Beijing Jiaotong University

D_style's template

Version 2.0

Mr.D_style
2013/10/16

目录

一、 数据结构

1.线段树

1.1二维线段树

1.2矩形面积并

```
#include <cstdio>
#include <set>
#include <cstring>
#include <iostream>
#include <vector>
#include <algorithm>
#define lson p<<1,l,mid
#define rson p<<1|1,mid+1,r
#define A first
#define B second
#define mp make_pair
using namespace std;
typedef pair<pair<double, double>, pair<double, int> > PDDDI;


vector<PDDDI> seg;
set<double> DX;
vector<double> D;
int cnt[1000];
double S[1000];
void update(int, int, int, int, int, int);
void push_up(int, int, int);
int main()
{
    int n, t=1;
    double x1, y1, x2, y2;
    while (~scanf("%d", &n))
    {
        if (!n)
            break;
        DX.clear();
        seg.clear();
        memset(cnt, 0, sizeof(cnt));
        memset(S, 0, sizeof(S));
        for (int i=0;i<n;i++)
        {
            scanf("%lf%lf%lf%lf", &x1, &y1, &x2, &y2);
            DX.insert(x1);
            DX.insert(x2);
            seg.push_back(mp(mp(y1, x1), mp(x2, 1)));
            seg.push_back(mp(mp(y2, x1), mp(x2, -1)));
        }
        D = *(new vector<double>(DX.begin(), DX.end()));
        sort(seg.begin(), seg.end());
        double ANS = 0, last;
        for (int i=0;i<2*n;i++)
        {
            if (i)
                ANS += S[1]*(seg[i].A.A-last);
            int l = lower_bound(D.begin(), D.end(),
seg[i].A.B)-D.begin();
            int r = lower_bound(D.begin(), D.end(),
seg[i].B.A)-D.begin();
            update(1, 1, D.size(), l+1, r, seg[i].B.B);
            last = seg[i].A.A;
        }
        printf("Test case #%d\nTotal explored
area: %.2lf\n\n",t++ , ANS);
    }
    return 0;
}
void update(int p, int l, int r, int x, int y, int c)
{
```

```
    if (x<=l && y>=r)
    {
        cnt[p] += c;
        push_up(p, l, r);
        return;
    }
    int mid = (l+r)>>1;
    if (x <= mid)
        update(lson, x, y, c);
    if (y > mid)
        update(rson, x, y, c);
    push_up(p, l, r);
}
void push_up(int p, int l, int r)
{
    if (cnt[p])
        S[p] = D[r] - D[l-1];
    else if (l==r)
        S[p] = 0;
    else
        S[p] = S[p<<1] + S[p<<1|1];
}
```

1.3矩形周长并

```
#include <cstdio>
#include <cstring>
#include <cctype>
#include <algorithm>
using namespace std;
#define lson l , m , rt << 1
#define rson m + 1 , r , rt << 1 | 1

const int maxn = 22222;
struct Seg{
    int l , r , h , s;
    Seg() {}
    Seg(int a,int b,int c,int d):l(a) , r(b) , h(c) , s(d) {}
    bool operator < (const Seg &cmp) const {
        if (h == cmp.h) return s > cmp.s;
        return h < cmp.h;
    }
}ss[maxn];
bool lbd[maxn<<2] , rbd[maxn<<2];
int numseg[maxn<<2];
int cnt[maxn<<2];
int len[maxn<<2];
void PushUP(int rt,int l,int r) {
    if (cnt[rt]) {
        lbd[rt] = rbd[rt] = 1;
        len[rt] = r - l + 1;
        numseg[rt] = 2;
    } else if (l == r) {
        len[rt] = numseg[rt] = lbd[rt] = rbd[rt] = 0;
    } else {
        lbd[rt] = lbd[rt<<1];
        rbd[rt] = rbd[rt<<1|1];
        len[rt] = len[rt<<1] + len[rt<<1|1];
        numseg[rt] = numseg[rt<<1] + numseg[rt<<1|1];
        if (lbd[rt<<1|1] && rbd[rt<<1]) numseg[rt] -= 2;//两条线重合
    }
}
void update(int L,int R,int c,int l,int r,int rt) {
    if (L <= l && r <= R) {
        cnt[rt] += c;
        PushUP(rt , l , r);
        return ;
    }
    int m = (l + r) >> 1;
    if (L <= m) update(L , R , c , lson);
    if (m < R) update(L , R , c , rson);
    PushUP(rt , l , r);
}
int main() {
    int n;
```

```
    while (~scanf("%d",&n)) {
        int m = 0;
        int lbd = 10000, rbd = -10000;
        for (int i = 0 ; i < n ; i ++) {
            int a , b , c , d;
            scanf("%d%d%d%d",&a,&b,&c,&d);
            lbd = min(lbd , a);
            rbd = max(rbd , c);
            ss[m++] = Seg(a , c , b , 1);
            ss[m++] = Seg(a , c , d , -1);
        }
        sort(ss , ss + m);
        int ret = 0 , last = 0;
        for (int i = 0 ; i < m ; i ++) {
            if (ss[i].l < ss[i].r) update(ss[i].l , ss[i].r - 1 ,
ss[i].s , lbd , rbd - 1 , 1);
            ret += numseg[1] * (ss[i+1].h - ss[i].h);
            ret += abs(len[1] - last);
            last = len[1];
        }
        printf("%d\n",ret);
    }
    return 0;
}
```

1.4可持久化线段树

```
#include <iostream>
#include <cstdio>
#include <cstring>
#include <algorithm>
using namespace std;

int idx[100010];
int ls[2000010], rs[2000010], ST[2000010], sz;
void build(int, int, int);
int add(int, int, int, int);
int query(int, int, int, int, int);
void push_up(int);
int main()
{
    int n, q, a, b, k, num[100010], D[100010];
    scanf("%d%d", &n, &q);
    for (int i=0;i<n;++i)
        scanf("%d", num+i);
    memcpy(D, num, sizeof(num));
    sort(D, D+n);
    int len = unique(D, D+n)-D;
    sz = 0;
    build(1, 1, len);
    idx[0] = 1;
    for (int i=0;i<n;++i)
        idx[i+1] = add(idx[i], 1, len, lower_bound(D, D+len,
num[i])-D+1);
    for (int i=0;i<q;++i)
    {
        scanf("%d%d%d", &a, &b, &k);
        printf("%d\n", D[query(idx[a-1], idx[b], 1, len, k)-1]);
    }
    return 0;
}
void build(int p, int l, int r)
{
    ST[p] = 0;
    ls[p] = l;
    rs[p] = r;
    ++sz;
    if (l == r)
        return;
    int mid = l+r>>1;
    build(p<<1, l, mid);
    build(p<<1|1, mid+1, r);
}
int add(int p, int l, int r, int c)
{
```

```
    int np = ++sz;
    if (l == r)
    {
        ST[np] = ST[p]+1;
        return np;
    }
    int mid = l+r>>1;
    if (c <= mid)
        ls[np] = add(ls[p], l, mid, c), rs[np] = rs[p];
    else
        ls[np] = ls[p], rs[np] = add(rs[p], mid+1, r, c);
    push_up(np);
    return np;
}
void push_up(int p)
{
    ST[p] = ST[ls[p]] + ST[rs[p]];
}
int query(int lp, int rp, int l, int r, int k)
{
    if (l == r)
        return l;
    int mid = l+r>>1;
    if (ST[ls[rp]] - ST[ls[lp]] >= k)
        return query(ls[lp], ls[rp], l, mid, k);
    else
        return query(rs[lp], rs[rp], mid+1, r,
k-ST[ls[rp]]+ST[ls[lp]]);
}
```

2.树状数组

2.1二维

```
#include<iostream>
#include<cstring>
```

```
#include<cstdio>
using namespace std;

void update(int,int,int);
int getsum(int,int);
int lowbit(int);
int tot,n,t,x1,y1,x2,y2,s[1010][1010];
int main()
{
    char cmd;
    scanf("%d",&tot);
    while(tot--)
    {
        scanf("%d%d\n",&n,&t);
        memset(s,0,sizeof(s));
        for(int i=1;i<=t;i++)
        {
            scanf("%c%d%d",&cmd,&x1,&y1);
            getchar();
            if(cmd=='C')
            {
                scanf("%d%d\n",&x2,&y2);
                update(x2+1,y2+1,1);
                update(x2+1,y1,1);
                update(x1,y2+1,1);
                update(x1,y1,1);
            }
            else
            {
                int SUM=getsum(x1,y1);
                printf("%d\n",SUM&1);
            }
        }
        if(tot)
          printf("\n");
    }
    return 0;
}
void update(int x,int y,int c)
```

```
{
    for(int i=x;i<=n;i+=lowbit(i))
       for(int j=y;j<=n;j+=lowbit(j))
          s[i][j]+=c;
}
int getsum(int x,int y)
{
    int SUM=0;
    for(int i=x;i;i-=lowbit(i))
       for(int j=y;j;j-=lowbit(j))
          SUM+=s[i][j];
    return SUM;
}
int lowbit(int x)
{
    return x&(-x);
}
```

