**Pdev数论模板**

**Ver 2.0 Alpha**

**Content：**

**---------数论部分----------**

**1. 同余**

**2. 逆元**

**3. CRT（解同余方程组）**

**4. gcd\extend\_gcd（解同余方程、解ax+by=c）**

**5. 欧拉函数**

**6. 唯一分解定理**

**7. 莫比乌斯反演**

**---------矩阵部分-------------**

1. **矩阵的基本运算、矩阵快速幂**
2. **高斯消元**

**---------博弈论部分------------**

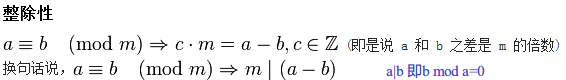
**---------概率论部分------------**

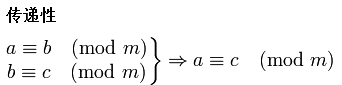
# A.数论部分

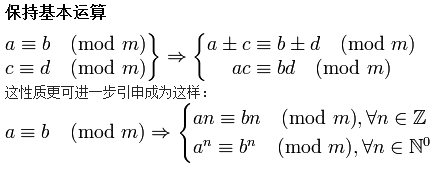
## 1.同余

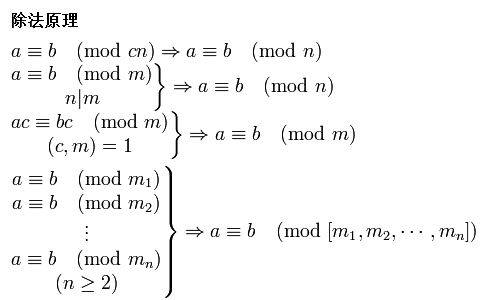
**同余：a≡b (mod m)，表示a % m==b % m**

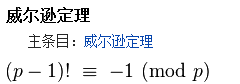
**同余式的运算法则：**

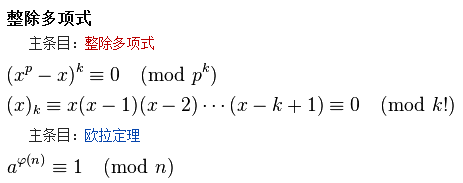


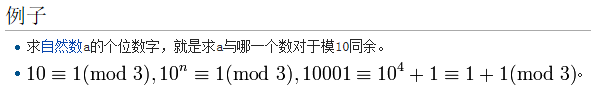












## 2.逆元

**HDU 1576**

**逆元的定义：例如a\*b≡1 (mod m)，则b就是a关于m的逆元。**

**若a和b互为逆元，则(x/a) mod m=(x\*b) mod m**

**同余方程中，若a\*x≡b (mod n)，左乘a的逆元A，得x≡A\*b(mod n)**

**求逆元方法也很简单，用扩展欧几里得解这个方程即可。**

void calc\_inv(LL n) //求1--n关于MOD的逆元，inv[i]=i的逆元

{

inv[1] = 1;

for(int i=2;i<n;i++)

{

if(i >= MOD) break;

inv[i] = (MOD - MOD / i) \* inv[MOD % i] % MOD;

}

}

## 3. 中国剩余定理

**POJ 1006**

**中国剩余定理：求解同余方程组**

**n≡a[1] (mod m[1])**

**n≡a[2] (mod m[2])**

**......**

**n≡a[i] (mod m[i])**

**该方程组有解的条件：m[i]两两互质**

int CRT(int a[],int m[],int n)

//中国剩余定理，返回方程x≡a[1..n] (mod m[1..n])的解x

{

int M=1;

for (int i=1;i<=n;i++) M\*=m[i];

int ret=0;

for (int i=1;i<=n;i++)

{

int x,y;

int tm=M/m[i];

extend\_gcd(tm,m[i],x,y);

ret=(ret+tm\*x\*a[i])%M;

}

return (ret+M)%M;

