Contents

https://vizle.offnote.co

Contact us: **vizle@offnote.co**

This document was generated automatically by **Vizle**

Your **Personal Video Reader Assistant**

Learn from Videos **Faster** and **Smarter**

VIZLE **PRO / BIZ**

PDF, PPT ~~Watermarks~~

- Convert *entire* videos
- *Customize* to retain all essential content
- Include Spoken *Transcripts*
- Customer support

Visit **https://vizle.offnote.co/pricing** to learn more

VIZLE **FREE PLAN**

PDF only Watermarks

- Convert videos *partially*
- Slides may be *skipped**
- Usage restrictions
- No Customer support

Visit **https://vizle.offnote.co** to try free

Login to Vizle to unlock more slides*

```
    while (x <= n) {
        sum[x] += val;
        x += (x&(-x)); //di chuyen toi nut cha
    }
}
int sum_t(int x) {
    int t = 0;
    while (x>0) {
        t += sum[x];
        x &= (x-1); //di chuyen toi nut ke
        //x-=(x&(-x));
    }
    return t;
}
```

2. BIT Tree 2D:

```
int sum[maxn+3][maxn+3],a[maxn+3][maxn+3];

void insert_t(int x,int y, int val) {
    while (x <= maxn) {
        int y1 = y;
        while (y1 <= maxn) {
            sum[x][y1] += val;
            y1 += (y1&(-y1));
        }
        x += (x&(-x)); //di chuyen toi nut cha
    }
}
int sum_t(int x, int y) {
    int t = 0;
    while (x>0) {
        int y1 = y;
        while (y1 > 0) {
            t += sum[x][y1];
            y1 &= (y1-1);
        }
        x &= (x-1); //di chuyen toi nut ke
    }
    return t;
}
void init() {
    FOR (i,1,maxn)
        FOR (j,1,maxn) {
            sum[i][j] = 0;
            a[i][j] = 0;
        }
}
```

```
        return;
    }
    int mid = (l+r) >> 1;
    build_it(i*2, l, mid);
    build_it(i*2+1, mid+1, r);
    it[i].min_ = min ( it[i*2].min_, it[i*2+1].min_);
    it[i].max_ = max ( it[i*2].max_, it[i*2+1].max_);
}
int get_min(int i, int u, int v, int l, int r) {
    if (l > v || r < u) return MAX;
    if (l >= u && r <=v) return it[i].min_;
    int mid = (l+r) >> 1;
    return min (get_min(i*2,u,v,l,mid), get_min(i*2+1,u,v,mid+1,r));
}
```

```
void update(int i, int u, int v, int l, int r) {
    if (r < u || l > v) return;
    if (l >= u && r <= v) {
        it[i]++;
        return;
    }
    int Mid = (l+r)>>1;
    update(i*2,u,v,l,mid);
    update(i*2+1,u,v,mid+1,r);
}
```

4. Segment Tree 2D:

(1081 - Light OJ) The game is played on a 2D N x N grid

Input

Input starts with an integer T (≤ 3), denoting the number of test cases.

The first line of a case is a blank line. The next line contains two integers N (1 ≤ N ≤ 500), Q (0 ≤ Q ≤ 50000).

Each of the next N lines will contain N space separated integers forming the grid. All the integers will be between 0 and 105.

Each of the next Q lines will contain a query which is in the form I J S (1 ≤ I, J ≤ N and 1 ≤ I + S, J + S < N and S > 0).

Output

For each test case, print the case number in a single line.

Then for each query you have to print the maximum integer found in the square whose top left corner is (I, J) and whose bottom right corner is (I+S-1, J+S-1).

```
freopen("test.inp","r",stdin);
freopen("test.out","w",stdout);
int
test;
("%d",&test);
te,1,test) {
canf("%d",&n);
OR (i,1,n) scanf("%d",&h[i]);
(n+1] = 0;
stack <int> S; S.push(0);
FOR (i,1,n+1) {
    while (h[i] < h[S.top()]) {
        R[S.top()] = i;
        S.pop();
    }
    if (h[i] == h[S.top()]) L[i] = L[S.top()]; else L[i] = S.top();
    S.push(i);
}
int ans = 0;
FOR (i,1,n) {
    //printf("L = %d R = %d\n", L[i],R[i]);
    ans = max (ans, h[i] * (R[i] - L[i] - 1));
}
printf("Case %d: %d\n", te, ans);
}
return 0;
}
```

