1 Template

```cpp
// :80 <enter>
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef pair<int, ll> pil;
typedef pair<ll, int> pli;
typedef pair<double,double> pdd;
#define SQ(i) ((i)*(i))
#define MEM(a, b) memset(a, (b), sizeof(a))
#define SZ(i) int(i.size())
#define FOR(i, j, k, in) for (int i=j ; i<(k) ; i+=in)
#define RFOR(i, j, k, in) for (int i=j ; i>=(k) ; i-=in)
#define REP(i, j) FOR(i, 0, j, 1)
#define REP1(i,j) FOR(i, 1, j+1, 1)
#define RREP(i, j) RFOR(i, j, 0, 1)
#define ALL(_a) _a.begin(),_a.end()
#define mp make_pair
#define pb push_back
#define eb emplace_back
#define X first
#define Y second

#define endl '\n'
#define IOS() ios_base::sync_with_stdio(0);cin.tie(0)

const ll MOD = 1000000007;
const ll INF = 0x3f3f3f3f3f3f3f3f;
const int iNF = 0x3f3f3f3f;
const ll MAXN = 100005;

void solve(){

}

/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  int t = 1;
  cin >> t;
  while(t--){
    solve();
  }

  return 0;
}
```

2 BIT

```cpp
class BIT{
private:
  vector<ll> bit;
public:
  BIT(): bit(vector<ll>(MAXN, 0)) {}
  ll lowbit(ll x){
    return x & (-x);
  }

  ll query(ll idx){
    if(idx == 0) return 0;
    ll ans = 0;
    for(; idx>0; idx-=lowbit(idx)){
      ans += bit[idx];
    }

    return ans;
  }

  void modify(ll idx, int val){
    for(; idx <= n; idx += lowbit(idx)){
      bit[idx] += val;
    }
  }
};
```

3 Dijkstra

```cpp
// djisktra with priority queue and memory optimize

#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<ll, ll> pll;
#define pb push_back
#define mp make_pair

const ll INF = 0x3f3f3f3f3f3f3f3f;

int n, m;
vector<vector<pll>> w;
vector<ll> d;
vector<int> parent;

void dijkstra(int src){
  // vector<bool> visit(n+1, false);
  priority_queue<pll, vector<pll>, greater<pll>> pq; // first: weight, second: vertex
  for(int i=1;i<n;i++){
    pq.push(mp(INF, i));
  }
  d[src] = 0;
  parent[src] = src;
  while(!pq.empty()){
    pll edge = pq.top();
    pq.pop();
    ll u = edge.second;
    for(auto i:w[u]){
      ll w = i.second;
      ll v = i.first;
      ll alt = d[u] + i.second;
      if(alt < d[v]){
        d[v] = alt;
        parent[v] = u;
        pq.push(mp(alt, v));
      }
    }
  }
}

```

```
int main() {

  cin >> n >> m;
  w.resize(n+1);
  d.resize(n+1, INF);
  parent.resize(n+1, -1);
  ll a, b, tmp;
  for(int i=0;i<m;i++){
    cin >> a >> b >> tmp;
    w[a].pb(mp(b, tmp));
    w[b].pb(mp(a, tmp));
  }
  dijkstra(1); // source = 1
  vector<int> ans;
  int cur;
  if(d[n] != INF){ // if d[n] = INF, there
    ↪ is no shortest path from vertex 1 to vertex n
    cur = n;
    while(cur != 1){
      ans.push_back(cur);
      cur = parent[cur];
    }
    ans.push_back(1);
    int sz = ans.size();
    for(int i=sz-1;i>=0;i--){
      cout << ans[i] << " ";
    }
    cout << endl;
  } else {
    cout << -1 << endl;
  }


  return 0;
}
```

4 Segment Tree

```
// 利用線段樹解決區間求和 & 單點修改
#include <bits/stdc++.h>

using namespace std;

struct Node{
  int val;
  Node *lc, *rc;
  void pull(){
    val = lc -> val + rc -> val;
  }
};

const int n = 5;

int v[n + 1] = {0, 1, 2, 3, 4, 5};

Node* build(int L, int R){
  Node *node = new Node();
  if(L == R){
    node -> val = v[L];
  }
  int mid = (L+R) >> 1;
  node -> lc = build(L, mid);
  node -> rc = build(mid + 1, R);
  node -> pull();

  return node;
}

void modify(Node*
  ↪ node, int L, int R, int i, int d){
  if(L == R){
    assert(L == i);
    node->val += d;
    return;
  }
  int mid = (L + R) >> 1;
  if(i <= mid){
    modify(node -> lc, L, mid, i, d);
  } else {
    modify(node -> rc, mid + 1, R, i ,d);
  }
  node -> pull();
}

int query(Node*
  ↪ node, int L, int R, int ql, int qr){
  if(ql > R || qr < L) return 0;
  if(ql <= L && R <= qr) return node -> val;
  int mid = (L + R) >> 1;
  return query(node -> lc, L, mid, ql, qr) +
     query(node -> rc, mid + 1, R, ql, qr);
}
```

5 Segment Tree w/ lazy tags

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef pair<int, ll> pil;
typedef pair<ll, int> pli;
typedef pair<double,double> pdd;

struct Node{
  int val, tag;
  Node *lc, *rc;
  Node(){
    tag = val = 0;
    lc = rc = nullptr;
  }
  void pull(){
    val = lc -> val + rc -> val;
  }
};

const int n = 5;
int v[n+1] = {0, 1, 16, 2, 8, 4}; // input, 1-base

Node* build(int L, int R){
  Node *node = new Node();
  if( L == R ){
    node -> val = v[L];
    return node;
  }
  int mid = (L + R) >> 1;
  node -> lc = build(L, mid);
  node -> rc = build(mid+1, R);
```

```cpp
  node->pull();
  return node;
}

void push(Node* node, int L, int R){
  if(!node->tag) return;
  if(L != R){
    int mid = ( L + R ) >> 1;
    node->lc->tag += node->tag;
    node->rc->tag += node->tag;
    node->lc->val += node->tag * (mid - L + 1);
    node->rc->val += node->tag * ( R - mid );
  }
  node->tag = 0;
}

void modify(Node*
  ↪ node, int L, int R, int ql, int qr, int d){
  debug(L, R);
  if(ql > R || qr < L) return;
  if(ql <= L && R <= qr){
    node->tag += d;
    node->val += d * (R - L + 1);
    return;
  }
  push(node, L, R);
  int mid = (L + R) >> 1;
  modify(node->lc, L, mid, ql, qr, d);
  modify(node->rc, mid+1, R, ql, qr, d);
  node->pull();
}

int query(Node*
  ↪ node, int L, int R, int ql, int qr){
  if(ql > R || qr < L) return 0;
  if(ql <= L && R <= qr) return node->val;
  push(node, L, R);
  int mid = (L + R) >> 1;
  return query(node->lc, L, mid, ql,
    ↪ qr) + query(node->rc, mid + 1, R, ql, qr);
}
/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  Node* root = build(1, n);
  // cout << query(root, 1, n, 1, 5) << endl;
  modify(root, 1, n, 1, 4, 3);

  // cout << query(root, 1, n, 3, 3) << endl;

  return 0;
}
```

6 Disjoint Set

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

const int n = 100;

vector<int> p(n);
vector<int> sz(n, 1);

void init(){
  for(int i=0;i<n;i++){
    p[i] = i;
  }
}

int find(int x){
  if(p[x] == x) return x;

  return p[x] = find(p[x]);
}

void merge(int x, int y){
  int fx = find(x);
  int fy = find(y);
  if(fx == fy) return;
  if(sz[fx] < sz[fy]) swap(fx, fy);

  p[fy] = fx;
  sz[fx] += sz[fy];
}

int main(){

}
```

7 Fast Power MOD

```cpp
// C++ program to find
// (a^b)%m for b very large.
#include <bits/stdc++.h>
#define ll long long int
using namespace std;

// Function to find power
ll power(ll x, ll y, ll p)
{
  ll res = 1; // Initialize result

  // Update x if it is more than or
  // equal to p
  x = x % p;

  while (y > 0) {
    // If y is odd, multiply x with the result
    if (y & 1)
      res = (res * x) % p;

    // y must be even now
    y = y >> 1; // y = y/2
    x = (x * x) % p;
  }
  return res;
}
// Driver Code
int main()
{
  ll a = 3;

  // String input as b is very large
  string b = "100000000000000000000000000";

  ll remainderB = 0;
  ll MOD = 1000000007;

  // Reduce the number B to a small number
  // using Fermat Little
  for (int i = 0; i < b.length(); i++)
    remainderB
      = (remainderB * 10 + b[i] - '0') % (MOD - 1);

  cout << power(a, remainderB, MOD) << endl;
  return 0;
}
```

8 Kosaraju

```cpp
// same scc will number

#include <bits/stdc++.h>

using namespace std;

const int N = 100; // # of vertex

vector<vector<int>> g; // graph
vector<vector<int>> r; // reversed graph
vector<int> order;
vector<int> scc(N);
vector<bool> vis(N);

void RevDfs(int cur){
  vis[cur] = true;
  for(int u: r[cur]){
    if(!vis[u]){
      RevDfs(u);
    }
  }
  order.push_back(cur); // topological order
}

void Dfs(int cur, int s){
  scc[cur] = s;
  for(int u: g[cur]){
    if(scc[u] == -1) Dfs(u, s);
  }
}

void Kosaraju(int n){
  fill(vis.begin(), vis.end(), false);
  fill(scc.begin(), scc.end(), -1);

  for(int i=0;i<n;i++){
    if(!vis[i]) RevDfs(i);
  }

  int n_scc = 0;
  for(int i=n-1;i>=0;i--){
    int cur = order[i];
    if(scc[cur] == -1){
      Dfs(cur, n_scc);
      n_scc++;
    }
  }
}
```

