```
#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
#include<set>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; --i)
#define REP(i,n) FOR(i,0,(n)-1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
#define maxi(a,b) a=max(a,b)
bool debug;
typedef vector<int> vi;
typedef long long ll;
typedef long double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 1e6 + 5;

inline int fastMax(int x, int y) { return (((y-x)>>(32-1))&(x^y))^y; }
inline int fastMin(int x, int y) { return (((y-x)>>(32-1))&(x^y))^x; }
inline int fastAbs(int n) { return (n ^ (n >> (32-1))) - (n >> (32-1)); }

int n;
int t[1000007];


/*Kod1, n = 100000, losowe dane:
        scanf("%d",&n);
        REP(i,n) scanf("%d",&t[i]);

        ll suma = 0;
        REP(i,n) FOR(j,i+1,n-1)
                suma += min(t[i],t[j]);
        printf("%lld\n",suma);
        return 0;

min, min: 5.128s
fastMin, fastMin: 6.258s
*/

/*Kod2, n = 100000, losowe dane:
        scanf("%d",&n);
        REP(i,n) scanf("%d",&t[i]);

        ll suma = 0;
        int x = 0;
        REP(i,n) FOR(j,i+1,n-1) if (min(t[i], t[j]) > x) {
                suma += min(t[i],t[j]);
                x = t[i] - t[j];
        }
        printf("%lld\n",suma);
        return 0;

min, min: 16.557s
min, fastMin: 5.101s
fastMin, fastMin: 5.555s
*/
```

```
/*Kod3, n = 100000, losowe dane:
        scanf("%d",&n);
        REP(i,n) scanf("%d",&t[i]);

        ll suma = 0;
        int x = 0;
        REP(i,n) FOR(j,i+1,n-1) if (min(t[i], t[j]) > x) {
                suma += min(t[i],t[j] - x);
                x = t[i] - t[j];
        }
        printf("%lld\n",suma);
        return 0;

min, min: 5.222s
min fastMin: 5.051s
fastMin, fastMin: 5.549s
*/

//Jak zamienic min -> max i fastMin -> fastMax wyniki sa analogiczne

/*Kod4, n = 100000, losowe dane:
        scanf("%d",&n);
        REP(i,n) scanf("%d",&t[i]);

        ll suma = 0;
        int x = 0;
        REP(i,n) FOR(j,i+1,n-1) if (abs(t[i] - t[j]) > abs(t[j] - x)) {
                suma += abs(t[i] - x);
                x = t[i] - t[j];
        }
        printf("%lld\n",suma);

abs, abs, abs: 14.527s
fastAbs, fastAbs, fastAbs: 16.609s
*/

//dla fastAbs nie znalazlem kodu kiedy bedzie szybsze
//wniosek ogolnie zwykle funkcje lepsze ale jak jest grubo to mozna sprobowac

int main(int argc, char * argv[]) {
        debug = argc > 1;
        return 0;
}
//testowana na:
//      http://codeforces.com/problemset/problem/286/E 4.5s/8s,
//                      wersja NTT 3^n za wolna ;_;


#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
#include<set>
#include<complex>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; --i)
#define REP(i,n) FOR(i,0,(n)-1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
```

```
#define maxi(a,b) a=max(a,b)
bool debug;
typedef vector<int> vi;
typedef long long ll;
typedef double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 1e5 + 5;
const int czapa = 2097152;

const ld PI = 3.14159265358979323846264338327950;
complex<ld> y2[czapa];

void fft(complex<ld> *a, complex<ld> *y, bool rev,
                int n, int p = 0, int s = 1, int q = 0) {

        if (n == 1) {
                y[q] = a[p];
                return;
        }

        complex<ld> e = 1, en = rev ? exp(complex<ld>(0,−2.*PI / (ld)n)) : exp(
complex<ld>(0,2.*PI / (ld)n));
        fft(a, y, rev, n/2, p, s*2, q);
        fft(a, y, rev, n/2, p+s, s*2, q+n/2);
        REP(k,n/2) {
                y2[k] = y[q+k] + e*y[q+n/2+k];
                y2[n/2 + k] = y[q+k] − e*y[q+n/2+k];
                e *= en;
        }
        REP(k,n)
                y[q+k] = y2[k];
}

complex<ld> y[czapa], a[czapa];

void kw(int n, bool *v) {
        while (n&(n−1)) ++n; n *= 2;
        REP(i,n) a[i] = complex<ld>((ld)v[i],0.0);
        fft(a,y,false,n);
        REP(i,n) y[i] = y[i]*y[i];
        fft(y,a,true,n);
        REP(i,n) a[i] /= ld(n);
}

int n,m,x;
bool v[czapa+7];

int main(int argc, char * argv[]) {
        debug = argc > 1;

        scanf("%d%d",&n,&m);
        REP(i,n) {
                int x;
                scanf("%d",&x);
                v[x] = true;
        }

        kw(m+1,v);
        vector<int> res;
        FOR(i,1,m) {
                if (v[i]) {
                        if (round(a[i].real()) < 0.5) res.pb(i);
                }
                else {
```

