A - A/B

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll ans=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return ans;
}

ll inv(ll a,ll mod)//存在逆元条件：gcd(a,mod)=1
{
    ll x,y;
    ll g=ex_gcd(a,mod,x,y);
    if(g!=1)return -1;
    return (x%mod+mod)%mod;
}

int main()
{
    ll T;
    scanf("%lld",&T);
    while(T--)
    {
        ll a,b;
        scanf("%lld%lld",&a,&b);
        printf("%lld\n",a*inv(b,9973)%9973);
    }
    return 0;
}
```

B - 青蛙的约会

根据题意可以得到同余方程$x+mt\equiv y+nt(mod\ L)$
化简得$(m-n)t\equiv y-x(mod\ L)$
由于$gcd(m-n,L)$不一定等于1，所以可能出现没有逆元的情况，就要化成，二元一次方程。
$(m-n)t+Lk=y-x$

然后用拓展欧几里得求解。

```
#include<bits/stdc++.h>
```

```
using namespace std;
typedef long long ll;

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll ans=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return ans;
}

int main()
{
    ll x,y,m,n,L;
    scanf("%lld%lld%lld%lld%lld",&x,&y,&m,&n,&L);
    ll a=m-n;
    ll b=L;
    ll c=y-x;
    ll x0,y0;
    ll g=ex_gcd(a,b,x0,y0);
    if(c%g!=0)
    {
        printf("Impossible\n");
        return 0;
    }
    x0*=c/g;
    b/=g;
    if(b<0)b=-b;
    x0=(x0%b+b)%b;
    printf("%lld\n",x0);
    return 0;
}
```

C - Strange Way to Express Integers

同余方程组，模数不要求两两互质，所以用扩展中国剩余定理。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

ll fmul(ll a,ll b,ll mod)
{
    ll sum=0,base=(a%mod+mod)%mod;
    while(b)
    {
        if(b%2)sum=(sum+base)%mod;
        base=(base+base)%mod;
        b/=2;
    }
```

```
    return sum;
}

ll gcd(ll a,ll b)
{
    return b?gcd(b,a%b):a;
}

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll g=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return g;
}

ll a[100005],m[100005];
ll ex_crt(ll *a,ll *m,ll n)//长度为0到n-1
{
    ll M=1,c=0;
    for(int i=0;i<n;i++)
    {
        ll t=(a[i]%m[i]-c%m[i]+m[i])%m[i];
        ll x,y;
        ll g=ex_gcd(M,m[i],x,y);
        if(t%g!=0)return -1;
        ll tM=M;
        M*=m[i]/gcd(M,m[i]);
        x=fmul(t/g,(x%M+M)%M,M);
        c=(c%M+fmul(x,tM,M)+M)%M;
    }
    return (c%M+M)%M;
}

int main()
{
    ll n;
    while(scanf("%lld",&n)!=EOF)
    {
        for(int i=0;i<n;i++)
        {
            scanf("%lld%lld",&m[i],&a[i]);
        }
        printf("%lld\n",ex_crt(a,m,n));
    }
    return 0;
}
```

D - Discrete Logging

指数同余方程，保证底数和模数互质
BSGS模板题

```
#include <stdio.h>
#include <unordered_map>
#include <math.h>
using namespace std;
typedef long long ll;
const ll mod=1e9+7;

ll fpow(ll a,ll n,ll mod)
{
    ll sum=1,base=a%mod;
    while(n!=0)
    {
        if(n%2)sum=sum*base%mod;
        base=base*base%mod;
        n/=2;
    }
    return sum;
}

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll ans=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return ans;
}

ll inv(ll a,ll mod)//存在逆元条件：gcd(a,mod)=1
{
    ll x,y;
    ll g=ex_gcd(a,mod,x,y);
    if(g!=1)return -1;
    return (x%mod+mod)%mod;
}

ll BSGS(ll a,ll b,ll p)
{
    b%=p;
    if(b==1||p==1)return 0;
    ll n=sqrt(p);
    static unordered_map<ll,ll>Bmp;
    Bmp.clear();
    ll inva=inv(fpow(a,n-1,p),p)*b%p;
    for(ll i=n-1;i>=0;i--)
    {
        Bmp[inva]=i;inva=inva*a%p;
    }
    ll ta=1,powa=fpow(a,n,p);
```

```
    for(ll k=0;k<=p;k+=n)
    {
        if(Bmp.count(ta))return k+Bmp[ta];
        ta=ta*powa%p;
    }
    return -1;
}

int main()
{
    ll p,b,n;
    while(scanf("%lld%lld%lld",&p,&b,&n)!=EOF)
    {
        ll ans=BSGS(b,n,p);
        if(ans==-1)printf("no solution\n");
        else printf("%lld\n",ans);
    }
    return 0;
}
```