2.2三维

```
#include<iostream>
#include<cstring>
using namespace std;

void update(int,int,int,int);
int getsum(int,int,int);
int lowbit(int);
int n,m,x1,y1,z1,x2,y2,z2,s[110][110][110],c;
int main()
{
    while(cin>>n>>m)
    {
        memset(s,0,sizeof(s));
        for(int i=1;i<=m;i++)
        {
            cin>>c>>x1>>y1>>z1;
            if(c)
            {
                cin>>x2>>y2>>z2;
                update(x1,y1,z1,1);
                update(x2+1,y1,z1,1);
                update(x1,y2+1,z1,1);
                update(x1,y1,z2+1,1);
                update(x2+1,y2+1,z1,1);
                update(x2+1,y1,z2+1,1);
                update(x1,y2+1,z2+1,1);
                update(x2+1,y2+1,z2+1,1);
            }
            else
            {
                int SUM=getsum(x1,y1,z1);
                cout<<(SUM&1)<<endl;
            }
        }
    }
    return 0;
}
void update(int x,int y,int z,int cc)
{
    for(int i=x;i<=n;i+=lowbit(i))
       for(int j=y;j<=n;j+=lowbit(j))
          for(int k=z;k<=n;k+=lowbit(k))
             s[i][j][k]+=cc;
}
int getsum(int x,int y,int z)
{
    int SUM=0;
    for(int i=x;i;i-=lowbit(i))
       for(int j=y;j;j-=lowbit(j))
          for(int k=z;k;k-=lowbit(k))
             SUM+=s[i][j][k];
    return SUM;
}
int lowbit(int x)
{
    return x&(-x);
```

```
57 }
```

3. RMQ & LCA

3.1 RMQ

```
 1 #include <cstdio>
 2 #include <cstring>
 3 #include <iostream>
 4 #include <algorithm>
 5 #define MAXN 50010
 6 #define _MAXN 20
 7 using namespace std;
 8
 9
10 class RMQ
11 {
12 private:
13     int val[MAXN], sz, idx[MAXN];
14     int Min[_MAXN][MAXN], Max[_MAXN][MAXN];
15 public:
16     void loadData(int _sz)
17     {
18         sz = _sz;
19         idx[0] = -1;
20         for (int i=1;i<=sz;i++)
21         {
22             scanf("%d", val+i);
23             Min[0][i] = Max[0][i] = val[i];
24             idx[i] = (i&(i-1)) ? idx[i-1] : idx[i-1]+1;
25         }
26         for (int i=1;i<=idx[sz];i++)
27         {
28             int limit = sz+1-(1<<i);
29             for (int j=1;j<=limit;j++)
30             {
31                 Min[i][j] = min(Min[i-1][j],
Min[i-1][j+(1<<i>>1)]);
32                 Max[i][j] = max(Max[i-1][j],
Max[i-1][j+(1<<i>>1)]);
33             }
34         }
35     }
36     int queryMax(int l, int r)
37     {
38         int t = idx[r-l+1];
39         r -= (1<<t)-1;
40         return max(Max[t][l], Max[t][r]);
41     }
42     int queryMin(int l, int r)
43     {
44         int t = idx[r-l+1];
45         r -= (1<<t)-1;
46         return min(Min[t][l], Min[t][r]);
47     }
48 }R;
49 int main()
50 {
51     int n, q;
52     int x, y;
53     while (~scanf("%d%d", &n, &q))
54     {
55         R.loadData(n);
56         for (int i=0;i<q;i++)
57         {
58             scanf("%d%d", &x, &y);
59             printf ("%d\n",R.queryMax(x, y) - R.queryMin(x, y));
60         }
61     }
62     return 0;
63 }
```

3.2 LCA

```cpp
//poj 1330
#include <cstdio>
#include <cstring>
#include <iostream>
#include <vector>
#define MAXN 10010
#define _MAXN 14


using namespace std;
class LCA
{
private:
    int fa[MAXN], lvl[MAXN], idx[MAXN], sz;
    int an[_MAXN][MAXN];
    vector<int> chd[MAXN];
public:
    LCA()
    {
        idx[0] = -1;
        for (int i=1;i<MAXN;i++)
            idx[i] = (i&(i-1)) ? idx[i-1] : idx[i-1]+1;
    }
    void reset(int _sz)
    {
        sz = _sz;
        memset(an, -1, sizeof(an));
        for (int i=0;i<MAXN;i++)
            chd[i].clear();
    }
    void loadData()
    {
        int u, v;
        int root = sz*(sz-1)/2;
        for (int i=0;i<sz-1;i++)
        {
            scanf("%d%d", &u, &v);
            u--;v--;
            fa[v] = u;
            chd[u].push_back(v);
            root -= v;
        }
        dfs(root, 0);
        for (int i=0;i<sz;i++)
            an[0][i] = fa[i];

        for (int i=1;1<<i < sz;i++)
            for (int j=0;j<sz;j++)
                if (an[i-1][j] >= 0)
                    an[i][j] = an[i-1][an[i-1][j]];
    }
    void dfs(int p, int d)
    {
        lvl[p] = d;
        for (int i=0;i<chd[p].size();i++)
            dfs(chd[p][i], d+1);
    }
    int query(int x, int y)
    {
        if (lvl[x] < lvl[y])
            x^=y^=x^=y;
        for (int i=idx[lvl[x]];i>=0;i--)
            if (lvl[x] - (1<<i) >= lvl[y])
                x = an[i][x];
        if (x == y)
            return x;
        for (int i=idx[lvl[x]];i>=0;i--)
            if (an[i][x] >= 0 && an[i][x] != an[i][y])
            {
                x = an[i][x];
                y = an[i][y];
            }
        return fa[x];
    }
```

```
}L;


int main()
{
    int tot, n, a, b;
    scanf("%d", &tot);
    while (tot--)
    {
        scanf("%d", &n);
        L.reset(n);
        L.loadData();
        scanf("%d%d", &a, &b);
        printf("%d\n", L.query(a-1, b-1)+1);
    }
    return 0;
}
```