```
                        if (round(a[i].real()) > 0.5) {
                                puts("NO");
                                return 0;
                        }
                }
        }
        puts("YES");
        printf("%d\n",(int)res.size());
        for (auto i: res) printf("%d ",i); puts("");
        return 0;
}
//testowana na:
//      http://codeforces.com/problemset/problem/286/E 1.7s/8s
//      http://codeforces.com/problemset/problem/300/D 0.186s/3s
//      http://codeforces.com/contest/472/problem/G 1.3s/7s
//      http://codeforces.com/contest/528/problem/D 0.5s/3s

#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
#include<set>
#include<cmath>
#include<complex>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; −−i)
#define REP(i,n) FOR(i,0,(n)−1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
#define maxi(a,b) a=max(a,b)
bool debug;
typedef vector<int> vi;
typedef long long ll;
typedef double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 1e5 + 5;
const int czapa = 2097152;
const int mod = 998244353; //mod − 1 = 2^23 * 7 * 17
const int generator = 15311432; // == 3 ** (7 * 17)
const int maxczapa = 1<<23;

const ld PI = 3.14159265358979323846264338327950;

struct zespo{
        ld re, im;
        zespo(ld r=0., ld i=0.) : re(r), im(i) {}

        zespo operator+(zespo const& x) const {
                return zespo(re+x.re,im+x.im);
        }

        zespo operator−(zespo const& x) const {
                return zespo(re−x.re,im−x.im);
        }

        zespo operator*(zespo const& x) const {
                return zespo(re*x.re−im*x.im,re*x.im+im*x.re);
        }
        zespo operator/(ld d) const {
                return zespo(re/d,im/d);
        }
};

void fft(zespo *a, bool rev, int n) {
        //najnizszy krok rekurencyjny, symulujemy dzialanie rekurencyjnej
        //funckji i w ten sposob wilyczamy cos sie bedzie odpierdzielac
        //nie ogarniam jak to dziala, ale jest szybkie
        for (int i=1, j=0; i<n; i++) {
                int bit = n >> 1;
                for (; j>=bit; bit>>=1)
                        j -= bit;
                j += bit;
                if (i < j)
                        swap (a[i], a[j]);
        }

        //wersja ktora ogarniam, troszeczke wolniejsza, (przewal bloki i rekure
ncyjnie nizej)
        /*for (int przedzial = n >> 1, blok = 1; blok < przedzial; przedzial >>
= 1, blok <<= 1) {
                for (int i = przedzial; i < n; i += przedzial<<1)
                        for (int j = i; j < i+przedzial; j += blok<<1)
                                for (int k = j; k < j+blok; k++)
                                        swap(a[k - przedzial + blok], a[k]);
        }*/


        //wykonujemy rekurencje od dolu
        for (int len=2; len<=n; len<<=1) {
                ld kat = 2.0*PI/len * (rev ? -1 : 1);
                zespo pierw(cos(kat),sin(kat));
                for (int i=0; i<n; i+=len) {
                        zespo w(1,0);
                        for (int j=0; j<len/2; ++j) {
                                zespo u = a[i+j], v = a[i+j+len/2]*w;
                                a[i+j] = u + v;
                                a[i+j+len/2] = u - v;
                                w = w * pierw;
                        }
                }
        }

        if (rev)
                for(int i=0;i<n;i++)
                        a[i] = a[i] / n;
}

zespo a[czapa];

void kw(int n, bool *v) {
        while (n&(n-1)) ++n; n *= 2;
        REP(i,n) a[i] = zespo((ld)v[i],0.0);
        fft(a,false,n);
        REP(i,n) a[i] = a[i]*a[i];
        fft(a,true,n);
}

int n,m,x;
bool v[czapa];

int main(int argc, char * argv[]) {
        debug = argc > 1;
```

```
        scanf("%d%d",&n,&m);
        REP(i,n) {
                int x;
                scanf("%d",&x);
                v[x] = true;
        }

        kw(m+1,v);
        vector<int> res;
        FOR(i,1,m) {
                if (v[i]) {
                        if (round(a[i].re) < 0.5) res.pb(i);
                }
                else {
                        if (round(a[i].re) > 0.5) {
                                puts("NO");
                                return 0;
                        }
                }
        }
        puts("YES");

        int x = (int)res.size();
        printf("%d\n",(int)res.size());