E - Power Modulo Inverted

指数同余方程，不要求互质
扩展BSGS

```
#include <bits/stdc++.h>
#include <unordered_map>
using namespace std;
typedef long long ll;
const ll mod=1e9+7;

ll fpow(ll a,ll n,ll mod)
{
    ll sum=1,base=a%mod;
    while(n!=0)
    {
        if(n%2)sum=sum*base%mod;
        base=base*base%mod;
        n/=2;
    }
    return sum;
}

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll ans=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return ans;
```

```
}

ll inv(ll a,ll mod)//存在逆元条件：gcd(a,mod)=1
{
    ll x,y;
    ll g=ex_gcd(a,mod,x,y);
    if(g!=1)return -1;
    return (x%mod+mod)%mod;
}

ll gcd(ll a,ll b)
{
    return b?gcd(b,a%b):a;
}

ll BSGS(ll a,ll b,ll p)
{
    b%=p;
    if(b==1||p==1)return 0;
    ll n=sqrt(p);
    static unordered_map<ll,ll>Bmp;
    Bmp.clear();
    ll inva=inv(fpow(a,n-1,p),p)*b%p;
    for(ll i=n-1;i>=0;i--)
    {
        Bmp[inva]=i;inva=inva*a%p;
    }
    ll ta=1,powa=fpow(a,n,p);
    for(ll k=0;k<=p;k+=n)
    {
        if(Bmp.count(ta))return k+Bmp[ta];
        ta=ta*powa%p;
    }
    return -1;
}

ll exBSGS(ll a,ll b,ll p)
{
    b%=p;
    if(a==0&&b==0)return 1;
    else if(a==0&&b!=0)return -1;
    if(b==1||p==1)return 0;
    ll d=gcd(a,p);
    if(b%d!=0)return -1;
    p=p/d;
    b=b/d*inv(a/d,p)%p;
    if(d!=1)
    {
        ll ans=exBSGS(a,b,p);
        if(ans==-1)return -1;
        return ans+1;
    }
    ll ans=BSGS(a,b,p);
    if(ans==-1)return -1;
    return ans+1;
}

int main()
```

```
{
    ll a,p,b;
    while(scanf("%lld%lld%lld",&a,&p,&b)!=EOF)
    {
        if(a==0&&p==0&&b==0)break;
        ll ans=exBSGS(a,b,p);
        if(ans==-1)printf("No Solution\n");
        else printf("%lld\n",ans);
    }
    return 0;
}
```

F - Xiao Ming's Hope

题意：给定一个n，对于i从0到n，求有多少个C_n^i为奇数。

为奇数就是$C_n^i \equiv 1(mod\ 2)$。

考虑卢卡斯定理$C_n^i = C_{n/2}^{i/2} C_{n\%2}^{i\%2}$

那么有$C_1^0 = 1, C_1^1 = 1, C_0^1 = 0, C_0^0 = 1$

我们不断用卢卡斯定理拆分C_n^i，发现就是上下都拆分成对应的二进制位。

那么要使它为奇数，要满足下面为0的位置，上面也要是0，而下面为1的位置，上面可以是任意的数。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

int main()
{
    int n;
    while(scanf("%d",&n)!=EOF)
    {
        int ans=0;
        while(n!=0)
        {
            if(n&1)ans++;
            n>>=1;
        }
        printf("%d\n",1<<ans);
    }
    return 0;
}
```

G - Unknown Treasure

题意：求组合数取模，不过模数是由多个不同的素数相乘。
做法：求出组合数在每个模数下的值，最后用中国剩余定理合并。

```
#include <bits/stdc++.h>
using namespace std;
```

```
typedef long long ll;

ll fpow(ll a,ll n,ll mod)
{
    ll sum=1,base=a%mod;
    while(n!=0)
    {
        if(n%2)sum=sum*base%mod;
        base=base*base%mod;
        n/=2;
    }
    return sum;
}