4. BST

4.1 Treap

```
#include <cstdio>
#include <cstdlib>
#include <iostream>
#include <cstring>
#define MAXN 20010
#define MAXM 60010
#define MAXQ 500010
using namespace std;

class TreapNode
{
public:
    int pri, val, sz;
    TreapNode* chd[2];
    TreapNode(int _val):val(_val)
    {
        chd[0] = chd[1] = NULL;
        pri = rand();
        sz = 1;
    }
    bool operator < (const TreapNode& para) const
    {
        return pri < para.pri;
    }
    int cmp(int x) const
    {
        if (x == val)
            return -1;
        return (x>=val);
    }
    void maintain()
    {
        sz = 1;
        for (int i=0;i<2;i++)
            if (chd[i] != NULL)
                sz += chd[i]->sz;
    }
};
void rotate(TreapNode* &rt, int d)
{
    TreapNode* k = rt->chd[d^1];
    rt->chd[d^1] = k->chd[d];
    k->chd[d] = rt;
    rt->maintain();
    k->maintain();
    rt = k;
}
void insert(TreapNode* &rt, int x)
{
    if (rt == NULL)
        rt = new TreapNode(x);
    else
```

```
    {
        int d = (x>=rt->val);
        insert(rt->chd[d], x);
        if (rt->chd[d]->pri > rt->pri)
            rotate(rt, 1^d);
    }
    rt->maintain();
}
void remove(TreapNode* &rt, int x)
{
    int d = rt->cmp(x);
    if (d == -1)
    {
        TreapNode* u = rt;
        if (rt->chd[0] != NULL && rt -> chd[1] != NULL)
        {
            int d2 = (rt->chd[0]->pri > rt->chd[1]->pri);
            rotate(rt, d2);
            remove(rt->chd[d2], x);
        }
        else
        {
            if (rt->chd[0] == NULL)
                rt = rt->chd[1];
            else
                rt = rt->chd[0];
            delete u;
        }
    }
    else
        remove(rt->chd[d], x);
    if  (rt != NULL)
        rt->maintain();
}
int kth(TreapNode* rt, int k)
{
    if (rt == NULL || k <= 0 || k > rt->sz)
        return 0;
    int s = rt->chd[1] == NULL ? 0 : rt->chd[1]->sz;
    if (k == s+1)
        return rt->val;
    else if (k <= s)
        return kth(rt->chd[1], k);
    else
        return kth(rt->chd[0], k-s-1);
}
TreapNode* root[MAXN];
int cmd[MAXQ][3];
int n, m;
int pa[MAXN], from[MAXM], to[MAXM], w[MAXN], rm[MAXM];
long long tot_q, cnt_q;
int getFa(int x)
{
    return pa[x]!=x ? (pa[x] = getFa(pa[x])) : x;
}
void mergeTo(TreapNode* &src, TreapNode* &des)
{
    for (int i=0;i<2;i++)
        if (src->chd[i] != NULL)
            mergeTo(src->chd[i], des);
    insert(des, src->val);
    delete src;
    src = NULL;
}
void addEdge(int x)
{
    int u = getFa(from[x]-1), v = getFa(to[x]-1);
    if (u != v)
    {
        if (root[u]->sz < root[v]->sz)
        {
            pa[u] = v;
            mergeTo(root[u], root[v]);
        }
        else
        {
            pa[v] = u;
            mergeTo(root[v], root[u]);
```

```
        }
    }
}
void query(int x, int k)
{
    cnt_q++;
    tot_q += kth(root[getFa(x)], k);
}
void modify(int x, int v)
{
    int u = getFa(x);
    remove(root[u], w[x]);
    insert(root[u], v);
    w[x] = v;
}
int main()
{
    int type[300], T=1;
    type['D'] = 0;
    type['Q'] = 1;
    type['C'] = 2;
    while (scanf("%d%d", &n, &m) && n)
    {
        for (int i=0;i<n;i++)
            scanf("%d", w+i);
        for (int i=0;i<m;i++)
            scanf("%d%d", from+i, to+i);
        memset(rm, 0, sizeof(rm));
        int cnt = 0;
        while (1)
        {
            char ch;
            int x, y, z;
            getchar();
            scanf("%c", &ch);
            if (ch == 'E')
                break;
            scanf("%d", &x);
            if (ch == 'D')
                rm[x-1] = 1;
            if (ch == 'Q')
                scanf("%d", &y);
            if (ch == 'C')
            {
                scanf("%d", &z);
                y = w[x-1];
                w[x-1] = z;
            }
            cmd[cnt][0] = type[ch];
            cmd[cnt][1] = x-1;
            cmd[cnt][2] = y;
            cnt++;
        }

        for (int i=0;i<n;i++)
        {
            pa[i] = i;
            if (root[i] != NULL)
                delete root[i];
            root[i] = new TreapNode(w[i]);
        }
        for (int i=0;i<m;i++)
            if (!rm[i])
                addEdge(i);

        tot_q = cnt_q = 0;
//        for (int i=0;i<cnt;i++)
//            cout << cmd[i][0] <<  ' ' << cmd[i][1] << ' ' << cmd[i][2]
<< endl;
        for (int i=cnt-1;i>=0;i--)
        {
            if (cmd[i][0] == 0)
                addEdge(cmd[i][1]);
            if (cmd[i][0] == 1)
                query(cmd[i][1], cmd[i][2]);
            if (cmd[i][0] == 2)
                modify(cmd[i][1], cmd[i][2]);
        }
```

```
        printf("Case %d: %.6lf\n", T++, tot_q/((double)cnt_q));
    }
    return 0;
}
```

4.2 Splay

```
#include <cstdio>
#include <iostream>
#include <vector>
#define MAXN 100000
using namespace std;

class SplayNode
{
public:
    SplayNode *chd[2], *pa;
    int val, cnt, lazy;
    SplayNode(int _val=0, int _cnt=1,int
_lazy=0):val(_val),cnt(_cnt),lazy(_lazy){}
    void maintain()
    {
        cnt = chd[0]->cnt + chd[1]->cnt + 1;
    }
    int lr()
    {
        return (pa->chd[1]==this);
    }
    void push_down()
    {//有 add 等标记注意更新
        if (lazy)
        {
            swap(chd[0], chd[1]);
            chd[0]->lazy ^= 1;
            chd[1]->lazy ^= 1;
            lazy = 0;
        }
    }
};
//null 值的设置必须对结果没有影响
//如果 splay 时修改了 null 中的值，注意要每次要还原
SplayNode *null = new SplayNode(0, 0);

class Splay
{
public:
    SplayNode *root;
    int sz;
    vector<int> seq;
    void init(int _sz)
    {
        sz = _sz;
        root = build(0, sz);
        root->pa = null;
    }
    SplayNode* build(int l, int r)
    {
        if (l > r)
            return null;
        int mid = (l+r)>>1;
        SplayNode *p = new SplayNode(mid);
        p->chd[0] = build(l, mid-1);
        p->chd[1] = build(mid+1, r);
        p->chd[0]->pa = p->chd[1]->pa = p;
        p->maintain();
        return p;
    }
    void splay(SplayNode* src, SplayNode* des=null)
    {
        while (src->pa != des)
        {
            if (src->pa->pa == des)
                rotate(src);
            else if (src->lr() == src->pa->lr())
            {
                rotate(src->pa);
```

```
            rotate(src);
        }
        else
        {
            rotate(src);
            rotate(src);
        }
    }
}
void rotate(SplayNode* x)
{
    SplayNode *p = x->pa;
    p->push_down();
    x->push_down();
    int d = x->lr();
    p->pa->chd[p->lr()] = x;x->pa = p->pa;
    p->chd[d] = x->chd[1^d];x->chd[1^d]->pa = p;
    x->chd[1^d] = p;p->pa = x;
    p->maintain();
    if (p == root)
        root = x;
}
SplayNode *find(int x)
{
    for (SplayNode* p = root;;)
    {
        p->push_down();
        int c = p->chd[0]->cnt;
        if (x == c)
            return p;
        if (x > c)
        {
            x -= c+1;
            p = p->chd[1];
        }
        else
            p = p->chd[0];
    }
}
```

```
    SplayNode *&subSeq(int x, int y)
    {
        SplayNode *l = find(x-1);
        SplayNode *r = find(y);
        splay(l);
        splay(r, l);
        return r->chd[0];
    }
    void getSeq(SplayNode *p)
    {
        if (p == null)
            return;
        p->push_down();
        getSeq(p->chd[0]);
        seq.push_back(p->val);
        getSeq(p->chd[1]);
    }
}SP;
```

