Contents

1 Data Structures

1.1 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
```

1.2 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
```

1.3 Dynamic Seg

```cpp
#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
```

1.4 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
```

```
// 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] = -0x3f3f3f3f3f3f3f3fll;
    // 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
```

1.5 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);
}
```

1.6 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);
  }
}
```

1.7 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
```

```
// 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 Geometry

2.1 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"

template <class T>
ostream &operator<<(ostream &os, point<T> p)
{
  return os << "(x=" << p.x << ", y=" << p.y << ")";
}

#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

// 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
```

2.2 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);
  }

}

/*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");


}
```

2.3 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 Graph

3.1 Biconnected Components

```
// 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;
}
```

```
// 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
```

3.2 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;
}
```

3.3 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
```

3.4 Max Flow (Dinic)

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

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

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

const int inf = 0x3f3f3f3f;
#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

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

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

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
```

3.5 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
```

```
// 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
```

3.6 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
}
```

3.7 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 Misc

NO APPLICATIONS

5 Number Theory

5.1 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
```

5.2 Pollard rho

```
 /*
  Description: Pollard-rho randomized factorization algorithm. Returns prime
  factors of a number, in arbitrary order (e.g. 2299 -> {11, 19, 11}).
  Time: O(n^1/4) gcd calls, less for numbers with small factors.

  Source: https://github.com/kth-competitive-programming/kactl/blob/master/content/
   number-theory/Factor.h
 */
typedef unsigned long long ull;
typedef long double ld;

ull mod_mul(ull a, ull b, ull M) {
  ll ret = a * b - M * ull(ld(a) * ld(b) / ld(M));
  return ret + M * (ret < 0) - M * (ret >= (ll)M);
}
ull mod_pow(ull b, ull e, ull mod) {
  ull ans = 1;
  for (; e; b = mod_mul(b, b, mod), e /= 2)
    if (e & 1) ans = mod_mul(ans, b, mod);
  return ans;
}

bool isPrime(ull n) {
  if (n < 2 || n % 6 % 4 != 1) return n - 2 < 2;
  ull A[] = {2, 325, 9375, 28178, 450775, 9780504, 1795265022},
      s = __builtin_ctzll(n-1), d = n >> s;
  for(auto &a : A) {   // ^ count trailing zeroes
    ull p = mod_pow(a, d, n), i = s;
    while (p != 1 && p != n - 1 && a % n && i--)
```

```
      p = mod_mul(p, p, n);
    if (p != n-1 && i != s) return 0;
  }
  return 1;
}

ull pollard(ull n) {
  auto f = [n](ull x) { return (mod_mul(x, x, n) + 1) % n; };
  if (!(n & 1)) return 2;
  for (ull i = 2;; i++) {
    ull x = i, y = f(x), p;
    while ((p = __gcd(n + y - x, n)) == 1)
      x = f(x), y = f(f(y));
    if (p != n) return p;
  }
}

vector<ull> factorize(ull n) {
  if (n == 1) return {};
  if (isPrime(n)) return {n};
  ull x = pollard(n);
  auto l = factorize(x), r = factorize(n / x);
  l.insert(l.end(), all(r));
  return l;
}
```

5.3 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;
}
```

5.4 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
```

5.5 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
```

6 Numerical

6.1 Big Int

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

// This code is not meant to be written in icpc contests. This is just here to fill
   a void for now.
// Source: someone on CF

