2020牛客暑期多校训练营（第七场）

比赛情况

题号	A	B	C	D	E	F	G	H	I	J
状态	Ø	O	O	O	-	-	-	O	-	-

O 在比赛中通过 Ø 赛后通过 ! 尝试了但是失败了 - 没有尝试

比赛时间

2020-08-01 12:00-17:00

题解

A - Social Distancing

在半径 r 的圆内取 n 个整数点问两两距离平方的最大值。

~~意识到正解是打表的时候有点晚了。~~哦不好像不是打表，看到了这个dp做法。

$\sum\limits_{i=1}^{n-1}\sum\limits_{j=i+1}^n(x_i-x_j)^2+(y_i-y_j)^2=\sum\limits_{i=1}^{n-1}\sum\limits_{j=i+1}^nx_i^2+y_i^2+x_j^2+y_j^2-2x_ix_j-2y_iy_j=n\sum\limits_{i=1}^n(x_i^2+y_i^2)-(\sum\limits_{i=1}^nx_i)^2-(\sum\limits_{i=1}^ny_i)^2$

我们可以用 $dp[n][j][k]$ 表示 n 个点 $\sum\limits_{i=1}^nx_i=j,\sum\limits_{i=1}^ny_i=k$ 时 $\sum\limits_{i=1}^n(x_i^2+y_i^2)$ 的最大值，则所求值为 $n\times dp[n][j][k]-j^2-k^2$ 。按半径由近至远加点，复杂度可以做到 $点数\times(nr)^2$ 。

```
#include<bits/stdc++.h>
#define ll long long
#define pii pair<int,int>
#define mp make_pair
#define fi first
#define se second
#define pb push_back
using namespace std;
int n,r,dp[10][505][505],res[10][35];
int sqr(int x){return x*x;}
struct Node{int x,y,dis;};
vector<Node>v;
int main()
{
    for(int i=-30;i<=30;i++)for(int
j=-30;j<=30;j++)v.pb(Node{i,j,i*i+j*j});
    sort(v.begin(),v.end(),[&](Node o1,Node o2){return o1.dis<o2.dis;});
```

```
    for(int r=1,p=0;r<=30;r++)
    {
        while(p<v.size()&&v[p].dis<=r*r)
        {
            for(int i=0;i<8;i++)
            for(int j=-r*i;j<=r*i;j++)
            for(int k=-r*i;k<=r*i;k++)
dp[i+1][j+v[p].x+250][k+v[p].y+250]=max(dp[i][j+250][k+250]+sqr(v[p].x)+sqr
(v[p].y),dp[i+1][j+v[p].x+250][k+v[p].y+250]);
            p++;
        }
        for(int i=1;i<=8;i++)for(int j=-r*i;j<=r*i;j++)for(int k=-
r*i;k<=r*i;k++)
        res[i][r]=max(res[i][r],i*dp[i][j+250][k+250]-sqr(j)-sqr(k));
    }
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n,r;
        scanf("%d%d",&n,&r);
        printf("%d\n",res[n][r]);
    }
    return 0;
}
```

