# 偶拉函数表

int phi[MAXN];

int n;

void initEularTable(){

memset(phi,0, sizeof(phi));

phi[1] = 1;

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

if (!phi[i]){

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

if (!phi[j]){

phi[j] = j;

}

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

}

}

}

}

# 唯一素数定理分解（带P统计）

void deconstruct(int num,int ans[],int ansNum[],int & cur){

cur = 0;

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

if (num % i == 0){

ans[cur] = i;

ansNum[cur] = 0;

while (num % i == 0){

ansNum[cur]++;

num /= i;

}

cur++;

}

}

if(num>1){

ans[cur] = num;

ansNum[cur++] = 1;

}

}

# 唯一素数分解（排列组合）

bool prime[MAXN];

int pstack[MAXN]; ／／素数表

int pcounter[MAXN]; ／／p数组

void init(){

memset(prime,1, sizeof(prime));

cur = 0;

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

if (prime[i]){

pstack[cur++] = i;

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

prime[j] = 0;

}

}

}

}

void addfactor(int n,int step){

for (int i = 0; i < cur && n!=1; ++i) {

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

pcounter[i] += step;

n /= pstack[i];

}

}

}

void addcfactor(int begin,int end,int step){

for (int i = begin; i <= end ; ++i) {

addfactor(i,step);

}

}

# 拓展GCD

LL exgcd(LL a,LL b,LL & x,LL & y){

LL d;

if (b){

d = exgcd(b,a%b,y,x);

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

} else{

d = a;

x = 1;

y = 0;

}

return d;

}

# 树形SG博弈

int dfs(int u){

visV[u] = true;

int ret = 0;

s[scur++] = u;

for (int i = head[u]; ~i; i = G[i].next) {

if (!visE[i]){

visE[i] = visE[i^1] = true;

int tt;

if (!visV[G[i].v]){

tt = dfs(G[i].v) + 1;

} else{

int q = s[--scur];

while (q != G[i].v){

w[q] = true;

q = s[--scur];

}

++scur;

return 1;

}

ret ^= w[G[i].v] ? tt % 2 : tt ;

}

}

return ret;

}