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll ans=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return ans;
}

ll inv(ll a,ll mod)//存在逆元条件：gcd(a,mod)=1
{
    ll x,y;
    ll g=ex_gcd(a,mod,x,y);
    if(g!=1)return -1;
    return (x%mod+mod)%mod;
}

ll jie[1000005],rjie[1000005];
void init_jie(ll n,ll mod)
{
    jie[0]=1;
    for(ll i=1;i<=n;i++)jie[i]=jie[i-1]*i%mod;
    for(ll i=0;i<=n;i++)rjie[i]=inv(jie[i],mod);
}
ll Lucas(ll n,ll k,ll mod)//返回n取k对mod取模
{
    if(n<k)return 0;
    if(n>=mod)return Lucas(n/mod,k/mod,mod)*Lucas(n%mod,k%mod,mod)%mod;
    else return jie[n]*rjie[n-k]%mod*rjie[k]%mod;
}

ll fmul(ll x,ll y,ll mod)
{
    ll tmp=(x*y-(ll)((long double)x/mod*y+1.0e-8)*mod);
    return tmp<0?tmp+mod:tmp;
}

ll a[100005],m[100005];
ll crt(ll *a,ll *m,ll n)//长度为0到n-1
```

```
{
    ll M=1;
    for(int i=0;i<n;i++)M=M*m[i];
    ll ans=0;
    for(int i=0;i<n;i++)
    {
        ll MM=M/m[i];
        ans=(ans+fmul(fmul(a[i],MM,M),inv(MM,m[i]),M))%M;
    }
    return ans;
}

int main()
{
    int T;
    scanf("%d",&T);
    while(T--)
    {
        ll n,mm,k;
        scanf("%lld%lld%lld",&n,&mm,&k);
        for(ll i=0;i<k;i++)scanf("%lld",&m[i]);
        for(ll i=0;i<k;i++)
        {
            init_jie(2*m[i],m[i]);
            a[i]=Lucas(n,mm,m[i]);
        }
        printf("%lld\n",crt(a,m,k));
    }
    return 0;
}
```

H - Fansblog

题意：给定一个质数P，找到最大的小于P的质数Q。然后求Q!(mod P)的值。
做法：根据素数密度定理，可以暴力找到Q，由于是大数判断素数，用素数测试。然后可以根据威尔逊定理知道(P-1)!(mod P)的值为P-1,然后只要把(P-1)!比Q!多乘的数全部取逆元乘上。注意模数较大，可能爆long long，所以要用快速乘。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

ll mul(ll a,ll b,ll mod){
    ll ret=0;
    while(b) {
        if(b & 1) ret=(ret+a)%mod;
        a=(a+a)%mod;
        b >>= 1;
    }
    return ret;
}

ll pow(ll a,ll b,ll mod) {
    ll ret = 1;
```

```
    while(b) {
        if(b & 1) ret = mul(ret,a,mod);
        a = mul(a,a,mod);
        b >>= 1;
    }
    return ret;
}
bool check(ll a,ll n){
    ll x = n - 1;
    int t = 0;
    while((x & 1) == 0) {
        x >>= 1;
        t ++;
    }
    x = pow(a,x,n);
    ll y;
    for(int i=1;i<=t;i++) {
        y = mul(x,x,n);
        if(y == 1 && x != 1 && x != n - 1) return true;
        x = y;
    }
    if(y != 1) return true;
    return false;
}
bool Miller_Rabin(ll n) {
    if(n == 2) return true;
    if(n == 1 || !(n & 1)) return false;
    const int arr[12] = {2,3,5,7,11,13,17,19,23,29,31,37};
    for(int i = 0; i < 12; i++) {
        if (arr[i] >= n) break;
        if(check(arr[i], n)) return false;
    }
    return true;
}

ll ex_gcd(ll a,ll b,ll& x,ll& y)
{
    if(b==0)
    {
        x=1;y=0;
        return a;
    }
    ll ans=ex_gcd(b,a%b,x,y);
    ll tmp=x;
    x=y;
    y=tmp-a/b*y;
    return ans;
}

ll inv(ll a,ll mod)//存在逆元条件：gcd(a,mod)=1
{
    ll x,y;
    ll g=ex_gcd(a,mod,x,y);
    if(g!=1)return -1;
    return (x%mod+mod)%mod;
}

int main()
```

```
{
    int T;
    scanf("%d",&T);
    while(T--)
    {
        ll P;
        scanf("%lld",&P);
        ll Q;
        for(Q=P-1;;Q--)
        {
            if(Miller_Rabin(Q))break;
        }
        ll ans=P-1;
        for(ll i=Q+1;i<=P-1;i++)
        {
            ans=mul(ans,inv(i,P),P);
        }
        printf("%lld\n",ans);
    }
    return 0;
}
```

