ws_hld(1, n, graph);
  heavy_light<segtree, true> bs_hld(1, n, graph);

  for (int i = 1; i <= n; i++)
    bs_hld.update_path(1, i, 1);
  for (int i = 1; i <= n; i++)
    ws_hld.update_path(i, i, 1);

  scanf("%d", &q);
  for (int i = 0; i < q; i++)
  {
    int x, y;
    scanf("%d %d", &x, &y);
    if (x == 1)
    {
      if (y == 1)
      {
        v[y] *= -1;
        hld.update(y, v[y] * y);
        continue;
      }

      cur_color = (v[y] > 0 ? 0 : 1);
      int first_diff = hld.p[abs(hld.query_path(1, y, y, [](int a, int b) {
        if (a > 0)
        {
          if (b == 0)
            return -a;
          return b;
        }
        return a;
      }))];

      int v1 = ws_hld.query<int>(y);
      int v2 = bs_hld.query<int>(y);

      if (v[y] == -1)
```

```
        ws_hld.update_path(hld.p[y], first_diff, -v1);
      else
        bs_hld.update_path(hld.p[y], first_diff, -v2);

      v[y] *= -1;
      hld.update(y, v[y] * y);
      cur_color = (v[y] > 0 ? 0 : 1);
      first_diff = hld.p[abs(hld.query_path(1, y, y, [](int a, int b) {
        if (a > 0)
        {
          if (b == 0)
            return -a;
          return b;
        }
        return a;
      }))];

      if (v[y] == 1)
        bs_hld.update_path(hld.p[y], first_diff, v2);
      else
        ws_hld.update_path(hld.p[y], first_diff, v1);
    }
    else
    {
      cur_color = (v[y] > 0 ? 0 : 1);
      int last_equal = abs(hld.query_path(1, y, y, [](int a, int b) {
        if (a > 0)
        {
          if (b == 0)
            return -a;
          return b;
        }
        return a;
      }));

      // debug(last_equal);
      if (v[y] == -1)
        printf("%d\n", ws_hld.query<int>(last_equal));
      else
        printf("%d\n", bs_hld.query<int>(last_equal));
    }
  }
}

7
1 2
1 3
2 4
2 5
3 6
3 7
4
0 1
```

```
1 1
0 2
0 3
```

4.12 Strongly Connected Components

```
// http://br.spoj.com/problems/CARDAPIO/

#include "../scc.cpp"

// x || y must be true
// Make graph ~x -> y, ~y -> x
// There is a solution if x and ~x are no in the same scc.

vector<int> graph[4123];

char s1[51], s2[51];
int a[1123], b[1123];
int neg[4123];

int
main(void)
{
  int n, t = 1;
  while(scanf("%d", &n) != EOF)
  {
    map<string,int> hash;
    for(int i = 0; i < 4123; i++)
      graph[i].clear();

    memset(neg, 0, sizeof(neg));
    int id = 1;

    for(int i = 0; i < n; i++)
    {
      scanf(" %s %s", s1, s2);
      if(hash[s1] == 0)
        hash[s1] = id++;
      if(hash[s2] == 0)
        hash[s2] = id++;
      a[i] = hash[s1], b[i] = hash[s2];
    }
    string no = "!";
    map<string,int> hash2 = hash;
    for(map<string,int>::iterator it = hash.begin(); it != hash.end(); it++)
      if((it->first)[0] != '!')
        neg[neg[it->second] = hash2[no + it->first]] = it->second;

    for(int i = 0; i < n; i++)
    {
      if(neg[a[i]] != 0)
        graph[neg[a[i]]].push_back(b[i]);
      if(neg[b[i]] != 0)
        graph[neg[b[i]]].push_back(a[i]);
    }

        scc_decomp rdm(id - 1, graph);
```

```
        debug(rdm.scc);

    bool ans = true;
    for(int i = 1; i < id; i++)
//      printf("%d\n", scc[i]);
      if(rdm.scc[neg[i]] == rdm.scc[i])
         ans = false;

    printf("Instancia %d\n%s\n\n", t++, ans ? "sim" : "nao");

  }
}
```

4.13 Transitive Closure

```
#include "../transitive_closure.cpp"

#define MAXN 312
#define MAXK 1123

bitset<MAXN> graph[MAXN];
vector<pii> doc[MAXK];
vector<int> l[MAXN][MAXN];
int doc_done[MAXK];
int n = 0;

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  map<string, int> rdm;
  int a, b;
  string s1, s2;

  cin >> s1 >> s2;
  rdm[s1] = ++n;
  rdm[s2] = ++n;
  tie(a, b) = {rdm[s1], rdm[s2]};

  graph[a][b] = 1;

  int k = 0, m;
  cin >> k;
  for (int i = 0; i < k; i++)
  {
    cin >> m;
    while (m--)
    {
      cin >> s1 >> s2;
      if (rdm[s1] == 0)
        rdm[s1] = ++n;

      if (rdm[s2] == 0)
        rdm[s2] = ++n;
      tie(a, b) = {rdm[s1], rdm[s2]};

      doc[i].push_back({a, b});
      l[a][b].push_back(i);
    }
  }

  transitive_closure<MAXN> t(graph);

  bool ans = true;
  while (ans)
```

```
  {
    int cur = -1;
    for (int i = 1; cur == -1 && i <= n; i++)
      for (int j = 1; cur == -1 && j <= n; j++)
        if (t.closure[i][j] && !l[i][j].empty())
        {
          int tmp = l[i][j].back();
          l[i][j].pop_back();
          if (!doc_done[tmp])
            cur = tmp;
          else
            j--;
        }

    if (cur == -1)
      break;

    doc_done[cur] = true;
    for (int i = 0; i < doc[cur].size(); i++)
      t.add_edge(doc[cur][i].first, doc[cur][i].second);

    for (int i = 1; i <= n; i++)
      if (t.closure[i][i])
        ans = false;
  }

  cout << (ans ? "Yes" : "No") << endl;
}
```

```
#include "../transitive_closure.cpp"

#define MAXN 312
#define MAXK 1123

bitset<MAXN> graph[MAXN];
vector<pii> doc[MAXK];
vector<int> l[MAXN][MAXN];
int doc_done[MAXK];
int n = 0;

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  map<string, int> rdm;
  int a, b;
  string s1, s2;

  cin >> s1 >> s2;
  rdm[s1] = ++n;
  rdm[s2] = ++n;
  tie(a, b) = {rdm[s1], rdm[s2]};

  graph[a][b] = 1;

  int k = 0, m;
  cin >> k;
  for (int i = 0; i < k; i++)
  {
    cin >> m;
    while (m--)
    {
      cin >> s1 >> s2;
      if (rdm[s1] == 0)
        rdm[s1] = ++n;

      if (rdm[s2] == 0)
        rdm[s2] = ++n;
      tie(a, b) = {rdm[s1], rdm[s2]};

      doc[i].push_back({a, b});
      l[a][b].push_back(i);
    }
  }

  transitive_closure<MAXN> t(graph);

  bool ans = true;
  while (ans)
  {
    int cur = -1;
```

```
    for (int i = 1; cur == -1 && i <= n; i++)
      for (int j = 1; cur == -1 && j <= n; j++)
        if (t.closure[i][j] && !l[i][j].empty())
        {
          int tmp = l[i][j].back();
          l[i][j].pop_back();
          if (!doc_done[tmp])
            cur = tmp;
          else
            j--;
        }

    if (cur == -1)
      break;

    doc_done[cur] = true;
    for (int i = 0; i < doc[cur].size(); i++)
      graph[doc[cur][i].first][doc[cur][i].second] = 1;

    t = transitive_closure<MAXN>(graph);

    for (int i = 1; i <= n; i++)
      if (t.closure[i][i])
        ans = false;
  }

  cout << (ans ? "Yes" : "No") << endl;
}
```

4.14 Tree Isomorphism

```
//https://www.urionlinejudge.com.br/judge/pt/problems/view/1229
#include "../../../contest/header.hpp"

vector<vi> graph_a, graph_b;
int label;
map<vi, int> map_labels;


pii get_roots(vector<vi> &graph)
{
    queue<int> q;
    vi vis(sz(graph));
    vi degree(sz(graph));