9 Treap

```cpp
#include <cstdio>
#include <algorithm>
#include <stack>
#include <ctime>
#include <cstdlib>
#include <queue>
#define MAXN 800000
#define INF 2147483647
using namespace std;
struct treap
{
  int v;
  int sz;
  int p;
  int mn;
  int rev;
  int add;
  treap *l, *r;
  treap() {}
  treap(int k) : v(k), sz(1), p(rand()),
    mn(k), rev(0), add(0), l(NULL), r(NULL) {}
};

treap mempool[MAXN];
treap* ptr;
treap* gc; //
    use treap as linked list to garbage collect

inline void init() {
  ptr = mempool;
  gc = NULL;
}

inline void Del(treap* t) {
  t->l = gc;
  gc = t;
}

inline treap* New(int v) {
  if (gc == NULL) {
    *ptr = treap(v);
    return ptr++;
  } else {
    treap* t = gc;
    gc = gc->l;
    *t = treap(v);
    return t;
  }
}

inline int size(treap* t) {
  return t != NULL ? t->sz : 0;
}

inline int small(treap* t) {
  return t != NULL ? t->mn + t->add : INF;
}

inline void pull(treap* t) {
  if (t == NULL)
    return;
  t->sz = 1 + size(t->l) + size(t->r);
  t->mn = min(t->v,
  ↪   min(small(t->l), small(t->r)));
}

inline void reverse(treap* t) {
  if (t != NULL)
    t->rev ^= 1;
}

inline void addn(treap* t, int v) {
  if (t != NULL)
    t->add += v;
}

inline treap* push(treap* t) {
  if (t != NULL) {
    if (t->rev) {
      swap(t->l, t->r);
      reverse(t->l);
      reverse(t->r);
      t->rev = 0;
    }

    if (t->add) {
      t->v += t->add;
      t->mn += t->add;
      addn(t->l, t->add);
      addn(t->r, t->add);
      t->add = 0;
    }
  }
  return t;
}

void split(treap*
↪  t, int k, treap*& a, treap*& b) {
↪  // split first k nodes from t to a, others to b
  push(t);

  if (t == NULL) {
    a = b = NULL;
  } else if (size(t->l) + 1 <= k) {
    a = t;
    split(t->r, k - size(t->l) - 1, a->r, b);
    pull(a);
  } else {
    b = t;
    split(t->l, k, a, b->l);
    pull(b);
  }
}

treap* merge(treap* a, treap* b) {
  if (a == NULL)
    return push(b);
  else if (b == NULL)
    return push(a);
  if (a->p > b->p) {
    push(a);
    a->r = merge(a->r, b);
    pull(a);
    return a;
  } else {
    push(b);
    b->l = merge(a, b->l);
    pull(b);
    return b;
  }
}

inline void slice(treap* t, int
↪   x, int y, treap*& l, treap*& m, treap*& r) {
  split(t, x - 1, l, r);
  split(r, y - x + 1, m, r);
}

treap* build(int n) {
  treap* r = NULL;
  int v;
  stack<treap*> rc;
  treap* nt;
  while (n--) {
    scanf("%d", &v);
    nt = New(v);
    r = NULL;
    while (!rc.empty() && rc.top()->p < nt->p) {
      pull(r = rc.top());
      rc.pop();
    }
    nt->l = r;
    if (!rc.empty())
      rc.top()->r = nt;
    rc.push(nt);
  }
  while (!rc.empty()) {
    pull(r = rc.top());
    rc.pop();
  }
  return r;
}

int main()
{
  srand(42);
  int n, q;
  char cmd[10];
  int x, y, v;
  treap *l, *m, *r;
  treap *ml, *mr;
  treap* root;

  while (scanf("%d", &n) == 1) {
    init();
    root = build(n);
    scanf("%d", &q);
    while (q--) {
      scanf("%s", cmd);
      switch (cmd[0]) {
      case 'A':
        scanf("%d%d%d", &x, &y, &v);
        slice(root, x, y, l, m, r);
        addn(m, v);
        root = merge(merge(l, m), r);
        break;
      case 'I':
        scanf("%d%d", &x, &v);
        split(root, x, l, r);
        root = merge(merge(l, New(v)), r);
        break;
      case 'D':
```

```cpp
        scanf("%d", &x);
        slice(root, x, x, l, m, r);
        Del(m);
        root = merge(l, r);
        break;
      case 'M':
        scanf("%d%d", &x, &y);
        slice(root, x, y, l, m, r);
        printf("%d\n", m->mn);
        root = merge(merge(l, m), r);
        break;
      case 'R':
        scanf("%d%d", &x, &y);
        switch (cmd[3]) {
        case 'E':
          slice(root, x, y, l, m, r);
          reverse(m);
          root = merge(merge(l, m), r);
          break;
        case 'O':
          scanf("%d", &v);
          int len = (y-x+1);
          v = (v % len + len) % len; // v could be negative?
          if (v) {
            slice(root, x, y, l, m, r);
            split(m, len-v, ml, mr);
            root = merge(merge(l, merge(mr, ml)), r);
          }
          break;
        }
        break;
      }
    }
  }
  return 0;
}
```

10 FFT

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

using namespace std;

typedef complex<double> cd;
const double PI = acos(-1);

void fft(vector<cd>& a, bool invert){
  int n = a.size();
  if(n == 1){
    return;
  }
  vector<cd> a0(n/2), a1(n/2);
  for(int i=0;2*i<n;i++){
    a0[i] = a[2*i];
    a1[i] = a[2*i+1];
  }
  fft(a0, invert);
  fft(a1, invert);

  double ang = 2 * PI / n * (invert ? -1 : 1);
  cd w(1), wn(cos(ang), sin(ang));
  for(int i=0;2*i<n;i++){
    a[i] = a0[i] + w * a1[i];
    a[i+n/2] = a0[i] - w * a1[i];
    if(invert){
      a[i] /= 2;
      a[i+n/2] /= 2;
    }
    w *= wn;
  }
}

vector<int> multiply(vector<int>& a, vector<int>& b){
  vector<cd> fa(a.begin(), a.end()), fb(b.begin(), b.end());
  int n = 1;
  while(n < (int)a.size() + (int)b.size()){
    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);

  vector<int> result(n);
  for(int i=0;i<n;i++){
    result[i] = round(fa[i].real());
  }

  return result;
}

int main(){
  vector<int> a{1, 2, 1};
  vector<int> b{2, 4, 6};

  auto res = multiply(a, b);
  for(auto i : res){
    cout << i << " ";
  }
  cout << endl;
}
```

11 pbds ordered set

```cpp
// C++ program to demonstrate the
// ordered set in GNU C++
#include <iostream>
using namespace std;

// Header files, namespaces,
// macros as defined above
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/pb_ds/tree_policy.hpp>
using namespace __gnu_pbds;

```

```cpp
#define ordered_set tree<int, null_type,less<int>, rb_tree_tag,tree_order_statistics_node_update

// Driver program to test above functions
int main()
{
  // Ordered set declared with name o_set
  ordered_set o_set;

  // insert function to insert in
  // ordered set same as SET STL
  o_set.insert(5);
  o_set.insert(1);
  o_set.insert(2);

  // Finding the second smallest element
  // in the set using * because
  // find_by_order returns an iterator
  cout << *(o_set.find_by_order(1))
    << endl;

  // Finding the number of elements
  // strictly less than k=4
  cout << o_set.order_of_key(4)
    << endl;

  // Finding the count of elements less
  // than or equal to 4 i.e. strictly less
  // than 5 if integers are present
  cout << o_set.order_of_key(5)
    << endl;

  // Deleting 2 from the set if it exists
  if (o_set.find(2) != o_set.end())
    o_set.erase(o_set.find(2));

  // Now after deleting 2 from the set
  // Finding the second smallest element in the set
  cout << *(o_set.find_by_order(1))
    << endl;

  // Finding the number of
  // elements strictly less than k=4
  cout << o_set.order_of_key(4)
    << endl;

  return 0;
}
```

12 Almost union-find

```cpp
vector<int> dsu(MAXN);
vector<int> convert(MAXN);
vector<ll> sz(MAXN, 1);
vector<ll> sum(MAXN, 0);
int n, q;
int find(int a){
  if(dsu[a] == a){
    return a;
  }
  return dsu[a] = find(dsu[a]);
}
void merge(int a, int b){
  int fa = find(a);
  int fb = find(b);
  if(fa == fb){
    return;
  } else {
    sum[fb] += sum[fa];
    sz[fb] += sz[fa];
    dsu[fa] = fb;
  }
}
void move(int a, int b){
  int fa = find(convert[a]);
  int fb = find(convert[b]);
  if(fa == fb){
    return;
  } else {
    sz[fa] -= 1;
    sum[fa] -= a;
    convert[a] = ++n;
    sum[convert[a]] = a;
    sz[convert[a]] = 1;
    merge(convert[a], convert[b]);
  }
}
void solve(){
  while(cin >> n >> q){
    for(int i=1;i<MAXN;i++){
      dsu[i] = sum[i] = convert[i] = i;
      sz[i] = 1;
    }
    int a, b, c;
    while(q--){
      cin >> c;
      if(c == 1){
        cin >> a >> b;
        merge(convert[a], convert[b]);
      } else if(c == 2){
        cin >> a >> b;
        move(a, b);
      } else {
        cin >> a;
        a = find(convert[a]);
        cout << sz[a] << " " << sum[a] << endl;
      }
    }
  }

