Test 20221115

T1 Reverse

题意

一个类最短路问题。

分析

因为 k 是定值，所以每个点能够一步到达的点的数量也是有限的，并且有左右上限，不难发现：

当 k 是偶数时，可以到达距离为 $1,3,5,\dots k-1$ 的点

当 k 是奇数时，可以到达距离为 $0,2,4,\dots k-1$ 的点

所以一测就写了个寄搜，然后真的寄了。

为什么呢？

坑点

改题的时候 kds 给了我一组 hack，我才发现自己推出了上面的规律之后就开始偏离题目做题，这个题本来是利用翻转区间的操作让一个点到达另一个，对于当前这个点，当我们把区间左右移动的时候，我们得到的点也是同向移动的，也就是说：

对于一个点 x ，把包含他的区间 $[l,r]$ 翻转， x 就会变到 $L+R-x$ 的位置。

但是对于边界上面的点，我们这个区间是取不满的，因为 l 最少就是 1 ，所以出于边界上面的点或者是靠近边界一定距离的点就不能满足我们上面的规律了，只能使用 $L+R-x$ 的方式计算，这就是我写寄了的原因。

然后改了一下，把寄搜换成了 bfs ，（感觉最短路也行？只是可能 T 得很凶）。

然后观察到每个点能到达的点一定是彼此同奇偶的，所以用一个 set 来维护奇偶的点所在的集合，然后保留每一个点第一次被更新到的值就行了，芝士 bfs 的性质，或者说这就是个 spfa 。

Code

```
#include<bits/stdc++.h>
#define Hanggoash
using namespace std;
inline int read()
{
    int x=0;
    int f=1;char c=getchar();
    for(;!isdigit(c);c=getchar())if(c=='-')f=-1;
    for(;isdigit(c);c=getchar())x=(x<<1)+(x<<3)+(c^48);
    return x*f;
}
template<typename T>inline void wr(T x)
{
    if(x<0)putchar('-'),x=-x;
    if(x>9)wr(x/10);
```

```
    putchar(x%10^48);
}
const int maxn=1e5+10;
int banned[maxn];
int dis[maxn];
int n,k,m,s;
set<int> odd,even;
inline void pre()
{
    odd.clear(),even.clear();
    n=read(),k=read(),m=read(),s=read();
    for(register int i=1;i<=m;++i)banned[read()]=1;
    memset(dis,-1,sizeof dis);
    k--;
    for(int i=1;i<=n;++i)
    {
        if(banned[i]||i==s)continue;
        if(i&1)odd.insert(i);
        else even.insert(i);
    }
}

typedef set<int>::iterator iter;
inline void bfs()
{
    queue<int> q;
    dis[s]=0,q.push(s);
    while(q.size())
    {
        int x=q.front();q.pop();
        int L=max(1,x-k),R=min(n,x+k);
//      printf("%d----\nL0:%d R0:%d\n",x,L,R);
        L=L+(L+k)-x,R=R+(R-k)-x;
//      printf("L:%d R:%d\n",L,R);
        set<int> &p = L&1?odd:even;
        if(L&1)
            for(iter i=odd.lower_bound(L);i!=odd.end()&&*i<=R;odd.erase(i++))
                dis[*i]=dis[x]+1,q.push(*i);
        else
            for(iter i=even.lower_bound(L);i!=even.end()&&*i<=R;even.erase(i++))
                dis[*i]=dis[x]+1,q.push(*i);
    }
}
int main()
{
    #ifdef Hanggoash
    freopen("Reverse.in","r",stdin);
    freopen("Reverse.out","w",stdout);
    #endif
    pre();
    bfs();
    for(register int i=1;i<=n;++i)
        wr(dis[i]),putchar(' ');
    return 0;
}
```

关于 STL 的一个小问题

就是如果在遍历的同时删除迭代器指向的内容的话，只能像代码里面写的那样`odd.erase(i++)`。

然而不能在循环里面写 `odd.erase(i)` ，然后再在`for`里面写 `i++` ，至于为什么，银牌佬感觉讲了一遍，但也感觉什么都没有讲，g。

T2 Silhouette

题意

三视图问题，看起来很简单，实际上这个难度的容斥并不是给我们这种凡人做的。

给出正视图和左视图，求有多少种立体图形能够满足条件，模 10^9+7

分析

俯视图

如果俯视图也确定的话，那么这个图就一定是确定的了~~(废话)~~，另外样例解释也用了俯视图，我们不妨从俯视图的角度出发考虑。

不难发现，如果我们定义俯视图为矩阵 $x_{i,j}$ 的话，那么我们要求的东西就是：

> $\forall i\in[1,n],\max_{j=1}^{n}x_{i,j}=A_i,max_{j=1}^{n}x_{j,i}=B_i$ 的解数
>
> 不妨先将A, B排序，显然这样并不会有什么影响。如果A的最大值不等于B的最大值则答 案是 0 ，否则一定有解。从大到小枚举每个A, B中出现的值s，我们先来考虑s为最大值的情况，此时我们要确定的位置构成了一个矩形。
>
> 可 以发现我们要解决这样一个子问题：一个a × b的矩形，每个位置的值在 $[0,s]$ 中，且每行每列的最大值均为 s ，需要计算有多少取值的方案。