    for (int i = 0; i < sz(graph); i++)
    {
        if (sz(graph[i]) == 1)
            q.push(i);
        degree[i] = sz(graph[i]);
    }

    int last = 0;
    while (!q.empty())
    {
        int u = q.front();
        q.pop();

        if (vis[u]) continue;
        vis[u] = 1;

        last = u;

        for (int v : graph[u])
        {
            if (degree[v] == 1)
            {
                return {u, v};
            }
            if (!vis[v])
            {
                degree[u]--;
                degree[v]--;

                if (degree[v] == 1)
                    q.push(v);
            }
        }

    }

    return {last, last};
```

```
}

int canonical(int u, int p, vector<vi> &graph)
{
    vi children_labels;
    for (int v : graph[u])
    {
        if (v != p)
            children_labels.push_back(canonical(v, u, graph));
    }

    sort(all(children_labels));
    if (map_labels.count(children_labels) == 0)
        map_labels[children_labels] = label++;
    return map_labels[children_labels];
}

int main()
{
    int n;
    while (scanf("%d", &n) != EOF)
    {
        graph_a.clear();
        graph_a.resize(n + 1);
        graph_b.clear();
        graph_b.resize(n + 1);
        map_labels.clear();
        label = 0;
        for (int i = 0; i < n - 1; i++)
        {
            int u, v;
            scanf("%d%d", &u, &v);
            graph_a[u].push_back(v);
            graph_a[v].push_back(u);
        }
        for (int i = 0; i < n - 1; i++)
        {
            int u, v;
            scanf("%d%d", &u, &v);
            graph_b[u].push_back(v);
            graph_b[v].push_back(u);
        }

        pii roots_a = get_roots(graph_a);
        pii cano_a = {canonical(roots_a.first, 0, graph_a), canonical(roots_a.second
   , 0, graph_a)};

        pii roots_b = get_roots(graph_b);
        pii cano_b = {canonical(roots_b.first, 0, graph_b), canonical(roots_b.second
   , 0, graph_b)};

        if (cano_a.first == cano_b.first ||
            cano_a.first == cano_b.second ||
```

```
            cano_a.second == cano_b.first ||
            cano_a.second == cano_b.second)
            printf("S\n");
        else
            printf("N\n");
    }

}
```

```
// https://codeforces.com/contest/1252/problem/F
#include "../../../contest/header.hpp"

#define MAXN 4123

int n;
vector<int> graph[MAXN];
vector<int> sub_size[MAXN];
bool block[MAXN];
int chosen_vertice = -1;
int degree[MAXN];
int id_label = 1;
int vis[MAXN];
map<vector<int>, int> map_label;
vector<vi> centroids;

int dfs_centroid(int a, int p, int sz, int &centroid, int &val)
{
  int sum = 0, mx = 0, pidx = -1;
  for (int i = 0; i < sz(graph[a]); i++)
    if (graph[a][i] != p && !block[graph[a][i]])
    {
      int x = dfs_centroid(graph[a][i], a, sz, centroid, val);

      sub_size[a][i] = x;
      mx = max(x, mx);
      sum += x;
    }
    else if (graph[a][i] == p && !block[graph[a][i]])
      pidx = i;

  if (pidx != -1)
  {
    sub_size[a][pidx] = sz - sum - 1;
    mx = max(mx, sub_size[a][pidx]);
  }

  if (mx < val)
    val = mx, centroid = a;

  return sum + 1;
}

void put_edge(int a, int b)
{
  degree[a]++;
  graph[a].push_back(b);
  sub_size[a].push_back(0);
  degree[b]++;
  graph[b].push_back(a);
  sub_size[b].push_back(0);
}
```

```
int canonical(int u, int p = 0)
{
    vi sub_labels;
    for (int v : graph[u])
        if (v != p)
        {
            sub_labels.push_back(canonical(v, u));
        }

    sort(all(sub_labels));
    if (!map_label.count(sub_labels))
        map_label[sub_labels] = id_label++;
    return map_label[sub_labels];
}


int main()
{
  scanf("%d", &n);
  for (int i = 0; i < n - 1; i++)
  {
    int u, v;
    scanf("%d%d", &u, &v);
    put_edge(u, v);
  }

    int val = inf;
    dfs_centroid(1, 1, n, chosen_vertice, val);

  if (chosen_vertice == -1)
    return !printf("-1\n");

    for (int v : graph[chosen_vertice])
    {
        --degree[v];
        for (int j = 0; j < sz(graph[v]); j++)
        {
            if (graph[v][j] == chosen_vertice)
            {
                graph[v].erase(graph[v].begin() + j);
                sub_size[v].erase(sub_size[v].begin() + j);
                break;
            }
        }
    }

    vector<int> roots_one;
    vector<pii> roots_two;

    queue<int> q;
  for (int i = 1; i <= n; i++)
  {
    if (degree[i] == 1)
```

```
    {
      q.push(i);
    }
        if (degree[i] == 0)
            roots_one.push_back(i);
  }

  while (!q.empty())
  {
    int u = q.front();
    q.pop();

        if (vis[u]) continue;
        vis[u] = 1;

        for (int k : graph[u])
            if (degree[k] == 1)
            {
                roots_two.push_back({u, k});
                vis[k] = 1;
                break;
            }

        bool last = true;
    for (int v : graph[u])
    {
      if (degree[v] && v != chosen_vertice)
      {
                last = false;
        degree[v]--;
                degree[u]--;

        if (degree[v] == 1)
        {
          q.push(v);
        }
      }
    }

        if (last)
        {
            roots_one.push_back(u);
        }
  }

    if (!roots_one.empty() && sz(roots_one) != sz(graph[chosen_vertice]))
        return !printf("-1\n");

    if (!roots_two.empty() && sz(roots_two) != sz(graph[chosen_vertice]))
        return !printf("-1\n");

    centroids.resize(sz(graph[chosen_vertice]));
    if (!roots_one.empty())
```

```
    {
        for (int i = 0; i < sz(centroids); i++)
        {
            centroids[i].push_back(roots_one[i]);
        }
    }
    if (!roots_two.empty())
    {
        for (int i = 0; i < sz(centroids); i++)
        {
            centroids[i].push_back(roots_two[i].first);
            centroids[i].push_back(roots_two[i].second);
        }
    }

    int target1 = canonical(centroids[0][0]), target2 = canonical(centroids[0].back
   ());
    bool poss = true;
    for (int i = 1; i < sz(graph[chosen_vertice]); i++)
    {
        if (canonical(centroids[i][0]) != target1 && canonical(centroids[i].back())
   != target1)
        {
            poss = false;
            break;
        }
    }

    if (poss)
        return !printf("%d\n", sz(graph[chosen_vertice]));

    poss = true;

    for (int i = 1; i < sz(graph[chosen_vertice]); i++)
    {
        if (canonical(centroids[i][0]) != target2 && canonical(centroids[i].back())
   != target2)
        {
            poss = false;
            break;
        }
    }

    if (poss)
        return !printf("%d\n", sz(graph[chosen_vertice]));

    printf("-1\n");
}
```

```cpp
// https://www.spoj.com/submit/TREEISO/id=24729943

#include "../../../contest/header.hpp"

vector<vector<int>> graph_a, graph_b;
int label;
map<vi, int> map_labels;


pii get_roots(vector<vi> &graph)
{
    queue<int> q;
    vi vis(sz(graph));
    vi degree(sz(graph));

    for (int i = 0; i < sz(graph); i++)
    {
        if (sz(graph[i]) == 1)
            q.push(i);
        degree[i] = sz(graph[i]);
    }

    int last = 0;
    while (!q.empty())
    {
        int u = q.front();
        q.pop();

        if (vis[u]) continue;
        vis[u] = 1;

        last = u;

        for (int v : graph[u])
        {
            if (degree[v] == 1)
            {
                return {u, v};
            }
            if (!vis[v])
            {
                degree[u]--;
                degree[v]--;

                if (degree[v] == 1)
                    q.push(v);
            }
        }

    }

    return {last, last};
}
```

```
int canonical(int u, int p, vector<vi> &graph)
{
    vi children_labels;
    for (int v : graph[u])
    {
        if (v != p)
            children_labels.push_back(canonical(v, u, graph));
    }