// NOTE:
// This code contains various bug fixes compared to the original version from
// indy256 (github.com/indy256/codelibrary/blob/master/cpp/numbertheory/bigint-full.
   cpp),
// including:
// - Fix overflow bug in mul_karatsuba.
// - Fix overflow bug in fft.
// - Fix bug in initialization from long long.
// - Optimized operators + - *.
//
// Tested:
// - https://www.e-olymp.com/en/problems/266: Comparison
// - https://www.e-olymp.com/en/problems/267: Subtraction
// - https://www.e-olymp.com/en/problems/271: Multiplication
// - https://www.e-olymp.com/en/problems/272: Multiplication
// - https://www.e-olymp.com/en/problems/313: Addition
// - https://www.e-olymp.com/en/problems/314: Addition/Subtraction
// - https://www.e-olymp.com/en/problems/317: Multiplication (simple / karatsuba /
   fft)
// - https://www.e-olymp.com/en/problems/1327: Multiplication
// - https://www.e-olymp.com/en/problems/1328
// - VOJ BIGNUM: Addition, Subtraction, Multiplication.
// - SGU 111: sqrt
// - SGU 193
// - SPOJ MUL, VFMUL: Multiplication.
// - SPOJ FDIV, VFDIV: Division.

const int BASE_DIGITS = 9;
const int BASE = 1000000000;

struct BigInt {
    int sign;
    vector<int> a;

    // -------------------- Constructors --------------------
    // Default constructor.
    BigInt() : sign(1) {}

    // Constructor from long long.
    BigInt(long long v) {
        *this = v;
    }
    BigInt& operator = (long long v) {
```

```
        sign = 1;
        if (v < 0) {
            sign = -1;
            v = -v;
        }
        a.clear();
        for (; v > 0; v = v / BASE)
            a.push_back(v % BASE);
        return *this;
    }

    // Initialize from string.
    BigInt(const string& s) {
        read(s);
    }

    // -------------------- Input / Output --------------------
    void read(const string& s) {
        sign = 1;
        a.clear();
        int pos = 0;
        while (pos < (int) s.size() && (s[pos] == '-' || s[pos] == '+')) {
            if (s[pos] == '-')
                sign = -sign;
            ++pos;
        }
        for (int i = s.size() - 1; i >= pos; i -= BASE_DIGITS) {
            int x = 0;
            for (int j = max(pos, i - BASE_DIGITS + 1); j <= i; j++)
                x = x * 10 + s[j] - '0';
            a.push_back(x);
        }
        trim();
    }
    friend istream& operator>>(istream &stream, BigInt &v) {
        string s;
        stream >> s;
        v.read(s);
        return stream;
    }

    friend ostream& operator<<(ostream &stream, const BigInt &v) {
        if (v.sign == -1 && !v.isZero())
            stream << '-';
        stream << (v.a.empty() ? 0 : v.a.back());
        for (int i = (int) v.a.size() - 2; i >= 0; --i)
            stream << setw(BASE_DIGITS) << setfill('0') << v.a[i];
        return stream;
    }

    // -------------------- Comparison --------------------
    bool operator<(const BigInt &v) const {
        if (sign != v.sign)
```

```
            return sign < v.sign;
        if (a.size() != v.a.size())
            return a.size() * sign < v.a.size() * v.sign;
        for (int i = ((int) a.size()) - 1; i >= 0; i--)
            if (a[i] != v.a[i])
                return a[i] * sign < v.a[i] * sign;
        return false;
    }

    bool operator>(const BigInt &v) const {
        return v < *this;
    }
    bool operator<=(const BigInt &v) const {
        return !(v < *this);
    }
    bool operator>=(const BigInt &v) const {
        return !(*this < v);
    }
    bool operator==(const BigInt &v) const {
        return !(*this < v) && !(v < *this);
    }
    bool operator!=(const BigInt &v) const {
        return *this < v || v < *this;
    }

    // Returns:
    // 0 if |x| == |y|
    // -1 if |x| < |y|
    // 1 if |x| > |y|
    friend int __compare_abs(const BigInt& x, const BigInt& y) {
        if (x.a.size() != y.a.size()) {
            return x.a.size() < y.a.size() ? -1 : 1;
        }

        for (int i = ((int) x.a.size()) - 1; i >= 0; --i) {
            if (x.a[i] != y.a[i]) {
                return x.a[i] < y.a[i] ? -1 : 1;
            }
        }
        return 0;
    }