I - Super A^B mod C

由于B巨大，所以用欧拉降幂，先求出$B(mod\ \phi(C))$的值，然后用快速幂。

```
#include <iostream>
#include <cmath>
#include <cstring>
using namespace std;
typedef __int64 ll;

ll fpow(ll a,ll n,ll mod)
{
    ll sum=1;
    while(n)
    {
        if(n&1)sum=sum*a%mod;
        a=a*a%mod;
        n>>=1;
    }
    return sum;
}

ll phi(ll n)
{
    ll i,rea=n;
    for(i=2;i*i<=n;i++)
    {
        if(n%i==0)
        {
            rea=rea-rea/i;
            while(n%i==0)
                n/=i;
         }
```

```
    }
    if(n>1)
        rea=rea-rea/n;
    return rea;
}
char b[1000005];

int main()
{
    ll a,c;
    while(scanf("%I64d%s%I64d",&a,&b,&c)!=EOF)
    {
        ll len=strlen(b);
        ll p=phi(c);
        ll sum=0;
        for(ll i=0;i<len;i++)
        {
            sum=(sum*10+(b[i]-'0'))%p;
        }
        printf("%I64d\n",fpow(a,sum+p,c));
    }
    return 0;
}
```

J - 求和

防AK，题解暂无。

K - GCD

莫比乌斯反演模板题。

```
#include<bits/stdc++.h>
#include<unordered_map>
using namespace std;
typedef long long ll;
const ll mod = 1e9+7;
typedef pair<int,int> P;
const ll MAXN=100000;

inline ll read()
{
   ll X=0,w=0; char ch=0;
   while(!isdigit(ch)) {w|=ch=='-';ch=getchar();}
   while(isdigit(ch)) X=(X<<3)+(X<<1)+(ch^48),ch=getchar();
   return w?-X:X;
}
inline void print(ll x)
{
   if(x<0){putchar('-');x=-x;}
```

```
    if(x>9) print(x/10);
    putchar(x%10+'0');
}

unordered_map<ll,ll>mpSmu;
ll prime[MAXN+10],notPrime[MAXN+10],mu[MAXN+10],tot;
ll Smu[MAXN+10];
void initMu(ll n)
{
    notPrime[1]=mu[1]=1;
    for(ll i=2;i<=n;i++)
    {
        if(!notPrime[i])prime[tot++]=i,mu[i]=-1;
        for(ll j=0;j<tot&&i*prime[j]<=n;j++)
        {
            notPrime[i*prime[j]]=1;
            if(i%prime[j])mu[i*prime[j]]=-mu[i];
            else {mu[i*prime[j]]=0;break;}
        }
    }
    for(ll i=1;i<=n;i++)Smu[i]=Smu[i-1]+mu[i];
}

ll getSmu(ll n)
{
    if(n<=MAXN)return Smu[n];
    if(mpSmu[n])return mpSmu[n];
    ll ans=1;
    for(ll l=2,r;l<=n;l=r+1)
    {
        r=n/(n/l);
        ans-=getSmu(n/l)*(r-l+1);
    }
    return mpSmu[n]=ans;
}

ll work(ll a,ll b,ll c,ll d,ll k)
{
    if(k==0)return 0;
    if(a>b||c>d)return 0;
    a=(a+k-1)/k;b=b/k;
    c=(c+k-1)/k;d=d/k;
    ll ans=0;
    for(ll g=1;g<=max(b,d);g++)
    {
        ans=ans+mu[g]*(b/g-(a+g-1)/g+1)*(d/g-(c+g-1)/g+1);
    }
    return ans;
}

int main()
{
    initMu(MAXN);
    ll T;
    scanf("%lld",&T);
    for(ll cas=1;cas<=T;cas++)
    {
        ll a,b,c,d,k;
```

```
        scanf("%lld%lld%lld%lld%lld",&a,&b,&c,&d,&k);
        ll ans=work(a,b,c,d,k)-work(max(a,c),min(b,d),max(a,c),min(b,d),k)/2;
        printf("Case %lld: ",cas);
        printf("%lld\n",ans);
    }
    return 0;
}
```