    sort(all(children_labels));
    if (map_labels.count(children_labels) == 0)
        map_labels[children_labels] = label++;
    return map_labels[children_labels];
}

int main()
{
    int t;
    scanf("%d", &t);

    while (t--)
    {
        int n;
        scanf("%d", &n);

        graph_a.clear();
        graph_a.resize(n + 1);
        graph_b.clear();
        graph_b.resize(n + 1);
        map_labels.clear();
        label = 0;
        for (int i = 0; i < n - 1; i++)
        {
            int u, v;
            scanf("%d%d", &u, &v);
            graph_a[u].push_back(v);
            graph_a[v].push_back(u);
        }
        for (int i = 0; i < n - 1; i++)
        {
            int u, v;
            scanf("%d%d", &u, &v);
            graph_b[u].push_back(v);
            graph_b[v].push_back(u);
        }

        pii roots_a = get_roots(graph_a);
        pii cano_a = {canonical(roots_a.first, 0, graph_a), canonical(roots_a.second
   , 0, graph_a)};

        pii roots_b = get_roots(graph_b);
        pii cano_b = {canonical(roots_b.first, 0, graph_b), canonical(roots_b.second
```

```
    , 0, graph_b)};

        if (cano_a.first == cano_b.first ||
            cano_a.first == cano_b.second ||
            cano_a.second == cano_b.first ||
            cano_a.second == cano_b.second)
            printf("YES\n");
        else
            printf("NO\n");
    }

}
```

```
#include "../../../contest/header.hpp"

vector<vector<int>> graph;
vector<vector<int>> sub_size;
vector<bool> block;

int dfs_centroid(int a, int p, int sz, pii &centroid, int &val)
{
  int sum = 0, mx = 0, pidx = -1;
  for (int i = 0; i < sz(graph[a]); i++)
    if (graph[a][i] != p && !block[graph[a][i]])
    {
      int x = dfs_centroid(graph[a][i], a, sz, centroid, val);

      sub_size[a][i] = x;
      mx = max(x, mx);
      sum += x;
    }
    else if (graph[a][i] == p && !block[graph[a][i]])
      pidx = i;

  if (pidx != -1)
  {
    sub_size[a][pidx] = sz - sum - 1;
    mx = max(mx, sub_size[a][pidx]);
  }

  if (mx < val)
  {
    val = mx;
    centroid = {a, a};
  }

  if (mx == val)
    centroid.second = a;

  return sum + 1;
}

void put_edge(int a, int b)
{
  graph[a].push_back(b);
  sub_size[a].push_back(0);
  graph[b].push_back(a);
  sub_size[b].push_back(0);
}

int tab_id;
map<vector<int>, int> tab;

int invariant(int a, int p)
{
  vector<int> v;
```

```
  v.reserve(sz(graph[a]));
  for (int i = 0; i < sz(graph[a]); i++)
    if (graph[a][i] != p && !block[graph[a][i]])
      v.push_back(invariant(graph[a][i], a));
  sort(v.begin(), v.end());
  int &x = tab[v];
  if (x == 0)
    x = ++tab_id;

  return x;
}

bool isometric(int a, int b, int sz)
{
  int val_a = inf;
  pair<int, int> centroid_a;
  dfs_centroid(a, a, sz, centroid_a, val_a);

  int val_b = inf;
  pair<int, int> centroid_b;
  dfs_centroid(b, b, sz, centroid_b, val_b);

  int x = invariant(centroid_a.first, centroid_a.first);

  if (x == invariant(centroid_b.first, centroid_b.first) || x == invariant(
   centroid_b.second, centroid_b.second))
    return true;
  return false;
}

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);
  int t;
  cin >> t;
  while (t--)
  {
    int n, a, b;
    cin >> n;
    tab_id = 0;
    tab.clear();
    graph.assign(2*n+1, vector<int>());
    sub_size.assign(2*n+1, vector<int>());
    block.assign(2*n+1, 0);

    for (int i = 1; i < n; i++)
    {
      cin >> a >> b;
      put_edge(a, b);
    }
    for (int i = 1; i < n; i++)
    {
```

```
      cin >> a >> b;
      put_edge(n + a, n + b);
    }

    cout << (isometric(1, n + 1, n) ? "YES" : "NO") << endl;
  }
}
```

5 Misc

5.1 MOs

```
// https://codeforces.com/contest/86/problem/D

#include "../mos.cpp"

#define MAXN 212345

int a[MAXN];
ll ans[MAXN];
ll val = 0;
int freq[1123456];

ll calc(int x)
{
  return freq[x] * 1ll * freq[x] * 1ll * x;
}

void add(int i)
{
  val += (long long)(2 * freq[a[i]] + 1) * a[i];
  freq[a[i]]++;
}

void remove(int i)
{
  val -= (long long)(2 * freq[a[i]] - 1) * a[i];
  freq[a[i]]--;
}

void output(int i)
{
  ans[i] = val;
}

int main(void)
{
  int n, m;
  scanf("%d %d", &n, &m);
  vector<query> q(m);
  for (int i = 1; i <= n; i++)
    scanf("%d", &a[i]);
  for (int i = 0; i < m; i++)
  {
    scanf("%d %d", &q[i].l, &q[i].r);
    q[i].id = i;
  }

  mos(n, q, add, remove, output);
  for (int i = 0; i < m; i++)
    printf("%lld\n", ans[i]);
```

```
}
```

```
//https://codeforces.com/contest/877/problem/F

#include <bits/stdc++.h>
using namespace std;
#define pb push_back
#define db(x) //cerr << #x << " = " << x << endl;
#define INF 0x3f3f3f3f3f3f3f3f
#define fi first
#define se second
#define vi vector<int>
#define vll vector<ll>
#define all(x) x.begin(), x.end()
#define pii pair<int, int>
#define pll pair<ll, ll>
#define vii vector<pii>
#define ll long long
#define ull unsigned long long
typedef long double ld;
#define sz(x) x.size()

#define MAXN 112345

struct query {
  int l, r, id;
};

int n, k;
int type[MAXN];
ll prefix[MAXN];
ll target[MAXN];
ll ans[MAXN];
ll wanted[2 * MAXN], available[2 * MAXN];
ll curr;


void add_l(int p)
{
    available[prefix[p]]++;
    curr += available[target[p]];
    wanted[target[p]]++;
}

void add_r(int p)
{
    available[prefix[p]]++;
    wanted[target[p]]++;
    curr += wanted[prefix[p]];
}

void remove_l(int p)
{
    curr -= available[target[p]];
    available[prefix[p]]--;
```

```
    wanted[target[p]]--;
}

void remove_r(int p)
{
    curr -= wanted[prefix[p]];
    available[prefix[p]]--;
    wanted[target[p]]--;
}

void output(int id)
{
    ans[id] = curr;
}

void mos(int n, vector<query> q)
{
  int bsize = 1 + n / sqrt(sz(q));
    sort(q.begin(), q.end(), [&](const query &lhs, const query &rhs) {
        if (lhs.l / bsize != rhs.l / bsize)
            return lhs.l < rhs.l;
        if ((lhs.l / bsize) & 1)
            return (lhs.r > rhs.r);
        return (lhs.r < rhs.r);
  });

  int l = 1, r = 0; // int l = 0, r = -1; (if indices starts at 0)
  for (int i = 0; i < sz(q); i++)
  {
    while (l > q[i].l)
      add_l(--l);
    while (r < q[i].r)
      add_r(++r);
    while (l < q[i].l)
      remove_l(l++);
    while (r > q[i].r)
      remove_r(r--);

    output(q[i].id);
  }
}

int main()
{
    scanf("%d%d", &n, &k);

    for (int i = 1; i <= n; i++)
    {
        scanf("%d", &type[i]);
    }

    int id = 0;
    set<ll> used;
```

```
    map<ll, int> compress;

    ll curr = 0;
    for (int i = 1; i <= n; i++)
    {
        ll v;
        scanf("%lld", &v);
        target[i] = curr + k;
        used.insert(target[i]);
        if (type[i] == 1)
            curr += v;
        else
            curr -= v;
        prefix[i] = curr;
        used.insert(prefix[i]);
    }

    for (ll i : used)
        compress[i] = id++;
    for (int i = 1; i <= n; i++)
    {
        prefix[i] = compress[prefix[i]];
        target[i] = compress[target[i]];
    }


    int q;
    scanf("%d", &q);
    vector<query> vet_q;
    for (int i = 0; i < q; i++)
    {
        int l, r;
        scanf("%d%d", &l, &r);
        vet_q.pb({l, r, i});
    }

    mos(n, vet_q);

    for (int i = 0; i < q; i++)
    {
        printf("%lld\n", ans[i]);
    }

}
```

```
// https://www.spoj.com/problems/COT2/
#include "../mos_tree_vertex_query.cpp"