Nếu u == v thì xong.
Nếu u!=v, nếu cha bậc 2^k của chúng khác nhau thì cả 2 cùng nhảy lên 2^k bước lên cha bậc 2^k của chúng. Ở đây k được xét lần lượt theo dãy [lgn], [lgn] - 1, ..., 4, 3, 2, 1, 0. Cụ thể là nếu f[u, k] != f[v, k] thì đặt u = f[u, k], v = f[v, k]. Sau khi duyệt xong dãy [lgn], [lgn] - 1, ..., 4, 3, 2, 1 thì u và v trở thành 2 nút cùng cha. Đáp số LCA(u, v) là cha (trực tiếp) của chúng.

```
/**Problem RENDEZVOUS - ACM Regional Vietnam 2010*/
#define maxn 30000005
#define MOD 1000000005

int n,k; // number of node & queries
vector <int> adj[maxn];      //danh sach ke
int parent[maxn];            //parent[i] = parent of node number i
bool Free[maxn];             //Free[i]=1 neu dinh i chua than, nguoc lai la 0
int f[maxn][20];             //f[u][k] = cha thu 2^k cua dinh u
                             //f[u][0] = cha truc tiep cua node u
                             //f[u][k] = f[v,k-1] voi v = f[u][k-1]
                             //Tuc la cha bac 2^k la cha bac 2^(k-1) cua cha bac 2^(k-1)
int h[maxn];                 //chieu cao (height) cua node i

void bfs(int u) {
    queue<int> Q;
    int v;
    Free[u] = 0;
    h[u] = 1;
    Q.push(u);
    while (!Q.empty()) {
        u = Q.front(); Q.pop();
        FO (i,0,adj[u].size()) {
```



```
            v = adj[u][i];
            if (Free[v]) {
                parent[v] = u;
                Free[v] = 0;
                h[v] = h[u] + 1;
                Q.push(v);
            }
```

```
            scanf("%d %d",&u,&v);
            adj[u].pb(v);
            adj[v].pb(u);
        }
        initParent(); //finish init parent
        initF();    //finish init lca
        printf("finish init lca");
```

```
void init_dijkstra(int n, int start) {
    FOR (i,1,n) {
        dist[i] = INF;
        dad[i] = -1;
    }
    dist[start] = 0;
    while (!Q.empty()) Q.pop();
    Q.push(make_pair(0,start));
}
int dijkstra(int type_query, int n, int start, int finish) {
    init_dijkstra(n,start);
    int u,v,d_u_v;
    while (!Q.empty()) {
        u = Q.top().second; Q.pop();
        if (u == finish) break;
        FO (i,0,adj[u].size()) {
            v = adj[u][i].second;
            d_u_v = adj[u][i].first;
            if (dist[u] + d_u_v < dist[v]) {
                dist[v] = dist[u] + d_u_v;
                dad[v] = u;
                Q.push(make_pair(dist[v], v));
            }
        }
    }
    /****TRACE MIN PATH*****/
    VI path;
    for (int i = finish; i != -1; i = dad[i]) path.push_back(i);
    reverse(path.begin(), path.end());
    /**********************/
    if (type_query == 0) printf("%d\n", dist[finish]);
    else {
        printf("%d",path.size());
        FORV (it,path) printf(" %d", *it);
        printf("\n");
    }
    return dist[finish];
}
```

```
                if (number[v]==0) {
                        visit(v);
                        low[u]=min(low[u],low[v]);
                } else
low[u]=min(low[u],number[v]);
                }
        }
        if (low[u]==number[u]) {
                ans++;  //tăng số thành phần lt mạnh
                set <int> t;  //tập các đỉnh trong thành
phần lt này
                set <int> :: iterator it;
                while (1) {
                        t.insert(s.top());
                        Free[s.top()]=1;
                        if (s.top()==u) {
                                s.pop();
                                break;
                        };
                        s.pop();
                }
                // in ra các đỉnh thuộc tplt
        for (it=t.begin();it!=t.end();it++) cout << *it;
                cout << endl;

        }
}
main() {
        scanf("%d %d",&n,&m);
        int u,v;
        FOR (i,1,m) {
                scanf("%d %d",&u,&v);
                a[u].pb(v);  // danh sách kề
        }
        FOR (i,1,n)
                if (Free[i]==0) visit(i);
        cout << ans;
}
```

```
        Output ← u; //In xau
    end;
until stack = ∅; //Lặp tới khi ngăn xếp rỗng
```