C - A National Pandemic

给一棵树，每个点有点权 $F(x)$ 三种操作，给所有点加上它们到 x 的距离/将某个 $F(x)$ 置为 0/询问 $F(x)$

树链剖分可以搞，辅助一堆记录的登西。给所有点加距离即加上 dep_x 和 $dep_{自己}$，然后需要减去 dep_{lca} 这个是树剖的部分。我们从 x 到根权值 $+1$，等到询问 y 时查询 y 到根的和即可。如果需要置零则查询一下，同样记录一个变量，减去当前的值。

```
#include <bits/stdc++.h>
using namespace std;
const int N = 5e4+5;
typedef long long ll;
int n,m;
struct E {
    int nxt,to;
}e[N<<1];
int head[N],tot;
void add(int x,int y) {
```

```
    e[++tot].to = y;e[tot].nxt = head[x];head[x] = tot;
}
ll tr[N<<2];
int lazy[N<<2];
ll decn[N];
ll adding,times;
void build(int p,int l,int r) {
    tr[p] = 0;
    lazy[p] = 0;
    if (l==r)return;
    int mid = (l+r)>>1;
    build(p<<1,l,mid);
    build(p<<1|1,mid+1,r);
}
void Push_Down(int p,int l,int r) {
    if (l==r||!lazy[p])return;
    int mid = (l+r)>>1;
    tr[p<<1] += (ll)(mid-l+1)*lazy[p];
    tr[p<<1|1] += (ll)(r-mid)*lazy[p];
    lazy[p<<1]+=lazy[p];
    lazy[p<<1|1]+=lazy[p];
    lazy[p] = 0;
}
void Update(int p,int l,int r,int a,int b,int c) {
    Push_Down(p,l,r);
    if (l>=a && r<=b) {
        tr[p] += (ll)(r-l+1)*c;
        lazy[p] += c;
        return;
    }
    int mid = (l+r)>>1;
    if (a <= mid)Update(p<<1,l,mid,a,b,c);
    if (b > mid)Update(p<<1|1,mid+1,r,a,b,c);
    tr[p] = tr[p<<1]+tr[p<<1|1];
}
ll Getans(int p,int l,int r,int a,int b) {
    Push_Down(p,l,r);
    if (l>=a && r<=b) return tr[p];
    ll ans = 0;
    int mid = (l+r)>>1;
    if (a <= mid)ans+=Getans(p<<1,l,mid,a,b);
    if (b > mid)ans+=Getans(p<<1|1,mid+1,r,a,b);
    return ans;
}
int size[N],son[N],w[N],top[N],fa[N],cnt,id[N],dpt[N];
void dfs1(int x,int f) {
    fa[x] = f;
    size[x] = 1;
    dpt[x] = dpt[f]+1;
    son[x] = 0;
    for (int i = head[x];i;i=e[i].nxt) {
```

```
        if (e[i].to == f)continue;
        dfs1(e[i].to,x);
        size[x]+=size[e[i].to];
        if (size[e[i].to] > size[son[x]])son[x] = e[i].to;
    }
}
void dfs2(int x,int tp) {
    top[x] = tp;
    w[x] = ++cnt;
    id[cnt] = x;
    if (son[x])dfs2(son[x],tp);
    for (int i = head[x];i;i=e[i].nxt)
        if (e[i].to!=son[x] && e[i].to!=fa[x])
            dfs2(e[i].to,e[i].to);
}
void update(int x,int y,int z)
{
    while (top[x]!=top[y])
    {
        if (dpt[top[x]]<dpt[top[y]])swap(x,y);
        Update(1,1,n,w[top[x]],w[x],z);
        x = fa[top[x]];
    }
    if (dpt[x]<dpt[y])swap(x,y);
    Update(1,1,n,w[y],w[x],z);
}
ll getans(int x,int y)
{
    ll ans = 0;
    while (top[x]!=top[y])
    {
        if (dpt[top[x]]<dpt[top[y]])swap(x,y);
        ans+=Getans(1,1,n,w[top[x]],w[x]);
        x = fa[top[x]];
    }
    if (dpt[x]<dpt[y])swap(x,y);
    ans+=Getans(1,1,n,w[y],w[x]);
    return ans;
}
ll getFx(int x) {
    ll ans = decn[x]-(ll)times*dpt[x];
    ans = ans+adding;
    ans = ans+getans(1,x);
    return ans;
}
void init() {
    memset(head,0,sizeof(head));
    memset(decn,0,sizeof(decn));
    tot = cnt = 0;
    adding = times = 0;
```

```
}
int main()
{
    int cas;
    scanf("%d",&cas);
    while(cas--) {
        init();
        scanf("%d%d",&n,&m);
        int x,y;
        for (int i = 1;i<n;i++) {
            scanf("%d%d",&x,&y);
            add(x,y);add(y,x);
        }
        build(1,1,n);
        dfs1(1,0);
        dfs2(1,1);
        while (m--) {
            int opt;
            scanf("%d",&opt);
            if (opt == 1) {
                scanf("%d%d",&x,&y);
                adding += y-dpt[x];
                times++;
                update(1,x,2);
            } else if(opt == 2) {
                scanf("%d",&x);
                ll tmp = getFx(x);
                if (tmp > 0) decn[x]-=tmp;
            } else {
                scanf("%d",&x);
                printf("%lld\n",getFx(x));
            }
        }
    }
}
```

D - Fake News

满足 1 到 n 的平方和是完全平方数的只有 1 和 24，打表之后可以猜得到。

H - Dividing

经过一顿推导之后可以发现对于同一个k可能的n只有k的倍数和k的倍数$+1$，（包括1），所以要求的就是下面这个式子

$N+\sum_{i=2}^{K}\lfloor \frac{N}{i}\rfloor +\lfloor\frac{N+1}{i}\rfloor+1$

此公式针对$K\leq N$的情况，如果$K>N$可以使$K=N$最后答案中加$K-N$因为对于大于N的k每个k都仅有1这个数字可以作为一组。

上面这个式子用数论分块可以直接求，另外要注意一些特殊情况。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll mod = 1e9+7;
ll getans(ll n,ll k) {
    ll ans = 0;
    for (ll l = 1,r;l<=k;l=r+1) {
        r = min(k,n/(n/l));
        ans += (r-l+1)*(n/l)%mod;
        ans = ans%mod;
    }
    return ans;
}
int main()
{
    ll ans = 0;
    ll n,k;
    scanf("%lld%lld",&n,&k);
    if (k >= n) {
        ans += k-n+2;
        k = n-1;
    }
    ans += getans(n,k)+getans(n-1,k)+k-n;
    ans = (ans%mod+mod)%mod;
    printf("%lld\n",ans);
    return 0;
}
```