#define MAXN 41234
#define MAXQ 112345

vector<int> graph[MAXN];

int x[MAXN];
int ans[MAXQ];
int freq[MAXN];
int val = 0;

void add(int a)
{
  if (++freq[x[a]] == 1)
    val++;
}

void remove(int a)
{
  if (--freq[x[a]] == 0)
    val--;
}

void output(int i)
{
  ans[i] = val;
}

int main(void)
{
  int n, m, a, b;
  scanf("%d %d", &n, &m);

  map<int, int> rdm;
  for (int i = 1; i <= n; i++)
  {
    scanf("%d", &x[i]);
    rdm[x[i]] = 1;
  }

  int tmp = 0;
  for (auto &kvp : rdm)
    kvp.second = tmp++;
  for (int i = 1; i <= n; i++)
    x[i] = rdm[x[i]];

  for (int i = 1; i < n; i++)
  {
    scanf("%d %d", &a, &b);
    graph[a].push_back(b);
    graph[b].push_back(a);
```

```
  }

  vector<pii> q(m);
  for (int i = 0; i < m; i++)
    scanf("%d %d", &q[i].first, &q[i].second);

  mos_tree(n, n, q, graph, add, remove, output);

  for (int i = 0; i < sz(q); i++)
    printf("%d\n", ans[i]);
}
```

```
8 2
105 2 9 3 8 5 7 7
1 2
1 3
1 4
3 5
3 6
3 7
4 8
2 5
7 8
```

```
//https://www.hackerearth.com/pt-br/practice/data-structures/advanced-data-
   structures/fenwick-binary-indexed-trees/practice-problems/algorithm/sherlock-and-
   inversions/
#include <bits/stdc++.h>

using namespace std;
#define pb push_back
#define db(x) //cerr << #x << " = " << x << endl;
#define INF 0x3f3f3f3f3f3f3f3f
#define fi first
#define se second
#define vi vector<int>
#define vll vector<ll>
#define all(x) x.begin(), x.end()
#define pii pair<int, int>
#define pll pair<ll, ll>
#define vii vector<pii>
#define ll long long
#define ull unsigned long long
typedef long double ld;

#define MAXN 112345
#define sz(x) x.size()
constexpr int logn = 20;
constexpr int maxn = 1 << logn;
ll hilbertorder(int x, int y)
{
  ll d = 0;
  for (int s = 1 << (logn - 1); s; s >>= 1)
  {
    bool rx = x & s, ry = y & s;
    d = d << 2 | rx * 3 ^ static_cast<int>(ry);
    if (!ry)
    {
      if (rx)
      {
        x = maxn - x;
        y = maxn - y;
      }
      swap(x, y);
    }
  }
  return d;
}

struct query {
  int l, r, id;
    ll ord() const
    {
        return hilbertorder(l, r);
    }
};
```

```
int n, q;
int vals[MAXN];
ll ans[MAXN];
ll curr;
ll BIT[MAXN];
vector<query> vet_q;

void update(int p, int delta)
{
    while (p <= MAXN)
    {
        BIT[p] += delta;
        p += p & -p;
    }
}

ll pref(int p)
{
    ll res = 0;
    while (p > 0)
    {
        res += BIT[p];
        p -= p & -p;
    }
    return res;
}

ll less_than(int v)
{
    return pref(v - 1);
}

ll greater_than(int v)
{
    return pref(n) - pref(v);
}

void add_l(int p)
{
    int v = vals[p];
    update(v, 1);
    curr += less_than(v);
}

void add_r(int p)
{
    int v = vals[p];
    update(v, 1);
    curr += greater_than(v);
}

void remove_l(int p)
{
```

```
    int v = vals[p];
    update(v, -1);
    curr -= less_than(v);
}

void remove_r(int p)
{
    int v = vals[p];
    update(v, -1);
    curr -= greater_than(v);
}

void output(int id)
{
    ans[id] = curr;
}

void mos(int n, vector<query> q)
{
  int bsize = 1 + n / sqrt(sz(q));
  sort(q.begin(), q.end(), [&](const query &lhs, const query &rhs) {
        return lhs.ord() < rhs.ord();
  });

  int l = 1, r = 0; // int l = 0, r = -1; (if indices starts at 0)
  for (int i = 0; i < sz(q); i++)
  {
    while (l > q[i].l)
      add_l(--l);
    while (r < q[i].r)
      add_r(++r);
    while (l < q[i].l)
      remove_l(l++);
    while (r > q[i].r)
      remove_r(r--);

    output(q[i].id);
  }
}

int main()
{
    scanf("%d%d", &n, &q);
    vi input(n + 1);
    set<int> s;
    for (int i = 1; i <= n; i++)
    {
        scanf("%d", &input[i]);
        s.insert(input[i]);
    }

    int id = 1;
    map<int, int> compress;
```

```
    for (int v : s)
    {
        compress[v] = id;
        ++id;
    }

    for (int i = 1; i <= n; i++)
    {
        vals[i] = compress[input[i]];
    }

    for (int i = 0; i < q; i++)
    {
        int l, r;
        scanf("%d%d", &l, &r);
        vet_q.pb({l, r, i});
    }

    mos(n, vet_q);

    for (int i = 0; i < q; i++)
    {
        printf("%lld\n", ans[i]);
    }

}
```

```
// https://codeforces.com/gym/100962
#include "../mos_tree_edge_query.cpp"

#define MAXN 112345
#define MAXQ 112345

vector<int> graph[MAXN];
vector<int> cost[MAXN];

int x[MAXN];
int ans[MAXQ];
int freq[MAXN];
int val[MAXN];

#define SQN 300

void add(int a)
{
  if (++freq[x[a]] == 1)
  {
    val[x[a] / SQN]++;
  }
}

void remove(int a)
{
  if (--freq[x[a]] == 0)
    val[x[a] / SQN]--;
}

void output(int k)
{
  for (int i = 0; ; i++)
    if (val[i] != SQN)
    {
      for (int j = i * SQN; ; j++)
        if (freq[j] == 0)
        {
          ans[k] = j;
          return;
        }
    }
}

void dfs(int a, int p)
{
  for (int i = 0; i < sz(graph[a]); i++)
    if (graph[a][i] != p)
    {
      dfs(graph[a][i], a);
      x[graph[a][i]] = cost[a][i];
    }
}
```

```
int main(void)
{
  int n, m, a, b, c;
  scanf("%d %d", &n, &m);

  unordered_set<int> s;
  vector<tuple<int,int,int>> e(n-1);
  for (int i = 0; i < n - 1; i++)
  {
    scanf("%d %d %d", &a, &b, &c);
    e[i] = {a, b, c};
    s.insert(c);
  }

  int first;
  for (first = 0; s.find(first) != s.end(); first++)
  {
  }

  for (int i = 0; i < n - 1; i++)
  {
    tie(a, b, c) = e[i];
    graph[a].push_back(b);
    cost[a].push_back(min(first + 1, c));
    graph[b].push_back(a);
    cost[b].push_back(min(first + 1, c));
  }

  int root = (rand() % n) + 1;

  dfs(root, root);

  vector<pii> q(m);
  for (int i = 0; i < m; i++)
    scanf("%d %d", &q[i].first, &q[i].second);

  mos_tree(root, n, q, graph, add, remove, output);

  for (int i = 0; i < sz(q); i++)
    printf("%d\n", ans[i]);
}
```

5.2 Ternary Search