    // -------------------- Unary operator - and operators +- --------------------
    BigInt operator-() const {
        BigInt res = *this;
        if (isZero()) return res;

        res.sign = -sign;
        return res;
    }

    // Note: sign ignored.
    void __internal_add(const BigInt& v) {
```

```
        if (a.size() < v.a.size()) {
            a.resize(v.a.size(), 0);
        }
        for (int i = 0, carry = 0; i < (int) max(a.size(), v.a.size()) || carry; ++i
    ) {
            if (i == (int) a.size()) a.push_back(0);

            a[i] += carry + (i < (int) v.a.size() ? v.a[i] : 0);
            carry = a[i] >= BASE;
            if (carry) a[i] -= BASE;
        }
    }

    // Note: sign ignored.
    void __internal_sub(const BigInt& v) {
        for (int i = 0, carry = 0; i < (int) v.a.size() || carry; ++i) {
            a[i] -= carry + (i < (int) v.a.size() ? v.a[i] : 0);
            carry = a[i] < 0;
            if (carry) a[i] += BASE;
        }
        this->trim();
    }

    BigInt operator += (const BigInt& v) {
        if (sign == v.sign) {
            __internal_add(v);
        } else {
            if (__compare_abs(*this, v) >= 0) {
                __internal_sub(v);
            } else {
                BigInt vv = v;
                swap(*this, vv);
                __internal_sub(vv);
            }
        }
        return *this;
    }

    BigInt operator -= (const BigInt& v) {
        if (sign == v.sign) {
            if (__compare_abs(*this, v) >= 0) {
                __internal_sub(v);
            } else {
                BigInt vv = v;
                swap(*this, vv);
                __internal_sub(vv);
                this->sign = -this->sign;
            }
        } else {
            __internal_add(v);
        }
        return *this;
    }
```

```
    // Optimize operators + and - according to
    // https://stackoverflow.com/questions/13166079/move-semantics-and-pass-by-
   rvalue-reference-in-overloaded-arithmetic
    template< typename L, typename R >
        typename std::enable_if<
            std::is_convertible<L, BigInt>::value &&
            std::is_convertible<R, BigInt>::value &&
            std::is_lvalue_reference<R&&>::value,
            BigInt>::type friend operator + (L&& l, R&& r) {
        BigInt result(std::forward<L>(l));
        result += r;
        return result;
    }
    template< typename L, typename R >
        typename std::enable_if<
            std::is_convertible<L, BigInt>::value &&
            std::is_convertible<R, BigInt>::value &&
            std::is_rvalue_reference<R&&>::value,
            BigInt>::type friend operator + (L&& l, R&& r) {
        BigInt result(std::move(r));
        result += l;
        return result;
    }

    template< typename L, typename R >
        typename std::enable_if<
            std::is_convertible<L, BigInt>::value &&
            std::is_convertible<R, BigInt>::value,
            BigInt>::type friend operator - (L&& l, R&& r) {
        BigInt result(std::forward<L>(l));
        result -= r;
        return result;
    }

    // -------------------- Operators * / % --------------------
    friend pair<BigInt, BigInt> divmod(const BigInt& a1, const BigInt& b1) {
        assert(b1 > 0);  // divmod not well-defined for b < 0.

        long long norm = BASE / (b1.a.back() + 1);
        BigInt a = a1.abs() * norm;
        BigInt b = b1.abs() * norm;
        BigInt q = 0, r = 0;
        q.a.resize(a.a.size());

        for (int i = a.a.size() - 1; i >= 0; i--) {
            r *= BASE;
            r += a.a[i];
            long long s1 = r.a.size() <= b.a.size() ? 0 : r.a[b.a.size()];
            long long s2 = r.a.size() <= b.a.size() - 1 ? 0 : r.a[b.a.size() - 1];
            long long d = ((long long) BASE * s1 + s2) / b.a.back();
            r -= b * d;
            while (r < 0) {
```