}
```

13 Mega inversion

```cpp
inline ll lowbit(ll x){
  return x & (-x);
}

int n;
vector<int> a;
class BIT{
private:
  vector<ll> sum;
public:
  BIT(): sum(vector<ll>(MAXN, 0)) {}
  void update(int idx, int val){
    for(int i=idx;i<=n;i+=lowbit(i)){
      sum[i] += val;
    }
  }
  ll query(int idx){
    ll res = 0;
    for(int i=idx;i>0;i-=lowbit(i)){
      res += sum[i];
    }

    return res;
  }
};

void solve(){
  cin >> n;
  a.resize(n);
  for(auto &i: a) cin >> i;
  BIT bit1, bit2;
  vector<ll> ans1(MAXN, 0);
  vector<ll> ans2(MAXN, 0);
  for(int i=n-1;i>=0;i--){
    ans1[i] = bit1.query(a[i]-1);
    bit1.update(a[i], 1);
  }
  for(int i=0;i<n;i++){
    ans2[i] = i - bit2.query(a[i]);
    bit2.update(a[i], 1);
  }
  ll ans = 0;
  for(int i=0;i<n;i++){
    ans += ans1[i] * ans2[i];
  }
  cout << ans << endl;

}

/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  int t = 1;
  // cin >> t;
  while(t--){
    solve();
  }

  return 0;
}
```

14 逆序數對

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int Case, n, a[100005], bit[100005];
long long ans;
vector <int> v;

int get_id(int x){
  return lower_bound(v.begin(), v.end(), x)-v.begin()+1;
}
void update(int x){
  while (x <= n){
    bit[x]++;
    x += x & (-x);
  }
}
int query(int x){
  int ret = 0;
  while (x){
    ret += bit[x];
    x -= x & (-x);
  }
  return ret;
}

int main() {
  ios_base::sync_with_stdio(0);
  cin.tie(0);
  while (cin >> n){
    if (n == 0) break;
    v.clear();
    ans = 0;
    for (int i = 0; i < n; i++){
      cin >> a[i];
      bit[i] = 0;
      v.push_back(a[i]);
    }
    bit[n] = 0;
    sort(v.begin(), v.end());
    v.erase(unique(v.begin(), v.end()), v.end());
    for (int i = 0; i < n; i++){
      ans += i-query(get_id(a[i]));
      update(get_id(a[i]));
    }
    Case++;
    cout << "Case #" << Case << ": " << ans << "\n";
  }
}
```

15 Batmanacci

```python
n, k = input().split()
n = int(n)
k = int(k)

fib = [0, 1, 1]

for i in range(3, n+1):
  fib.append(fib[i-1] + fib[i-2])

while n > 2:
  if k > fib[n-2]:
    k -= fib[n-2]
    n -= 1
  else:
    n -= 2

if n == 1:
  print("N")
else:
  print("A")
```

16 模逆元

```cpp
#include <iostream>
using namespace std;
#define ll long long

const int mod = 17, maxn = 20;

ll pre[maxn+1];
ll inv[maxn+1];
ll prei[maxn+1];

void build(int n){
  pre[1] = pre[0] = 1, inv[1] = inv[0] = 1, prei[1] = prei[0] = 1;
  for(int i = 2 ; i <= n ; i++){
    pre[i] = pre[i-1] * i % mod;
    // i 的逆元 inv[i]=-(p/i)*inv[p%i] (mod p)
    inv[i] = mod - (mod / i * inv[mod % i]) % mod;
    prei[i] = prei[i-1] * inv[i] % mod;
  }
}

ll C(int n, int k){
 return pre[n] * prei[k] % mod * prei[n-k] % mod;
}

int main(){
  build(maxn);
  cout << inv[10] << endl;
  cout << C(6, 3) << endl;
}
```

17 樹重心

```cpp
#include <iostream>
#include <vector>
using namespace std;
const int maxn = 100005;
int N;
vector<int> g[maxn];
int cost[maxn]; //慘度

int dfs(int now, int pre){
  //tot: 以 now 為 root 的子樹 size
  int tot = 1, ret = 0;
  // now 下方的子樹
  for (auto nxt : g[now]){
    if (nxt != pre){
      ret = dfs(nxt, now);
      tot += ret;
      cost[now] = max(cost[now], ret);
    }
  }
  // now 頭上的子樹
  cost[now] = max(cost[now], N - tot);
  return tot;
}

int main() {
  ios_base::sync_with_stdio(0);
  cin.tie(0);
  int T, a, b;
  cin >> T;
  while (T--){
    cin >> N;
    for (int i = 0; i < N; i++){
      cost[i] = 0;
      g[i].clear();
    }
    for (int i = 0; i < N-1; i++){
      cin >> a >> b;
      g[a].push_back(b);
      g[b].push_back(a);
    }
    dfs(0, -1); //將 0 當作 root
    int mn = 0x7FFFFFFF;
    int ans = -1;
    for (int i = 0; i < N; i++){
      if (cost[i] < mn){
        mn = cost[i];
        ans = i;
      }
    }
    cout << ans << "\n";
  }
  return 0;
}
```

```cpp

// ⊠份代⊠默⊠⊠点⊠⊠⊠ 1 ⊠始，即 i □ [1,n]
int size[MAXN],  // ⊠⊠⊠点的 “大小”（所有子⊠上⊠点⊠ + ⊠⊠点）
  weight[MAXN], // ⊠⊠⊠点的 “重量”
  centroid[2];  // 用于⊠⊠⊠的重心（存的是⊠点⊠⊠）

void GetCentroid(int cur, int fa) { // cur 表示⊠前⊠点 (current)
 size[cur] = 1;
 weight[cur] = 0;
 for (int i = head[cur]; i != -1; i = e[i].nxt) {
  if (e[i].to != fa) { // e[i].to 表示⊠⊠有向⊠所通向的⊠点。
   GetCentroid(e[i].to, cur);
   size[cur] += size[e[i].to];
   weight[cur] = max(weight[cur], size[e[i].to]);
  }
 }
 weight[cur] = max(weight[cur], n - size[cur]);
 if (weight[cur] <= n / 2) { // 依照⊠的重心的定⊠⊠⊠
  centroid[centroid[0] != 0] = cur;
 }
}
```

18 pbds hashtable

```cpp
#include <ext/pb_ds/assoc_container.hpp>
using namespace __gnu_pbds;
gp_hash_table<int, int> table;
cc_hash_table<int, int> table;
```

19 0/1 背包

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

void solve(int c, int n){
  vector<int> v(n);
  vector<int> w(n);
  REP(i, n){
    cin >> v[i] >> w[i];
  }
  vector<vector<int>>
   ↪ dp(n+1, vector<int>(c+1, 0));
  vector<vector<bool>>
   ↪ has(n+1, vector<bool>(c+1, false));
  for(int i=0;i<n;i++){
    for(int j=0;j<=c;j++){
      if(j - w[i] < 0){
        dp[i+1][j] = dp[i][j];
      } else {
        if(dp[i][j-w[i]] + v[i] > dp[i][j]){
          dp[i+1][j] = dp[i][j-w[i]] + v[i];
          has[i][j] = true;
        } else {
          dp[i+1][j] = dp[i][j];
        }
      }
    }
  }
  vector<int> ans;
  for(int i=n-1, j=c;i>=0;i--){
    if(has[i][j]){
      ans.push_back(i);
      j -= w[i];
    }
  }
  cout << ans.size() << endl;
  for(auto i: ans) cout << i << " ";
  cout << endl;
}

/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  // cin >> t;
  int c, n;
  while(cin >> c >> n){
    solve(c, n);
  }

  return 0;
}
```

20 有限背包

```cpp
const int N = 100, W = 100000;
int cost[N], weight[N], number[N];
int c[W + 1];

void knapsack(int n, int w)
{
  for (int i = 0; i < n; ++i)
  {
    int num = min(number[i], w / weight[i]);
    for (int k = 1; num > 0; k *= 2)
    {
      if (k > num) k = num;
      num -= k;
      for (int j = w; j >= weight[i] * k; --j)
        c[j] = max(c[j],
          ↪ c[j - weight[i] * k] + cost[i] * k);
    }
  }
  cout << "最高的價值為" << c[w];
}
```

21 無限背包

$$c(n,w)=max(c(n-1,w),c(n,w-weight[n])+cost[n])$$

```cpp
const int N = 100, W = 100000;
int cost[N], weight[N];
int c[W + 1];

void knapsack(int n, int w)
{
  memset(c, 0, sizeof(c));

  for (int i=0; i<n; ++i)
    for (int j = weight[i]; j <= w; ++j)
      c[j] = max(c[j], c[j - weight[i]] + cost[i]);

  cout << "最高的價值為" << c[w];
}
```

22 換錢問題

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

void solve(){
  vector<int> dp(MAXN, iNF);
  int x;
  cin >> x;
  int n;
  cin >> n;
  vector<int> a(n);
  REP(i, n) cin >> a[i];
  dp[0] = 0;
  for(int i=0;i<n;i++){
    for(int j=MAXN;j>=0;j--){
      if(j - a[i] >= 0){
        dp[j] = min(dp[j], dp[j-a[i]] + 1);
      }
    }
  }
  for(int i=x;i<MAXN;i++){
    if(dp[i] != iNF){
      cout << i << " " << dp[i] << endl;
      return;
    }
  }
}

```

```cpp
/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  int t = 1;
  cin >> t;
  while(t--){
    solve();
  }

  return 0;
}
```