4.3 SBT

5. Tree Query

5.1 树链剖分

```
#include <cstdio>
//#include <iostream>
#include <cstring>
#include <algorithm>
#define lson p<<1,l,mid
#define rson p<<1|1,mid+1,r
#define MAXN 10010
using namespace std;

class Tree
```

```
{
public:
    int root, n;
    int adj[MAXN<<1][3], adj_cnt, last[MAXN];
    int sz[MAXN], top[MAXN], fw[MAXN], pos[MAXN];
    int ST[MAXN<<2], STN, ST_val[MAXN];
    int dep[MAXN], fa[MAXN];
    int vis[MAXN], data[MAXN][2];
    void reset()
    {
        memset(adj, 0, sizeof(adj));
        memset(vis, 0, sizeof(vis));
        memset(last, 0, sizeof(last));
        adj_cnt = 0;
        STN = 0;
    }
    void addEdge(int u, int v, int w)
    {
        adj[++adj_cnt][0] = v;
        adj[adj_cnt][1] = w;
        adj[adj_cnt][2] = last[u];
        last[u] = adj_cnt;
    }
    void loadData()
    {
        scanf("%d", &n);
        int u, v, w;
        for (int i=0;i<n-1;++i)
        {
            scanf("%d%d%d", data[i], data[i]+1, &w);
            data[i][0]--;data[i][1]--;
            addEdge(data[i][0], data[i][1], w);
            addEdge(data[i][1], data[i][0], w);
        }
        root = n/2;
    }
    void dfs(int p, int d)
    {
        vis[p] = 1;
        sz[p] = 1;
        dep[p] = d;
        for (int i=last[p];i;i=adj[i][2])
        {
            int v = adj[i][0];
            if (!vis[v])
            {
                fa[v] = p;
                fw[v] = adj[i][1];
                dfs(v, d+1);
                sz[p] += sz[v];
            }
        }
    }
    void bfs()
    {
        int que[MAXN], pre[MAXN];
        int chain_last[MAXN], chain[MAXN], chain_cnt=0;
        int f = 0, r = 1;
        int MAX, flag;
        memset(vis, 0, sizeof(vis));
        que[f] = root;
        top[root] = root;
        pre[root] = -1;
        chain[root] = 0;
        chain_last[0] = root;
        while (f < r)
        {
            int u = que[f++];
            vis[u] = 1;
            MAX = -1;
            for (int i=last[u];i;i=adj[i][2])
            {
                int v = adj[i][0];
                if (!vis[v] && MAX < sz[v])
                {
                    MAX = sz[v];
                    flag = v;
                }
```

```
89          }
90          if (MAX == -1) continue;
91          pre[flag] = u;
92          top[flag] = top[u];
93          chain[flag] = chain[u];
94          chain_last[chain[u]] = flag;
95          que[r++] = flag;
96          for (int i=last[u];i;i=adj[i][2])
97          {
98              int v = adj[i][0];
99              if (!vis[v] && v!=flag)
100             {
101                 chain[v] = ++chain_cnt;
102                 chain_last[chain_cnt] = v;
103                 pre[v] = -1;
104                 top[v] = v;
105                 que[r++] = v;
106             }
107         }
108     }
109     for (int i=0;i<=chain_cnt;++i)
110     {
111         int len = dep[chain_last[i]]-dep[top[chain_last[i]]];
112         for (int j=STN+len, k=chain_last[i];j>STN;--j, k=pre[k])
113         {
114             ST_val[j] = fw[k];
115             pos[k] = j;
116         }
117         STN+=len;
118     }
119 }
120 void ST_build(int p, int l, int r)
121 {
122     if (l==r)
123     {
124         ST[p] = ST_val[l];
125         return;
126     }
127     int mid = l+r>>1;
```

```
128     ST_build(lson);
129     ST_build(rson);
130     ST[p] = max(ST[p<<1], ST[p<<1|1]);
131 }
132 void ST_update(int p, int l, int r, int x, int y, int c)
133 {
134     if (x<=l && y>=r)
135     {
136         ST[p] = c;
137         return;
138     }
139     int mid = l+r>>1;
140     if (x <= mid)
141         ST_update(lson, x, y, c);
142     if (y > mid)
143         ST_update(rson, x, y, c);
144     ST[p] = max(ST[p<<1], ST[p<<1|1]);
145 }
146 int ST_query(int p, int l, int r, int x, int y)
147 {
148     if (x<=l && y>=r)
149         return ST[p];
150     int mid = l+r>>1;
151     if (y <= mid)
152         return ST_query(lson, x, y);
153     if (x > mid)
154         return ST_query(rson, x, y);
155     int p1=ST_query(lson, x, y), p2=ST_query(rson, x, y);
156     return max(p1, p2);
157 }
158 void update(int p, int w)
159 {
160     int u = data[p][0], v = data[p][1];
161     if (dep[u] < dep[v])
162         u^=v^=u^=v;
163     if (top[u] == u)
164         fw[u] = w;
165     else
166         ST_update(1, 1, STN, pos[u], pos[u], w);
```

```
    }
    int query(int u, int v)
    {
        int MAX = -1, tmp;
        while (u!=v)
        {
            if (top[u] != top[v])
            {
                if (dep[top[u]] < dep[top[v]])
                    u^=v^=u^=v;
                if (top[u] == u)
                {
                    tmp = fw[u];
                    u = fa[u];
                }
                else
                {
                    tmp = ST_query(1, 1, STN,
pos[u]-dep[u]+dep[top[u]]+1, pos[u]);
                    u = top[u];
                }
            }
            else
            {
                if (dep[u] < dep[v])
                    u^=v^=u^=v;
                tmp = ST_query(1, 1, STN, pos[u]-dep[u]+dep[v]+1,
pos[u]);
                u = v;
            }
            MAX = max(MAX, tmp);
        }
        return MAX;
    }
    void work()
    {
        char buf[10];
        int x, y;
        loadData();
        dfs(root, 0);
        bfs();
        ST_build(1, 1, STN);
        while(scanf("%s", buf))
        {
            if (buf[0] == 'D')
                break;
            scanf("%d%d", &x, &y);
            if (buf[0] == 'Q')
                printf("%d\n", query(x-1, y-1));
            else
                update(x-1, y);
        }
    }
}T;
int main()
{
    int tot;
    scanf("%d", &tot);
    while (tot--)
    {
        T.reset();
        T.work();
    }
    return 0;
}
```

5.2 Link-Cut-Tree

二、 字符串

1. KMP & exKMP

1.1 Kmp

```
int kmp(const char *S, int lenS, const char *T, int lenT)
{
    next[0]=-1;
    int ret = 0;
    for(int j=-1,i=1;i<lenT;i++)
    {
       while(j>=0&&T[j+1]!=T[i])
          j= next [j];
       if(T[j+1]==T[i])
          j++;
        next[i]=j;
    }
    for(int j=-1,i=0;i<lenS;i++)
    {
       while(j>=0&&T[j+1]!=S[i])
          j= next[j];
       if(T[j+1]==S[i])
          j++;
       if(j==lenT-1)
       {
          j=next[j];
          ret++;
       }
    }
    return ret;
}
```

1.2 exKmp

```
ex_next[1]=0;
for(int j=0;j<lenT-1&&T[j]==T[j+1];j++)
    ex_next[1]++;
ex_next[0]=lenT;
for(int i=2,k=1;i<lenT;i++)
{
    int l=ex_next[i-k],max_l=k+ex_next[k]-1;
    if(l<max_l-i+1)
        ex_next[i]=l;
    else
    {
        int j=max(0,max_l-i+1);
        while(i+j<lenT&&T[i+j]==T[j])
            j++;
        ex_next[i]=j;
        k=i;
    }
}

for(int j=0;j<lenS&&j<lenT&&T[j]==S[j];j++)
    extend[0]++;
for(int i=1,k=0;i<lenS;i++)
{
    int l=ex_next[i-k],max_l=k+extend[k]-1;
    if(l<max_l-i+1)
        extend[i]=l;
    else
    {
        int j=max(0,max_l-i+1);
        while(i+j<lenS&&j<lenT&&S[i+j]==T[j])
            j++;
        extend[i]=j;
        k=i;
    }
}
```

2. Palindrome

2.1 Mannacher

```cpp
#include<iostream>
#include<cstdio>
#define MAXN 110010
using namespace std;

char* iniString(char *);
int Manacher(char*);
int main()
{
    char s[MAXN],*str;
    while(~scanf("%s",s))
    {
        str=iniString(s);
        int ANS=Manacher(str);
        printf("%d\n",ANS);
    }
    return 0;
}
char* iniString(char *s)
{
    char str[MAXN<<1];
    for(int i=0,j=0;s[i];i++)
    {
        if(!i)
           str[j++]='~';
        str[j++]='#';
        str[j++]=s[i];
        if(!s[i+1])
        {
            str[j++]='#';
            str[j++]='`';
            str[j++]=0;
        }
    }
    return str;
}
int Manacher(char *str)
{
//    printf("%s\n",str);
    int rd[MAXN<<1],r=0,p,MAX=1;
    for(int i=1;str[i+1];i++)
    {
        if(r>i)
           rd[i]=min(rd[2*p-i],r-i);
        else
           rd[i]=1;
        while(str[i+rd[i]]==str[i-rd[i]])
           rd[i]++;
        p=rd[i]+i>r?i:p;
        r=max(rd[i]+i,r);
        MAX=max(MAX,rd[i]-1);
    }
    return MAX;
}
```