        REP(i,x) printf("%d ",res[i]); puts("");
        return 0;
}
#include <cstdio>
#include <vector>
#include <queue>
#include <stack>
#include <cstring>
#include <iostream>
#include <algorithm>
#include <set>
#define MAXN 5007
#define INF
#define PB push_back
#define MP make_pair
#define ST first
#define ND second

#define REP(i,n) for(int i=0;i<(n);i++)
#define FOR(a,b,c) for(int a=b;a<=(c);a++)
#define FORD(a,b,c) for (int a=b;a>=(c);a--)
#define VAR(v,n) __typeof(n) v=(n)
#define ALL(c) c.begin(),c.end()
#define FOREACH(i,c) for(VAR(i,(c).begin());i!=(c).end();i++)

using namespace std;

typedef long long LL;
typedef long double LD;

const LD EPS = 1e-6;
int n;
int zmienna[MAXN];
LD m[MAXN][MAXN],wyn[MAXN];

bool IsZero(LD x) {
        return x >= -EPS && x <= EPS;
}

int szukaj(int w, int k) {
        FOR(i,w,n-1) if (!IsZero(m[i][k])) return i;
        return -1;
}

int Gauss() {
        int w = 0, k = 0, res = 0;
        while (k < n) {
                int p = szukaj(w,k);
                if (p == -1) res = 1,zmienna[w] = -1;
                else {
                        swap(m[p],m[w]);
                        zmienna[w] = k;
                        FOR(i,w+1,n-1) {
                                LD stos = m[i][k]/m[w][k];
                                FOR(j,k,n) m[i][j] -= stos*m[w][j];
                        }
                        ++w;
                }
                ++k;
        }
        FOR(i,w,n-1) if (!IsZero(m[i][n])) return -1;
        FORD(i,w-1,0) {
                int x = zmienna[i];
                wyn[x] = m[i][n];
                FOR(j,x+1,n-1) wyn[x] -= wyn[j]*m[i][j];
                wyn[x] /= m[i][x];
        }
        return res;
}

int main(){
        scanf("%d",&n);
        REP(i,n) REP(j,n+1) scanf("%Lf",&m[i][j]);

        //-1 sprzeczny, 0 jednoznaczny, 1 niejednoznaczny
        int res = Gauss();
        printf("%d\n",res);
        if (res != -1) {
                REP(i,n) printf("x%d = %Lf\n",i+1,wyn[i]);
        }
        return 0;
}
//sprawdzone:
//      http://main.edu.pl/pl/archive/pa/2012/ren 0.28s/25s
//      http://community.topcoder.com/stat?c=problem_statement&pm=7726&rd=10787
//      http://community.topcoder.com/stat?c=problem_statement&pm=8143&rd=10789

#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
#include<set>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; --i)
#define REP(i,n) FOR(i,0,(n)-1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
#define maxi(a,b) a=max(a,b)
bool debug;
```

```
typedef vector<int> vi;
typedef long long ll;
typedef long double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 607;

//maksymalnego skojarzenia, indeksowanie od 1, wszystko na intach
//w skojx[] bedzie skojarzenie

namespace Hung{
        int n;
        int m[nax][nax];
        int skojx[nax], skojy[nax];
        int lx[nax], ly[nax];
        bool S[nax], T[nax];
        int q[nax], qbeg, qend;
        int slack[nax], prev[nax];

        void init(int size) {
                n = size;
                FOR(i,1,n) FOR(j,1,n) m[i][j] = -inf;
                FOR(i,1,n) skojx[i] = skojy[i] = 0;
        }

        void AddEdge(int a, int b, int c) {
                m[a][b] = max(m[a][b], c);
        }

        int hungarian() {
                FOR(i,1,n) {
                        ly[i] = 0; lx[i] = -inf;
                        FOR(j,1,n) lx[i] = max(lx[i], m[i][j]);
                }

                REP(k,n) {
                        qbeg = qend = 0;
                        FOR(i,1,n) S[i] = T[i] = false;
                        FOR(i,1,n) if (!skojx[i]) {
                                q[qend++] = i; break;
                        }

                        S[q[0]] = true;
                        FOR(i,1,n) slack[i] = lx[q[0]] + ly[i] - m[q[0]][i], pr
ev[i] = q[0];

                        int y = 0;
                        while (true) {
                                while (qbeg < qend) {
                                        int x = q[qbeg++]; S[x] = true;
                                        FOR(i,1,n) if (!T[i]) {
                                                if (slack[i] > lx[x] + ly[i] -
m[x][i]) {
                                                        slack[i] = lx[x] + ly[i
] - m[x][i];
                                                        prev[i] = x;