```cpp
// https://codeforces.com/contest/626/problem/E

#include "../ternary_search_discrete.cpp"

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int n;
  cin >> n;
  vector<int> a(n);
  for (int i = 0; i < n; i++)
    cin >> a[i];

  sort(a.begin(), a.end());

  vector<ll> pref(n + 1);
  for (int i = 1; i <= n; i++)
    pref[i] = pref[i-1] + a[i-1];

  pii sol = {0, 0};
  double val = -infll;
  for (int i = 0; i < n; i++)
  {
    auto f = [&a, &i, &n, &pref](int x) {
      return (pref[n] - pref[n-x] + pref[i+1] - pref[i-x] - (2ll*x + 1)*a[i]) / (2.0
    * x + 1.0);
    };

    int x = ternary_search(f, 0, min(i, n - i - 1));
    if (f(x) > val)
    {
      val = f(x);
      sol = {i, x};
    }
  }

  cout << (2 * sol.second + 1) << endl;
  for (int i = sol.first; i >= sol.first - sol.second; i--)
    cout << a[i] << " ";

  for (int i = n - 1; i >= n - sol.second; i--)
    cout << a[i] << " ";

  cout << endl;
}
```

6 Number Theory

6.1 CRT

```
// https://www.spoj.com/problems/FACTMUL/
#include "../crt.cpp"

#define MOD1 186583
#define MOD2 587117
#define MOD 109546051211

int main()
{
    int n;
    scanf("%d", &n);
    ll fat1 = 1, fat2 = 1;
    ll res1 = 1, res2 = 1;;
    for (int i = 2; i <= n; i++)
    {
        fat1 = (fat1 * i) % MOD1;
        res1 = (res1 * fat1) % MOD1;

        fat2 = (fat2 * i) % MOD2;
        res2 = (res2 * fat2) % MOD2;
    }

    printf("%lld\n", crt(res1, MOD1, res2, MOD2));

}
```

```
// https://www.urionlinejudge.com.br/judge/pt/problems/view/2908
#include "../crt_system.cpp"

#define MAXB 11
#define MAXZ 112

int B, Z;
int curr_pos[MAXB];
int next_pos[MAXB][MAXZ];
ll a[MAXZ][MAXB], m[MAXZ][MAXB];

int main()
{
    scanf("%d%d", &B, &Z);
    for (int i = 0; i < B; i++)
    {
        scanf("%d", &curr_pos[i]);
        --curr_pos[i];
        for (int j = 0; j < Z; j++)
        {
            scanf("%d", &next_pos[i][j]);
            --next_pos[i][j];
        }
    }

    int time;
    for (time = 0; time < 300; time++)
    {
        if (count(curr_pos, curr_pos + B, curr_pos[0]) == B)
        {
            return !printf("%d %d\n", curr_pos[0] + 1, time);
        }

        for (int i = 0; i < B; i++)
        {
            curr_pos[i] = next_pos[i][curr_pos[i]];
        }
    }

    vi possible_zoo(Z, 1);

    for (int i = 0; i < B; i++)
    {
        vi vis(Z, -1);
        int cycle_sz = 0;

        while (vis[curr_pos[i]] == -1)
        {
            vis[curr_pos[i]] = cycle_sz;
            cycle_sz++;
            curr_pos[i] = next_pos[i][curr_pos[i]];
        }
```

```
        for (int j = 0; j < Z; j++)
        {
            if (vis[j] == -1)
            {
                possible_zoo[j] = 0;
            }
            else
            {
                a[j][i] = vis[j];
                m[j][i] = cycle_sz;
            }
        }
    }

    ll best = 2 * infll;
    int best_zoo;
    for (int i = 0; i < Z; i++)
    {
        if (possible_zoo[i])
        {
            ll x = crt_system(a[i], m[i], B);
            if (x != -1)
            {
                if (x < best)
                {
                    best = x;
                    best_zoo = i;
                }
            }
        }
    }

    if (best == 2 * infll)
        printf("*\n");
    else
        printf("%d %lld\n", best_zoo + 1, best + time);
}
```

```cpp
// https://open.kattis.com/problems/generalchineseremainder
#include "../crt.cpp"

int main()
{
    int t;
    scanf("%d", &t);
    while (t--)
    {
        ll rem[2], mod[2];
        scanf("%lld%lld%lld%lld", &rem[0], &mod[0], &rem[1], &mod[1]);
        ll lcm = (mod[0] * mod[1]) / __gcd(mod[0], mod[1]);
        ll ans = crt(rem[0], mod[0], rem[1], mod[1]);
        if (ans != -1)
        {
            printf("%lld %lld\n", ans, lcm);
        }
        else
            printf("no solution\n");
    }
}
```

6.2 Euclid

```
// https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=
   show_problem&problem=1045

#include "../euclid.cpp"

int main(void)
{
  int a, b, x, y;
  while (scanf("%d %d", &a, &b) != EOF)
  {
    int d = gcd(a, b, x, y);

    printf("%d %d %d\n", x, y, d);
  }
}
```

```
4 6
17 17
```

6.3 Modular Inverse

```
// https://codeforces.com/contest/300/problem/C

#include "../mod_inverse.cpp"

#define MAXN 1123456

const ll mod = 1e9+7;

ll fat[MAXN];

ll choose(ll n, ll k)
{
  return (fat[n] * mod_inverse((fat[n-k]*fat[k]) % mod, mod)) % mod;
}

int a, b;

bool good(ll x)
{
  while (x)
  {
    if (x % 10 != a && x % 10 != b)
      return false;

    x /= 10;
  }

  return true;
}

int main(void)
{
  ll n;
  cin >> a >> b >> n;

  fat[0] = 1;
  for (int i = 1; i <= n; i++)
    fat[i] = (fat[i-1] * i) % mod;

  ll ans = 0;
  for (int i = 0; i <= n; i++)
  {
    ll sum = a*i + b*(n-i);
    if (good(sum))
      ans = (ans + choose(n, i)) % mod;
  }

  cout << ans << endl;
}
```

6.4 Modular Arithmetic

```
// https://codeforces.com/contest/1236/problem/F

#include "../modular_arithmetic.cpp"

#define MAXN 512345
#define mod_num mod_num<1000000007>

vector<int> graph[MAXN];
int been[MAXN];
vector<int> st;

vector<vector<int>> rings;
vector<pii> edges;

mod_num vertex_rings_factor[MAXN];

void dfs(int a, int p)
{
  been[a] = 1;
  st.push_back(a);
  for (int i = 0; i < sz(graph[a]); i++)
  {
    if (graph[a][i] != p)
    {
      if (been[graph[a][i]] == 0)
        dfs(graph[a][i], a);
      else if (been[graph[a][i]] == 1)
      {
        vector<int> tmp;
        for (int j = sz(st) - 1; st[j] != graph[a][i]; j--)
          tmp.push_back(st[j]);
        tmp.push_back(graph[a][i]);

        rings.push_back(tmp);
      }
    }
  }

  st.pop_back();
  been[a] = 2;
}

bool intersect(const vector<int> &v1, const vector<int> &v2)
{
  for (int i = 0; i < sz(v1); i++)
    if (find(v2.begin(), v2.end(), v1[i]) != v2.end())
      return true;
  return false;
}

int main()
```

```
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int n, m, a, b;
  cin >> n >> m;
  for (int i = 0; i < m; i++)
  {
    cin >> a >> b;
    edges.push_back({min(a, b), max(a, b)});
    graph[a].push_back(b);
    graph[b].push_back(a);
  }

  dfs(1, 1);

  vector<mod_num> pot2div(max(11, n + 1));
  pot2div[0] = mod_num(1);
  pot2div[1] = pot2div[0] / mod_num(2);

  for (int i = 2; i < sz(pot2div); i++)
    pot2div[i] = pot2div[i-1] * pot2div[1];

  mod_num sum_rings_factors(0);

  for (int i = 0; i < sz(rings); i++)
  {
    sum_rings_factors += pot2div[sz(rings[i])];
    for (int x : rings[i])
    {
      vertex_rings_factor[x] += pot2div[sz(rings[i])];
    }
  }