3. AC_Automaton

```cpp
#include<iostream>
#include<cstdio>
#include<cstring>
#define MAXN 5000010
#define MAX_CHD 26
using namespace std;


class ACAutomaton
{
private:
    int chd[MAXN][MAX_CHD];
```

```
13      int fail[MAXN];
14      int que[MAXN];
15      int ID[MAX_CHD+300];
16      int val[MAXN];
17      int sz;
18  public:
19      ACAutomaton()
20      {
21          fail[0]=0;
22          for(int i=0;i<MAX_CHD;i++)
23             ID[i+'a']=i;
24          sz=1;
25      }
26      void Reset()
27      {
28          memset(chd[0],0,sizeof(chd[0]));
29          sz=1;
30      }
31      void Insert(char *s,int key)
32      {
33          int p=0;
34          for(int i=0;s[i];i++)
35          {
36              int c=ID[s[i]];
37              if(!chd[p][c])
38              {
39                  memset(chd[sz],0,sizeof(chd[sz]));
40                  val[sz]=0;
41                  chd[p][c]=sz++;
42              }
43              p=chd[p][c];
44          }
45          val[p]+=key;
46      }
47      void Build()
48      {
49          int front=0,rear=1;
50          que[front]=0;
51          while(rear-front)
52          {
53  //              cout<<front<<' '<<rear<<endl;
54              int u=que[front++];
55              for(int i=0;i<MAX_CHD;i++)
56              {
57                  int &v=chd[u][i];
58                  if(v)
59                  {
60                      que[rear++]=v;
61                      fail[v]=u?chd[fail[u]][i]:0;
62                  }
63                  else if(u)
64                      v=chd[fail[u]][i];
65              }
66          }
67      }
68      ///HDU 2222
69      int Solve(char *str)
70      {
71          int ret=0,p=0;
72          for(int i=0;str[i];i++)
73          {
74              int c=ID[str[i]];
75              while(p&&!chd[p][c])
76                 p=fail[p];
77              if(chd[p][c])
78                 p=chd[p][c];
79              for(int now=p;now&&val[now]!=-1;now=fail[now])
80              {
81                  //cout<<str[i]<<' '<<val[now]<<endl;
82                  ret+=val[now];
83                  val[now]=-1;
84              }
85          }
86          return ret;
87      }
88  }AC;
89
90  int main()
```

```
{
    char s[1000010];
    int tot,n;
    scanf("%d",&tot);
    while(tot--)
    {
        scanf("%d",&n);
        AC.Reset();
        for(int i=1;i<=n;i++)
        {
            scanf("%s",s);
            AC.Insert(s,1);
        }
        AC.Build();
        scanf("%s",s);
        printf("%d\n",AC.Solve(s));
    }
}
```

4. Suffix Array

```
#define F(x) ((x)/3+((x)%3==1?0:tb))
#define G(x) ((x)<tb?(x)*3+1:((x)-tb)*3+2)
#define cmp1(r,a,b) (r[a]==r[b]&&r[a+1]==r[b+1]&&r[a+2]==r[b+2])
#define cmp3(r,a,b) (r[a]<r[b]||r[a]==r[b]&&wv[a+1]<wv[b+1])
#define cmp2(k,r,a,b)
(k==2?(r[a]<r[b]||r[a]==r[b]&&cmp3(r,a+1,b+1)):cmp3(r,a,b))
const int M=20;
const int N=(1<<M);
///sa 数组从 sa[1]到 sa[n],存储的是 0 到 n-1 的排列
///sa[i]记录的是排名为 i 的后缀的起始位置
///rank 数组从 rank[0]到 rank[n-1],存储的是 1 到 n 的排列
///rank[i]记录的是以 i 为起点的后缀的排名
///high[i]记录 lcp(i,i-1)
class suffix_array
{
public:
    int rank[N], sa[3*N], init[3*N], high[N], n;
    int buc[N], m, wv[N], i, j ,k;
    int log[N],rmq[M][N];
    suffix_array()
    {
        log[0] = -1;
        for(i = 1; i < N ; ++i)log[i] = (i & (i - 1)) ? log[i-1] :
log[i-1] + 1 ;
    }
    inline void sort(int *r, int *a, int *b, int n, int m)
    {
        for(i = 0; i < n; ++i) wv[i] = r[a[i]];
        for(i = 0; i < m; ++i) buc[i] = 0;
        for(i = 0; i < n; ++i) buc[wv[i]]++;
        for(i = 1; i < m; ++i) buc[i] += buc[i-1];
        for(i = n - 1; i >= 0; --i) b[--buc[wv[i]]] = a[i];
        return;
    }
    inline void suffix_dc3(int *r, int *sa, int n, int m)
    {
        int *rn = r + n;
        int *san = sa + n, ta = 0, tb = (n + 1) / 3, tbc = 0, p, *wa
= rank ,*wb = high;
        r[n] = r[n+1] = 0;
        for(i = 0; i < n; ++i) if(i % 3 != 0) wa[tbc++] = i;
        sort(r + 2, wa, wb, tbc, m);
        sort(r + 1, wb, wa, tbc, m);
        sort(r    , wa, wb, tbc, m);
        for(p = 1, rn[F(wb[0])] = 0, i = 1; i < tbc; ++i)
            rn[F(wb[i])] = cmp1(r, wb[i-1], wb[i]) ? p - 1 : p++;
        if(p < tbc) suffix_dc3(rn, san, tbc, p);
        else for(i = 0; i < tbc; ++i) san[rn[i]] = i;
        for(i = 0; i < tbc; ++i) if(san[i] < tb) wb[ta++] = san[i]
* 3;
        if(n % 3 == 1) wb[ta++] = n - 1;
        sort(r, wb, wa, ta, m);
        for(i = 0; i < tbc; ++i) wv[wb[i] = G(san[i])] = i;
        for(i = 0, j = 0, p = 0; i < ta && j < tbc; ++p)
```

```
51          sa[p] = cmp2(wb[j] % 3, r, wa[i], wb[j]) ? wa[i++] :
wb[j++];
52      for(; i < ta; sa[p++] = wa[i++]);
53      for(; j < tbc; sa[p++] = wb[j++]);
54    }
55    inline int exec(char *in)
56    {
57      for(int &p=n=m=0; in[p]; ++p)///注意结束符
58      {
59        init[p] = in[p];
60        m = max(m, init[p] + 1);
61      }
62      init[n] = 0;
63      suffix_dc3(init, sa, n + 1, m);
64      for(i = 1; i <= n ; ++i) rank[sa[i]] = i;
65      for(i = 0, k = 0; i < n ; high[rank[i++]] = k)
66        for(k ? k-- : 0 , j = sa[rank[i] - 1] ; init[i+k] ==
init[j+k] ; ++k);
67      for(i = 1; i <= n ; ++i) rmq[0][i] = high[i];
68      for(i = 1; i <= log[n] ; ++i)
69        for(j = 1; j <= n - (1 << i) + 1; ++j)
70          rmq[i][j] = min(rmq[i-1][j] ,
rmq[i-1][j+(1<<i>>1)]);
71      return n;
72    }
73    inline int lcp(int a, int b)/// lcp(rank[i],rank[j])询问 i,j 后
缀的最长公共前缀
74    {
75      if (a==b) return n-sa[a];
76      if(a > b) swap(a, b);
77      int t = log[b - a];
78      return min(rmq[t][a + 1] , rmq[t][b - (1<<t) + 1]);
79    }
80 }SA;
```

5. Suffix Automanton

```
1  #include <cstdio>
2  #include <cstring>
3
4  const int MAXN = 250010;
5  const int MAX_CHD = 26;
6  class suffix_automaton
7  {
8  public:
9    int chd[MAXN<<1][MAX_CHD];
10   int cnt[MAXN<<1];
11   int fa[MAXN<<1], len[MAXN<<1];
12   int sz, last, n;
13   int hd[MAXN], f[MAXN], next[MAXN<<1];
14   void reset()
15   {
16     last = n = 0;
17     sz = 1;
18     fa[0] = -1;
19     memset(chd[0], 0, sizeof(chd[0]));
20   }
21   void add(int w)
22   {
23     int p = last, np = sz++; ++n;
24     len[np] = len[p]+1;
25     cnt[np] = 1;
26     memset(chd[np], 0, sizeof(chd[np]));
27     for (;p>=0 && !chd[p][w];p=fa[p])
28       chd[p][w] = np;
29     if (p<0)
30       fa[np] = 0;
31     else
32     {
33       int q = chd[p][w];
34       if (len[p]+1 == len[q])
35         fa[np] = q;
```

```
36          else
37          {
38              int nq = sz++;
39              memcpy(chd[nq], chd[q], sizeof(chd[q]));
40              len[nq] = len[p]+1;
41              fa[nq] = fa[q];
42              fa[q] = nq;
43              fa[np] = nq;
44              for (;p>=0 && chd[p][w] == q;p=fa[p])
45                  chd[p][w] = nq;
46          }
47      }
48      last = np;
49    }
50 }SAM;
```