20. KMP:

```
int next[maxn];
int kq[maxn],dem=0;
char T[maxn],P[maxn];
int M,N;

void initKMP() {
    next[0] = -1;
    int j = -1;
    FO (i,1,M) {
        while (j>-1 && P[i]!=P[j+1]) j=next[j];
        if (P[i] == P[j+1]) j++;
        next[i] =j;
    }
}

void process() {  //T=Text; P=Pattern
    int j = -1;
    FO (i,0,N) {
        while (j>-1 && T[i]!=P[j+1]) j = next[j];
    //tim xau P[1..j] la suffix cuaT[1..i-1] va P[j+1] ==
T[i]
        if (T[i] == P[j+1]) j++;
        if (j>=M-1) {
            kq[++dem] = i-j+1;
            j = next[j];  //khi tim thay roi thi dich luon
        }
    }
}

int main() {
    gets(T);gets(P);
    N =  strlen(T);
    M = strlen(P);
    initKMP();
    process();
    FOR (i,1,dem) printf("%d ",kq[i]);
```

```
    cout << 1 << endl;//return 1
```

```
template <class ForwardIterator, class T>
  ForwardIterator lower_bound (ForwardIterator first,
ForwardIterator last, const T& val)
{
  ForwardIterator it;
  iterator_traits<ForwardIterator>::difference_type count, step;
  count = distance(first,last);
  while (count>0)
  {
    it = first; step=count/2; advance (it,step);
    if (*it<val) {                 // or: if (comp(*it,val)), for
version (2)
      first=++it;
      count-=step+1;
    }
    else count=step;
  }
  return first;
}
```

22. Template:

```
#include <set>
#include <map>
#include <list>
#include <cmath>
#include <queue>
#include <stack>
#include <cstdio>
#include <string>
#include <vector>
#include <cstdlib>
#include <cstring>
#include <sstream>
#include <iomanip>
#include <iostream>
#include <algorithm>
#include <ctime>
#include <deque>
#include <bitset>
```

```
LL get(int x, LL bit[]){
    LL ans = 0;
    while(x > 0){
        ans = (ans + bit[x]) % MOD;
        x -= (x&(-x));
    }
    return ans;
}

void Init(){
    POW[0] = 1;
    FOR(i,1,100005) POW[i] = (POW[i-1]*10) % MOD;
    FOR(i,1,n) a[i] = (s[i]*POW[i]) % MOD;
    FOR(i,1,n) b[i] = (s[i]*POW[n+1-i]) % MOD;

    memset(tree1,0,sizeof(tree1));
    memset(tree2,0,sizeof(tree2));
    FOR(i,1,n) update(i,a[i],tree1);
    FOR(i,1,n) update(i,b[i],tree2);
}

int main(){
    scanf("%s", &c);
    s =  string(c);
    n = s.size();
    s = '0' + s;
    Init();
    cin >> m;
    FOR(i,1,m)
    {
        cin >> x;
        if(x == "palindrome?"){
        scanf("%d %d", &L, &R);
        LL tong1 = get(R,tree1) - get(L-1,tree1) + MOD; tong1 %=
MOD;
        LL tong2 = get(R,tree2) - get(L-1,tree2) + MOD; tong2 %=
MOD;
        if((tong1*POW[n-R]) % MOD == (tong2*POW[L-1]) % MOD)
printf("Yes\n");
        else printf("No\n");
        //cout << tong1 << " " << tong2 << endl;
```