                                                }
                                                if (lx[x] + ly[i] == m[x][i]) {
                                                        T[i] = true;
                                                        if (skojy[i])
                                                                q[qend++] = sko
jy[i];
                                                        else {
                                                                y = i; goto kon
iec;
```

```
                                                        }
                                                }
                                        }
                                }

                                int minslack = inf;
                                FOR(i,1,n) if (!T[i]) minslack = min(minslack,
slack[i]);
                                FOR(i,1,n) {
                                        if (S[i]) lx[i] -= minslack;
                                        if (T[i])
                                                ly[i] += minslack;
                                        else
                                                slack[i] -= minslack;
                                }
                                FOR(i,1,n) if (!T[i] && !slack[i]) {
                                        T[i] = true;
                                        if (skojy[i]) {
                                                q[qend++] = skojy[i];
                                        }
                                        else {
                                                y = i; goto koniec;
                                        }
                                }
                        }

                koniec:
                        while (y) {
                                skojy[y] = prev[y];
                                int pom = skojx[prev[y]];
                                skojx[prev[y]] = y;
                                y = pom;
                        }
                }

                int res = 0;
                FOR(i,1,n) res += m[i][skojx[i]];
                return res;
        }
}

int n,m;
bool in[nax], out[nax];
int cost[nax][nax];

int main(int argc, char * argv[]) {
        debug = argc > 1;

    scanf("%d%d",&n,&m);
    for (int i = 0; i < 2 * n; i++) {
        for (int j = 0; j < 2 * n; j++) {
            if (i >= n && j >= n) {
                cost[i][j] = 0;
            } else {
                cost[i][j] = -inf;
            }
        }
    }
    for (int i = 0; i < m; i++) {
        int x,y,k;
        scanf("%d%d%d", &x, &y, &k);
        cost[x-1][y-1] = -k;
        in[x-1] = true;
        out[y-1] = true;
    }
```

```
    for (int i = 0; i < n; i++) {
        if (!in[i] || !out[i]) {
            puts("NIE");
            return 0;
        }
    }
    for (int i = 0; i < n; i++) {
        int outcost = -inf;
        int incost = -inf;
        for (int j = 0; j < n; j++) {
            outcost = max(outcost, cost[i][j]);
            incost = max(incost, cost[j][i]);
        }
        for (int j = n; j < 2 * n; j++) {
            cost[i][j] = outcost;
            cost[j][i] = incost;
        }
    }
    n = 2 * n;
    Hung::init(n);
    REP(i,n) REP(j,n) {
                Hung::AddEdge(i+1,j+1,cost[i][j]);
        }

        printf("%d\n",-Hung::hungarian());
    return 0;
}
#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
#include<set>
#include<queue>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; --i)
#define REP(i,n) FOR(i,0,(n)-1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
#define maxi(a,b) a=max(a,b)
bool debug;
typedef vector<int> vi;
typedef long long ll;
typedef long double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 6405;

//sprawdzone:
//      http://main.edu.pl/pl/archive/pa/2012/ren 4.5s/25s
//      http://codeforces.com/contest/321/problem/B 0.06/2s
//      http://codeforces.com/problemset/problem/277/E 1s/5s

//c - capacity, v - value
//wierzcholki sa od 0 do n, wszystko na intach
//wiec wtedy musisz ogarnac odlegloci i takie tam
namespace MinCost{
        struct Edge{
                int w,c,v,rev;
                Edge(int _w, int _c, int _v, int _rev) :
                        w(_w), c(_c), v(_v), rev(_rev)
                {}