我们发现有两个维度，我们很难直接计算出：在一个确切的行数下，有多少种方案满足条件或者是多少种不满足条件。

容斥

所以考虑容斥，定义 $f(i)=C_a^i(s^i\times(s+1)^{a-i}-s^a)^b$ 为当前 $a\times b$ 的矩阵内至少有 i 行不满足条件的方案数。

解释一下上面计算式的意义：（必须满足每一列都要合法）

$$s^i\times(s+1)^{a-i}\text{：有}i\text{个数不满足条件（取不到}s\text{），其余}a-i\text{个随便取}$$
$$-s^a\text{：减去所有 数都不满足条件（取不到}s\text{）的情况，为了满足每一列都合法的条件}$$
$$()^b\text{：一共有}b\text{列}$$
$$C_a^i\text{：从}a\text{行里选出}i\text{行}$$

既然是至少，就意味着：

$f(0)$ 包含了有 $0,1,2,3,\dots a$ 行不满足条件的情况

$f(1)$ 包含了有 $1,2,3\dots a$ 行不满足条件的情况

$f(2)$ 包含了有，$2, 3 ... a$ 行不满足条件的情况

......

$f(a)$ 包含了有 a 行不满足条件的情况。

这就很完美地对应了标准容斥的使用条件，也就是：

$$|\bigcup_{i=1}^{n} s_i| = \sum_{i=1}^{n} |s_i| - \sum_{1 \le i < j \le n}^{n} |s_i \cap s_j| + \sum_{1 \le i < j < k \le n}^{n} |s_i \cap s_j \cap s_k| - ...... + (-1)^{n+1} \sum_{1 \le ... \le n}^{n} |s_1 \cap ... \cap s_n|$$

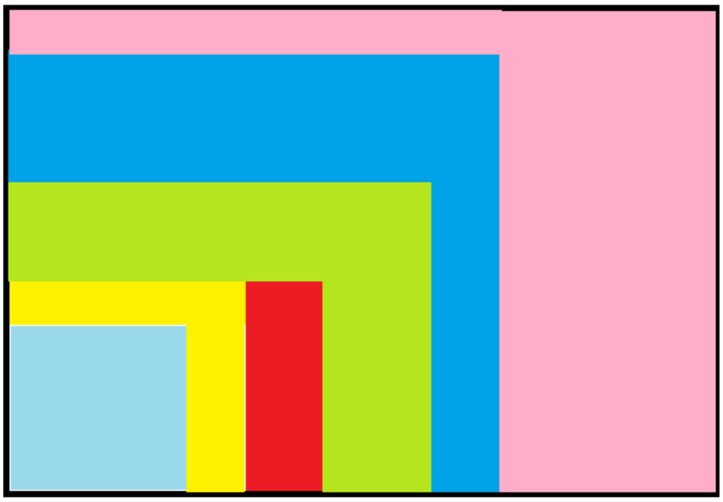
所以本题中一个 $a \times b$ 矩形的方案数就是 $\sum_{i=0}^{a} (-1)^i f(i)$

所以我们就可以愉快地开始容斥了。

计算

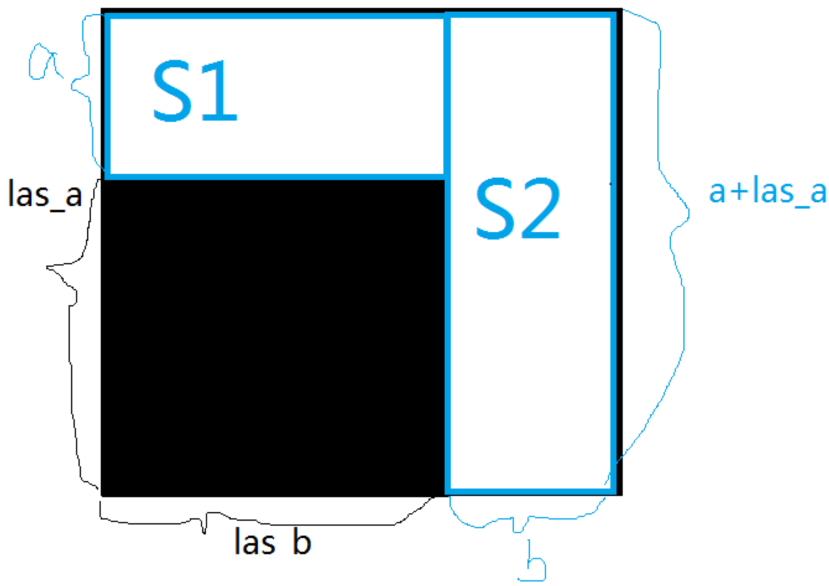
我们把所有 A, B 数组里的值存到一个数组 S 里，然后把 A, B, S 都降序排序，并把 S 去重，然后从大到小枚举值 S_i，把我们的俯视图按照 S_i 的大小分成不同的块，然后根据上面的式子计算。

