```
/*
ID: dnkihot1
LANG: C++
TASK: combo
*/
#include <iostream>
#include <fstream>
#include <set>
using namespace std;


#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()


void gen_co(int cc[3], int N, set<int> &s) {
    for(int i=-2; i<=2; ++i)
        for(int j=-2; j<=2; ++j)
            for(int k=-2; k<=2; ++k) {
                int co[3] = {(cc[0]-1+N+i)%N+1, (cc[1]-1+N+j)%N+1, (cc[2]-1+N+k)%N+1};
                s.insert(co[0]*100 + co[1]*10 + co[2]);
            }
}


int main() {
    cin.sync_with_stdio(false);
    ifstream in("combo.in");
    ofstream out("combo.out");


    int N; in >> N;
    int fc[3]; in >> fc[0] >> fc[1] >> fc[2];
    int mc[3]; in >> mc[0] >> mc[1] >> mc[2];


    set<int> co;
    gen_co(&fc[0], N, co);
    gen_co(&mc[0], N, co);


    out << SZ(co) << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: wormhole
*/
#include <iostream>
#include <fstream>
#include <algorithm>
#include <vector>
#include <set>
using namespace std;
```

```cpp
#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()


const int MAXN = 13;


struct Wh {
    int x, y, o;
    Wh(int x, int y, int o): x(x), y(y), o(o) {}
    bool operator<(const Wh &ot) const {
        return (y < ot.y) ||
               (y == ot.y && x < ot.x);
    }
};
ostream& operator<<(ostream &os, const Wh &w) {
    os << "[" << w.x << ", " << w.y << ", ord=" << w.o << "]";
    return os;
}


int N, sol;
vector<Wh> wh;
vector<int> ne(MAXN), pr(MAXN);
set<int> us;


int circ() {
    for(int i=0; i<N; ++i) {
        int n=ne[pr[i]];
        while(n!=i && n!=-1) {
            n = ne[pr[n]];
        }
        if(n == i) return 1;
    }
    return 0;
}


void rec(int x) {
    if(SZ(us) == N) {
        sol += circ();
        //for(auto i: us) show(i);
        //for(int i=0; i<N; ++i) show(pr[i]);
    } else {
        us.insert(x);
        for(int i=x+1; i<N; ++i)
            if (!us.count(i)) {
                us.insert(i);
                pr[i] = x;
                pr[x] = i;


                int mn=0;
                while(us.count(mn)) ++mn;
                rec(mn);


                us.erase(i);
            }
        us.erase(x);
    }
}


int main() {
    cin.sync_with_stdio(false);
    ifstream in("wormhole.in");
    ofstream out("wormhole.out");


    wh.clear();
    fill(ALL(ne), -1);
    fill(ALL(pr), -1);
    us.clear();


    in >> N;
    for(int i=0; i<N; ++i) {
        int tx, ty; in >> tx >> ty;
        wh.push_back(Wh(tx, ty, i));
    }
    sort(ALL(wh));
    //for (auto i: wh) show(i);


    int p=0;
    while(p<N) {
        int cy=wh[p].y;
        int r=p;
        while(r<N && wh[r].y==cy) ++r;


        for(int i=p; i<r-1; ++i)
            ne[wh[i].o] = wh[i+1].o;
        p=r;
    }
    //for(int i=0; i<4; ++i) show(ne[i]);


    sol=0;
    rec(0);
    out << sol << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: skidesign
*/
#include <iostream>
#include <fstream>
#include <algorithm>
using namespace std;


#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()
```

```
const int MAXN=1010;
const int MAXH=101;
const int INF=(int) 1e9;
int h[MAXN];
int c[MAXN][MAXH];


int main() {
    ifstream in("skidesign.in");
    ofstream out("skidesign.out");


    fill(&c[0][0], &c[0][0]+MAXN*MAXH, 10000);


    int N; in >> N;
    int minh=100, maxh=0;
    for(int i=0; i<N; ++i) {
        in >> h[i];
        if(h[i] > maxh) maxh=h[i];
        if(h[i] < minh) minh=h[i];
    }


    for(int i=0; i<N; ++i) {
        for(int ub=max(17,minh); ub<=max(17,maxh); ++ub) {
            int lb=max(ub-17,0);
            if(h[i] > ub) c[i][ub]=(h[i]-ub)*(h[i]-ub);