        };

        int odl[nax], pot[nax], pop[nax], pop_kraw[nax];
        int q[nax], qbeg, qend;
        vector<Edge> v[nax];
        bool bylo[nax];
        queue<int> kolej;
        //priority_queue<pair<int,int> > kolej;

        void init(int n) {
                FOR(i,0,n) v[i].clear();
        }

        void AddEdge(int a, int b, int cap, int cost) {
                v[a].pb(Edge(b,cap,cost,int(v[b].size()) + (a == b)));
                v[b].pb(Edge(a,0,-cost,int(v[a].size()-1)));
        }

        pair<int,int> MinCostMaxFlow(int s, int t, int n) {
                int flow = 0, cost = 0;
                //FOR(i,0,n) pot[i] = 0;
                //potencjaly zerowe dzialaja dla samych nieujemnych kosztow
                //jak sa ujemne krawedzie to bellman

                while (true) {
                        FOR(i,0,n) {
                                odl[i] = inf;
                                bylo[i] = false;
                        }
                        bylo[s] = true;
                        odl[s] = 0;
                        kolej.push(s);

                        //djikstra, mozna napisac na kolejce
                        while(!kolej.empty()) {
                                int x = kolej.front();
                                kolej.pop();
                                bylo[x] = false;

                                //if (bylo[x])
                                //      continue;

                                /*int najm = inf;
                                FOR(i,0,n) if (!bylo[i] && najm > odl[i]) {
                                        x = i; najm = odl[i];
                                }*/

                                //bylo[x] = true;
                                int dl = v[x].size();
                                REP(i,dl) if (v[x][i].c > 0 && odl[v[x][i].w] >
 odl[x] + pot[x] - pot[v[x][i].w] + v[x][i].v) {
                                        odl[v[x][i].w] = odl[x] + pot[x] - pot[
v[x][i].w] + v[x][i].v;
                                        if (!bylo[v[x][i].w]) {
                                                kolej.push(v[x][i].w);
                                                bylo[v[x][i].w] = true;
                                        }
                                        //kolej.push(mp(-odl[v[x][i].w], v[x][i
].w));
                                        pop[v[x][i].w] = x; pop_kraw[v[x][i].w]
 = i;
                                }
                        }
```

```
                        if (odl[t] == inf)
                                break;

                        //FOR(i,0,n) pot[i] += odl[i];

                        int x = t;
                        int cap = inf;
                        while (x != s) {
                                cap = min(cap, v[pop[x]][pop_kraw[x]].c);
                                x = pop[x];
                        }

                        flow += cap;
                        x = t;
                        while (x != s) {
                                cost += v[pop[x]][pop_kraw[x]].v*cap;
                                v[pop[x]][pop_kraw[x]].c -= cap;
                                v[x][v[pop[x]][pop_kraw[x]].rev].c += cap;
                                x = pop[x];
                        }
                }

                return mp(flow, cost);
        }
};

using namespace MinCost;

const int naxn = 87;

int n,m;
int t[naxn][naxn];

int main(int argc, char * argv[]) {
        debug = argc > 1;
    scanf("%d%d",&n,&m);
    init(n*m+1);
    FOR(i,1,n) FOR(j,1,m)
                scanf("%d",&t[i][j]);

        FOR(i,1,n) FOR(j,1,m) {
                int val = (i-1)*m + j;
                if ((i+j)&1) {
                        MinCost::AddEdge(0,val,1,0);
                        if (i > 1)
                                AddEdge(val, val-m, 1, t[i][j] != t[i-1][j]);
                        if (j > 1)
                                AddEdge(val, val-1, 1, t[i][j] != t[i][j-1]);
                        if (i < n)
                                AddEdge(val, val+m, 1, t[i][j] != t[i+1][j]);
                        if (j < m)
                                AddEdge(val, val+1, 1, t[i][j] != t[i][j+1]);
                }
                else
                        MinCost::AddEdge(val,n*m+1,1,0);
        }