然后过程就像下面的这张图被染色的过程一样。（从左下角往右上角）



可以发现实际上有些被染色的时候是一个规则的矩形，那么它的方案数就是简单的我们的计算式.

但是有一些又是 "L" 形的，所以要分割来算，如下：



分别计算 S_1,S_2 两个矩形的方案数，然后乘起来，按照式子计算：

$$ans_1=(S^i(S+1)^{a-i}-S^a)^{lasb}$$
$$ans_2=(S^i(S+1)^{a+lasa}-S^{a+lasa})^b$$

但是这样是错误的，因为我们在考虑 S_2 的时候减去的 S^{a+lasa} 已经包含了 S_1 中全部不合法的方案数，所以 S_1 就不用减去 S^a 了，更形式地说就是这样：

$$ans_1=(S^i(S+1)^{a-i})^{lasb}$$
$$ans_2=(S^i(S+1)^{a+lasa}-S^{a+lasa})^b$$

然后最后这个蓝色部分的答案就是 $\sum_{i=0}^{a}C_a^i\times ans1_i\times ans2_i$

细节

上面那个就是最大的细节了，如果还硬要说有的话应该就是组合数 C_n^m 要特判 $m=0$ 的时候组合数等于 1 ，不然会 G。

Code

```
#include<bits/stdc++.h>
#define int long long
#define Hanggoash
using namespace std;
const int maxn=1e6;
const int MOD=1e9+7;
int jc[maxn],inv[maxn];
int A[maxn],B[maxn];
int S[maxn],cnt;
int n;
inline int power(int a,int b)
{
    a%=MOD,b%=MOD-1;
    int ans=1;
    while(b)
    {
        if(b&1)ans=ans*a%MOD;
```

```
        a=a*a%MOD;
        b>>=1;
    }
    return ans;
}
inline void pre()
{
    scanf("%lld",&n);
    for(register int i=1;i<=n;++i)scanf("%lld",&A[i]),S[++cnt]=A[i];
    for(register int i=1;i<=n;++i)scanf("%lld",&B[i]),S[++cnt]=B[i];
    jc[0]=jc[1]=1;
    for(int i=1;i<=n;++i)jc[i]=jc[i-1]*i%MOD;
    inv[n]=power(jc[n],MOD-2);
    for(register int i=n-1;i>=1;--i)inv[i]=inv[i+1]*(i+1)%MOD;
    sort(A+1,A+n+1,greater<int>());
    sort(B+1,B+n+1,greater<int>());
    sort(S+1,S+cnt+1,greater<int>());
    cnt=unique(S+1,S+cnt+1)-S-1;
}
int C(int n,int m)
{
    if(n==m||m==0)return 1;
    return jc[n]*inv[m]%MOD*inv[n-m]%MOD;
}
inline int calc(int a,int b,int las_a,int las_b,int val)
{
//  printf("%lld %lld %lld %lld %lld\n",a,b,las_a,las_b,val);
    int ans=0;
    for(register int i=0;i<=a;++i)
    {
        int c=C(a,i);
        int s1=power(power(val,i)*power(val+1,a+las_a-i)%MOD-
power(val,a+las_a)+MOD,b);
        int s2=power(power(val,i)*power(val+1,a-i),las_b);
        int s=s1*s2%MOD*c%MOD;
        if(i&1)ans=(ans-s+MOD)%MOD;
        else ans=(ans+s)%MOD;
    }
    return ans;
}
signed main()
{
    #ifdef Hanggoash
    freopen("Silhouette.in","r",stdin);
    freopen("Silhouette.out","w",stdout);
    #endif
    pre();
    int las_a=0,las_b=0;
    int F_ans=1;
    for(register int i=1;i<=cnt;++i)
    {
        int a=las_a,b=las_b;
        while(a<n&&A[a+1]==S[i])a++;
        while(b<n&&B[b+1]==S[i])b++;
        F_ans=F_ans*calc(a-las_a,b-las_b,las_a,las_b,S[i])%MOD;
```

```
        las_a=a,las_b=b;
    }
    printf("%lld\n",F_ans%MOD);
    return 0;
}
```

T3 Seat

聪明人才看得到这篇题解

题意

 桤? ? 栎? 恴旹癏戼(H岓癏9?劼 H泮H塇枳? 脲H罄H岓纇壛鑛d 隑H罄H岓餟壛鑠d ?H罄H岓酷壛鐳d ?H罄H岓辪壛?d ?H罄H岓驢壛?