            if(h[i] < lb) c[i][ub]=(lb-h[i])*(lb-h[i]);
            if(lb <= h[i] && h[i] <= ub) c[i][ub] = 0;
        }
    }


    int sol=INF;
    for(int j=0; j<=MAXH; ++j) {
        int acc=0;
        for(int i=0; i<N; ++i) acc+=c[i][j];
        if(acc < sol) sol = acc;
    }


    out << sol << endl;
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: preface
*/
#include <iostream>
#include <fstream>
using namespace std;
```

```
#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()


const char dig[7] = {'I', 'V', 'X', 'L', 'C', 'D', 'M'};
int num[7];


int get_num(int n, int d) {
    if(d==6) {//M
        int x=n/1000;
        int y=(n%1000)/100;
        return x + (y==9);
    }
    if(d==5) {//D
        int x=(n%1000)/100;
        return (4<=x) && (x<9);
    }
    if(d==4) {//C
        int x=((n%1000)/100)%5;
        int y=(n%100)/10;
        return ((1<=x) && (x<4) ? x :
                (x==4) ? 1 : 0) +
               (y==9);
    }
    if(d==3) {//L
        int x=(n%100)/10;
        return (4<=x) && (x<9);
    }
    if(d==2) {//X
        int x=((n%100)/10)%5;
        int y=n%10;
        return ((1<=x) && (x<4) ? x :
                (x==4) ? 1 : 0) +
               (y==9);
    }
    if(d==1) {//V
        int x=n%10;
        return (4<=x) && (x<9);
    }
    if(d==0) {//I
        int x=n%5;
        return (1<=x) && (x<4) ? x :
               (x==4) ? 1 : 0;
    }
}


int main() {
    ifstream in("preface.in");
    ofstream out("preface.out");


    int N; in >> N;
    for(int i=1; i<=N; ++i)
        for(int t=0; t<7; ++t)
            num[t] += get_num(i, t);


    for(int i=0; i<7; ++i)
        if(num[i] > 0)
            out << dig[i] << ' ' << num[i] << endl;
```

```
}



/******************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: lamps
*/
#include <iostream>
#include <fstream>
#include <algorithm>
#include <set>
using namespace std;


#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()


#define SW0
"1111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
111"
#define SW1
"1010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
010"
#define SW2
"0101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010
101"
#define SW3
"1001001001001001001001001001001001001001001001001001001001001001001001001001001001001001001001001
001"
#define EMP
"0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000"


const int MAXN = 101;


struct Lamps {
    int e[MAXN];
    int sz;


    Lamps(): sz(MAXN) { fill(e,e+MAXN,1); };
    Lamps(int sz): sz(sz) { fill(e,e+MAXN,1); }
    Lamps(int sz, string pa): sz(sz) {
        fill(e,e+MAXN,1);
        for(int i=0; i<pa.length() && i<sz; ++i)
            if(pa[i] == '0') e[i]=0;
            else e[i]=1;
    }


    int& operator[](int i) { return e[i]; }
    Lamps operator&(const Lamps &ot) const {
        Lamps r(sz);
        for(int i=0; i<MAXN; ++i) r.e[i] = e[i] & ot.e[i];
```

```cpp
        return r;
    }
    Lamps operator|(const Lamps &ot) const {
        Lamps r(sz);
        for(int i=0; i<MAXN; ++i) r.e[i] = e[i] | ot.e[i];
        return r;
    }
    Lamps operator^(const Lamps &ot) const {
        Lamps r(sz);
        for(int i=0; i<MAXN; ++i) r.e[i] = e[i] ^ ot.e[i];
        return r;
    }
    bool operator<(const Lamps &ot) const {
        for(int i=0; i<sz; ++i)
            if(e[i] != ot.e[i])
                if(e[i] < ot.e[i])
                    return true;
                else return false;
    }
    bool operator==(const Lamps &ot) const {
        for(int i=0; i<sz; ++i)
            if(e[i] != ot.e[i])
                return false;
        return true;