```
            r += b, --d;
        }
        q.a[i] = d;
    }

    q.sign = a1.sign * b1.sign;
    r.sign = a1.sign;
    q.trim();
    r.trim();
    auto res = make_pair(q, r / norm);
    if (res.second < 0) res.second += b1;
    return res;
}
BigInt operator/(const BigInt &v) const {
    return divmod(*this, v).first;
}

BigInt operator%(const BigInt &v) const {
    return divmod(*this, v).second;
}

void operator/=(int v) {
    assert(v > 0);  // operator / not well-defined for v <= 0.
    if (llabs(v) >= BASE) {
        *this /= BigInt(v);
        return ;
    }
    if (v < 0)
        sign = -sign, v = -v;
    for (int i = (int) a.size() - 1, rem = 0; i >= 0; --i) {
        long long cur = a[i] + rem * (long long) BASE;
        a[i] = (int) (cur / v);
        rem = (int) (cur % v);
    }
    trim();
}

BigInt operator/(int v) const {
    assert(v > 0);  // operator / not well-defined for v <= 0.

    if (llabs(v) >= BASE) {
        return *this / BigInt(v);
    }
    BigInt res = *this;
    res /= v;
    return res;
}
void operator/=(const BigInt &v) {
    *this = *this / v;
}

long long operator%(long long v) const {
    assert(v > 0);  // operator / not well-defined for v <= 0.
```

```
        assert(v < BASE);
        int m = 0;
        for (int i = a.size() - 1; i >= 0; --i)
            m = (a[i] + m * (long long) BASE) % v;
        return m * sign;
    }

    void operator*=(int v) {
        if (llabs(v) >= BASE) {
            *this *= BigInt(v);
            return ;
        }
        if (v < 0)
            sign = -sign, v = -v;
        for (int i = 0, carry = 0; i < (int) a.size() || carry; ++i) {
            if (i == (int) a.size())
                a.push_back(0);
            long long cur = a[i] * (long long) v + carry;
            carry = (int) (cur / BASE);
            a[i] = (int) (cur % BASE);
            //asm("divl %%ecx" : "=a"(carry), "=d"(a[i]) : "A"(cur), "c"(base));
            /*
             int val;
             __asm {
             lea esi, cur
             mov eax, [esi]
             mov edx, [esi+4]
             mov ecx, base
             div ecx
             mov carry, eax
             mov val, edx;
             }
             a[i] = val;
             */
        }
        trim();
    }

    BigInt operator*(int v) const {
        if (llabs(v) >= BASE) {
            return *this * BigInt(v);
        }
        BigInt res = *this;
        res *= v;
        return res;
    }

    // Convert BASE 10^old --> 10^new.
    static vector<int> convert_base(const vector<int> &a, int old_digits, int
    new_digits) {
        vector<long long> p(max(old_digits, new_digits) + 1);
        p[0] = 1;
        for (int i = 1; i < (int) p.size(); i++)
```

```
            p[i] = p[i - 1] * 10;
        vector<int> res;
        long long cur = 0;
        int cur_digits = 0;
        for (int i = 0; i < (int) a.size(); i++) {
            cur += a[i] * p[cur_digits];
            cur_digits += old_digits;
            while (cur_digits >= new_digits) {
                res.push_back((long long)(cur % p[new_digits]));
                cur /= p[new_digits];
                cur_digits -= new_digits;
            }
        }
        res.push_back((int) cur);
        while (!res.empty() && !res.back())
            res.pop_back();
        return res;
    }

    void fft(vector<complex<double> > & a, bool invert) const {
        int n = (int) a.size();

        for (int i = 1, j = 0; i < n; ++i) {
            int bit = n >> 1;
            for (; j >= bit; bit >>= 1)
                j -= bit;
            j += bit;
            if (i < j)
                swap(a[i], a[j]);
        }