23 Increasing Subsequences

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

void print(vector<int>& a, vector<int>& prev, int idx){
  vector<int> tmp;
  while(idx != -1){
    tmp.push_back(a[idx]);
    idx = prev[idx];
  }
  reverse(ALL(tmp));
  for(auto i: tmp) cout << i << " ";
  cout << endl;
}

void solve(){
  int n;
  cin >> n;
  if(n == 0) exit(0);
  vector<int> a(n);
  for(auto &i: a) cin >> i;
  vector<int> dp(n, 1);
  vector<int> prev(n, -1);
  for(int i=0;i<n;i++){
    for(int j=0;j<i;j++){
      if(a[j] < a[i] and (dp[j] + 1 > dp[i] or (dp[j] + 1 == dp[i] and (prev[i] != -1 and a[prev[i]] > a[j])))){
        dp[i] = dp[j] + 1;
        prev[i] = j;
      }
    }
  }
  int ans = 0;
  int ans_val = iNF;
  for(int i=0;i<n;i++){
    if(ans < dp[i] or (ans == dp[i] and ans_val > a[i])){
      ans = dp[i];
      ans_val = a[i];
    }
  }
  cout << ans << " ";
  for(int i=0;i<n;i++){
    if(dp[i] == ans and ans_val == a[i]){
      print(a, prev, i);
      break;
    }
  }
}

/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  // cin >> t;
  while(true){
    solve();
  }

  return 0;
}
```

24 Exgcd

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef pair<int, ll> pil;
typedef pair<ll, int> pli;
typedef pair<double,double> pdd;
#define SQ(i) ((i)*(i))
#define MEM(a, b) memset(a, (b), sizeof(a))
#define SZ(i) int(i.size())
#define FOR(i, j, k, in) for (int i=j ; i<(k) ; i+=in)
#define RFOR(i, j, k, in) for (int i=j ; i>=(k) ; i-=in)
#define REP(i, j) FOR(i, 0, j, 1)
#define REP1(i,j) FOR(i, 1, j+1, 1)
#define RREP(i, j) RFOR(i, j, 0, 1)
#define ALL(_a) _a.begin(),_a.end()
#define mp make_pair
#define pb push_back
#define eb emplace_back
#define X first
#define Y second
#ifdef tmd
#define TIME(i) Timer i(#i)
#define debug( ... ) do{\
    fprintf(stderr,"%s - %d (%s) = ",__PRETTY_FUNCTION__,__LINE__,#__VA_ARGS__);\
  _do(__VA_ARGS__);\
}while(0)
template<typename T>void _do(T &&_x){cerr<<_x<<endl;}
template<typename T,typename ...S> void _do(T &&_x,S &&..._t){cerr<<_x<<", ";_do(_t...);}
template<typename _a,typename _b> ostream& operator << (ostream &_s,const pair<_a,_b> &_p) {return _s<<"("<<_p.X<<","<<_p.Y<<")";}
template<typename It> ostream& _OUTC(ostream &_s,It _ita,It _itb)
```

```cpp
{
  _s<<"{";
  for(It _it=_ita;_it!=_itb;_it++)
  {
    _s<<(_it==_ita?"":",")<<*_it;
  }
  _s<<"}";
  return _s;
}
template<typename _a> ostream &operator << (ostream &_s,vector<_a> &_c){return _OUTC(_s,ALL(_c));}
template<typename _a> ostream &operator << (ostream &_s,set<_a> &_c){return _OUTC(_s,ALL(_c));}
template<typename _a> ostream &operator << (ostream &_s,deque<_a> &_c){return _OUTC(_s,ALL(_c));}
template<typename _a,typename _b> ostream &operator << (ostream &_s,map<_a,_b> &_c){return _OUTC(_s,ALL(_c));}
template<typename _t> void pary(_t _a,_t _b){_OUTC(cerr,_a,_b);cerr<<endl;}
#define IOS()
class Timer {
private:
  string scope_name;
  chrono::high_resolution_clock::time_point start_time;
public:
  Timer (string name) : scope_name(name) {
    start_time = chrono::high_resolution_clock::now();
  }
  ~Timer () {
    auto stop_time = chrono::high_resolution_clock::now();
    auto length = chrono::duration_cast<chrono::microseconds>(stop_time - start_time).count();
    double mlength = double(length) * 0.001;
    debug(scope_name, mlength);
  }
};
#else
#define TIME(i)
#define debug( ... )
#define pary( ... )
#define endl '\n'
#define IOS() ios_base::sync_with_stdio(0);cin.tie(0)
#endif

const ll MOD = 1000000007;
const ll INF = 0x3f3f3f3f3f3f3f3f;
const int iNF = 0x3f3f3f3f;
const ll MAXN = 100005;

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());


/*
find solution of ax + by = gcd(a, b)

gcd(a, b) = gcd(b, a % b), therefore

bx + (a % b)y = gcd(b, a % b) = gcd(a, b) = ax + by
bx' + (a % b)y' = bx' + (a - floor(a/b) * b)y' = ay' + b(x' - floor(a/b)y') = ax + by

x = y', y = x' - floor(a/b) * y'
*/

void exgcd(int a, int b, int &x, int &y){
  if(b == 0){
    x = 1;y = 0;
    return;
  }
  int x1, y1;
  exgcd(b, a%b, x1, y1);
  x=y1;
  y=x1-(a/b)*y1;

}

// if x < 0, ans = (x + b) % b

/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();

  return 0;
}
```

25 計算幾何模板

```cpp
#include <bits/stdc++.h>
using namespace std;
#define X first
#define Y second
typedef pair<double, double> Pt;


Pt operator+(const Pt& p1, const Pt& p2){
  return Pt(p1.X + p2.X, p1.Y + p2.Y);
}
Pt operator-( const Pt& p1 , const Pt& p2 ){
  return Pt( p1.X - p2.X , p1.Y - p2.Y );
}
double operator*( const Pt& p1 , const Pt& p2 ){
  return p1.X * p2.X + p1.Y * p2.Y;
}

double operator^( const Pt& p1 , const Pt& p2 ){
  return p1.X * p2.Y - p1.Y * p2.X;
}

int main(){

}
```

26 Monotone Chain

```cpp
// :80 <enter>
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;
typedef pair<int, ll> pil;
typedef pair<ll, int> pli;
typedef pair<double,double> pdd;
#define SQ(i) ((i)*(i))
#define MEM(a, b) memset(a, (b), sizeof(a))
#define SZ(i) int(i.size())
#define FOR(i, j, k, in) for (int i=j ; i<(k) ; i+=in)
#define RFOR(i, j, k, in) for (int i=j ; i>=(k) ; i-=in)
#define REP(i, j) FOR(i, 0, j, 1)
#define REP1(i,j) FOR(i, 1, j+1, 1)
#define RREP(i, j) RFOR(i, j, 0, 1)
#define ALL(_a) _a.begin(),_a.end()
#define mp make_pair
#define pb push_back
#define eb emplace_back
#define X first
#define Y second
#ifdef jayinnn
#define TIME(i) Timer i(#i)
#define debug(...) do{\
    fprintf(stderr,"%s - %d (%s) = ",__PRETTY_FUNCTION__,__LINE__,#__VA_ARGS__)\
  _do(__VA_ARGS__);\
}while(0)
template<typename T>void _do(T &&_x){cerr<<_x<<endl;}
template<typename T,typename ...S> void _do(T &&_x,S &&..._t){cerr<<_x<<", ";_do(_t...);}
template<typename _a,typename _b> ostream& operator << (ostream &_s,const pair<_a,_b> &_p){return _s<<"("<<_p.X<<","<<_p.Y<<")";}
template<typename It> ostream& _OUTC(ostream &_s,It _ita,It _itb)
{
  _s<<"{";
  for(It _it=_ita;_it!=_itb;_it++)
  {
    _s<<(_it==_ita?"":",")<<*_it;
  }
  _s<<"}";
  return _s;
}
template<typename _a> ostream &operator << (ostream &_s,vector<_a> &_c){return _OUTC(_s,ALL(_c));}
template<typename _a> ostream &operator << (ostream &_s,set<_a> &_c){return _OUTC(_s,ALL(_c));}
template<typename _a> ostream &operator << (ostream &_s,deque<_a> &_c){return _OUTC(_s,ALL(_c));}
template<typename _a,typename _b> ostream &operator << (ostream &_s,map<_a,_b> &_c){return _OUTC(_s,ALL(_c));}
template<typename _t> void pary(_t _a,_t _b){_OUTC(cerr,_a,_b);cerr<<endl;}
#define IOS()
class Timer {
private:
  string scope_name;
  chrono::high_resolution_clock::time_point start_time;
public:
  Timer (string name) : scope_name(name) {
    start_time = chrono::high_resolution_clock::now();
  }
  ~Timer () {
    auto stop_time = chrono::high_resolution_clock::now();

      auto length = chrono::duration_cast<chrono::microseconds>(st
      - start_time).count();
    double mlength = double(length) * 0.001;
    debug(scope_name, mlength);
  }
};
#else
#define TIME(i)
#define debug(...)
#define pary(...)
#define endl '\n'
#define IOS() ios_base::sync_with_stdio(0);cin.tie(0)
#endif

const ll MOD = 1000000007;
const ll INF = 0x3f3f3f3f3f3f3f3f;
const int iNF = 0x3f3f3f3f;
const ll MAXN = 100005;

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

int cross(pii o, pii a, pii b){
  int u = a.X - o.X;
  int v = a.Y - o.Y;
  int s = b.X - o.X;
  int t = b.Y - o.Y;

  return u * t - s * v;
}

bool cmp(pii &a, pii &b){
  return (a.X < b.X) || (a.X == b.X && a.Y < b.Y);
}