 祕E?E|-婨?E|%婨饗M鍮 ??婨鲂M鍮(??? ?? H菂0]肬H攵H汷0?EH婨H巟 ,E鵞黎 壛H?f? 袗H菂0]肬H攵H汷0?嵱塙D巟 H岓餟壛H?? 泄d H?/? 祕Ei? 婨拎壛玪)搿?睸E ?香?防i? 香?防拎A爈A拎A)繢爈? 香?防?羮弜H菂0]肬H攵嵱H塙L巟 ?]肬H攵H汷0?嵱H塙L巟

 蠬壜H婨H塒H婨鳫?? H壛H亣? H咿t遖8/u贖H黀H墐 @?L9頏埢 ? H= ?u虾 苾 L嫋 H塇 ? 儍8 ? 1译掂嫋 A嫢H= ? H峆H墦 @?@堑 ?? M婡D夆H塇椏? L婨D夆H塇栌? 殂? H嫋 H? 紛 L?脢 ?€ H壜H尬H黀H墐 @?L9頏埢 剋 H= ?u虾 苾 L嫋 H塇 ? 儍8 ? 1译 ? 閅? I婡? 9卤? H娽?:區? M媥L媓H?鰤 ? L媟H婷I婫H?蒡刌 H?趭 H塇钀? I儅 tD夆H塇M娄?4 ? H塇栉? M姓D夆H塇枥? H咗勬? I娄D夆H塇梔3 楣? @ 莽0 棣? I婡?7吀? M婡I婬H?€zc? A?1? H壷L?0? ? L婡L尋螃椓挄8製

A?甥 D夆H塇鑩3 L婨H?麍 ? I婡H? H巃螃劚 ? H巃L尋螃匑 H婨D夆H塇L婡?3 H婨?1咁? H婡儀呧? H婡€8>呌? ? H塇璞? 榱?娲0 呉t菉菂H\A]A^A*肔?I嵡嬻H嚣鑸 L墻 胝L?I嵡嬻H嚣柰 L墻 膗H嫣(H 涫(嬻M婡H嚣?? @鯡H壘(H媜 ujH? L?_ ?H壜H嵡劼 ?*

 H塇?H I姓I壛? H塇?? H巃H哷墈T?? 1簛? f? H塇D嫫(D媜8嫫@嫍P楮 H婼€:I刪 H墲D墶(D塩8墶@墈PH婦8?? ?(琚 L婦(I壛?H塇塁? I爈械? H婰8H呋匃? 婼8;S<嶋? L婥0Lc穛?K?葔S8??

 H塇?? H巃?? 禔<r勀 <p?? H尫H媿H塇桉E E1莡爈箮 H塇柁? I爈鑱? H塇栉? 嵱呟? 壛T1篱脌

 H塇?? H巃楁? €yp叩? H屹H壂€yT鹵 H尫1禶媿耍(;K,峢 Hc睪1纲?Hk?塊(HC H吚勓? ? 壴I爈槔?

 婥湮?? 庡 1韮?濎 湮卄 H?箛 H奤鐵1 吚? €} n? 禘<at<w? 箺 H塇桤= H塇H龔鐻? H 婼I壞?<E羚 <p剳 <i哈? €zI呺? H塇瑗D I壛M觅? H塇钑? I娄I壛? H塇鑲? I姓I壛? H塇鑟? H巃閗? 1鯚?區 H儏簌 H塇鐺= I姓I壛? H塇?? H巃?? H媥H?香 H?婡夵CPH奤鑈0 吚刌 H媥婡湮刟 幀 湮 吋? H媥H? €xc刌 H塇柩C I壞H?\E H奤?0 吚刘 H?/H H奤?0 吚tH?H H奤粸/ 吚叓 H塇瑗 H龔H婥€8I 句 I奤M觅? H塇鑒? I姓I壛? H塇鐿? H巃镵? 吚彺? I姓E1珊4 H塇?? H巃?? H塇?? 嵱壜原? E1 篱~? H塇? I姓I壛? H塇桀? H巃殛? H媥H?mD H?婡夵CPH夰?/ 吚柎 H媥婡湮劉 巒 湮劼 湮u臮?/D H夰梏. 吚劆 €?n卝 禛<at<w匵 箺 H塇鑩; H塇H罄杳? H婼I壞?<E勊 <p剻 <i? €zI? H 塇?B I壛M觅? H塇?? I夬I壛? H塇棂? I姓I壛? H塇桄? I爈鷫? H尫H媿H塇?? H塇H巃梡 H龔H婥€8I? I奤I姓? H塇锠? H巃閶? 禘 <mt<p叩? :E卐? H婥€8*劚 H塇鑚A ? I爈I壛H塇鐪? 鎠? 1鯚?厰 H 儏簌 H塇?: I姓I壛? H塇?? I爈閰? H塇鑪? I爈闆? H塇梵? I爈問? 婥湮几? 幛 1 湮?? H媥H? €xc 刦 H塇栶@ I壞H?PB H夰?- 吚勝 H?#E H夰?- 吚tH?E H夰梭, 吚毱 H塇钢 H罄H婥€8I顛 I夰M觅? H塇鍇? I姓I壛? H塇鐲? I爈椝? 吚?? I姓E1珊4 H塇?? I爈閮? ?<mt<p卄? :G靈? H婥€8刬* H塇桦? ? I爈I 壛H塇柝? ?? H塇杷? H塇H龔枥? H塇I壞壂? ?? H塇?? 鬩? H塇杷 I娄I壛? H塇鑸? I壛樾? ? 峆 蔬?v夵c<嚐? H塇枰? I壞鋼? 箃 H塇鑛8 H龔樾? H尫H媿H塇瑗? H塇H巃鍋 H罄H婥€8I劮 I夰I姓? H塇?? I爈閳? H塇鏴? H巃闍? H塇鏑? 锳? H塇柃> H龔閉? 箃