    }
};


ostream& operator<<(ostream &os, const Lamps &la) {
    for(int i=0; i<la.sz; ++i) os << la.e[i];
    return os;
}


int cntb(int a) {
    int r=0;
    while(a) {
        r += (a&1);
        a >>= 1;
    }
    return r;
}


int main() {
    ifstream in("lamps.in");
    ofstream out("lamps.out");


    int N; in >> N;
    int C; in >> C;
    Lamps sw[] = { Lamps(N, SW0), Lamps(N, SW1), Lamps(N, SW2), Lamps(N, SW3) };


    Lamps eon(N, EMP);
    for(;;) {
        int t; in >> t;
        if(t == -1) break;
        eon[t-1]=1;
    }
    Lamps eof(N, EMP);
    for(;;) {
        int t; in >> t;
        if(t == -1) break;
        eof[t-1]=1;
    }


    Lamps emp(N, EMP);


    set<Lamps> un;
    for(int i=0; i<16; ++i) {
        if(cntb(i) <= C) {
            Lamps st(N);
            for(int j=0; j<4; ++j)
                if((i >> j) & 1)
                    st = st ^ sw[j];


            if (((st & eon) == eon) && ((st & eof) == emp))
                un.insert(st);
        }
    }


    if(SZ(un)) {
        for(set<Lamps>::iterator it=un.begin(); it != un.end(); ++it)
            out << *it << endl;
    } else {
        out << "IMPOSSIBLE" << endl;
    }
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: subset
*/
#include <iostream>
#include <fstream>
#include <map>
using namespace std;


#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()


const int MAXN=40;
const int MAXS=(MAXN-1)*MAXN/2;


typedef long long ll;
typedef pair<int,int> pii;
typedef map<pii, ll> hashmap;
hashmap dp[MAXN];
```

```
int main() {
    ifstream in("subset.in");
    ofstream out("subset.out");


    int N; in >> N;


    int cs=0;
    dp[0][make_pair(0,0)] = 1LL;
    for(int i=1; i<=N; ++i) {
        cs+=i;
        for(int j=0; j<=cs/2; ++j) {
            int n1=min(j,cs-j), n2=max(j,cs-j);
            int o1, o2;
            if(cs-j-i>=0) {
                o1=min(j,cs-j-i);
                o2=max(j,cs-j-i);
                dp[i][make_pair(n1,n2)] += dp[i-1][make_pair(o1,o2)] * (o1==o2 ? 2LL : 1LL);
                //cerr << i << ' ' << n1 <<  ' ' << n2 << ' ' << dp[i][n1][n2]/2 << endl;
            }
            if(j-i>=0 && j-i!=cs-j-i) {
                o1=min(j-i,cs-j);
                o2=max(j-i,cs-j);
                dp[i][make_pair(n1,n2)] += dp[i-1][make_pair(o1,o2)];
                //cerr << i << ' ' << n1 << ' ' << n2 << ' ' << dp[i][n1][n2]/2 << endl;
            }
        }
    }


    out << (cs%2 ? 0 : dp[N][make_pair(cs/2,cs/2)]/2LL) << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: runround
*/
#include <iostream>
#include <fstream>
#include <vector>
#include <set>
#include <algorithm>
using namespace std;


#define show(x) cerr << "# " << #x << " = " << (x) << endl
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) (int) (a).size()


typedef unsigned long ulong;


ulong next_candidate(ulong m) {
    //cerr << "m=" << m << endl;
    vector<int> r;
    int l=0;
    while(m) {
        int d=m%10u;
        r.push_back(d);
        m/=10u;
        ++l;
    }
    int i=l-1;
    int u=0;
    for(;;) {
        for(int j=i+1; j<l && !u; ++j)
            u=(r[i]==r[j]);
        if(i==0 || r[i]==0 || u) break;
        --i;
    }
    //cerr << "i=" << i << endl;


    int n=10;
    while(i<l && n>9) {
        n=r[i]+1;
        for(; n<=9; ++n) {
            int j, e;
            for(j=i+1, e=0; j<l && !e; ++j)
                e=(r[j]==n);
            if(!e) break;
        }
        ++i;
    }


    if(i==l && n>9) {
        r.push_back(0);
        ++l;
        for(int j=l-1; j>=0; --j) r[j]=l-j;
    } else {
        r[i-1] = n;
