A　玉桑的环星球

出题人：爆哥
考察点：数学推导，解方程

首先我们对于这个题，忽略掉第 i 天走 i^2 步这个条件，假定要是一天走一步的话，那么我们可以分析，对于不同的星球，每次回到了自己原来的出生点，那么必然是已经走了 $k * A_i$ $(k \geq 1)$ 步，那么对于 n 个星球的话，他们要同时满足走 X 步回到各自出生点，那么就是 $k_1 * A_1 = k_2 * A_2 ... = k_n * A_n = X$ 那么就发现最小的 $X = LCM(A_1, A2, ..., A_n)$

现在开始用 lcm 代替 $LCM(A_1, A_2, ..., A_n)$

然后继续回到问题，那么我们可以得出结论：走的步数最后一定是这些数的最小公倍数的倍数。

那么我假设最少走了 m 天之后便可以满足他们各自重新回到了出生点，根据平方和公式得出 m 天走了 $\frac{m*(m+1)*(2m+1)}{6}$ 步，那么根据前面推导，我们可以得出 $\frac{m*(m+1)*(2m+1)}{6} = k * lcm$ $(k \geq 1)$

移项之后 $m * (m+1) * (2m+1) = 6 * lcm * k$ $(k \geq 1)$

然后这个时候我们可以得出 $m * (m+1) * (2m+1)$ 是 $6 * lcm$ 的倍数这个结论

然后此时你可以发现 m $m+1$ $2m+1$ 这三个数俩俩互质，这一点可以通过求辗转相减求最大公约数得出。

这个时候我们假设 $6 * lcm$ 一共有 q 个质数，第 i 个质数有 b_i 个那么 $6 * lcm = \prod_{i=1}^{q} p_i^{b_i}$

那么通过上面得出的互质结论，每一个 $p_i^{b_i}$ 是 m $m+1$ $2m+1$ 其中一个的因子

然后题目给定 $lcm <= 10^{12}$ 那么 $6 * lcm < 10^{13}$ 所以 q 最大不超过 11

那么我们可以暴力枚举每一个 $p_i^{b_i}$ 是这三个数中的哪一个因子

假设暴力分配后的为 a_1, a_2, a_3 ，那么就有
$a_1 * k_1 = m, a_2 * k_2 = m+1, a_3 * k_3 = 2m+1$ $(k_1, k_2, k_3 \geq 1)$

那么接下来只要解这三个方程即可

时间复杂度 $O(3^{11}log(\lambda))$

```
#include<bits/stdc++.h>
using namespace std;
using ll=__int128;
ll Lcm(ll c,ll x){
    if(!c) return x;
    return c/__gcd(x,c)*x;
}
ll exgcd(ll a,ll b,ll &x,ll &y){
    if(!b){
        x=1;y=0;
        return a;
    }
    ll x1,y1,d;
    d=exgcd(b,a%b,x1,y1);
    x=y1;y=x1-a/b*y1;
    return d;
}
const int N=25;
ll pr[N];
int cnt[N];
int top;
ll ans;
ll res[3];
void dfs(int id){
    if(id==top){
        ll x,y;
        exgcd(res[0],res[1],x,y);
        ll k1=-x/res[1],k2=y/res[0];
        while(x+res[1]*k1>=0) k1--;
        while(y-res[0]*k2<=0) k2--;
        k1=min(k1,k2);
        ll c=res[0]*(-x)+y*res[1];
        ll a=2ll*res[0]*res[1],b=res[2];
```

```
        ll x1,y1;
        ll d=exgcd(b,a,x1,y1);
        if(c%d) return;
        x1*=c/d;y1*=c/d;
        ll b1=b/d,a1=a/d;
        ll k3=-x1/a1;
        ll k4=(y1-k1)/b1;
        while(y1-k4*b1>k1) k4++;
        while(x1+k3*a1<=0) k3++;
        k1=y1-max(k4,k3)*b1;
        ans=min(ans,-(x+res[1]*k1)*res[0]);
        return;
    }
    for(int i=0;i<3;i++)
    {
        res[i]=res[i]*pr[id];
        dfs(id+1);
        res[i]=res[i]/pr[id];
    }
}

long long x[N];
void solve(){
    int n;
    cin>>n;
    ll lc=0;
    for(int i=1;i<=n;i++){
        cin>>x[i];
        lc=Lcm(lc,x[i]);
    }

    top=0;
    lc*=6;
    ans=lc;
    for(int i=2;1ll*i*i<=lc;i++){
        if(lc%i==0){
            pr[top]=1;
```

```
                cnt[top]=0;
                while(lc%i==0){
                    lc/=i;cnt[top]++;
                    pr[top]*=i;
                }
                top++;
            }
        }
        if(lc>1){
            pr[top]=lc;
            cnt[top]=1;
            top++;
        }
        res[0]=res[1]=res[2]=1;
        dfs(0);
        cout<<(long long)ans<<'\n';
    }


    int main(){
        ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);
        int t=1;
        while(t--){
            solve();
        }
        return 0;
    }
```