分析

@?L9頏埢 刓 H=   ?u虾   苾   L嫪 H堾 ? 傻8 ? 1译狭婇H媢M吚tD夆H堾杋? H嫪 H=   劑 H峆H墦 I姓?{D夆苾 {H堾鏞? H嫪 H=   凜 H峆H墦 ?}苾 }閅? M婇M吚tD夆H堾鐱? H儅 厛? H嫵H俩? 喂 H?覇 L?蛦 ? H姓H嬑H亾H壋 D?L9鬣堎 ? H
? I9@嚣 D夆H堾鑵? H嫇 H佲   ? H峄H墐 ?苾 f儅 匀 H?黱 H?飬 ?H壜H尮H嬑H墐 @?H9頏埢 ?? H=   ?u虾   苾   L嫪 H堾 ? 傻8 ? 1译狤?灡 H堾?? 轼? H堾箵 鏢? H媥D夆H堾L婇?? 钱 H堾?? 镚? 儁吘? H婭€9>叡? ? H堾?? H媥H婇H?H媥
奔0 M婇D夆H堾鏞 ?? I媂?厴? I媢?)LD疳y? L媥D夆H堾? H?j H堾铇? ?? 綤 軛? 1? 橐? H婭屜? M媂D夆H堾枛 M娄D夆H堾? 槭? 壔X 奔0 榈? H婇? <0t^<1?? H?}i H堾?? 闀? L9鍁?*产 佲 唬 疽H堾L叐鏢? 閴? ? H堾鍩? L媥槲? H?!i H堾枞? ?? AUATUWVSH冹(A? 冭湮*wKL?k L簟壼H墐 Ic両冞 酘媥 H= 勦 H峆H墦 ? 夲苾 L媥H堾H葩([^]A\A]*榄? H媥 H? L? ?fD H壜H尮H嬑H墐
D?L9頓垡 勷 H= D?u魏 苾 L嫪 H堾 ? 傻8 ? 1译ゝ恷婇閟 € H媥 H?舜 H?郠 ?f? H壜H尮H嬑H墐 @?H9頏埢 剳 H= ?u虾 苾 L嫪 H堾 ? 傻8 ? 1译? H媥 H?hi H?ji ?f? H壜H尮H嬑H墐 @?H9頏埢 ? H= ?u虾 苾 L嫪 H堾 ? 傻8 ? 1译? H媥 H?騢 H?犸 ?f? H壜H尮H嬑H墐 @?H9頏埢 剳 H= ?u虾 苾 L嫪 H堾 ? 傻8 ? 1译? H媄 H佲 劒 H峄H墐 ?苾
H= ? H峆H墦 ?&苾 &@ H葩([^_]A\A]? H媄 H佲 劉 H峄H墐 ?苾 H?鯤 H?駁 ?fD H壜H尮H嬑H墐 @?H9頏埢 t嶩= ?u雍 苾 L嫪 H堾 ? 傻8 ? 1译?€ 冪匛 H媥 H= 凴 H峆H墦 ?
*苾 ? f? H媥 檗? @ H媥 ? @ H媥 H?g H?g ?f? H壜H尮H嬑H墐 @?H9頏埢 劒? H= ?u 虾 苾 L嫪 H堾 ? 傻8 ? 1译? H媥 H? H?╣ ?f? H壜H尮H嬑H墐 @?H9頏埢 ?? H= ?u虾 苾 L嫪 H堾 ? 傻8 ? 1译? €? (t)H媥 H= 劼 H峆H墦 ? 苾 L媥伞H堾H?鶤 H?鳫 栩? H嫪 ?f? H壜H尮H嬑H墐 @?H9頏埢 刞? H= ?u虾 苾 L嫪 H堾 ? 傻8 ? 1译? L媥伞H堾鏞? H嫪 H= 劍 H峆H墦 ?)苾)轸? @ ? 苾 L嫪 H堾 ? 傻8 ? 1篱 ? 苼 L媥 ? 傻8 ? 1篱? ? 苼 L媥 ? 傻8 ? 1议1? ? 苾 L嫪 H堾 ? 傻8 ? 1篱/ ? 苼 L媥 ? 傻8 ? 1议6? ? 苼 L媥 ? 傻8 ? 1篱? ? 苼 L媥 ? 傻8 ? 1篱嚏 fD AWAVAUATUWVSH冹HM吚H壩茾L壖D墵u]雁婥吚 uLL婥?A? u峆鍪?v9H要湮)誇 L媫 H墡 t>湮tS湮th孌H皊鏞? L墻 H?H呟t