        for(int j=i-2; j>=0; --j) {
            for(int k=1; k<=9; ++k) {
                int e=0;
                for(int jj=j+1; jj<l && !e; ++jj)
                    e=(r[jj]==k);
                if(e) continue;
                r[j] = k;
                break;
            }
        }
    }


    ulong s=0;
    while(l--) {
        s *= 10;
        s += r[l];
    }
    return s;
}


bool is_runround(ulong m) {
    ulong tm=m;
    vector<int> r;
    int l=0;
    while(tm) {
        int d=tm%10u;
        r.push_back(d);
        tm/=10u;
        ++l;
    }
    reverse(ALL(r));


    int mg=(1<<l)-1;
    int mc=0;


    int p=0;
    for(int i=0; i<l; ++i) {
        mc |= (1<<p);
        p = (p+r[p])%l;
    }


    return (mc==mg) && (p==0);
}


int main() {
    ifstream in("runround.in");
    ofstream out("runround.out");


    //srand(time(NULL)); for(auto r: { 802, 81361, 979, rand()%200000000+1 }) {
        //cerr << r << " " << next_candidate(r) << endl;
    //}
    //cerr << 81362 << " " << is_runround(81362) << endl;


    ulong M; in >> M;
    M = next_candidate(M);


    while(!is_runround(M)) {
        M = next_candidate(M);
    }


    out << M << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: prefix
*/
#define NDEBUG
#include <cassert>
#include <fstream>
```

```
#include <iostream>
#include <string>
#include <queue>
#include <set>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())


const int ALPHSZ=26;


inline int c2i(char c) { return c-'A'; }


typedef struct tnode tnode_t;
struct tnode {
    int v;
    bool e;
    tnode_t *c[ALPHSZ];
};


tnode_t *tinit(int v, bool e) {
    tnode_t *t = new tnode_t;
    t->v = v;
    t->e = e;
    for(int i=0; i<ALPHSZ; ++i) t->c[i]=nullptr;
    return t;
}


void tinsert(tnode_t *r, const string &s) {
    assert(r);
    int sz=SZ(s);


    for(int i=0; i<sz; ++i) {
        int j=c2i(s[i]);
        if(!r->c[j])
            r->c[j]=tinit(s[i], false);
        if(i==sz-1)
            r->c[j]->e=true;
        r=r->c[j];
    }
}


void tcheck(tnode_t *r, const string &s, int j, deque<int> &d) {
    int sz=SZ(s);
    tnode_t *n = r->c[c2i(s[j])];
    while(n && j<sz) {
        if(n->e) d.push_back(j+1);
        ++j;
        n = n->c[c2i(s[j])];
    }
}


int main() {
    ifstream in("prefix.in");
    ofstream out("prefix.out");


    tnode_t *ro = tinit(' ', false);


    for(;;) {
        string s; in >> s;
        if(s == ".") break;
        tinsert(ro, s);
    }
    string p, S;
    while(in >> p) S+=p;
    //cerr << S << endl;
    int sz=SZ(S);


    deque<int> ni; ni.push_back(0);
    set<int> wi;
    int maxl=0;
    while(!ni.empty()) {
        int ci=ni.front(); ni.pop_front();
        if(wi.count(ci)) continue;
        if(ci > maxl) maxl=ci;
        if(sz == maxl) break;
        wi.insert(ci);
        tcheck(ro, S, ci, ni);
    }


    out << maxl << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: money
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())


typedef long long ll;
const int MAXV=26;
const int MAXN=10010;


int co[MAXV];
ll cv[MAXV][MAXN];
```

```
int main() {
    ifstream in("money.in");
    ofstream out("money.out");


    int V; in >> V;
    int N; in >> N;
    for(int i=0; i<V; ++i) in >> co[i];


    for(int k=0; k<=N; k+=co[0])
        ++cv[0][k];


    for(int i=1; i<V; ++i) {
        for(int j=0; j<N; ++j)
            for(int k=j+co[i]; k<=N; k+=co[i])
                cv[i][k] += cv[i-1][j];
        for(int j=0; j<=N; ++j)
            cv[i][j] += cv[i-1][j];
    }


    out << cv[V-1][N] << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++
TASK: zerosum