  vector<mod_num> intersecting_rings_factors(sz(rings));
  for (int i = 0; i < sz(rings); i++)
    for (int x : rings[i])
      intersecting_rings_factors[i] += vertex_rings_factor[x] - pot2div[sz(rings[i])
   ];

  mod_num ex = (mod_num(2)^0) * mod_num(n) / mod_num(2) - mod_num(m) / mod_num(4);
  for (int i = 0; i < sz(rings); i++)
    ex += pot2div[sz(rings[i])];

  mod_num ex2(0);

  mod_num ev2 = mod_num(n) * mod_num(n - 1) / (mod_num(2)^2) + mod_num(n) / (mod_num
   (2)^1);

  mod_num ee2(0);
  for (int i = 0; i < m; i++)
  {
    ee2 += pot2div[2];
```

```
    ee2 += mod_num(sz(graph[edges[i].first]) + sz(graph[edges[i].second]) - 2) *
   pot2div[3];
    ee2 += mod_num(m - (sz(graph[edges[i].first]) + sz(graph[edges[i].second]) - 1))
    * pot2div[4];
  }

  mod_num er2(0);
  for (int i = 0; i < sz(rings); i++)
  {
    er2 += pot2div[sz(rings[i])];
    er2 += pot2div[sz(rings[i]) - 1] * intersecting_rings_factors[i];
    er2 += pot2div[sz(rings[i])] * (sum_rings_factors - intersecting_rings_factors[i
   ] - pot2div[sz(rings[i])]);
  }

  mod_num eve;
  for (int i = 1; i <= n; i++)
  {
    eve += mod_num(sz(graph[i])) * pot2div[2];
    eve += mod_num(m - sz(graph[i])) * pot2div[3];
  }

  mod_num eer;
  for (int j = 0; j < sz(rings); j++)
  {
    int k = sz(rings[j]);

    eer += mod_num(sz(rings[j])) * pot2div[sz(rings[j])];
    for (int x : rings[j])
    {
      eer += mod_num(sz(graph[x]) - 2) * pot2div[sz(rings[j]) + 1];
      k += sz(graph[x]) - 2;
    }

    eer += mod_num(m - k) * pot2div[sz(rings[j]) + 2];
  }

  mod_num evr;
  for (int i = 1; i <= n; i++)
  {
    evr += vertex_rings_factor[i];
    evr += (sum_rings_factors - vertex_rings_factor[i]) * pot2div[1];
  }

  ex2 = ev2 + ee2 + er2 - mod_num(2) * eve - mod_num(2) * eer + mod_num(2) * evr;

  cout << (mod_num(2)^(1000000007)) * (ex2 - ex * ex) * (mod_num(2)^(1000000005)) <<
    endl;
}
```

6.5 Phi

```
// https://www.spoj.com/problems/ETF/

#include "../phi.cpp"

int main(void)
{
  int T, n;
  scanf("%d", &T);

  totient::init();
  while(T--)
  {
    scanf("%d", &n);
    // printf("%d\n", totient::phi[n]);
    printf("%d\n", phi(n));
  }
}
```

```
5
1
2
3
4
5
```

6.6 Sieve

```
// https://www.spoj.com/problems/NFACTOR/

#include "../sieve.cpp"

const int MAXN = sieve::MAXP;

int nfactor[MAXN + 1];

int main(void)
{
  int T, a, b, n;
  scanf("%d", &T);

  sieve::init();

  for (int i = 2; i <= MAXN; i++)
  {
    int x = i;
    while (sieve::lp[x] == sieve::lp[i])
      x /= sieve::lp[x];
    nfactor[i] = 1 + nfactor[x];
  }

  for (int i = 0; i < sieve::p.size(); i++)
    assert(nfactor[sieve::p[i]] == 1);

  vector<int> tab[11];
  for (int i = 1; i <= MAXN; i++)
    if (nfactor[i] <= 10)
      tab[nfactor[i]].push_back(i);

  while (T--)
  {
    scanf("%d %d %d", &a, &b, &n);
    printf("%d\n", (int) (upper_bound(tab[n].begin(), tab[n].end(), b) - lower_bound
   (tab[n].begin(), tab[n].end(), a)));
  }
}
```

```
6
1 3 1
1 10 2
1 10 3
1 100 3
1000 1000 0
1 1000000 10
```

7 Numerical

7.1 FFT

```
// https://open.kattis.com/problems/kinversions

#include "../fft.cpp"

int main(void)
{
  string s;
  cin >> s;
  int n = s.size();
  vector<int> a(n, 0), b(n, 0);
  for (int i = 0; i < n; i++)
    if (s[i] == 'A')
      a[i] = 1;
    else
    {
      b[n-i-1] = 1;
    }

  vector<int> c = multiply(a, b);
  for (int i = n; i < 2*n - 1; i++)
    printf("%d\n", c[i]);
}
```

```
// https://www.spoj.com/problems/POLYMUL/

#include "../fft.cpp"

int main(void)
{
  int T, n;
  cin >> T;
  while(T--)
  {
    scanf("%d", &n);
    vector<ll> a(n + 1), b(n + 1);
    for (int i = 0; i < n + 1; i++)
      scanf("%lld", &a[i]);
    for (int i = 0; i < n + 1; i++)
      scanf("%lld", &b[i]);
    vector<ll> c = multiply(a, b);

    for (int i = 0; i < 2*n + 1; i++)
      printf("%lld%c", c[i], (i + 1 < 2*n + 1) ? ' ' : '\n');
  }
}
```

7.2 Fraction

```
// https://codeforces.com/group/kZPk3ZTzR5/contest/249481

#include "../../frac/frac.cpp"
#include "../../../number_theory/mod_inverse/mod_inverse.cpp"
#include "../../bigint/bigint.cpp"
#include "../../linalg/mat.cpp"

int n, k;

#define frac frac<BigInt>
#define mat mat<frac>
#define vec vec<frac>

bool been[20][20][20];
frac tab[20][20][20];

frac pd(int cur, int nex, int pack)
{
  if (pack == 0 || cur == 0)
    return cur == nex ? frac(1) : frac(0);

  if (been[cur][nex][pack])
    return tab[cur][nex][pack];

  been[cur][nex][pack] = true;
  frac p_success(cur, n);
  return tab[cur][nex][pack] = p_success * pd(cur - 1, nex, pack - 1) + (frac(1) -
   p_success) * pd(cur, nex, pack - 1);
}

int main()
{
  cin >> n >> k;
  mat p(n + 1, n + 1);
  for (int i = 0; i <= n; i++)
    for (int j = i; j >= 0; j--)
      p[n - i][n - j] = pd(i, j, k);

  mat q(n, n);
  for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
      q[i][j] = p[i][j];

  mat id(n, n);
  for (int i = 0; i < n; i++)
    id[i][i] = frac(1);

  debug(p);
  debug(q);
  debug(id - q);
  mat N = (id - q).inverse();
```

```
  frac t(0);
  for (int i = 0; i < n; i++)
    t = t + N[0][i];

  debug(N);
  debug(t);

  BigInt mod(1000000007ll);

  cout << (t.a / t.b) << " " << ((t.a % t.b) * mod_inverse(t.b, mod)) % mod << endl;
}
```

7.3 Integration

```
// https://www.spoj.com/problems/VCIRCLES/

#include "../../../geometry/2d/2d.cpp"
#include "../simpson.cpp"

#define point point<double>
#define circle circle<double>
#define MAXN 1123

circle c[MAXN];

int main(void)
{
  int n;
  scanf("%d", &n);
  for (int i = 0; i < n; i++)
    scanf("%lf %lf %lf", &c[i].center.x, &c[i].center.y, &c[i].r);

  set<double> inx;
  for (int i = 0; i < n; i++)
  {
    for (int j = i + 1; j < n; j++)
    {
      pair<point, point> intersections;
      if (!(c[i].center == c[j].center) && c[i].intersect(c[j], intersections) > 0)
      {
        inx.insert(intersections.first.x);
        inx.insert(intersections.second.x);
      }
    }

    inx.insert(c[i].center.x + c[i].r);
    inx.insert(c[i].center.x - c[i].r);
  }

  vector<double> good_x(inx.begin(), inx.end());

  double ans = 0;
  int total = 500000;
  double range_x = good_x.back() - good_x[0];
  vector<pair<double, pair<int,int>>> v;
  for (int i = 0; i + 1 < good_x.size(); i++)
  {
    v.clear();
    for (int j = 0; j < n; j++)
    {
      double d = abs((good_x[i] + good_x[i+1])/2 - c[j].center.x);
      if (d + EPS < c[j].r)
      {
        double h = sqrt(c[j].r * c[j].r - d * d);
        v.push_back({c[j].center.y - h, {j, 1}});
```