```

```cpp
void solve(int n){
  vector<pii> a;
  set<pii> s;
  int x, y;
  for(int i=0;i<n;i++){
    cin >> x >> y;
    s.insert({x, y});
  }
  for(auto i: s){
    a.push_back(i);
  }
  n = a.size();
  if(n <= 2){
    cout << n << endl;
    for(auto i: a){
      cout << i.X << " " << i.Y << endl;
    }
    return;
  }
  sort(ALL(a), cmp);
  debug(a);
  debug(a);
  int idx = 0;
  vector<pii> ch(2*n);
  for(int i=0;i<n;i++){
    while(idx >= 2 && cross(ch[idx-2], ch[idx-1], a[i]) <= 0){
      idx --;
    }
    ch[idx++] = a[i];

  }
  for(int i=n-2,t=idx+1;i>=0;i--){
    while(idx >= t && cross(ch[idx-2], ch[idx-1], a[i]) <= 0){
      idx --;
    }
    ch[idx++] = a[i];
  }
  idx --;
  cout << idx << endl;
  for(int i=0;i<idx;i++){
    cout << ch[i].X << " " << ch[i].Y << endl;
  }

}



/********** Good Luck :) **********/
int main () {
  TIME(main);
  IOS();
  int n;

  while(cin >> n){
    if(n == 0) break;
    solve(n);
  }

  return 0;
}

```

27 旋轉卡尺

```cpp
#include <iostream>
#include <cmath>
#include <algorithm>
#include <vector>
#include <map>
#include <stack>
#include <set>
#include <queue>
#include <list>
#include <string.h>
#include <complex>
#include <sstream>
using namespace std;
#define INITIO() ios_base::sync_with_stdio(false);cin.tie(NULL);
#define FILE() freopen("a.in","r",stdin);freopen("out.txt","w",stdout);
#define endl   '\n'
#define F     first
#define S     second
#define pb     push_back
#define pf     push_front
#define all(a)   a.begin(),a.end()
#define rall(a)  a.rbegin(),a.rend()
#define sz(a)   (int)a.size()
#define FOR(i,a,b) for(int i = (a); i < (b); i++)
using vii = vector<vector<int>>;
using vi = vector<int>;
using mii = map<int,int>;
using cd = complex<double>;
typedef long long ll;
typedef unsigned long long ull;
typedef pair<int,int> pii;
typedef pair<long,long> pll;
const double PI  = acos(-1);
const ll Mod   = 1e9+7;
//const ll Inf  = 2e18+9;
//const int N   = 1e5+9;
//const int dx[4] = {0, 1, 0, -1};
//const int dy[4] = {1, 0, -1, 0};
//const int dx8[8] = {-1, -1, -1, 0, 1, 1, 1, 0};
//const int dy8[8] = {-1, 0, 1, 1, 1, 0, -1, -1};
ll gcd(ll a, ll b){if( b==0 )return a;return gcd( b, a%b );}
ll lcm(ll a,ll b){return a * b / gcd(a, b);}
int mul(int a, int b) {return (1LL * a * b) % Mod;}
ll add(ll a, ll b) {a += b;if (a >= Mod) a -= Mod;if (a < 0) a += Mod;return a;}
ll sub(ll a, ll b) {return (a - b) % Mod + ((a >= b) ? 0 : Mod);}
// --------------------------------------------------------

ll cross(pll a,pll b,pll c){
  return (a.F-b.F)*(a.S-c.S) - (a.S-b.S)*(a.F-c.F);
}

void solve(){
  int n;
  scanf("%d", &n);
  vector<pll> a;
  set<pii> s;
  FOR(i,0,n){
   int x,y;
   scanf("%d %d", &x, &y);
   if(s.count({x,y}))continue;
   a.pb({x,y});
   s.insert({x,y});
  }
  n=sz(a);
  sort(all(a));
  vector<pll> ans;
  FOR(i,0,n){
   while(sz(ans)>=2 &&
    ↪  cross(ans[sz(ans)-1], ans[sz(ans)-2],
    ↪  a[i]) <= 0)ans.pop_back();
   ans.pb(a[i]);
  }
  for(int i=n-2, t=sz(ans)+1;i>=0;i--){
   while(sz(ans) >= t &&
    ↪  cross(ans[sz(ans)-1], ans[sz(ans)-2],
    ↪  a[i]) <= 0)ans.pop_back();
   ans.pb(a[i]);
  }
  if(sz(ans)>1)ans.pop_back();
  a = ans;
  n=sz(a);
  ll best=0;
  //FOR(i,0,n)cout<<a[i].F<<'
   ↪  '<<a[i].S<<endl;
  for(int i=0;i<n-2;i++){
   int k=i+1;
   for(int j=i+2;j<n;j++){
    ll area = abs(cross(a[i],a[k],a[j]));
    while(1){
     k++;
     ll newarea = abs(cross(a[i],a[k],a[j]));
     if(newarea <= area || k >= j)break;
     area = newarea;
    }
    k = max(i+1, k-1);
    best = max(area, best);
   }
  }
  printf("%.5f\n",0.5*best);
 }

 int main(){
  INITIO()
  //int t;cin>>t;
  //while(t--){
   solve();
  //}
  return 0;
 }
```

28 Optimal Triangulation

d(i, j) = max(d(i, k) + d(k, j) + w(i,j,k)|i<k<j) ,w(i,j,k) is the weight function

```
int n;
vector<pii> a;
double area(int i,int j,int k){
  return abs((a[j].F
   ↪  - a[i].F)*(a[k].S - a[i].S)*1.0 - (a[j].S
   ↪  - a[i].S)*(a[k].F - a[i].F)*1.0)/2.0;
}

bool judge(int a,int b,int c){
  for(int i=0;i<n;i++){
    if(i==a || i==b || i==c)continue;
    double s = area(a,b,i) +
     ↪  area(a,c,i) + area(b,c,i) - area(a,b,c);
    if(fabs(s) < 0.01)return false;
  }
  return true;
}

double d[55][55];

double dp(int i,int j){
  if(i+1 >= j)return d[i][j]=0;
  if(d[i][j] != -1.0)return d[i][j];
  double ans=1e9*1.0;
  for(int k=i+1;k<j;k++){
    if(judge(i,j,k))ans
     ↪  = min(ans, max(dp(i,k),max(dp(k,j),
     ↪  area(i,k,j))));
  }
  return d[i][j] = ans;
}

int main(){
  ios_base::sync_with_stdio(false);cin
   ↪  .tie(NULL);
  //freopen("a.in","r",stdin);freopen("out
   ↪  .txt","w",stdout);
  int t;cin>>t;
  while(t--){
    cin>>n;
    a.resize(n);FOR(i,0,n)cin>>a[i]
     ↪  .F>>a[i].S;
    FOR(i,0,n)FOR(j,0,n)d[i][j] = -1.0;
    printf("%.1f\n", dp(0,n-1));

  }
  return 0;
}
```

29 Catalan Number

When counting the way of triangulation of a convex hull. f(n) = f(n-1)*f(2) + f(3)*f(n-2) + ⋯+ f(n-1)*f(2)

30 Optimal Matrix Chain Multiplication

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

int p[105];

int dp(int i, int j){
  if(j <= i+1) return 0;
  int mn = 10000000;
  for(int k=i+1;k<j;k++){
    int cost = dp(i,k) + dp(k,j) + p[i]*p[k]*p[j];
    mn = min(cost, mn);
  }
  return mn;
}

int main(){
  int n;
  cin>>n;
  for(int i=0;i<n;i++)cin>>p[i];
  cout<<dp(0,n-1)<<endl;
  return 0;
}
```

31 Check if point belongs to the convex polygon (logN for each query)

```
struct pt {
  long long x, y;
  pt() {}
  pt(long long _x, long long _y) : x(_x), y(_y) {}
  pt operator+(const
  ↪   pt &p) const { return pt(x + p.x, y + p.y); }
  pt operator-(const
  ↪   pt &p) const { return pt(x - p.x, y - p.y); }
  long long cross(const
  ↪   pt &p) const { return x * p.y - y * p.x; }
  long long dot(const
  ↪   pt &p) const { return x * p.x + y * p.y; }
  long long cross(const pt &a, const pt &b) const
  ↪   { return (a - *this).cross(b - *this); }
  long long dot(const pt &a, const pt &b)
  ↪   const { return (a - *this).dot(b - *this); }
  long long sqrLen()
  ↪   const { return this->dot(*this); }
};

bool lexComp(const pt &l, const pt &r) {
  return l.x < r.x || (l.x == r.x && l.y < r.y);
}

int sgn(long long
 ↪  val) { return val > 0 ? 1 : (val == 0 ? 0 : -1); }

vector<pt> seq;
pt translation;
int n;

bool pointInTriangle(pt a, pt b, pt c, pt point) {
  long long s1 = abs(a.cross(b, c));
  long long s2 = abs(point.cross(a,
  ↪   b)) + abs(point.cross(b,
  ↪   c)) + abs(point.cross(c, a));
  return s1 == s2;
}

void prepare(vector<pt> &points) {
  n = points.size();
  int pos = 0;
  for (int i = 1; i < n; i++) {
    if (lexComp(points[i], points[pos]))
      pos = i;
  }
  rotate(points.begin(),
  ↪   points.begin() + pos, points.end());

  n--;
  seq.resize(n);
  for (int i = 0; i < n; i++)
    seq[i] = points[i + 1] - points[0];
  translation = points[0];
}

bool pointInConvexPolygon(pt point) {
  point = point - translation;
  if (seq[0].cross(point) != 1 &&
      sgn(seq[0].cross(point))
       ↪   != sgn(seq[0].cross(seq[n - 1])))
    return false;
  if (seq[n - 1].cross(point) != 0 &&
      sgn(seq[n - 1].cross(point))
       ↪   != sgn(seq[n - 1].cross(seq[0])))
    return false;

  if (seq[0].cross(point) == 0)
    return seq[0].sqrLen() >= point.sqrLen();

  int l = 0, r = n - 1;
  while (r - l > 1) {
    int mid = (l + r) / 2;
    int pos = mid;
    if (seq[pos].cross(point) >= 0)
      l = mid;
    else
      r = mid;
  }
  int pos = l;
  return pointInTriangle(seq[pos],
   ↪   seq[pos + 1], pt(0, 0), point);
}
```

32 Bungee Builder (monotonic stack)

