/\*

Towfiqul Islam

CSE, IIUC.

Bangladesh.

\*/

/\*\*\*\*\*\*\* header files \*\*\*\*\*\*\*/

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

#include <cstdio>

#include <cstring>

#include <cmath>

#include <algorithm>

#include <map>

#include <queue>

#include <stack>

#include <vector>

#include <deque>

#include <functional>

#include <string>

#include <iostream>

#include <cctype>

#include <set>

#include <climits>

#include <iomanip>

#include <cassert>

#include <unordered\_map>

/\*\*\*\*\*\*\* header files end \*\*\*\*\*\*\*/

/\*\* sieve prime \*\*/

#define M 100000000

int marked[M/64 + 2];

#define on(x) (marked[x/64] & (1<<((x%64)/2)))

#define mark(x) marked[x/64] |= (1<<((x%64)/2))

void sieve(int n)

{ for (int i = 3; i \* i < n; i += 2) {

if (!on(i)) {

for (int j = i \* i; j <= n; j += i + i) {

mark(j); } } } }

/\*\* sieve normal\*\*/

bool marked[M];

void sieve2(int n){

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

if (marked[i] == false) // i is a prime {

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

marked[j] = true; } } } }

/\*\* Divisor \*\*/

vector<int> primes; // we'll preload primes once at the beginning

int countDivisor(int n){

int divisor = 1;

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

if (n % primes[i] == 0) {

int cnt = 1;

while (n % primes[i] == 0) {

n /= primes[i];

cnt++; }

divisor \*= cnt; } }

return divisor; }

/\*\* GCD \*\*/

int gcd(int a, int b){

return b == 0 ? a : gcd(b, a % b); }

/\*\*extended euclid where ax+by = gcd (a,b) \*\*/

typedef pair<int, int> pii;

#define fs first

#define se second

pii extendedEuclid(int a, int b) /\*\*returns x, y | ax + by = gcd(a,b)\*\*/

{ if(b == 0) return pii(1, 0); else { pii d = extendedEuclid(b, a % b); return pii(d.se, d.fs - d.se \* (a / b)); } }

/\*\* LCM \*\*/

int lcm(int a, int b){

return (a / gcd(a, b)) \* b;}

/\*\*Modular Inverse \*\*/

int modularInverse(int a, int n) {

pii ret = extendedEuclid(a, n);

return ((ret.fs % n) + n) % n;}

/\*\* euler toient function \*\*/

int phi[M];

void calculatePhi(){

for (int i = 1; i < M; i++) {

phi[i] = i; }

for (int p = 2; p < M; p++) {

if (phi[p] == p) // p is a prime {

for (int k = p; k < M; k += p) {

phi[k] -= phi[k] / p; } } } }

/\*\* euler toient function 2\*\*/

int phi(int n){

int ret = n;

for (int i = 2; i \* i <= n; i++) {

if (n % i == 0) {

while (n % i == 0) {

n /= i; }

ret -= ret / i; } } /\*\* this case will happen if n is a prime number in that case we won't find any prime that divides n that's less or equal to sqrt(n) \*\*/

if (n > 1) ret -= ret / n; return ret; }

/\*\*n factorial mod m\*\*/

int nFactModm(int n,int m) {

int factorial = 1;

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

factorial \*= i;

factorial %= m; }

return factorial;}

/\*\* n sum mod m\*\*/

int nSumModm(int n,int m) {

int sum = 0;

for (int i = 1; i <= n; i++) { sum += i; sum %= m; } return sum; }

/\*\*\*\*\*\*\*\*\*\*\*Template Starts Here\*\*\*\*\*\*\*\*\*\*\*/

#define pb push\_back

#define nl puts ("")

#define sp printf ( " " )

#define phl printf ( "hello\n" )

#define ff first

#define ss second

#define POPCOUNT \_\_builtin\_popcountll

#define RIGHTMOST \_\_builtin\_ctzll

#define LEFTMOST(x) (63-\_\_builtin\_clzll((x)))