```
        v.push_back({c[j].center.y + h, {j, -1}});
      }
    }
    sort(v.begin(), v.end());

    ans += simpsons([&](double x) {

      double prev = -1e9;
      double retv = 0;
      int open = 0;
      for (int j = 0; j < v.size(); j++)
      {
        double d = abs(x - c[v[j].second.first].center.x);
        double h = sqrt(c[v[j].second.first].r * c[v[j].second.first].r - d * d);
        double tmp = c[v[j].second.first].center.y - v[j].second.second * h;

        retv += (tmp - prev) * (open ? 1 : 0);
        open += v[j].second.second;
        assert(tmp + EPS >= prev);
        prev = tmp;
      }

      return retv;
    },
            2*max(1, int(((good_x[i+1] - good_x[i]) / range_x) * total)) , good_x[i
   ], good_x[i+1]);
  }

  printf("%.5lf\n", ans);
}

2
5 6 3
5 5 5
```

7.4 linalg

```
// https://codeforces.com/group/kZPk3ZTzR5/contest/249481

#include "../../frac/frac.cpp"
#include "../../../number_theory/mod_inverse/mod_inverse.cpp"
#include "../../bigint/bigint.cpp"
#include "../../linalg/mat.cpp"

int n, k;

#define frac frac<BigInt>
#define mat mat<frac>
#define vec vec<frac>

bool been[20][20][20];
frac tab[20][20][20];

frac pd(int cur, int nex, int pack)
{
  if (pack == 0 || cur == 0)
    return cur == nex ? frac(1) : frac(0);

  if (been[cur][nex][pack])
    return tab[cur][nex][pack];

  been[cur][nex][pack] = true;
  frac p_success(cur, n);
  return tab[cur][nex][pack] = p_success * pd(cur - 1, nex, pack - 1) + (frac(1) -
   p_success) * pd(cur, nex, pack - 1);
}

int main()
{
  cin >> n >> k;
  mat p(n + 1, n + 1);
  for (int i = 0; i <= n; i++)
    for (int j = i; j >= 0; j--)
      p[n - i][n - j] = pd(i, j, k);

  mat q(n, n);
  for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
      q[i][j] = p[i][j];

  mat id(n, n);
  for (int i = 0; i < n; i++)
    id[i][i] = frac(1);

  debug(p);
  debug(q);
  debug(id - q);
  mat N = (id - q).inverse();
```

```
  frac t(0);
  for (int i = 0; i < n; i++)
    t = t + N[0][i];

  debug(N);
  debug(t);

  BigInt mod(1000000007ll);

  cout << (t.a / t.b) << " " << ((t.a % t.b) * mod_inverse(t.b, mod)) % mod << endl;
}
```

7.5 NTT

```
// https://codeforces.com/contest/1251/problem/F

#include "../ntt.cpp"

#define MAXN 312345

vector<ll> pot(vector<ll> b, ll e)
{
  vector<ll> ans = {1};
  for (; e; b = conv(b, b), e /= 2)
    if (e & 1)
      ans = conv(ans, b);
  return ans;
}

int main()
{
  int t = 1;
  while (t--)
  {
    int n, k, a;
    scanf("%d %d", &n, &k);
    vector<int> tab(MAXN, 0);
    for (int i = 0; i < n; i++)
    {
      scanf("%d", &a);
      tab[a]++;
    }

    vector<ll> ans(4 * MAXN);
    vector<int> b(k);
    for (int i = 0; i < k; i++)
      scanf("%d", &b[i]);
    sort(b.begin(), b.end());

    int k1 = 0, k2 = 0, j = 0;
    vector<ll> p = {1};
    for (int i = 0; i < k; i++)
    {
      for (; j < b[i]; j++)
        if (tab[j] >= 2)
          k2++;
        else if (tab[j] == 1)
          k1++;

      p = conv(p, conv(pot({1, 2}, k1), pot({1, 2, 1}, k2)));
      k1 = 0, k2 = 0;
      for (int l = 0; l < sz(p); l++)
        ans[l + b[i] + 1] = (p[l] + ans[l + b[i] + 1]) % mod;
    }
```

```
    int q;
    scanf("%d", &q);
    while (q--)
    {
      scanf("%d", &a);
      printf("%lld\n", ans[a / 2]);
    }
  }
}
```

7.6 Simplex

```cpp
// https://codeforces.com/gym/101492/problem/I

#include "../simplex.cpp"

int main(void)
{
  int n, m;
  cin >> n >> m;

  int num_constraints = m, num_vars = n;

  // maximize c*x, s.t. a*x <ops> b. x >= 0.
  mat<double> a(num_constraints, num_vars);
  vec<double> b(num_constraints);
  vec<simplex::op> ops(num_constraints);
  vec<double> c(num_vars);
  vec<double> res(num_vars);

  for (int i = 0; i < n; i++)
    cin >> c[i];

  for (int i = 0; i < m; i++)
  {
    int l, r, x;
    cin >> l >> r >> x;
    for (int j = l - 1; j <= r - 1; j++)
      a[i][j] = 1;
    b[i] = x;
    ops[i] = simplex::op::le;
  }

  double ans;
  simplex::run_simplex(num_constraints, num_vars, a, ops, b, c, res, ans);

  cout << ((long long)(ans + 0.5)) << endl;
}
```

```cpp
// https://icpc.kattis.com/problems/roadtimes

#include "../simplex.cpp"

int edge_num[51][51];
int d[51][51];
int prox[51][51];

int main(void)
{
    int n;
    cin >> n;
    int m = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
        {
            cin >> d[i][j];
            if (d[i][j] == -1)
                d[i][j] = 0x3f3f3f3f;
            else
            {
                if (d[i][j] > 0)
                    edge_num[i][j] = m++;
                prox[i][j] = j;
            }
        }

    int r, u, v, t;
    cin >> r;

    int num_constraints = 2 * m + r, num_vars = m;

    // maximize c*x, s.t. a*x <= b.
    mat<double> a(num_constraints, num_vars);
    vec<double> b(num_constraints);
    vec<simplex::op> ops(num_constraints);
    vec<double> c(num_vars);
    vec<double> res(num_vars);

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
        {
            if (d[i][j] > 0 && d[i][j] < 0x3f3f3f3f)
            {
                // d[i][j] <= x[edge_num[i][j]] <= 2 * d[i][j]
                a[2 * edge_num[i][j]][edge_num[i][j]] = 1;
                b[2 * edge_num[i][j]] = 2 * d[i][j];
                ops[2 * edge_num[i][j]] = simplex::op::le;

                a[2 * edge_num[i][j] + 1][edge_num[i][j]] = 1;
                b[2 * edge_num[i][j] + 1] = d[i][j];
                ops[2 * edge_num[i][j] + 1] = simplex::op::ge;
            }
```

```
        }

    for (int k = 0; k < n; k++)
        for (int i = 0; i < n; i++)
            for (int j = 0; j < n; j++)
                if (d[i][k] + d[k][j] < d[i][j])
                {
                    d[i][j] = d[i][k] + d[k][j];
                    prox[i][j] = prox[i][k];
                }

    for (int i = 0; i < r; i++)
    {
        cin >> u >> v >> t;

        while (u != v)
        {
            int w = prox[u][v];
            a[2 * m + i][edge_num[u][w]] = 1;
            u = w;
        }

        ops[2 * m + i] = simplex::op::eq;
        b[2 * m + i] = t;
    }

    cout << fixed << setprecision(12);
    int q;
    cin >> q;
    while (q--)
    {
        cin >> u >> v;
        cout << u << " " << v;
        c = vec<double>(num_vars);
        while (u != v)
        {
            int w = prox[u][v];
            c[edge_num[u][w]] = 1;
            u = w;
        }

        double ans = 0;
    vec<double>::linear_comb(c, -1, c, 0, c);
        simplex::run_simplex(num_constraints, num_vars, a, ops, b, c, res, ans);
    vec<double>::linear_comb(c, -1, c, 0, c);

        cout << " " << -ans;

        simplex::run_simplex(num_constraints, num_vars, a, ops, b, c, res, ans);

        cout << " " << ans << endl;
    }
}
```

8 String

8.1 KMP

```
// https://www.spoj.com/problems/NHAY/

#include "../kmp.cpp"

int main(void)
{
  int n;
  string key;
  while (scanf("%d", &n) != EOF)
  {
    string text;
    cin >> key;
    scanf(" ");
    char c;
    while ((c = getchar()) != '\n')
      text += c;

    for (int x : kmp(text, key))
      printf("%d\n", x);
    printf("\n");
  }
}
```