三、 DP

四、 经典问题

1.第 K 大不同子串

```
 1 #include <cstdio>
 2 #include <cstring>
 3 #include <iostream>
 4
 5 using namespace std;
 6
 7 const int MAXN = 90010;
 8 const int MAX_CHD = 26;
 9 class suffix_automaton
10 {
11 public:
12     int chd[MAXN<<1][MAX_CHD];
13     int cnt[MAXN<<1];
14     int fa[MAXN<<1], len[MAXN<<1];
15     int sz, last, n;
16     int hd[MAXN], f[MAXN], next[MAXN<<1];
17     int hash[MAXN<<1][MAX_CHD], hash_cnt[MAXN<<1],
ch[MAXN<<1][MAX_CHD];
18     void reset()
19     {
20         last = n = 0;
21         sz = 1;
22         fa[0] = -1;
23         memset(chd[0], 0, sizeof(chd[0]));
24     }
25     void add(int w)
26     {
27         int p = last, np = sz++; ++n;
28         len[np] = len[p]+1;
29         cnt[np] = 1;
30         memset(chd[np], 0, sizeof(chd[np]));
31         for (;p>=0 && !chd[p][w];p=fa[p])
32             chd[p][w] = np;
33         if (p<0)
34             fa[np] = 0;
35         else
36         {
37             int q = chd[p][w];
38             if (len[p]+1 == len[q])
39                 fa[np] = q;
40             else
41             {
42                 int nq = sz++;
43                 cnt[nq] = 1;
44                 memcpy(chd[nq], chd[q], sizeof(chd[q]));
45                 len[nq] = len[p]+1;
46                 fa[nq] = fa[q];
47                 fa[q] = nq;
48                 fa[np] = nq;
49                 for (;p>=0 && chd[p][w] == q;p=fa[p])
50                     chd[p][w] = nq;
```

```
            }
        }
        last = np;
    }
    void init()
    {
        memset(hd, -1, sizeof(hd));
        for (int i=0;i<sz;++i)
        {
            next[i] = hd[len[i]];
            hd[len[i]] = i;
        }
        for (int i=n;i>=0;--i)
            for (int p=hd[i];p>=0;p=next[p])
                for (int c=0;c<MAX_CHD;++c)
                    if (chd[p][c])
                    {
                        cnt[p] += cnt[chd[p][c]];
                        hash[p][hash_cnt[p]] = chd[p][c];
                        ch[p][hash_cnt[p]++] = c;
                    }
    }
    void query(int k)
    {
        for (int p=0;k;)
            for (int i=0;i<hash_cnt[p];++i)
                if (k > cnt[hash[p][i]])
                    k-=cnt[hash[p][i]];
                else
                {
                    putchar('a'+ch[p][i]);
                    p = hash[p][i];
                    --k;
                    break;
                }
        putchar('\n');
    }
}SAM;

```

```
char buf[MAXN];
int main()
{
    int q, k;
    scanf("%s%d", buf, &q);
    SAM.reset();
    for (int i=0;buf[i];++i)
        SAM.add(buf[i]-'a');
    SAM.init();
    while (q--)
    {
        scanf("%d", &k);
        SAM.query(k);
    }
    return 0;
}
```

2.第 K 大子串(包括相同)

```
#include <cstdio>
#include <cstring>
#include <algorithm>
#define LL long long
using namespace std;

int n, m, v[100005], len, len_;
long long t, sum, fq[30], st[30];
char ch, s[100005];

int main(){
scanf("%s%d", s, &m);
n = strlen(s);

if (m > (LL)n * (LL)(n+1) / (LL)2)
{
    printf("No such line.\n");
```

```
    return 0;
}
for (int i=0; i<n; i++) v[i] = i;
len = n;
while (m > 0)
{
    memset(fq, 0, sizeof(fq));
    memset(st, 0, sizeof(st));
    for (int j=0, i; j<len; j++)
    {
        i = v[j];
        fq[s[i]-'a'] += n - i;
        st[s[i]-'a']++;
    }
    for (char c='a'; c<='z'; c++){
        if (fq[c-'a'] >= m)
        {
            ch = c;
            break;
        }
        else
            m -= fq[c-'a'];
    }
    printf("%c", ch);
    m -= st[ch-'a'];
    len_ = len;
    len = 0;
    for (int i=0; i<len_; i++)
        if (s[v[i]] == ch && v[i] < n-1)
            v[len++] = v[i] + 1;
}
printf("\n");
return 0;
}
```

3.树上每点出发的最长路

```
int n, m, md[1000010][2], top[1000010], d[1000010];
int hd[1000010], ecnt;
void dfs(int);
void work(int, int);
int main()
{
    while(...)
    {
        memset(hd, -1, sizeof(hd));
        memset(e, 0, sizeof(e));
        ecnt = 0;

        ///input...

        dfs(1);///看情况加vis[]或者pre
        work(1, 0);///之前清空vis[]
        ///每个点出发的最长路记录在d[]
    }
    return 0;
}
void dfs(int p)
{
    md[p][0] = 0;
    top[p] = 1;
    for (int i=hd[p];i!=-1;i=e[i].next)
    {
        dfs(e[i].t);
        int v = md[e[i].t][0] + e[i].w;
        if (v > md[p][0])
        {
            md[p][1] = md[p][0];
            md[p][0] = v;
            top[p] = 2;
        }
        else if (top[p] == 2 && v > md[p][1])
```

```
            md[p][1] = v;
        else if (top[p] == 1)
            md[p][top[p]++] = v;
    }
}
void work(int p, int r)
{
    d[p] = max(md[p][0], r);
    int down;
    for (int i=hd[p];i!=-1;i=e[i].next)
    {
        if (top[p] == 1)
            down = r + e[i].w;
        else if(md[p][0] == md[e[i].t][0]+e[i].w)
            down = max(r, md[p][1]) + e[i].w;
        else
            down = max(r, md[p][0]) + e[i].w;
        work(e[i].t, down);
    }
}
```

4.树上距离小于 k 的点对(带求重心)

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

struct edge
{
    int s, t, w;
    int next;
}e[20010];

int hd[10010], ecnt;

int n, k, vis[10010], sz[10010], d[10010], idx[10010], m_sz[10010];
int ANS;
int get_root(int);
void dfs(int, int, int &);
void dfs_calc(int, int, int &, int);
void addEdge(int, int, int);
int calc(int *d, int l, int r, int c);
void solve(int);

int main()
{
    int u, v, w;
    while (scanf("%d%d", &n, &k) && n && k)
    {
        memset(hd, -1, sizeof hd);
        memset(e, 0, sizeof e);
        ecnt = 0;
        for (int i=0;i<n-1;++i)
        {
            scanf("%d%d%d", &u, &v, &w);
            addEdge(u, v, w);
            addEdge(v, u, w);
        }
        memset(vis, 0, sizeof(vis));
        ANS = 0;
        solve(1);
        printf("%d\n", ANS);
    }
    return 0;
}
int get_root(int p)
{
    int flag, MIN = n;
    int tot = 0;
    dfs(p, 0, tot);
    for (int i=0;i<=tot;++i)
    {
        int MAX = max(m_sz[i], sz[0]-sz[i]);
```

```
        if (MAX < MIN)
        {
            MIN = MAX;
            flag = idx[i];
        }
    }
    return flag;
}
void dfs(int p, int pre, int &tot)
{
    sz[tot] = 1;
    m_sz[tot] = 0;
    idx[tot] = p;
    int now = tot;
    for (int i=hd[p];i!=-1; i = e[i].next)
        if (!vis[e[i].t] && e[i].t!=pre)
        {
            int last = tot;
            dfs(e[i].t, p, ++tot);
            m_sz[now] = max(sz[last+1], m_sz[now]);
            sz[now] += sz[last+1];
        }
}
void solve(int p)
{
    int root = get_root(p), last, time = 0;
//    printf("%d-%d\n", p, root);
    vis[root] = 1;
    d[0] = 0;
    for (int i=hd[root];i != -1; i = e[i].next)
        if (!vis[e[i].t])
        {
            last = time;
            dfs_calc(e[i].t, root, ++time, e[i].w);
            sort(d+last+1, d+time+1);
            ANS -= calc(d, last+1, time+1, k);
        }
    sort(d, d+time+1);
    ANS += calc(d, 0, time+1, k);
    for (int i=hd[root];i != -1; i = e[i].next)
        if (!vis[e[i].t])
            solve(e[i].t);
}
void dfs_calc(int p, int pre, int &time, int dis)
{
    d[time] = dis;
    for (int i=hd[p];i != -1; i = e[i].next)
        if (!vis[e[i].t] && e[i].t != pre)
            dfs_calc(e[i].t, p, ++time, dis+e[i].w);
}
int calc(int *d, int l, int r, int c)
{
    int ret = 0;
    for (;l+1<r;++l)
    {
        while (r>l+1 && d[r-1]+d[l]>c)
            --r;
        ret += r-l-1;
    }
    return ret;
}
void addEdge(int u, int v, int w)
{
    e[ecnt].s = u, e[ecnt].t = v, e[ecnt].w = w, e[ecnt].next = hd[u];
hd[u] = ecnt++;
}
```

5.树上路径第 K 大

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

