// Pollard\_Rho大数质因数分解

inline LL mod(LL x,LL MO) {while (x>=MO) x-=MO;while (x<0) x+=MO;return x;}

inline LL gcd(LL x,LL y) {return y==0?x:gcd(y,x%y);}

inline LL myabs(LL x) {return x>=0?x:-x;}

inline LL quick\_mul(LL x,LL y,LL MO)

{

x%=MO;LL res=0;

while (y)

{

if (y&1) res=mod(res+x,MO),y--;

x=mod(x+x,MO);y>>=1;

}

return res;

}

inline LL quick\_pow(LL x,LL y,LL MO)

{

x%=MO;LL res=1;

while (y)

{

if (y&1) res=quick\_mul(res,x,MO),y--;

x=quick\_mul(x,x,MO);y>>=1;

}

return res;

}

int lp[]={2,3,5,7,11,13,17,19,23,29,31,37};

LL x[148];

inline bool Miller\_Rabin(LL n)

{

int i;

for (i=0;i<=11;i++)

{

if (n==lp[i]) return true;

if (n%lp[i]==0) return false;

}

LL tmp=n-1;int k=0,ti=20;

while (!(tmp&1)) k++,tmp>>=1;

while (ti--)

{

LL a=rand()%(n-2)+2;

x[0]=quick\_pow(a,tmp,n);

for (i=1;i<=k;i++)

{

x[i]=quick\_mul(x[i-1],x[i-1],n);

if (x[i]==1 && x[i-1]!=1 && x[i-1]!=n-1) return false;

}

if (x[k]!=1) return false;

}

return true;

}

inline LL pollard\_rho(LL n,LL c)

{

LL x=rand()%(n-1)+1,y=x,i=1,k=2;

while (true)

{

i++;

x=mod(quick\_mul(x,x,n)+c,n);

LL p=gcd(myabs(x-y),n);

if (p!=1 && p!=n) return p;

if (y==x) return n;

if (i==k)

{

y=x;

k<<=1;

}

}

return n;

}

//fac里面存了所有的因子，要获得次数要再写一个函数

vector<LL> fac;

inline void findfac(LL n)

{

if (n==1) return;

if (Miller\_Rabin(n)) {fac.pb(n);return;}

LL p=n;

while (p>=n) p=pollard\_rho(n,rand()%(n-1));

findfac(p);findfac(n/p);

}