25. Sàng nguyên tố + Phi hàm Euler:

```
// Sang Eratosthenes O(n loglog n)
    memset(d,0,sizeof(d));
    FOR(i,2,sqrt(n)){
        if(d[i] == 0){
            for(int j = i*i; j <= n; j += i)
                d[j] = i;
        }
    }
    FOR(i,2,n) if(d[i] == 0) d[i] = i;

// Euler phi function
    phi[1] = 1;
    FOR(i,2,n){
        int j = i;
        int k = d[j];
        while(j % k == 0) j /= k;
        phi[i] = phi[j] * (i/j - i/(j*k));
    }

int phi(int n){
    int res = n;
    for(int i = 2; i*i <= n; i++){
        if(n % i == 0) res -= res/i;
        while(n % i == 0) n /= i;
    }
    if(n > 1) res -= res/n;
    return res;
}
```

26. Kiểm tra nguyên tố Miller – Rabin

```
ULL power_mod(ULL a,ULL b,ULL c){
    long long x=1,y=a; // long long is taken to avoid overflow of
intermediate results
    while(b > 0){
        if(b%2 == 1){
            x = (x%c) * (y%c); x%= c;
        }
        y = (y%c) * (y%c); // squaring the base
        y %= c;
        b /= 2;
    }
```

```
    } else {
        x = c/d * mod_inverse(a/d, b/d);
        y = (c-a*x)/b;
    }
    no nghiệm: x = x_0 + k*b/gcd(a,b), y = y_0 - k*a/gcd(a,b)
```

28. Số nhỏ nhất có N ước:

```
LL ans = 1e18 + 5; int n;
int p[] = { 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41 };
void dfs(int i, LL x, int c){
    if (c > n ) return;
    if (c == n && x < ans) ans = x;
    FOR(j, 1, 60){
        if (ans / p[i] < x) break;
        x *= p[i];
        if ((n % (j + 1)) == 0){
            c *= (j+1);
            dfs(i + 1, x, c );
            c /= (j+1);
        }
    }
}
cin >> n; dfs(0, 1, 1); cout << ans;
```

29. Bài toán lũy thừa tầng:

Áp dụng định lý Fermat nhỏ và định lý số dư Trung Hoa, giải hệ phương trình đồng dư:

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \dots \\ x \equiv a_k \pmod{m_k} \end{cases}$$

```
int limited(int heso, int uoc, int start, int finish, int base){
    //kiem tra xem luy thua tang co chia het cho base không? Yes ->
-1; No -> reminder
    double tu = base, mau = uoc;
    FOR(i,start+1,finish){
        if(heso > (double)log(tu)/log(mau)) {
            return -1;
```

```
    phân tích thành các thừa số nguyên tố + số mũ
}
LL Chinese_systems_equation(vector<int> B, vector<int> R, int base){
    /// sub-bases and reminders
    LL ans = 0;
    int length = B.size();
    vector<int> M, Y; M.resize(length); Y.resize(length);
    FOR(i,0,length-1) M[i] = base/B[i];
    FOR(i,0,length-1) Y[i] = mod_inverse(M[i], B[i]);

    FOR(i,0,length-1) {
        LL tmp = 1LL*R[i]*Y[i]*M[i];
        ans += tmp;
    }
    ans %= base;
    return ans;
}

LL dequy(int a, int start, int finish, int base){
    if(start == finish+1) return 1;
    if(start == finish) return a;
    vector<int> V;
    vector<int> ans;
    phantich(V, base);
    /// base = product of all V[i].fi ^ V[i].se
    FOR(i,0,V.size()-1){
        int tmp;
        int uoc = GCD(V[i], a);
        if(uoc != 1){
            tmp = limited(a, uoc, start, finish, V[i]);
            if(tmp == -1) ans.pb(0);
            else ans.pb(tmp);
        }
        else{
            if(phi[V[i]] == 1) ans.pb(1);
            else{
                tmp = dequy(a, start+1, finish, phi[V[i]]);
                ans.pb(powerMod(a, tmp, V[i]));
            }
        }
    }
    LL res = 0;
    res = Chinese_systems_equation(V, ans, base);
    return res;
}
```

```
            if(segmentPos(p3,P[i],P[i+1]) == 0)
                if(segmentPos(p3,Q[j],Q[j+1]) == 0)
                    ans[dem++] = p3;
        }
    }
    dem--;
    grahamScan(ans,dem);
}
```