```
void solve(){
  int n;cin>>n;
  vector<int> a(n);FOR(i,0,n)cin>>a[i];
  stack<pii> s;
  int ans=0;
  FOR(i,0,n){
    int bot=a[i];
    while(sz(s)){
      pii x = s.top();s.pop();
      bot = min(bot, x.S);
      if(x.F > a[i]){
        s.push({x.F,bot});
        ans=max(ans,a[i]-bot);
        break;
      }
      ans=max(ans,x.first-bot);
    }
    s.push({a[i],a[i]});
  }
  cout<<ans<<endl;
}
```

33 Arachnophobia (dijkstra + binary search)

```
int n,m,t,st,ed,k;
struct Edge{
 int v;ll w;
 bool operator<(const Edge &cmp) const {
  return cmp.w < w;
 }
};

vector<Edge> g[100005];
ll dis[100005],spiderdis[100005];

bool check(ll val){
 if(spiderdis[st] < val)return 0;
 if(spiderdis[ed] < val)return 0;
 FOR(i,0,n)dis[i]=Inf;
 priority_queue<Edge> pq;
 pq.push({st,0});
 while(pq.size()){
  auto node = pq.top(); pq.pop();
  if(dis[node.v] <= node.w)continue;
  dis[node.v] = node.w;
  for(auto it:g[node.v]){
   if(spiderdis[it.v]
   ↪ >= val && dis[it.v] > it.w + node.w){
    pq.push({it.v, it.w + node.w});
   }
  }
 }
 //FOR(i,0,n)cout<<dis[i]<<endl;
 return dis[ed] <= t;
}

void solve(){
 cin>>n>>m>>t;
 while(m--){
  int u,v,w;
  cin>>u>>v>>w;
  g[u].pb({v,w});
  g[v].pb({u,w});
 }
 cin>>st>>ed;
 cin>>k;
 priority_queue<Edge> pq;
 while(k--){
  int x;cin>>x;
  pq.push({x,0});
 }
 FOR(i,0,n)spiderdis[i]=Inf;
 while(pq.size()){
  auto node = pq.top(); pq.pop();
  if(spiderdis[node.v] <= node.w)continue;
  spiderdis[node.v] = node.w;
  for(auto it:g[node.v]){
   if(spiderdis[it.v] > it.w + node.w){
    pq.push({it.v,it.w + node.w});
   }
  }
 }
 ll ans=0,step=1;
 FOR(i,0,62)step*=2;
 while(step > 0){
  if(check(ans+step)) ans += step;
  step=(step>>1);
 }
 cout<<ans<<endl;
}
```

34 Ascending Photo (Do some transition and DP)

```
void solve(){
 int n,cnt=0;cin>>n;
 vector<int> h;
 set<int> st;
 map<int,int> mp;
 FOR(i,0,n){
  int x;cin>>x;
  st.insert(x);
  if(h.empty() || h.back()!=x)h.pb(x);
 }
 for(auto it:st){
  if(!mp.count(it))mp[it] = cnt++;
 }
 n=sz(h);
 vector<vector<int>> pos(cnt);
 FOR(i,0,n){
  h[i] = mp[h[i]];
  pos[h[i]].pb(i);
 }
 pii best[2] = {{0,n}, {0,n}};
 FOR(i,0,cnt-1){
  pii nbest[2] = {best[0], best[1]};
  FOR(j,0,sz(pos[i])){
   int p = pos[i][j];
   if(p == n-1 || h[p]+1!=h[p+1])continue;
   pii s(0,n);
   if(p != best[0].second) s = best[0];
   else s = best[1];
   s.first++;
   s.second = p+1;
   if(pos[i+1].size() == 1) s.second = n;
   if(s > nbest[0]){
    nbest[1] = nbest[0];
    nbest[0] = s;
   }
   else if(s > nbest[1]) nbest[1] = s;
  }
  best[0] = nbest[0];
  best[1] = nbest[1];
 }
 cout<<n-1-best[0].first<<endl;
}
```

35 Pokemongogo (TSP)

```
int n,ans,cnt=0 , g[22][22]={}, dp[1<<22][22];
map<string,int> mp;
vector<int> pok[22];
vector<pii> p, v;

int dfs(int i,int j){
 if(dp[i][j] != -1)return dp[i][j];
 if(i ==
 ↪ (1 << (n+1))-1 && j==0)return dp[i][j]=0;
 int res = 1e9+5;
 for(int k=0;k<=n;k++){
  if(!((i >> k) & 1)){
```

```cpp
      res=min(res, dfs((i|(1<<k)), k)+g[j][k]);
    }
  }
  /*
  int b=i;
  while(b){
    cout<<(b&1);
    b/=2;
  }
  cout<<' '<<j<<' ';
  cout<<res<<endl;
  */

  return dp[i][j]=res;
}

void f(vector<int> a, int id){
  if(a.size()==n+1){

    for(int i=0;i<=n;i++){
      if(a[i])continue;
      a[i]=1;

      int cn=0;
      for(int i=0, k=1;i<sz(a);i++, k*=2)
        cn+=a[i]*k;
      if(dp[cn][i]!=-1){
        ans=min(ans, dp[cn][i]+g[0][i]);
        //for(int i=0;i<sz(a);i++)cout<<a[i];
        //cout<<' '<<i<<' ';
        //cout<<dp[cn][i]<<' ';
        //cout<<endl;
      }
      a[i]=0;
    }
    //cout<<endl;

  }
  int k=sz(a);
  FOR(i,0,sz(pok[id]))a.pb(1);
  for(int i=0;i<pok[id].size();i++){
    a[i+k]=0;
    f(a, id+1);
    a[i+k]=1;
  }
}

void solve(){
  cin>>n;
  p.resize(n+1);
  v.resize(n);
  FOR(i,0,n){
    int r,c;cin>>r>>c;
    string s;cin>>s;
    if(mp.count(s)==0){
      mp[s]=cnt++;
    }
    pok[mp[s]].pb(i);
    v[i]={r,c};
  }
  p[0]={0,0};
  for(int i=0,k=1;i<cnt;i++){
    for(int j=0;j<pok[i].size();j++){
      p[k++]=v[pok[i][j]];
    }
  }
  FOR(i,0,n+1){
    FOR(j,0,n+1){
      g[i][j]=abs(p[i].F-p[j].F)+abs(p[i].S-p[j].S);
      //cout<<g[i][j]<<' ';
    }
    //cout<<endl;
  }
  memset(dp, -1, sizeof(dp));
  dfs(0,0);
  ans=1e9+5;
  f({0}, 0);
  cout<<ans<<endl;
}
```

36 British Menu (Dp + SCC)

```cpp
#include <cmath>
#include <cstdio>
#include <cstring>
#include <iostream>
#include <algorithm>
#include <vector>
#define MAX_V 200005
using namespace std;
int n,ans,ans1[MAX_V],ans2[MAX_V],c[MAX_V][6],cnum[MAX_V],id[MAX_V],dis[MAX_V][6],m;
vector<int> G[MAX_V];
vector<int> rG[MAX_V];
vector<int> vs;
bool used[MAX_V];
int cmp[MAX_V];
void add_edge(int from,int to)
{
  G[from].push_back(to);
  rG[to].push_back(from);
}
void dfs(int v)
{
  used[v]=1;
  for(int i=0;i<G[v].size();++i)
  {
    if(!used[G[v][i]])
      dfs(G[v][i]);
  }
  vs.push_back(v);
}
void rdfs(int v,int k)
{
  used[v]=1;
  cmp[v]=k;
  for(int i=0;i<rG[v].size();++i)
  {
    if(!used[rG[v][i]])
    rdfs(rG[v][i],k);
  }
}
int scc()
{
  memset(used,0,sizeof used);
  vs.clear();
  for(int v=1;v<=n;v++)
  {
   if(!used[v]) dfs(v);
  }
  memset(used,0,sizeof used);
  int k=1;
  for(int i=vs.size()-1;i>=0;--i)
  {
   if(!used[vs[i]])
   rdfs(vs[i],k++);
  }
  return k;
 }
void cal(int now,int nowid,int f,int nowdis)
{
  used[now]=1;
  dis[now][nowid]=max(dis[now][nowid],nowdis);
  for(int i=0;i<G[now].size();++i)
  if(cmp[G[now][i]]==f&&!used[G[now][i]])
  {
   cal(G[now][i],nowid,f,nowdis+1);
  }
  used[now]=0;
  return;
}
int get2(int now);
int get1(int now)
{
  if(ans1[now]!=-1) return ans1[now];
  int nowans=1;
  for(int i=0;i<rG[now].size();i++)
  if(cmp[rG[now][i]]!=cmp[now])
  {
   nowans=max(nowans,get2(rG[now][i])+1);
  }
  return ans1[now]=nowans;
}
int get2(int now)
{
  if(ans2[now]!=-1) return ans2[now];
  int nowans=-1;
  for(int i=1;i<=cnum[cmp[now]];++i)
  {
   nowans=max(nowans,get1(c[cmp[now]][i])+dis[now][i]);
  }
  return ans2[now]=nowans;
}
int main()
{
  scanf("%d%d",&n,&m);
  int u,v;
  for(int i=1;i<=m;++i)
  {
   scanf("%d%d",&u,&v);
   add_edge(u,v);
  }
  scc();
  memset(used,0,sizeof used);
  for(int i=1;i<=n;++i)
  {
   ans1[i]=ans2[i]=-1;
   c[cmp[i]][++cnum[cmp[i]]]=i;
   id[i]=cnum[cmp[i]];
   cal(i,id[i],cmp[i],0);
  }
  for(int i=1;i<=n;++i)
  {
   ans=max(ans,get2(i));
  }
  cout<<ans;
  return 0;
}

```

37 Flight (Tree diameter/radius)