        for (int len = 2; len <= n; len <<= 1) {
            double ang = 2 * 3.14159265358979323846 / len * (invert ? -1 : 1);
            complex<double> wlen(cos(ang), sin(ang));
            for (int i = 0; i < n; i += len) {
                complex<double> w(1);
                for (int j = 0; j < len / 2; ++j) {
                    complex<double> u = a[i + j];
                    complex<double> v = a[i + j + len / 2] * w;
                    a[i + j] = u + v;
                    a[i + j + len / 2] = u - v;
                    w *= wlen;
                }
            }
        }
        if (invert)
            for (int i = 0; i < n; ++i)
                a[i] /= n;
    }

    void multiply_fft(const vector<int> &a, const vector<int> &b, vector<int> &res)
    const {
        vector<complex<double> > fa(a.begin(), a.end());
```

```
        vector<complex<double> > fb(b.begin(), b.end());
        int n = 1;
        while (n < (int) max(a.size(), b.size()))
            n <<= 1;
        n <<= 1;
        fa.resize(n);
        fb.resize(n);

        fft(fa, false);
        fft(fb, false);
        for (int i = 0; i < n; ++i)
            fa[i] *= fb[i];
        fft(fa, true);

        res.resize(n);
        long long carry = 0;
        for (int i = 0; i < n; ++i) {
            long long t = (long long) (fa[i].real() + 0.5) + carry;
            carry = t / 1000;
            res[i] = t % 1000;
        }
    }

    BigInt mul_simple(const BigInt &v) const {
        BigInt res;
        res.sign = sign * v.sign;
        res.a.resize(a.size() + v.a.size());
        for (int i = 0; i < (int) a.size(); ++i)
            if (a[i])
                for (int j = 0, carry = 0; j < (int) v.a.size() || carry; ++j) {
                    long long cur = res.a[i + j] + (long long) a[i] * (j < (int) v.a
.size() ? v.a[j] : 0) + carry;
                    carry = (int) (cur / BASE);
                    res.a[i + j] = (int) (cur % BASE);
                }
        res.trim();
        return res;
    }

    typedef vector<long long> vll;

    static vll karatsubaMultiply(const vll &a, const vll &b) {
        int n = a.size();
        vll res(n + n);
        if (n <= 32) {
            for (int i = 0; i < n; i++)
                for (int j = 0; j < n; j++)
                    res[i + j] += a[i] * b[j];
            return res;
        }

        int k = n >> 1;
        vll a1(a.begin(), a.begin() + k);
```

```
        vll a2(a.begin() + k, a.end());
        vll b1(b.begin(), b.begin() + k);
        vll b2(b.begin() + k, b.end());

        vll a1b1 = karatsubaMultiply(a1, b1);
        vll a2b2 = karatsubaMultiply(a2, b2);

        for (int i = 0; i < k; i++)
            a2[i] += a1[i];
        for (int i = 0; i < k; i++)
            b2[i] += b1[i];

        vll r = karatsubaMultiply(a2, b2);
        for (int i = 0; i < (int) a1b1.size(); i++)
            r[i] -= a1b1[i];
        for (int i = 0; i < (int) a2b2.size(); i++)
            r[i] -= a2b2[i];

        for (int i = 0; i < (int) r.size(); i++)
            res[i + k] += r[i];
        for (int i = 0; i < (int) a1b1.size(); i++)
            res[i] += a1b1[i];
        for (int i = 0; i < (int) a2b2.size(); i++)
            res[i + n] += a2b2[i];
        return res;
    }