Code

瑙 H岼蝰堝?} 閼 H壋H岼瓌堝鏞 ?H壋H岼縟堝?} H岼歑堝鑽 ?H壋H岼疙堝棂| H壸H堝棹 H壋H岼蠬堝? ? H壋H岼逍堝栉| H岼緆堝? ?H壋H岼蝰堝瑛| H壸H堝瓒 忘葩h[]肬SH冹hH崿$€ H嵱 H岼疙堝鎫| H峌疙岼營帀H? H堝鍘 H岼縟堝?| H峌縟岼癐帀H?湽 H堝? H峌歑岼瘮? A? H堝梣? H岼瓌堝鑨 H岼縟堝?| H岼歑堝鑐 H岼疙堝梏{ H?瞰 H?鄂 柩 H媮 H堝? H? H堝杵? H岼蝰堝鑺{ H峌蝰岼繧帀H?琛 H堝鑶 H岼逍堝鑗{ H峌逍岼鑫帀H?取 H堝鑽 H峌緆岼瞽? A? H堝?? H岼蠬堝瑹 H岼逍堝鐵{ H岼緆堝铓 H岼蝰堝?{ 閼 H壋H岼瓌堝鑪 ?H壋H岼縟堝?{ H岼歑堝瀴 ?H壋H岼疙堝桁z H壸H堝桉 H壋H岼蠬堝? ?H壋H岼逍堝枇z H岼緆堝? ?H壋H岼蝰堝瑜z H壸H堝瑭 忘葩h[]肬SH冹8H崿$€ H嵱袮裒E?禘貓E 甀岼疙堝?z H峌疙岼甀帀H壜H婆需@ H岼疙堝鏂z ?H壋H岼疙堝?z H壸H堝? H媥蠬葩8[]肬H攵H冹 ?E? M?C? ?Y硫H,缐罠?1? 袮H葩]肬SH冹8H崿$€ H嵱袮U鉷垙邢嵱鏁岼 堝鑾y H岼 墟H?悫 H婆需? H岼 堝铚y 婨亓?踈鄭鑸鑹E E? ?僻?H?皑 H婆需 傻?媥瑑?吚y迅?)?嚛 媥琀?? H?L? ?Hc蠬??? H?鲃?g? H婆需? 僻?龔H?R? H婆需? 鉎H?C? H婆需? 嚌H?4? H婆需? 隍H?%? H婆需v ?H?? H婆需d ?H?? H婆需R ?H? H婆需@ 惛傻?H?洄 H婆需) 媥?E肄潆1H壋H岼 堝鑠x H壸H堝鑥 H壋H婆需? H壸H堝鑸 H媥蠬葩8[]肬H攵H冹0H?D? 泻 H堝H?;? 蠬垙凲媥鐠? 箽? H堝H?
蠬岼皅堝H?^? 需/? 壈市柳凌睲?)睨袈?uO?? A墟篤UUUD墟麝D墟柳壯)翍??茼堝)置媥旱彡D墟麝龙D墟柳)睨袓媥?袮E隨杳? 髫A墟篤UUUD墟麝D墟柳壯)翍??茼堝)羖E髫A墟旱彡D墟麝龙D墟柳)睨袓媥?袮E鑠? 壈市柳凌睲?)睨袈?uO鑼? A墟篤UUUD墟麝D墟柳壯)翍??茼
? ?? ?H?睞 ?? ?? 堝箽fff壢麝龙壢柳)睨辛???翍秝?斃韧tn?? 壈市柳凌?兑)睨?独A? A? ? 堝H?:? 需蘼 壈市柳凌?兑)睨?独A? A? ? 堝H?? 袈E亇? ?? 荅 ?H?A? 鑺? 僂儅d~闔島盎 H岼颣堝鑐f H岼颕墟 H?? H皊鏈? H岼颣堝鐮f H崀H泮H岼餟堝?f H岼餓墟H?皼 H夰?? H岼餟堝?f H亾H泮H岼酫堝枥e H岼酻墟 H?貧 H夰榧? H岼酫堝梡e H亾H泮H岼駻堝琚e H岼駥墟H?蠺 H夰瓔? H岼駻堝瑄e H亾H泮H岼驢堝鑎e H 岼驣墟H?厲 H夰鐨? H岼驢堝鑪e 荅 隩谷 H?コ