```cpp
int n,vis[2505],dis[2505],siz[2505],h1[2505],h2[2505],c1[2505],c2[2505],p[2505];
vector<int> edge[2505];

pii treedia(int x){
  for(int i=1;i<=n;i++){
    vis[i]=0;
    dis[i]=0;
  }
  queue<int> q;
  q.push(x);
  while(q.size()){
    int top = q.front();
    q.pop();
    vis[top]=1;
    for(auto it:edge[top]){
      if(!vis[it]){
        dis[it] = dis[top]+1;
        q.push(it);
      }
    }
  }
  for(int i=1;i<=n;i++){
    if(dis[x] < dis[i])x = i;
  }
  for(int i=1;i<=n;i++){
    vis[i]=0;
    dis[i]=0;
  }
  q.push(x);
  while(q.size()){
    int top = q.front();
    q.pop();
    vis[top]=1;
    for(auto it:edge[top]){
      if(!vis[it]){
        dis[it] = dis[top]+1;
        q.push(it);
      }
    }
  }
  int id=0;
  for(int i=1;i<=n;i++){
    if(dis[id] < dis[i])id=i;
  }
  int ans=0;
  for(int i=1;i<=n;i++){
    if(dis[i] == dis[id]/2 + (dis[id]%2?1:0))ans = i;
  }
  if(dis[id]==0)ans=x;
  return {ans, id};
}

void record(int x, int height, int child)
{
  if (height>h1[x])
  {
    h2[x]=h1[x]; c2[x]=c1[x];
    h1[x]=height; c1[x]=child;
  }
  else if (height>h2[x])
  {
    h2[x]=height; c2[x]=child;
  }
}

void dfs1(int x){
  h1[x]=h2[x]=0;
  for(auto it:edge[x]){
    if(p[x]!=it){
      p[it]=x;
      dfs1(it);
      record(x,h1[it]+1,it);
    }
  }
}

void dfs2(int x){
  if(p[x]!=x){
    int y=p[x];
    if(c1[y]==x)record(x,h2[y]+1,y);
    else record(x,h1[y]+1,y);
  }
  for(auto it:edge[x]){
    if(it!=p[x]){
      dfs2(it);
    }
  }
}

int treecentroid(int x){
  int ans=x;
  FOR(i,1,n+1)
    h1[i]=h2[i]=c1[i]=c2[i]=p[i]=0;
  p[x]=x;
  dfs1(x);
  dfs2(x);
  for(int i=1;i<=n;i++){
    //cout<<h1[i]<<' ';
    if(h1[i] && h1[i]<h1[ans]){
      ans=i;
    }
  }
  //cout<<endl;
  return ans;
}

void solve(){
  cin>>n;
  vector<pii> ed;
  FOR(i,0,n-1){
    int u,v;cin>>u>>v;
    edge[u].pb(v);
    edge[v].pb(u);
    ed.pb({u,v});
  }
  int ans=1e9+5, a[4]={};
  for(auto [u,v]:ed){
    for(auto it=edge[u].begin();it!=edge[u].end();it++){
      if(*it==v){
        edge[u].erase(it);
        break;
      }
    }
    for(auto it=edge[v].begin();it!=edge[v].end();it++){
      if(*it==u){
        edge[v].erase(it);
        break;
      }
    }
    int k=treecentroid(u);
    int j=treecentroid(v);
    //cout<<k<<' '<<j<<endl;
    edge[k].pb(j);
    edge[j].pb(k);
    int y=treedia(u).second;
    if(ans>dis[y]){
      ans=dis[y];
      a[0]=u;
      a[1]=v;
      a[2]=k;
      a[3]=j;
    }
    for(auto it=edge[k].begin();it!=edge[k].end();it++){
      if(*it==j){
        edge[k].erase(it);
        break;
      }
    }
    for(auto it=edge[j].begin();it!=edge[j].end();it++){
      if(*it==k){
        edge[j].erase(it);
        break;
      }
    }
    edge[u].pb(v);
    edge[v].pb(u);
  }
  cout<<ans<<endl;
  cout<<a[0]<<' '<<a[1]<<endl;
  cout<<a[2]<<' '<<a[3]<<endl;
}
```

38 Rooted Subtree (Combinatorics + LCA)

```cpp
vector<int> e[N];
int n,m,dep[N]={},siz[N]={},p[20][N]={};

void dfs(int x){
  siz[x]=1;
  for(auto it:e[x]){
    if(p[0][x]!=it){
      p[0][it]=x;
      dep[it]=dep[x]+1;
    dfs(it);
    siz[x] += siz[it];
  }
 }
}

int lca(int a,int b){
 if(dep[a] > dep[b])swap(a,b);
 int u=a,v=b;
 if(dep[a] != dep[b]){
  int dif = dep[b] - dep[a];
  for(int i=0;i<20;i++){
   if(dif&1)b = p[i][b];
   dif>>=1;
  }
 }
 if(a==b)return dep[v]-dep[u];
 for(int i=19;i>=0;i--){
  if(p[i][a] != p[i][b]){
   a = p[i][a];
   b = p[i][b];
  }
 }
 return dep[u]+dep[v]-2*dep[p[0][a]];
}

void solve(){
 cin>>n>>m;
 FOR(i,0,n-1){
  int u,v;
  cin>>u>>v;
  e[u].pb(v);
  e[v].pb(u);
 }
 p[0][1]=1;
 dfs(1);
 for(int i=1;i<20;i++){
  for(int j=1;j<=n;j++){
   p[i][j] = p[i-1][p[i-1][j]];
  }
 }
 while(m--){
  int u,v;cin>>u>>v;
  ll x = 1ll*lca(u,v)+1;
  cout<<x*(x+1)/2 + n - x<<endl;
 }
}
```

39 Stogovi (LCA + DSU)

```
vector<int> e[N];
int n,m,dep[N]={},siz[N]={},p[20]
 ↪  [N]={},f[N]={};

void dfs(int x){
 siz[x] = 1;
 for(auto it:e[x]){
  dep[it] = dep[x] + 1;
  dfs(it);
  siz[x] += siz[it];
 }
}

int lca(int a,int b){
 if(dep[a] > dep[b])swap(a,b);
 if(dep[a] != dep[b]){
  int dif = dep[b] - dep[a];
  for(int i=0;i<20;i++){
   if(dif&1)b = p[i][b];
   dif>>=1;
  }
 }
 if(a==b)return a;
 for(int i=19;i>=0;i--){
  if(p[i][a] != p[i][b]){
   a = p[i][a];
   b = p[i][b];
  }
 }
 return p[0][a];
}

void prelca(){
 dfs(0);
 for(int i=1;i<20;i++){
  for(int j=1;j<=n;j++){
   p[i][j] = p[i-1][p[i-1][j]];
  }
 }
}

int find(int x){
 if(x == f[x])return x;
 return f[x] = find(f[x]);
}

void solve(){
 cin>>n;
 FOR(i,1,n+1)f[i]=i;
 vector<pii> ans;
 FOR(i,1,n+1){
  char c;
  cin>>c;
  int v,w;
  if(c=='a'){
   cin>>v;
   v = find(v);
   p[0][i] = v;
   e[v].pb(i);
  }
  else if(c=='b'){
   cin>>v;
   v = find(v);
   f[i] = p[0][v];
   ans.pb({v,-1});
  }
  else {
   cin>>v>>w;
   v = find(v);
   w = find(w);
   f[i] = v;
   ans.pb({v,w});
  }
 }
 prelca();
 for(auto [v,w]:ans){
  if(w == -1){
   cout<<v<<endl;
  }
  else{
   int x = lca(v,w);
   cout<<dep[x]<<endl;
  }
 }
}
```

40 LCA

```
// :80 <enter>
#include <bits/stdc++.h>
using namespace std;

vector<vector<int>> adj;
vector<vector<int>> p;
vector<int> d;

void dfs(int cur, int par, int dep){
 d[cur] = dep;
 p[cur][0] = par;
 for(auto i: adj[cur]){
  if(i != par){
   dfs(i, cur, dep+1);
  }
 }
}

int lca(int x, int y){
 if(d[x] > d[y]) swap(x, y);
 if(d[x] != d[y]){
  int diff = d[y] - d[x];
  for(int i=0;i<20;i++){
   if(diff & 1) y = p[y][i];
   diff >>= 1;
  }
 }
 if(x == y) return x;
 for(int i=19;i>=0;i--){
  if(p[x][i] != p[y][i]){
   x = p[x][i];
   y = p[y][i];
  }
 }

 return p[x][0];
}

void solve(){
 int n;
 cin >> n;
 adj.resize(n+1);
 p.resize(n+1, vector<int>(20, 0));
 d.resize(n+1);
 int x, y;
 for(int i=0;i<n-1;i++){
  cin >> x >> y;
  adj[x].pb(y);
  adj[y].pb(x);
 }
 dfs(1, 0, 1);
 for(int i=1;i<20;i++){
  for(int j=1;j<=n;j++){
   p[j][i] = p[p[j][i-1]][i-1];
  }
 }
 ll ans = 0;
 debug(d);
 for(int i=1;i<=n;i++){
  for(int j=2*i;j<=n;j+=i){
   debug(i, j, lca(i, j));
   ll dis = d[i] - 2 * d[lca(i, j)] + d[j] + 1;
   debug(dis);
   ans += dis;
  }
 }
 cout << ans << endl;
}
```

41 Edmonds-Karp

Time complexity: $O(E^2V)$