```
2
na
bananobano
6
foobar
foo
9
foobarfoo
barfoobarfoobarfoobarfoobarfoo
```

```cpp
// https://www.spoj.com/problems/PERIOD/

#include "../kmp.cpp"

int main(void)
{
  int t, n;
  scanf("%d", &t);
  for (int k = 1; k <= t; k++)
  {
    scanf("%d", &n);
    string s;
    cin >> s;

    vector<int> pi = prefix_function(s);

    printf("Test case #%d\n", k);
    for (int i = 1; i <= n; i++)
      if (pi[i] % (i - pi[i]) == 0 && i / (i - pi[i]) != 1)
        printf("%d %d\n", i, i / (i - pi[i]));
    printf("\n");
  }
}
```

8.2 Acho Corasick

```
// https://codeforces.com/problemset/problem/963/D

#include "../aho_corasick.cpp"

#define MAXN 112345

int k[MAXN];

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  string text;
  cin >> text;

  int n;
  cin >> n;
  vector<string> pats(n);
  for (int i = 0; i < n; i++)
    cin >> k[i] >> pats[i];

  aho_corasick aho(pats);
  auto tmp = aho.find_all(text);

  vector<vector<int>> m(n);
  for (int i = 0; i < sz(tmp); i++)
    for (auto x : tmp[i])
      m[x].push_back(i);

  for (int i = 0; i < n; i++)
  {
    int r = 0;
    int ans = inf;
    for (int j = 0; j + k[i] <= sz(m[i]); j++)
    {
      while (r < sz(m[i]) && r - j + 1 < k[i])
        r++;
      if (r - j + 1 == k[i])
        ans = min(ans, m[i][r] - m[i][j] + sz(pats[i]));
    }

    cout << (ans == inf ? -1 : ans) << "\n";
  }
}
```

```cpp
// https://br.spoj.com/problems/GROWIN10/
#include "../aho_corasick.cpp"

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int n;

  while (cin >> n && n)
  {
    vector<string> s(n);

    for (int i = 0; i < n; i++)
      cin >> s[i];

    sort(s.begin(), s.end(), [](const string &lhs, const string &rhs) { return lhs.
   size() < rhs.size(); });

    aho_corasick aho(s);

    vector<int> tab(n, 0);
    int ans = 0;
    for (int i = 0; i < n; i++)
    {
      tab[i] = 1;

      auto v = aho.find(s[i]);
      for (int j = 0; j + 1 < sz(s[i]); j++)
        if (v[j] >= 0)
          tab[i] = max(tab[v[j]] + 1, tab[i]);

      v = aho.find_all_at_pos(s[i], sz(s[i]) - 1);
      for (int j = 1; j < sz(v); j++)
        tab[i] = max(tab[v[j]] + 1, tab[i]);

      ans = max(ans, tab[i]);
    }

    cout << ans << endl;
  }
}
```

```
6
plant
ant
cant
decant
deca
an
2
supercalifragilisticexpialidocious
```

```
rag
0

3
plant
an
ant
0
```

8.3 Hash

```
// https://codeforces.com/contest/7/problem/D
#include "../hash.cpp"

char tmp[5123456];

int main(void)
{
  string s;
  scanf("%s", tmp);
  s = tmp;
  int n = sz(s);
  vector<int> tab(n+1);
  hash_interval hash(s, 137, 1000000007);
  reverse(s.begin(), s.end());
  hash_interval rev_hash(s, 137, 1000000007);
  ll ans = 0;
  for (int i = 1; i <= n; i++)
  {
    if (hash.get(0, (i-1)/2) == rev_hash.get(n - i, n - (i+2)/2))
      tab[i] = tab[i/2] + 1;
    else
      tab[i] = 0;

    ans += tab[i];
  }

  cout << ans << endl;
}
```

```cpp
// https://www.spoj.com/problems/ADACLEAN/
#include "../hash.cpp"

int main()
{
  cin.sync_with_stdio(0);
  cin.tie(0);

  int t;
  cin >> t;
  while (t--)
  {
    int n, k;
    string s;
    cin >> n >> k;
    cin >> s;

    hash_interval hash1(s, 137, 1000000007);
    hash_interval hash2(s, 137, 1000000009);
    set<pair<ll, ll>> rdm;
    for (int i = 0; i + k - 1 < n; i++)
      rdm.insert({hash1.get(i, i + k - 1), hash2.get(i, i + k - 1)});
    cout << rdm.size()<<endl;
  }
}
```

8.4 Suffix Array

```cpp
//https://www.spoj.com/problems/ADASTRNG/

#include "../sa.cpp"

long long ans[30];

int main(void)
{
  string s;
  cin >> s;
  auto sa = suffix_array(s);
  for (int i = 1; i <= s.size(); i++)
    ans[s[sa.sa[i]] - 'a'] += s.size() - sa.sa[i] - sa.lcp[i];

  for (int i = 0; i < 26; i++)
    printf("%lld%c", ans[i], i + 1 == 26 ? '\n' : ' ');
}
```

```
// https://www.spoj.com/problems/LONGCS/

#include "../sa.cpp"

int findsrc(vector<int> &v, int j)
{
  for (int i = 0; i < v.size(); i++)
    if (j < v[i])
      return i;
    else
    {
      j -= v[i];
    }

  assert(false);
}

int main(void)
{
  int T;
  scanf("%d", &T);
  while(T--)
  {
    int n;
    scanf("%d", &n);
    string s, tmp;
    vector<int> v;
    for (int i = 0; i < n; i++)
    {
      cin >> tmp;
      v.push_back(tmp.size() + 1);
      s += tmp;
      s += i + 1;
    }

    int ans = 0;
    auto sa = suffix_array(s);
    int j = 0;
    vector<int> cnt(n, 0);
    multiset<int> rdm;
    for (int i = 1; i <= s.size(); i++)
    {
      while (j + 1 <= s.size() && count(cnt.begin(), cnt.end(), 0) != 0)
      {
        j++;

        cnt[findsrc(v, sa.sa[j])]++;
        rdm.insert(sa.lcp[j]);
      }

      rdm.erase(rdm.find(sa.lcp[i]));

      if (count(cnt.begin(), cnt.end(), 0) == 0)
```

```
        ans = max(ans, *rdm.begin());

      cnt[findsrc(v, sa.sa[i])]--;
    }

    cout << ans << endl;
  }
}
```

```
2
2
aaabbb
bbaabb
3
icode
coder
contest
```

```cpp
// https://www.spoj.com/problems/SARRAY/

#include "../sa.cpp"

int main(void)
{
  string s;
  cin >> s;
  auto sa = suffix_array(s);
  for (int i = 1; i <= s.size(); i++)
    printf("%d\n", sa.sa[i]);
}
```

8.5 Suffix Tree

```
/*https://onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=
   show_problem&problem=1620
* Verify if strings are subvstrings of each other
*/
#include "../suffix_tree.cpp"

int main(){
    int t;
    scanf("%d", &t);

    while(t--){
    string s;
    cin >> s;

    suffix_tree st = suffix_tree(s);

    //st.print();

    int q;
    scanf("%d", &q);

    for(int i = 0; i < q; i++){
      string s;
      cin >> s;

      if(st.verify_substring(s))
        printf("y\n");
      else
        printf("n\n");
    }

    }

    return 0;
}
```