需>? 壩簞fff壢麝漾壢柳)睍辛??翂薂岴癳c稹萎H蠬壩?c H壩栳? 僂 儅 d~ ?? 杷? 荅? 隣媥魤铝?醒鳴猣E 鳴铝?醒鳴蕡罛?? 祕E鼉?咿tH?鉤 鑵? ?H?鍟 鑌? 僂?儅龎~

T4 Ancient

题意

给定 n,g ，求 $g^{\sum_{i=1,i|n}^{n} C_n^{n/i}}$， $n,g\le 10^9$

分析

首先肯定能观察到模数是非常大的，所以一定会用到扩欧降幂，具体来说就是
$a^m\equiv a^{m \mod \phi(p)} \mod p$

然后在这道题里面，模数是一个质数 999911659 ，于是我们指数取模的对象就变成了 999911658 ，这里会发现在计算组合数的时候如果暴力预处理阶乘的话是开不下的，而且时间复杂度同样顶不住，更何况涉及到求一个不是质数的数的欧拉函数。

总结一下，现阶段的问题只有两个：

1. 降低模数的数据范围使得阶乘能够被存下，并且处理阶乘的时间复杂度能够接受。
2. 让模数变成质数，让逆元变得好求一些。

所以我们考虑要求的是什么：

即 $x\equiv \sum_{i=1,i|n}^{n} C_n^{n/i}(\mod 999911658)$ ，然后再是 G^x

发现把 999911658 分解以后是一个 $SquareFreeNumber$ ，即
$999911658=2\times 3\times 4679\times 35617$，而且质因数个数非常少，然后就想到能不能求出以下方程组的一组通解：

$$\begin{cases}x\equiv \sum_{i=1,i|n}^{n} C_n^{n/i}(\mod 2)\\ x\equiv \sum_{i=1,i|n}^{n} C_n^{n/i}(\mod 3)\\ x\equiv \sum_{i=1,i|n}^{n} C_n^{n/i}(\mod 4679)\\ x\equiv \sum_{i=1,i|n}^{n} C_n^{n/i}(\mod 35617)\end{cases}$$

然后因为这几个模数之间都是彼此互质的，所以就一定能得到：

$$x-\sum_{i=1,i|n}^{n} C_n^{n/i}\equiv 0(\mod 2\times 3\times 4679\times 35617=999911658)$$
$$\iff$$
$$x\equiv \sum_{i=1,i|n}^{n} C_n^{n/i}(\mod 2\times 3\times 4679\times 35617=999911658)$$

刚刚提到的方程组，我们用中国剩余定理即可求解。

这样一来，我们便做到了缩小了模数的数据范围的同时把它们变成了质数，再加上卢卡斯定理，就能够在极小范围内求出阶乘然后 $O(p\log_p^n)$ 的效率内求出组合数了。

所以最后求出 x 之后再把模数换成 999911659 求 G^x 的快速幂即可。

Code

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
const int MOD=999911659;
const int maxn=40000;
int prime[]={0,2,3,4679,35617};
inline int power(int a,int b,int p)
{
    int ans=1;
    while(b)
    {
        if(b&1)ans=(ans*a)%p;
        a=a*a%p;
        b>>=1;
    }
    return ans%p;
}
int jc[maxn],ans[maxn];
int F_ans;
inline void pre(int idx){for(register int i=2;i<=prime[idx];++i)jc[i]=jc[i-
1]*i%prime[idx];}
inline int C(int n,int m,int p)
{
    if(n<m)return 0;
    if(n==m)return 1;
    return jc[n]*power(jc[m]*jc[n-m]%p,p-2,p)%p;
}
inline int Lucas(int n,int m,int p)
{
    if(n<m)return 0;
    if(!m)return 1;
    return Lucas(n/p,m/p,p)*C(n%p,m%p,p)%p;
}
inline void CRT()
{
    for(register int i=1;i<=4;++i)
    {
        int M=(MOD-1)/prime[i];
        F_ans=(F_ans+ans[i]*M%(MOD-1)*power(M,prime[i]-2,prime[i]))%(MOD-1);
    }
}
int n,G;
signed main()
{
    #ifdef Hanggoash
    freopen("Ancient.in","r",stdin);
    freopen("Ancient.out","w",stdout);
    #endif
    scanf("%lld%lld",&n,&G);
    jc[0]=jc[1]=1;
    for(register int i=1;i<=4;++i)
    {
        pre(i);
```