        printf("%d\n",MinCost::MinCostMaxFlow(0,n*m+1,n*m+1).nd);
    return 0;
}
//sprawdzane recznie na kilku przykladach
#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
```

```
#include<set>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; --i)
#define REP(i,n) FOR(i,0,(n)-1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
#define maxi(a,b) a=max(a,b)
bool debug;
typedef vector<int> vi;
typedef long long ll;
typedef long double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 1e6 + 5;

//vector<ll> witness = {2, 7, 61}; // < 4759123141
vector<ll> witness = {2, 325, 9375, 28178, 450775, 9780504, 1795265022}; // < 2
^64

ll mnoz(ll a, ll b, ll mod) {
        return (__int128(a)*b)%mod;

        /*ll res = 0;
        while (b) {
                if (b&1) res = (res+a)%mod;
                a = (a+a)%mod;
                b /= 2;
        }
        return res;*/
}

ll pot(ll a, ll b, ll mod) {
        ll res = 1;
        while (b) {
                if (b&1)
                        res = mnoz(res,a,mod);
                a = mnoz(a,a,mod);
                b /= 2;
        }
        return res;
}

bool test(ll n) {
        if (n == 2)
                return true;
        if (n < 2 || n%2 == 0)
                return false;

        ll d = n-1;
        ll s = 0;
        while (d%2 == 0) {
                d /= 2;
                ++s;
        }

        for (auto i: witness) if (i%n) {
                ll x = pot(i,d,n);
                if (x != 1) {
                        bool zlozona = true;
```

```
                        REP(j,s) {
                                if (x == n-1) {
                                        zlozona = false;
                                        break;
                                }
                                x = (x*x)%n;
                        }
                        if (zlozona)
                                return false;
                }
        }

        return true;
}

ll nwd(ll a, ll b) {
        return a ? nwd(b%a,a) : b;
}

ll f(ll x, ll mod, ll c) {
        ll y = mnoz(x,x,mod) + c;
        if (y > mod)
                y -= mod;
        return y;
}

void rho(ll n, vector<ll> &v) {
        if (n <= 1) return;
        if (test(n)) {
                v.pb(n);
                return;
        }

        ll c = 1;
        while(true) {
                ll x = 2, y = 2, d = 1;
                while (d == 1) {
                        x = f(x,n,c);
                        y = f(f(y,n,c),n,c);
                        d = nwd(abs(x-y),n);
                }
                if (d < n) {
                        rho(d, v);
                        rho(n/d,v);
                        return;
                }
                ++c;
        }
}

void rozklad(ll n, vector<ll> &v) {
        int BLOK = 100;
        FOR(i,2,BLOK) while (n%i == 0) {
                n /= i;
                v.pb(i);
        }

        rho(n,v);
        sort(v.begin(),v.end());
}

int main(int argc, char * argv[]) {
        debug = argc > 1;
        while(true) {
                ll x;
                scanf("%lld",&x);

                vector<ll> v;
                rozklad(x,v);
                printf("rozklad %lld to:\n",x);
                for (auto i: v) printf("%lld ",i); puts("");
                break;
        }
        return 0;
}
//sprawdzone recznie wiec swag
#include<cstdio>
#include<algorithm>
#include<vector>
#include<cstring>
#include<set>
#include<assert.h>
using namespace std;
#define FOR(i,a,b) for(int i = a; i <= b; ++i)
#define FORD(i,a,b) for(int i = a; i >= b; --i)
#define REP(i,n) FOR(i,0,(n)-1)
#define RI(i,n) FOR(i,1,n)
#define pb push_back
#define mp make_pair
#define st first
#define nd second
#define mini(a,b) a=min(a,b)
#define maxi(a,b) a=max(a,b)
bool debug;
typedef vector<int> vi;
typedef long long ll;
typedef long double ld;
typedef pair<int,int> pii;
const int inf = 1e9 + 5;
const int nax = 1e6 + 5;

void eukl(int &x, int &y, int a, int b) {
        if (!a) {
                x = 0;
                y = 1;
                return;
        }

        eukl(y,x,b%a,a);
        x -= y*(b/a);
}

int main(int argc, char * argv[]) {
        debug = argc > 1;
        int a,b;
        scanf("%d%d",&a,&b);
        int x,y;
        eukl(x,y,a,b);
        printf("%d %d nwd to %d\n",x,y,x*a + y*b);
        return 0;
}
```