    BigInt mul_karatsuba(const BigInt &v) const {
        vector<int> a6 = convert_base(this->a, BASE_DIGITS, 6);
        vector<int> b6 = convert_base(v.a, BASE_DIGITS, 6);
        vll a(a6.begin(), a6.end());
        vll b(b6.begin(), b6.end());
        while (a.size() < b.size())
            a.push_back(0);
        while (b.size() < a.size())
            b.push_back(0);
        while (a.size() & (a.size() - 1))
            a.push_back(0), b.push_back(0);
        vll c = karatsubaMultiply(a, b);
        BigInt res;
        res.sign = sign * v.sign;
        long long carry = 0;
        for (int i = 0; i < (int) c.size(); i++) {
            long long cur = c[i] + carry;
            res.a.push_back((int) (cur % 1000000));
            carry = cur / 1000000;
        }
        res.a = convert_base(res.a, 6, BASE_DIGITS);
        res.trim();
        return res;
    }

    void operator*=(const BigInt &v) {
```

```
        *this = *this * v;
    }
    BigInt operator*(const BigInt &v) const {
        if (a.size() * v.a.size() <= 1000111) return mul_simple(v);
        if (a.size() > 500111 || v.a.size() > 500111) return mul_fft(v);
        return mul_karatsuba(v);
    }

    BigInt mul_fft(const BigInt& v) const {
        BigInt res;
        res.sign = sign * v.sign;
        multiply_fft(convert_base(a, BASE_DIGITS, 3), convert_base(v.a, BASE_DIGITS,
    3), res.a);
        res.a = convert_base(res.a, 3, BASE_DIGITS);
        res.trim();
        return res;
    }

    // -------------------- Misc --------------------
    BigInt abs() const {
        BigInt res = *this;
        res.sign *= res.sign;
        return res;
    }
    void trim() {
        while (!a.empty() && !a.back())
            a.pop_back();
        if (a.empty())
            sign = 1;
    }

    bool isZero() const {
        return a.empty() || (a.size() == 1 && !a[0]);
    }

    friend BigInt gcd(const BigInt &a, const BigInt &b) {
        return b.isZero() ? a : gcd(b, a % b);
    }
    friend BigInt lcm(const BigInt &a, const BigInt &b) {
        return a / gcd(a, b) * b;
    }

    friend BigInt sqrt(const BigInt &a1) {
        BigInt a = a1;
        while (a.a.empty() || a.a.size() % 2 == 1)
            a.a.push_back(0);

        int n = a.a.size();

        int firstDigit = (int) sqrt((double) a.a[n - 1] * BASE + a.a[n - 2]);
        int norm = BASE / (firstDigit + 1);
        a *= norm;
        a *= norm;
```

```
        while (a.a.empty() || a.a.size() % 2 == 1)
            a.a.push_back(0);

        BigInt r = (long long) a.a[n - 1] * BASE + a.a[n - 2];
        firstDigit = (int) sqrt((double) a.a[n - 1] * BASE + a.a[n - 2]);
        int q = firstDigit;
        BigInt res;

        for(int j = n / 2 - 1; j >= 0; j--) {
            for(; ; --q) {
                BigInt r1 = (r - (res * 2 * BigInt(BASE) + q) * q) * BigInt(BASE) *
   BigInt(BASE) + (j > 0 ? (long long) a.a[2 * j - 1] * BASE + a.a[2 * j - 2] : 0);
                if (r1 >= 0) {
                    r = r1;
                    break;
                }
            }
            res *= BASE;
            res += q;

            if (j > 0) {
                int d1 = res.a.size() + 2 < r.a.size() ? r.a[res.a.size() + 2] : 0;
                int d2 = res.a.size() + 1 < r.a.size() ? r.a[res.a.size() + 1] : 0;
                int d3 = res.a.size() < r.a.size() ? r.a[res.a.size()] : 0;
                q = ((long long) d1 * BASE * BASE + (long long) d2 * BASE + d3) / (
   firstDigit * 2);
            }
        }

        res.trim();
        return res / norm;
    }
};
```

6.2 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');
  }
}
```

6.3 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;
}
```

6.4 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
```

6.5 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;
}
```

6.6 Simplex

```
// 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;
}

// 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;
    }
}
```

7 String

7.1 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
```

```
// 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");
  }
}
```

7.2 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
```

7.3 Suffix Array

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

#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
```

```
// 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]);
}
```