```
        for(register int j=1;j*j<=n;++j)
        {
            if(n%j!=0)continue;
            ans[i]=(ans[i]+Lucas(n,j,prime[i]))%prime[i];
            if(j*j!=n)ans[i]=(ans[i]+Lucas(n,n/j,prime[i]))%prime[i];
        }
    }
    CRT();
    cout<<power(G,F_ans,MOD);
    return 0;
}
```

注意

卢卡斯在求组合数的时候有非常大的可能调用 0 的阶乘，所以不止要把 1 的阶乘初始化为 1 ，0 的阶乘也是。

后置知识

卢卡斯定理

内容

当 p 是质数的时候：

$$C_n^m = C_{n/p}^{m/p} \times C_{n \mod p}^{m \mod p} (\mod p)$$

当模数很小，而 C_n^m 中的 n, m 很大的时候，就可以在 $O(p\log_p^n)$ 的时间内求出组合数了。

代码实现

用代码来表示就是：

```
inline int power(int a,int b,int p)
{
    int ans=1;
    while(b)
    {
        if(b&1)ans=(ans*a)%p;
        a=a*a%p;
        b>>=1;
    }
    return ans;
}
inline int C(int n,int m,int p)
{
    if(n<m)return 0;
    if(n==m)return 1;
    return jc[n]*power(jc[m]*jc[n-m]%p,p-2,p)%p;
}
inline int Lucas(int n,int m,int p)
{
    if(n<m)return 0;
    if(!m)return 1;
    return Lucas(n/p,m/p,p)*C(n%p,m%p,p)%p;
}
```

至于证明的话，涉及到生成函数，现阶段背结论就是了

中国剩余定理

内容

求：

$$\begin{cases} x \equiv a_1 (\mod m_1) \\ x \equiv a_2 (\mod m_2) \\ x \equiv a_3 (\mod m_3) \\ x \equiv a_4 (\mod m_4) \\ \ldots\ldots \\ x \equiv a_n (\mod m_n) \end{cases}$$

的一组解。

我们考虑构造一个数 x ，使得对于任意一个 i ，都有上述方程成立。

然后想到每一个数贡献一个答案 ans_i ，最后累加起来成为 x。

构造思路

设当前这个方程组编号是 i ，

1.对于 $j! = i$ ，我们只要让 $ans_j \equiv 0 (\mod m_j)$ ；

2.对于 i ，我们让 $ans_i \equiv a_i (\mod m_i)$

那么就能保证累加起来的答案一定是满足所有方程组的。

构造过程

初始所有 $ans_i = a_i$

对于 1 ，不难想到把每一个 m_i 累乘起来为 M，然后对于当前的，ans_i 要乘的数就是 M/m_i ，记为 M_i。

对于 2 ，我们只要让每个 ans_i 乘上一个 M_i 在模 m_i 意义下的逆元，变成 $a_i \times M_i \times inv_{M_i} (\mod m_i)$

答案

最后的答案 x 就是 $\sum_{i=1}^{n} ans_i = \sum_{i=1}^{n} a_i \times M_i \times inv_{M_i}$

最小非负整数解

用 x 模上 $m = lcm(m_1, m_2, m_3 \ldots m_n)$ 即可。

代码实现

```
inline int CRT()//返回最小非负整数解
{
    long long F_ans=0;
    long long M0=1,LCM=1;
    for(register int i=1;i<=n;++i)M0*=m[i],LCM=lcm(LCM,m[i]);
    for(register int i=1;i<=n;++i)
    {
        M[i]=M0/m[i];
        F_ans=(F_ans+a[i]*M[i]%LCM*power(M[i],m[i]-2,m[i]))%LCM;
    }
    return F_ans%LCM;
}
```

总结

要硬说总结的话，感觉这几天在给 NOIP 攒 rp 了，真的天天暴力都能打挂掉，然后天天保龄，菜得离谱了属于是。

感觉自己很想打暴力，但是又很想打正解，但是自己真的不是那种天天写正解的人（明示某 qmh20061363），要是一下思路不对就 G 了。

另外，感觉自己还是太菜了，不知道为什么看到第一题第一眼是 dp ，幸好没过样例让我发现这个做法是假的。

这么明显的最短路竟然没有看出来。。。

I 真是 SFW（上分王）。

还有呢，就是之前的数学差不多忘了，卢卡斯当时都不会打了，还有 CRT 也是，对我来说完全是新知识，不够牢固。

只不过，菜才是原罪。