35. Tam giác:

35.1. Diện tích tam giác

```
double areaTriangle(point A, point B, point C){
    double ans = abs(A.x*(B.y-C.y) + B.x*(C.y-A.y) + C.x*(A.y-B.y));
    return ans/2;
}
```

35.2. Tính góc BAC theo radian

```
double getAngle(point A, point B, point C){
    double a = dist(B,C);
    double b = dist(C,A);
    double c = dist(A,B);
    double agoc = (b*b + c*c - a*a) / (2*b*c);
    return acos(agoc);
}
```

35.3. Kiểm tra điểm p0 nằm trong tam giác ABC

```
int insideTriangle(point p0, point A, point B, point C){
    double S1 = areaTriangle(p0,A,B);
    double S2 = areaTriangle(p0,B,C);
    double S3 = areaTriangle(p0,C,A);
    double Sum = areaTriangle(A,B,C);
    if(cmp(Sum, S1+S2+S3) == 0) return 1;
    return 0;
}
```

35.4.Kiểm tra điểm p0 nằm trong góc tạo bởi tia AB, AC

```
int insideAngle(point p0, point A, point B, point C){
    if(p0 == A) return 1;
    if(ccw(p0,A,B) * ccw(p0,A,C) > 0) return 0;
    //return (getAngle(A,p0,B) + getAngle(A,p0,C) < PI+eps);
    if(cmp(getAngle(A,B,C), getAngle(A,p0,B) + getAngle(A,p0,C)) == 0)
return 1;
    return 0;
}
```

```
    return insideTriangle(p0,P[1],P[low],P[high]);
}
```

36.2. Bài toán cặp điểm gần nhất

```
Dữ liệu: chạy từ i = 0 → n-1
struct toCompare{
    bool operator() (point A, point B){
        if(A.y != B.y) return A.y < B.y;
        return A.x < B.x;
    }
};
// chia để trị
double closestPairDist(point P[], int n){
    sort(P,P+n);
    set<point, toCompare> b;
    set<point, toCompare> ::iterator lowerIt, upperIt, it;
    int j = 0;
    double ans = inf1;

    FO(i,0,n){
        while(P[i].x - P[j].x > ans){
            b.erase(P[j++]);
        }
        point c = P[i];
        c.y -= ans;
        lowerIt = b.lower_bound(c);
        c.y += 2* ans;
        upperIt = b.upper_bound(c);
        for(it = lowerIt; it != upperIt; it++){
            ans = min(ans, dist(P[i], *it));
        }
        b.insert(P[i]);
    }
    return ans;
}
```

36.3. Đường tròn nhỏ nhất phủ kín n điểm.

```
// Lấy đường trung trực của p1p2
int getCenterLine(point p0, point p1, double &a, double &b, double &c){
    if(p0 == p1) return 0;
    a = p1.x - p0.x;
    b = p1.y - p0.y;
    point p2;
    p2.x = (p0.x + p1.x) / 2;
    p2.y = (p0.y + p1.y) / 2;
    c = -(p2.x * a + p2.y * b);
    return 1;
```

https://vizle.offnote.co

Contact us: **vizle@offnote.co**

This document was generated automatically by **Vizle**

Your **Personal Video Reader Assistant**

Learn from Videos **Faster** and **Smarter**

VIZLE **PRO / BIZ**

PDF, PPT ~~Watermarks~~

- Convert *entire* videos
- *Customize* to retain all essential content
- Include Spoken *Transcripts*
- Customer support

Visit **https://vizle.offnote.co/pricing** to learn more

VIZLE **FREE PLAN**

PDF only Watermarks

- Convert videos *partially*
- Slides may be *skipped**
- Usage restrictions
- No Customer support

Visit **https://vizle.offnote.co** to try free

Login to Vizle to unlock more slides*