```

```
 8 const int MAXN = 100010;
 9 const int _MAXN = 20;
10 void build(int, int, int);
11 int add(int, int, int, int);
12 int query(int, int, int, int, int, int, int);
13 void push_up(int);
14 class Tree
15 {
16 private:
17     int fa[MAXN], lvl[MAXN], idx[MAXN], sz;
18     int an[_MAXN][MAXN], val[MAXN], D[MAXN];
19     int tot, n;
20     int ls[_MAXN*MAXN], rs[_MAXN*MAXN], ST[_MAXN*MAXN];
21     int rt[MAXN];
22     vector<int> chd[MAXN];
23 public:
24     Tree()
25     {
26         idx[0] = -1;
27         for (int i=1;i<MAXN;i++)
28             idx[i] = (i&(i-1)) ? idx[i-1] : idx[i-1]+1;
29     }
30     void reset()
31     {
32         tot = 0;
33         memset(an, -1, sizeof(an));
34         for (int i=0;i<MAXN;i++)
35             chd[i].clear();
36     }
37     void solve()
38     {
39         int u, v, q;
40         reset();
41         scanf("%d%d", &sz, &q);
42         for (int i=0;i<sz;++i)
43             scanf("%d", val+i);
44         memcpy(D, val, sizeof(val));
45         sort(D, D+sz);
46         n = unique(D, D+sz)-D;
47         build(1, 1, n);
48         for (int i=0;i<sz-1;i++)
49         {
50             scanf("%d%d", &u, &v);
51             chd[u].push_back(v);
52             chd[v].push_back(u);
53         }
54         rt[0] = 1;
55         dfs(1, 0, 0);
56         for (int i=1;1<<i < sz;i++)
57             for (int j=1;j<=sz;j++)
58                 if (an[i-1][j] >= 0)
59                     an[i][j] = an[i-1][an[i-1][j]];
60         int a, b, c, k;
61         for (int i=0;i<q;++i)
62         {
63             scanf("%d%d%d", &a, &b, &k);
64             c = lca(a, b);
65 //            cout << "<" << a <<','<<b<<','<<c<<">"<<endl;
66             printf("%d\n", D[query(rt[a], rt[b], rt[c],
rt[an[0][c]], 1, n, k)-1]);
67         }
68     }
69     void dfs(int p, int pre, int d)
70     {
71         lvl[p] = d;
72         an[0][p] = pre;
73         rt[p] = add(rt[pre], 1, n, lower_bound(D, D+n,
val[p-1])-D+1);
74         for (int i=0;i<chd[p].size();i++)
75             if (chd[p][i] != pre)
76                 dfs(chd[p][i], p, d+1);
77     }
78     int lca(int x, int y)
79     {
80         if (lvl[x] < lvl[y])
81             x^=y^=x^=y;
82         for (int i=idx[lvl[x]];i>=0;i--)
83             if (lvl[x] - (1<<i) >= lvl[y])
```

```
            x = an[i][x];
        if (x == y)
            return x;
        for (int i=idx[lvl[x]];i>=0;i--)
            if (an[i][x] >= 0 && an[i][x] != an[i][y])
            {
                x = an[i][x];
                y = an[i][y];
            }
        return an[0][x];
    }
    void build(int p, int l, int r)
    {
        ST[p] = 0;
        ls[p] = l;
        rs[p] = r;
        ++tot;
        if (l == r)
            return;
        int mid = l+r>>1;
        build(p<<1, l, mid);
        build(p<<1|1, mid+1, r);
    }
    int add(int p, int l, int r, int c)
    {
        int np = ++tot;
        if (l == r)
        {
            ST[np] = ST[p]+1;
            return np;
        }
        int mid = l+r>>1;
        if (c <= mid)
            ls[np] = add(ls[p], l, mid, c), rs[np] = rs[p];
        else
            ls[np] = ls[p], rs[np] = add(rs[p], mid+1, r, c);
        push_up(np);
        return np;
    }
```

```
    void push_up(int p)
    {
        ST[p] = ST[ls[p]] + ST[rs[p]];
    }
    int query(int lp, int rp, int ca, int caa, int l, int r, int k)
    {
        if (l == r)
            return l;
        int mid = l+r>>1;
        int tmp = ST[ls[rp]]+ST[ls[lp]]-ST[ls[ca]]-ST[ls[caa]];
        if (tmp >= k)
            return query(ls[lp], ls[rp], ls[ca], ls[caa], l, mid, k);
        else
            return query(rs[lp], rs[rp], rs[ca], rs[caa], mid+1, r,
k-tmp);
    }
}L;

int main()
{
    L.solve();
    return 0;
}
```

6.最大回文子正方形

```
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <cmath>
#include <iostream>
#include <algorithm>
#include <queue>
using namespace std;

#define N 805
```

```
11 void doManacher(int len, int str[], int rad[])
12 {
13     int i, mx = 0, id;
14     for(i = 1; i < len - 1; i++)
15     {
16         if(mx > i)
17             rad[i] = min(rad[2 * id - i], rad[id] + id - i);
18         else
19             rad[i] = 1;
20         while(str[i + rad[i]] == str[i - rad[i]])
21             rad[i]++;
22         if(rad[i] + i > mx)
23         {
24             mx = rad[i] + i;
25             id = i;
26         }
27     }
28 }
29 int initString(int len, int str[], int fstr[])
30 {
31     int i;
32     int T1 = 31415926 + 100, T2 = 31415926 + 105, T3 = 31415926 +
110;
33     fstr[0] = T1, fstr[1] = T2, fstr[2 * len + 2] = T3;
34     for(i = 0; i < len; i++)
35     {
36         fstr[i * 2 + 2] = str[i];
37         fstr[i * 2 + 3] = T2;
38     }
39     len = len * 2 + 3;
40     return len;
41     //fstr[len] = '\0';
42 }
43
44
45 int g[N][N], str[N], fstr[2 * N];
46 int n, m, rad[2 * N], row[N][N][2], col[N][N][2];
47 int main()
48 {
```

```
49     int t, i, j, k;
50     scanf("%d", &t);
51     while(t--)
52     {
53         scanf("%d%d", &n, &m);
54         for(i = 0; i < n; i++)
55         {
56             for(j = 0; j < m; j++)
57                 scanf("%d", &g[i][j]);
58             int nlen = initString(m, g[i], fstr);
59             doManacher(nlen, fstr, rad);
60             for(j = 0; j < m; j++)
61             {
62                 row[i][j][0] = (rad[2 * j + 1] - 1) / 2;
63                 row[i][j][1] = rad[2 * j + 2] / 2;
64             }
65         }
66         for(i = 0; i < m; i++)
67         {
68             for(j = 0; j < n; j++)
69                 str[j] = g[j][i];
70             //str[j] = '\0';
71             int nlen = initString(n, str, fstr);
72             doManacher(nlen, fstr, rad);
73             for(j = 0; j < n; j++)
74             {
75                 col[j][i][0] = (rad[2 * j + 1] - 1) / 2;
76                 col[j][i][1] = rad[2 * j + 2] / 2;
77             }
78         }
79         int sx, sy, tx, ty, ans = 0, tl;
80         for(i = 0; i < n; i++)
81         {
82             for(j = 0; j < m; j++)
83             {
84                 //odd first
85                 tl = min(min(i + 1, n - i), min(j + 1, m - j));
86                 for(k = 1; k <= tl && 2 * tl - 1 > ans; k++)
87                 {
```

```
                tl = min(tl, row[i - k + 1][j][1]);
                tl = min(tl, row[i + k - 1][j][1]);
                tl = min(tl, col[i][j - k + 1][1]);
                tl = min(tl, col[i][j + k - 1][1]);
                if(k > tl) break;
            }
            if(ans < 2 * (k - 1) - 1)
            {
                ans = 2 * (k - 1) - 1;
                sx = i - (k - 1) + 1, sy = j - (k - 1) + 1;
                tx = i + (k - 1) - 1, ty = j + (k - 1) - 1;
            }
            //even
            tl = min(min(i, n - i), min(j, m - j));
            for(k = 1; k <= tl && 2 * tl > ans; k++)
            {
                tl = min(tl, row[i - k][j][0]);
                tl = min(tl, row[i + k - 1][j][0]);
                tl = min(tl, col[i][j - k][0]);
                tl = min(tl, col[i][j + k - 1][0]);
                if(k > tl) break;
            }
            if(ans < 2 * (k - 1))
            {
                ans = 2 * (k - 1);
                sx = i - (k - 1), sy = j - (k - 1);
                tx = i + (k - 1) - 1, ty = j + (k - 1) - 1;
            }
        }
    }
    printf("%d\n", ans);
    //sx++, sy++, tx++, ty++;
    //printf("%d %d %d %d\n", sx, sy, tx, ty);
  }
  return 0;
}
```