B 干饭豪的不定方程

出题人：爆哥
考察点：暴力枚举

对于 $n <= 10^9, a, t, b, k \geq 2$ 那么我们可以发现单暴力枚举 x^y 这种形式不会有很多，所以我们可以先预处理出来这些情况方案之后，在对于每个询问，枚举 b^k 来看之前多少种 $a^t = n - b^k$ 情况

```
/*
干饭豪的a^k+b^t=n
数据未更新
*/
/*
    有多少个四元组(a,t,b,k)满足下面式子
    a^t+b^k=n
    且a,b,t,k>=2
    n<=1e9
*/
#include<bits/stdc++.h>
using namespace std;

void solve(){
    int n;
    cin>>n;
    long long ans=0;
    map<long long,int>mp;
    mp.clear();

    for(int a=2;;a++){
        long long c=1ll*a*a;
        if(c>=n) break;
        for(int t=2;c<n;t++,c=c*a){
            mp[c]++;
        }
    }
    for(int a=2;;a++){
        long long c=1ll*a*a;
```

```
            if(c>=n) break;
            for(int t=2;c<n;t++,c=c*a){
                if(mp.count(n-c)){
                    ans+=mp[n-c];
                }
            }
        }
        cout<<ans<<'\n';
}
int main(){
        ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);
        int t=1;
        cin>>t;
        while(t--){
            solve();
        }
        return 0;
}
/*
*/
```

C　狗头豪的数列求和

出题人：爆哥
考察点：数学推导，矩阵快速幂
本题需要求 $1^k+2^k+...+n^k \quad (n \leq 10^{18})$

发现 n 可能会很大不能正常的通过遍历来求和，那么我们来思考每一位

定义 $sum(x,k)$ 为前 x 项 k 次方的和

首先可以得出 $sum(x,k)=sum(x-1,k)+x^k \quad (x \geq 2)$

发现 $sum(x,k)$ 是用一个递推关系的，那么思考 $(x-1)^k$ 和 x^k 会有什么关系

可以通过高中学过的二项式展开得到 $(x-1)^k=\sum_{i=0}^{k} C_k^i x^i(-1)^{k-i}$

得出 $x^k=(x-1)^k-\sum_{i=0}^{k-1} x^i(-1)^{k-i}$

那么我们发现这显然是一个线性递推关系的

$k<=10$ 那么我们可以维护一个矩阵的信息分别处理 $1,x^1,x^2,...,x^k,sum(x,k)$ 的递推联系

便可以通过矩阵快速幂来解决这道题

时间复杂度 $O(k^3log(n))$

当然，如果你知道拉格朗日插值法，那么这个题就变的很裸，不过要注意这个题模数不一定会有逆元，就看你通过什么方式来处理了。

```

#include<bits/stdc++.h>
using namespace std;
const int N=15;
using ll=long long;
int k,mod;
long long n;
struct mat{
```

```
    int m;
    ll mp[N][N];
    mat(){
        memset(mp,0,sizeof(mp));
    }
    void re(){
        for(int i=0;i<m;i++){
            mp[i][i]=1;
        }
    }
    mat operator *(const mat &w)const{
        mat res;
        res.m=m;
        for(int k=0;k<m;k++){
            for(int i=0;i<m;i++){
                for(int j=0;j<m;j++){
                    res.mp[i][j]+=mp[i][k]*w.mp[k][j]%mod;
                    res.mp[i][j]%=mod;
                }
            }
        }
        return res;
    }
};

mat ksm(mat a,long long b){
    mat res;
    res.m=a.m;
    res.re();
    while(b){
        if(b&1) res=res*a;
        a=a*a;
        b>>=1;
    }
    return res;
}

```

```
ll c[N][N];

void solve(){
    cin>>n>>k>>mod;
    int m=k+2;
    for(int i=0;i<=m;i++){
        for(int j=0;j<=i;j++){
            if(i==j||j==0){
                c[i][j]=1;
            }
            else{
                c[i][j]=c[i-1][j-1]+c[i-1][j];
                if(c[i][j]>=mod) c[i][j]-=mod;
            }
        }
    }
    mat a;
    a.m=m;
    for(int j=m-1;j>=1;j--){
        a.mp[j][j]=1;
        int k1=k-j+1;
        for(int i=j+1,k2=k1-1,f=1;i<m;i++,k2--,f=mod-f){
            for(int p=i;p<m;p++){
                a.mp[p][j]+=f*c[k1][k2]%mod*a.mp[p][i];
                a.mp[p][j]%=mod;
            }
        }
    }
    a.mp[0][0]=1;
    for(int i=1;i<m;i++){
        a.mp[i][0]=a.mp[i][1];
    }
    mat res;
    res.m=m;
    for(int j=0;j<m;j++){
        res.mp[0][j]=1;
    }
```

```
    res=res*ksm(a,n-1);
    cout<<res.mp[0][0]<<'\n';
}

int main(){
    ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);
    int t=1;
    while(t--){
        solve();
    }
    return 0;
}
```

D 草莓豪的斐波那契

出题人：爆哥
考察点：思维

可以通过观察斐波那契序列值是指数形势增长，所以直接模拟即可

```
#include<bits/stdc++.h>
using namespace std;
void solve(){
    long long n;
    cin>>n;
    long long now=0;
    long long a=1,b=1;
    int t=0;
    while(now<n){
        now+=b;
        t++;
        a=a+b;
        swap(a,b);
    }
    cout<<t<<'\n';
}
int main(){
    ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);
    int t=1;
    while(t--){
        solve();
    }
    return 0;
}
```

E 小刻觉得应该来个平衡树

小刻觉得应该来个平衡树

** 宣传一下我的模板库：**

[Release ACM 模板 · zxyDEDIRE/ACM-Code-Library (github.com)](https://github.com/zxyDEDIRE/ACM-Code-Library/releases/tag/QAQ)

[zxyDEDIRE/ACM-Code-Library at QAQ (github.com)](https://github.com/zxyDEDIRE/ACM-Code-Library/tree/QAQ)

有 LaTeX 源码也有 PDF 版本