//快速乘法

int qmul(int a,int b){// 根据数据范围可选择long long

int ans=0;

while(b){

if( b&1)ans+=a;//按位与完成位数为1的判断

b>>=1;a<<=1;//位运算代替/2和\*2

}return ans;}

//快速乘法取模

int qmul\_mod(int a,int b,int mod){

int ans=0;

while(b){

if((b%=mod)&1)ans+=a%=mod;//这里需要b%=mod 以及a%=mod

b>>=1;a<<=1;

}return ans%mod; //ans也需要对mod取模}

//快速幂 a^b

int qpow(int a,int b){

if(a==0)return 0;//这是个坑，校赛被坑过，很多网上的实现都没写这一点

int ans=1;

while(b){

if(b&1)ans\*=a;//和快速乘法的区别

b>>=1;a\*=a;//区别，同上

}return ans;}

int qpow\_mod(int a,int b,int mod){

if(a==0)return 0;

int ans=1;

while(b){

if(b&1)ans=(ans%mod)\*(a%mod);//如果确定数据不会爆的话，可写成 ans\*=a%=mod;

b>>=1;a\*=a%=mod;//等价于a=(a%mod)\*(a%mod)，且将一个模运算通过赋值代替，提高了效率

}

return ans%mod;//数据不会爆的话，这里的%运算会等价于第5中不断重复的 ans%mod

}

ll poww(ll a, ll b, ll mod)

{

ll ans = 1, base = a;

while( b )

{

if( b & 1 ) /// you 1

ans \*= base;

base \*= base;/// 基数

b >>= 1;/// 除二}return ans;}