}

## 4.欧几里得：gcd、extern\_gcd

**POJ 1061、2142、2891（迭代解同余方程组）**

**对于同余方程A≡B (mod L)，**

**令A=x\*L+m，B=y\*L+m，稍微YY下即可转化为x\*L-y\*L=A-B，即ax+by=c的形式。**

**对于方程ax+by=c**

**设tm=gcd(a,b)**

**若c%tm!=0，则该方程无整数解。STOP**

**否则，先用扩展欧几里得求出方程ax+by=tm的解x0、y0**

**然后有a\*x0+b\*y0=tm**

**令x1=x0\*(c/tm)，y1=y0\*(c/tm)**

**则a\*x1+b\*y1=c**

**x1、y1即原方程的一个特解**

**这个方程的通解：xi=x1+k\*(b/m)，yi=y1-k\*(a/m)**

**另：如果要求yi的最小非负解？令r=a/tm，则解y2=(y1%r+r)%r**

**其他类似的要求自己用纸算一算，YY一下就行了**

int gcd(int a,int b){ //辗转相除法，返回gcd(a,b)

if (b==0) return a;

return gcd(b,a%b);

}

int extend\_gcd(int a,int b,int &x,int &y){

//扩展欧几里得，求解a\*x+b\*y=gcd(a,b)，返回值为gcd(a,b)

if (b==0){

x=1;y=0;

return a;

}

else{

int r=extend\_gcd(b,a%b,y,x);

y=y-x\*(a/b);

return r;

}

}

## 5.欧拉函数

**关于gcd的8题.txt**

**·欧拉函数：一般记作φ(n)，表示1-n中与n互质的数的数量。**

**·欧拉函数是积性函数，即φ(m\*n)=φ(m)\*φ(n)**

**//这条定理基友面试时还遇到了= =**

**·欧拉函数的值φ(n)=n\*(1-p[1])\*(1-p[2])\*...\*(1-p[n])**

**//p[i]是小于等于n的所有素数**

**·若n是m的倍数，则小于等于n且与m互质的数的个数为(n/m)\*φ(m)**

**//证明不难理解：设k小于等于m且与m互质，则k+m、k+2m......也与m互质**

**·若n是质数p的k次幂，则φ(n)=(p-1)\*(p^(k-1))**

**//关于欧拉函数wikipedia上讲的很详细，此处不赘述了**

void calc\_phi(int n)

//求1--n的欧拉函数，phi[i]=φ(i)

{

for (int i=2;i<=n;i++)

phi[i]=0;

phi[1]=1;

for (int i=2;i<=n;i++)

if (!phi[i])

for (int j=i;j<=n;j+=i)

{

if (!phi[j]) phi[j]=j;

phi[j]=phi[j]/i\*(i-1);

}

}

## 6.整数的唯一分解定理：

**POJ 1845**

**任意正整数都有且只有一种方式写出其素因子的乘积表达式。**

**A=(p1^k1)\*(p2^k2)\*(p3^k3)\*....\*(pn^kn) 其中pi均为素数**

# B．矩阵部分

## 0．矩阵的基本运算、矩阵快速幂

typedef unsigned long int ULL;

typedef vector<ULL> vec;

typedef vector<vec> mat;

const ULL P=9973;

mat mul(mat &A,mat &B) //return A\*B

{

mat C(A.size(),vec(B[0].size()));

for (int i=0;i<(int)A.size();i++)

{

for (int k=0;k<(int)B.size();k++)

{

for (int j=0;j<(int)B[0].size();j++)

{

C[i][j]=(C[i][j]+A[i][k]\*B[k][j])%P;

}

}

}

return C;

}

mat m\_pow(mat A,int m) //return A^m

{

mat B(A.size(),vec(A.size()));

for (int i=0;i<(int)A.size();i++)

B[i][i]=1;

while (m>0)

{

if (m&1) B=mul(B,A);

A=mul(A,A);

m>>=1;

}

return B;

}

int main()

{

cin>>n>>m;

mat A(n,vec(n));

for (int i=0;i<n;i++)

for (int j=0;j<n;j++)

cin>>A[i][j];

A=m\_pow(A,m);

for (int i=0;i<n;i++)

{

for (int j=0;j<n;j++)

cout<<A[i][j]<<" ";

cout<<endl

}

return 0;

}