```
// :80 <enter>
#include <bits/stdc++.h>
using namespace std;

class Edge{
public:
 int to, cap, rev;
 Edge(int _to, int _cap, int
  ↪  _rev): to(_to), cap(_cap), rev(_rev) {}
};

vector<vector<Edge>> G(MAXN);
vector<bool> vis(MAXN, false);


void add_edge(int u, int v, int cap){
 G[u].push_back(Edge(v, cap, G[v].size()));
 G[v].push_back(Edge(u, 0, G[u].size()-1));
}

int dfs(int cur, int t, int f){
 if(cur == t){
  return f;
 }
 vis[cur] = true;
 int cur_sz = G[cur].size();
 for(int i=0;i<cur_sz;i++){
  Edge& e = G[cur][i];
  if(vis[e.to] == false && e.cap > 0){
   int d = dfs(e.to, t, min(f, e.cap));
   if(d > 0){
    e.cap -= d;
    G[e.to][e.rev].cap += d;
    return d;
   }
  }
 }
 return 0;
}

void solve(){
 int n, m, k;
 cin >> n >> m >> k;
 /**
  * node 0: s
  * node 5000: t
```

```cpp
   *node 1: tmp
   *node 1001 ~ 1999: hero
   *node 2001 ~ 2999: monster
   *s -> tmp: n+k
   *tmp -> heros: 2
   *heros -> monster: 1
   *monster -> t: 1
   */
  int q;
  for(int i=1;i<=n;i++){
    cin >> q;
    int tmp;
    while(q--){
      cin >> tmp;
      add_edge(1000+i, 2000+tmp, 1);
    }
  }
  add_edge(0, 1, n+k);
  for(int i=1;i<=n;i++){
    add_edge(1, 1000+i, 2);
  }
  for(int i=1;i<=m;i++){
    add_edge(2000+i, 5000, 1);
  }
  int ans = 0;
  while(1){
    fill(ALL(vis), false);
    int f = dfs(0, 5000, iNF);
    if(f == 0){
      cout << ans << endl;
      return;
    }
    ans += f;
  }

}
```

42 Dinic

Time complexity: $O(V^2E)$

```cpp
// :80 <enter>
#include <bits/stdc++.h>
using namespace std;

int n, m;

class Edge{
public:
  int to, rev, flow, cap;
  Edge(int _to, int _rev, int _flow, int _cap): to(_to), rev(_rev), flow(_flow), cap(_cap) {}
};

class Graph{
public:
  int v;
  vector<int> level;
  vector<vector<Edge>> adj;
  vector<bool> vis;
  Graph(int _v){
    v = _v;
    adj = vector<vector<Edge>>(v);
    level = vector<int>(v);
    vis = vector<bool>(v, false);
  }

  void addEdge(int x, int y, int cap){
    Edge forward(y, adj[y].size(), 0, cap);
    Edge backward(x, adj[x].size(), 0, 0);
    adj[x].push_back(forward);
    adj[y].push_back(backward);
  }

  bool bfs(int s, int t){
    fill(level.begin(), level.end(), -1);
    level[s] = 0;
    queue<int> q;
    q.push(s);
    while(!q.empty()){
      int cur = q.front();
      q.pop();
      for(auto &i: adj[cur]){
        if(level[i.to] == -1 && i.flow < i.cap){
          level[i.to] = level[cur] + 1;
          q.push(i.to);
        }
      }
    }

    return level[t] != -1;
  }

  int sendFlow(int u, int flow, int t, vector<int>& idx){
    if(u == t) return flow;
    for(; idx[u]<(int)adj[u].size();idx[u]++){
      Edge& e = adj[u][idx[u]];

      if(level[e.to] == level[u]+1 && e.flow < e.cap){
        int cur_flow = min(flow, e.cap - e.flow);
        int tmp_flow = sendFlow(e.to, cur_flow, t, idx);

        if(tmp_flow > 0){
          e.flow += tmp_flow;
          adj[e.to][e.rev].flow -= tmp_flow;
          return tmp_flow;
        }
      }
    }

    return 0;
  }

  int dinic(int s, int t){
    for(int i=0;i<v;i++){
      for(Edge& j: adj[i]){
        j.flow = 0;
      }
    }
    if(s == t) return -1;
    int total = 0;
    while(bfs(s, t)){
      vector<int> idx(v, 0);
      while(int flow = sendFlow(s, iNF, t, idx)){
        total += flow;
    }
  }

  return total;
 }
};

void solve(){
 cin >> n >> m;
 Graph g(2*n+1);
 int u, v;
 // for(int i=1;i<=n;i++){
 //  g.addEdge(i, n+i, 1);
 //  g.addEdge(n+i, i, 1);
 // }
 for(int i=0;i<m;i++){
  cin >> u >> v;
  // g.addEdge(n+u, v, 1);
  g.addEdge(v, u, 1);
  g.addEdge(u, v, 1);
 }
 int ans = iNF;
 for(int i=1;i<=n;i++){
  for(int j=1;j<=n;j++){
   if(i != j)
    ans = min(ans, g.dinic(i, j));
  }
 }

 cout << ans << endl;
}

/********** Good Luck :) **********/
int main () {
 TIME(main);
 IOS();
 int t = 1;
 // cin >> t;
 while(t--){
  solve();
 }

 return 0;
}
```

43 KMP

```cpp
int F[MAXN];
vector < int > match(auto A, auto B) {
 const int Asz = A.size(), Bsz = B.size();
 vector < int > ans {};
 F[0] = -1, F[1] = 0;
 for (int i = 1, j = 0; i < Bsz; F[++i] = ++j) {
  if (B[i] == B[j]) F[i] = F[j]; // optimize
  while (j != -1 and B[i] != B[j]) j = F[j];
 }
 for (int i = 0, j = 0; i < Asz; ++i) {
  while (j != -1 and A[i] != B[j]) j = F[j];
  if (++j
   ↪ == Bsz) ans.emplace_back(i - j), j = F[j];
 }
 return ans;
}
```

44 Bipartie Matching

```cpp
array < int, SZ > mp;
array < bool, SZ > vis;
bool dfs(int now) {
 if (vis[now]) return false;
 vis[now] = true;
 for (int i = 0; i < n; i++) {
  if (!G[now][i]) continue;
  if (mp[i] == -1 or dfs(mp[i]))
   return mp[i] = now, true;
 }
 return false;
}
int solve() {
 mp.fill(-1);
 int r = 0;
 for (int i = 0; i < n; i++) {
  vis.fill(false);
  if (dfs(i)) r++;
 }
 return r;
}
```

45 Josephus

```cpp
int josephus(int n, int k) {
 int res = 0;
 for (int i = 1; i <= n; ++i) res = (res + k) % i;
 return res;
}
```

46 Undo Disjoint Set

```cpp
struct DisjointSet
{
 ^^I// save() is like recursive
 ^^I// undo() is like return
 int n, fa[MXN], sz[MXN];
 vector<pair<int*, int>> h;
 vector<int> sp;
 void init(int tn) {
  n = tn;
  for (int i = 0; i < n; i++) sz[fa[i] = i] = 1;
  sp.clear();
  h.clear();
 }
 void assign(int *k, int v) {
  h.PB({k, *k});
  *k = v;
 }
 void save() {
  sp.PB(SZ(h));
 }
 void undo() {
  assert(!sp.empty());
  int last = sp.back();
  sp.pop_back();
  while (SZ(h) != last)
  {
   auto x = h.back();
   h.pop_back();
   *x.F = x.S;
```

```
    }
  }

  int f(int x) {
    while (fa[x] != x) x = fa[x];
    return x;
  }

  void uni(int x, int y) {
    x = f(x);
    y = f(y);
    if (x == y) return;
    if (sz[x] < sz[y]) swap(x, y);
    assign(&sz[x], sz[x] + sz[y]);
    assign(&fa[y], x);
  }
} djs;
```

47 Fast GCD

```
ll fast_gcd(ll x, ll y)
{
  ll g = 1;
  while (x && y)
  {
    const int c = __builtin_ctzll(x | y);
    g <<= c;
    x >>= c;
    y >>= c;
    x >>= __builtin_ctzll(x);
    y >>= __builtin_ctzll(y);
    if (x < y) swap(x, y);
    x -= y;
  }

  return g * (x + y);
}
```

48 CRT

```
ll solve_crt(ll x1, ll m1, ll x2, ll m2){
  ll g = __gcd(m1, m2);
  if ((x2 - x1) % g) return -1;^^I// no sol
  m1 /= g;
  m2 /= g;
  auto [pf, ps] = extgcd(m1, m2);
  ll lcm = m1 / g * m2;
  ll res = pf * (x2 - x1) * m1 + x1;
  return (res % lcm + lcm) % lcm;
}
```

49 Binary Search

```
int binary_search(const vector<int> &data, int key) {
  int low = 0;
  int high = data.size() - 1;
  while (low <= high) {
    int mid = int((low + high) / 2);
    if (key == data[mid])
      return mid;
    else if (key > data[mid])
      low = mid + 1;
    else
      high = mid - 1;
  }
  return -1;
}
```