#define MP make\_pair

#define FOR(i,x,y) for(int i = (x) ; i <= (y) ; ++i)

#define ROF(i,x,y) for(int i = (y) ; i >= (x) ; --i)

#define CLR(x,y) memset(x,y,sizeof(x))

#define UNIQUE(V) (V).erase(unique((V).begin(),(V).end()),(V).end())

#define MIN(a,b) ((a)<(b)?(a):(b))

#define MAX(a,b) ((a)>(b)?(a):(b))

#define NUMDIGIT(x,y) (((int)(log10((x))/log10((y))))+1)

#define SQ(x) ((x)\*(x))

#define ABS(x) ((x)<0?-(x):(x))

#define FABS(x) ((x)+eps<0?-(x):(x))

#define ALL(x) (x).begin(),(x).end()

#define LCM(x,y) (((x)/gcd((x),(y)))\*(y))

#define SZ(x) ((int)(x).size())

using namespace std;

#define LL long long

typedef long long vlong;

typedef unsigned long long uvlong;

typedef pair < int, int > pii;

typedef pair < vlong, vlong > pll;

typedef vector<pii> vii;

typedef vector<int> vi;

const vlong inf = 2147383647;

const double pi = 2 \* acos ( 0.0 );

const double eps = 1e-9;

#ifdef alfa106

#include <ctime>

clock\_t tStart = clock();

#define debug(args...) {dbg,args; cerr<<endl;}

#define timeStamp printf("Execution Time: %.2fs\n", (double)(clock() - tStart)/CLOCKS\_PER\_SEC)

#else

#define debug(args...) // Just strip off all debug tokens

#define timeStamp

#endif

struct debugger

{

template<typename T> debugger& operator , (const T& v)

{

cerr<<v<<" ";

return \*this;

}

}dbg;

//int knightDir[8][2] = { {-2,1},{-1,2},{1,2},{2,1},{2,-1},{-1,-2},{1,-2},{-2,-1} };

//int dir4[4][2] = {{-1,0},{0,1},{1,0},{0,-1}};

inline vlong gcd ( vlong a, vlong b ) {

a = ABS ( a ); b = ABS ( b );

while ( b ) { a = a % b; swap ( a, b ); } return a;

}

int ext\_gcd ( int A, int B, int \*X, int \*Y ){

int x, y, u, v, m, n, a, b, q, r;

x = 0; y = 1;

u = 1; v = 0;

for (a = A, b = B; 0 != a; b = a, a = r, x = u, y = v, u = m, v = n) {

q = b / a;

r = b % a;

m = x - (u \* q);

n = y - (v \* q);

}

\*X = x; \*Y = y;

return b;

}

inline vlong modInv ( int a, int m ) {

int x, y;

ext\_gcd( a, m, &x, &y );

if ( x < 0 ) x += m; //modInv is never negative

return x;

}

inline vlong power ( vlong a, vlong p ) {

vlong res = 1, x = a;

while ( p ) {

if ( p & 1 ) res = ( res \* x );

x = ( x \* x ); p >>= 1;

}

return res;

}

inline vlong bigmod ( vlong a, vlong p, vlong m ) {

vlong res = 1 % m, x = a % m;

while ( p ) {

if ( p & 1 ) res = ( res \* x ) % m;

x = ( x \* x ) % m; p >>= 1;

}

return res;

}

inline LL lcm(LL a, LL b) {

LL g = gcd(a,b);

a = a/g;

b \*= a;

return b;

}

inline prm() {

for (LL i=2; i<=100000; i++) {

if (vis[i]) continue;

pr.pb(i);

for (LL j=i; j\*i<=100000; j++) {

vis[j\*i] = 1;

}

} return;

}

/\*\*\*\*\*\*\*\*\*\*\*Template Ends Here\*\*\*\*\*\*\*\*\*\*\*/