五、 其他

7.Java

7.1输入

```
import java.io.*
import java.util.*
public class Main {
    public static void main(String args[])
    {
        Scanner cin = new Scanner(new BufferedInputStream(System.in));
        //…
    }
}

//读一个整数:    int n = cin.nextInt();
//读一个字符串:  String s = cin.next();
//读一个浮点数:  double t = cin.nextDouble();
//读一整行:      String s = cin.nextLine();
//判断是否有下一个输入可以用 cin.hasNext() 或 cin.hasNextInt() 或
cin.hasNextDouble()
```

7.2输出

```
//保留几位小数,用 DecimalFormat
import java.text.*;
    DecimalFormat f = new DecimalFormat("#.00#");
    DecimalFormat g = new DecimalFormat("0.000");
    double a = 123.45678, b = 0.12;
    System.out.println(f.format(a));
    System.out.println(f.format(b));
    System.out.println(g.format(b));
```

7.3大数

```
//BigInteger 整数
//BigDecimal 浮点数
import java.math.*  // 需要引入 java.math 包
BigInteger a = BigInteger.valueOf(100);
BigInteger b = BigInteger.valueOf(50);
BigInteger c = a.add(b)   // c = a + b;
//主要有以下方法可以使用:
BigInteger add(BigInteger other)
BigInteger subtract(BigInteger other)
BigInteger multiply(BigInteger other)
BigInteger divide(BigInteger other)
BigInteger mod(BigInteger other)
int compareTo(BigInteger other)
static BigInteger valueOf(long x)
//输出数字时直接使用 System.out.println(a) 即可
```

7.4 Poj1001(计算 num^n)

```
import java.io.*;
import java.util.*;
import java.math.BigDecimal;

public class Main {
    public static void main(String args[]){
        Scanner cin = new Scanner(System.in);

        BigDecimal num;
        int n;
        String r;

        while(cin.hasNextBigDecimal()){
            num = cin.nextBigDecimal();
            n = cin.nextInt();
            num = num.pow(n);
            r = num.stripTrailingZeros().toPlainString();
            if(r.startsWith("0.")){
              r = r.substring(1);
            }
            System.out.println(r);
        }
    }
}
```

8. GCD

```
long long gcd(long long a,long long b)
{
    while(b^=a^=b^=a%=b);
    return a;
}
```

9. 输入挂

```
char ch;
void read_int(int &a)
{
    int p=1;
    for(ch=getchar();(ch<'0'||ch>'9') && (ch!='-');)
     ch=getchar();
    if(ch=='-')a=0,p=-1;else a=ch-'0';
    for(ch=getchar();ch>='0'&& ch<='9';ch=getchar())
     a=a*10+ch-'0';
    a*=p;
}
```

10. 常见类

10.1 Matrix

```
#define mod 10007
typedef ull unsigned long long;
class Matrix
{
private:
    int r, c;
    ull ele[MAXS][MAXS];
public:
    Matrix(int _r=0, int _c=0):r(_r),c(_c)
    {
        memset(ele, 0, sizeof(ele));
    }
    Matrix(int _r, int _c, ull v)//单位矩阵
    {
        new (this)Matrix(_r, _c);
        for (int i=0;i<_r && i<_c;i++)
                ele[i][i] = v%mod;
    }
    friend Matrix operator*(Matrix a, Matrix b)
    {
        Matrix product(a.r, b.c);
        for (int i=0;i<a.r;i++)
            for (int j=0;j<a.c;j++)
                for (int k=0;k<b.c;k++)
                    product.ele[i][j] = (product.ele[i][j] +
a.ele[i][k]*b.ele[k][j])%mod;
        return product;
    }
    void update(int x, int y, ull c)
    {
        ele[x][y] = (c+ele[x][y])%mod;
    }
    void modify(int x, int y, ull c)
    {
        ele[x][y] = c%mod;
    }
    ull getVal(int x, int y)
    {
        return ele[x][y];
    }
    int getRow()
    {
        return r;
    }
    int getCol()
    {
        return c;
    }
    Matrix pow(ull n)
    {
        Matrix ret(r, c, 1), base = (*this);
        for (ull i=n;i;base = base*base,i>>=1)
            if (i&1)
                ret = ret*base;
        return ret;
    }
    void print()
    {
        printf("===\n");
        for (int i=0;i<r;i++,printf("\n"))
            for (int j=0;j<c;j++)
                printf("%d ", ele[i][j]);
    }
    Matrix powSum(ull n)
    {
        Matrix temp(2*r, 2*c), ret(r, c);
        for (int i=0;i<r;i++)
            for (int j=0;j<c;j++)
                temp.modify(i, j, ele[i][j]);
        for (int i=r;i<2*r;i++)
        {
```

```
            temp.modify(i-r, i, 1);
            temp.modify(i, i, 1);
        }
//        temp.print();
        temp = temp.pow(n+1);
//        temp.print();
        for (int i=0;i<r;i++)
            for (int j=0;j<c;j++)
                ret.modify(i, j, temp.getVal(i, j+c));
        //得到m^0+m^1+m^2+...+m^n 返回前注意根据题目是否要减去一个单位矩阵
        return ret;
    }
};
```

10.2 Cube

```
class Cube{
//  +-+
//  |0|
//+-+-+-+-+
//|2|4|3|5|
//+-+-+-+-+
//  |1|
//  +-+
public:
    int faces[6];
    Cube(){}
    Cube(int *data)
    {
        for (int i=0;i<6;++i)
            faces[i] = data[i];
    }
    void ro_z(int t = 1){while(t--) ro(2, 4, 3, 5);}
    void ro_x(int t = 1){while(t--) ro(5, 1, 4, 0);}
    void ro_y(int t = 1){while(t--) ro(0, 3, 1, 2);}
    void ro(int a, int b, int c, int d)
    {
        int t = faces[d];faces[d] = faces[c];faces[c] = faces[b];faces[b] = faces[a];faces[a] = t;
    }
    int equals(Cube para)
    {
        for (int i=0;i<6;++i)
            if (faces[i] != para.faces[i])
                return false;
        return true;
    }
    bool operator == (Cube para)
    {
        Cube t = *this;
        for (int i=0;i<6;++i)
        {
            for (int j=0;j<4;++j,t.ro_z())
                if (t.equals(para))
                    return true;
            if (i & 1)
                t.ro_x();
            else
                t.ro_y();
        }
        return false;
    }
};
```

10.3 Date (Bate)

```
//日期函数
int days[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
struct date {
    int year, month, day;
};
//判闰年
```

```
int leap (int year) {
    return (year % 4 == 0 && year % 100 != 0) || year % 400 == 0;
}
//判合法性
int legal (date a) {
    if (a.month < 0 || a.month > 12)
        return 0;
    if (a.month == 2)
        return a.day > 0 && a.day <= 28 + leap (a.year);
    return a.day > 0 && a.day <= days[a.month - 1];
}
//比较日期大小
int datecmp (date a, date b) {
    if (a.year != b.year)
        return a.year - b.year;
    if (a.month != b.month)
        return a.month - b.month;
    return a.day - b.day;
}
//返回指定日期是星期几
int weekday (date a) {
    int tm = a.month >= 3 ? (a.month - 2) : (a.month + 10);
    int ty = a.month >= 3 ? a.year : (a.year - 1);
    return (ty + ty / 4 - ty / 100 + ty / 400 + (int) (2.6 * tm - 0.2)
+ a.day) % 7;
}
//日期转天数偏移
int date2int (date a) {
    int ret = a.year * 365 + (a.year - 1) / 4 - (a.year - 1) / 100
+ (a.year - 1) / 400, i;
    days[1] += leap (a.year);
    for (i = 0; i < a.month - 1; ret += days[i++]);
    days[1] = 28;
    return ret + a.day;
}
//天数偏移转日期
date int2date (int a) {
    date ret;
    ret.year = a / 146097 * 400;
    for (a %= 146097; a >= 365 + leap (ret.year); a -= 365 + leap
(ret.year), ret.year++);
    days[1] += leap (ret.year);
    for (ret.month = 1; a >= days[ret.month - 1]; a -= days[ret.month
- 1], ret.month++);
    days[1] = 28;
    ret.day = a + 1;
    return ret;
}
```

11. Notes

11.1 点的分治

每次把重心删掉，更新答案，再分治剩下的子树。

11.2 dp[1<<n][n]最短路

如果点可以重复到达，先最短路预处理，用最短路代替边转移。

11.3 排名串

匹配条件：两个串的排名串相等当且仅当这两个串的每个位置上的元素前面等于它的元素个数和小于它的元素个数都相等。

11.4 upper/lower_bound

在数组中都是寻找第一个**大于等于**关键字的元素，区别是lower_bound找第一个下标，upper_bound找最后一个下标。

在 set 和 map 中的 upper_bound 是找第一个**大于**的元素。