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <vector>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())


inline int pow3(int n) {
    int r=1;
    while(n--) r*=3;
    return r;
}


inline int con(int x, int y) {
    int c=0;
    int t=y;
    while(t) { t/=10; ++c; }
    int d=1;
    while(c--) d*=10;
    return x*d + y;
}


char os[20];


int main() {
    ifstream in("zerosum.in");
    ofstream out("zerosum.out");


    int N; in >> N;
    for(int i=0; i<pow3(N-1); ++i) {
        vector<int> a;
        int o=i;
        int c=N;
        for(int j=N-1; j>=1; --j) {
            if(o%3==0) { c=con(j,c); }
            else if (o%3==1) { a.push_back(c); c=j; }
            else {  a.push_back(-c); c=j; }
            o/=3;
        }
        a.push_back(c);


        int s=0;
        while(!a.empty()) { s+=a.back(); a.pop_back(); }
        if(s==0) {
            fill(os,os+20,0);
            os[2*N-2]=N+'0';
            int e=i;
            for(int j=N-1; j>=1; --j) {
                if(e%3==0) os[2*j-1]=' ';
                else if(e%3==1) os[2*j-1]='+';
                else os[2*j-1]='-';
                os[2*j-2]=j+'0';
                e/=3;
            }
            out << os << endl;
        }
    }
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: concom
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <queue>
#include <functional>
using namespace std;
```

```
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())


const int MAXC=101;
int ow[MAXC][MAXC];
int oc[MAXC];


int main() {
    ifstream in("concom.in");
    ofstream out("concom.out");


    int N; in >> N;
    int mc=0;
    for(int i=0; i<N; ++i) {
        int o, t, p; in >> o >> t >> p;
        ow[o][t] += p;
        mc=max(mc,max(o,t));
    }


    for(int i=1; i<=mc; ++i) {
        copy(ow[i],ow[i]+MAXC,oc);
        deque<int> tp;
        priority_queue<int, vector<int>, greater<int> > pr;
        for(int j=1; j<=mc; ++j) {
            if(i==j) continue;
            if(ow[i][j] > 50) {
                tp.push_back(j);
                pr.push(j);
            }
        }


        while(!tp.empty()) {
            int c=tp.front(); tp.pop_front();
            for(int j=1; j<=mc; ++j) {
                if(oc[j] <= 50) {
                    oc[j] += ow[c][j];
                    if(oc[j] > 50) {
                        tp.push_back(j);
                        pr.push(j);
                    }
                }
            }
        }


        while(!pr.empty()) {
            int c=pr.top(); pr.pop();
            if(c==i) continue;
            out << i << " " << c << endl;
        }
    }
}



/******************************************************************************/
```

```cpp
/*
ID: dnkihot1
LANG: C++11
TASK: nocows
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())


typedef long long ll;
const int MAXN=200;
const int MAXK=100;
const ll MOD=9901;


ll fc[MAXN][MAXK], uc[MAXN][MAXK];


ll guc(int n, int k) {
    ll &r=uc[n][k];
    if(r == -1) {
        r=0;
        for(int i=0; i<=n-2; ++i)
            r += guc(i,k-1)*guc(n-2-i,k-1);
        assert(r>=0);
        r%=MOD;
    }
    return r;
}


ll gfc(int n, int k) {
    ll &r=fc[n][k];
    if(r == -1) {
        r=0;
        for(int i=0; i<=n-2; ++i)
            r += gfc(i,k-1)*gfc(n-2-i,k-1) +
                 gfc(i,k-1)*guc(n-2-i,k-1) +
                 guc(i,k-1)*gfc(n-2-i,k-1);
        assert(r>=0);
        r%=MOD;
    }
    return r;
}


int main() {
    ifstream in("nocows.in");
    ofstream out("nocows.out");


    fill(&fc[0][0],&fc[0][0]+MAXN*MAXK,-1);
    for(int i=1; i<MAXN; ++i) fc[i][0]=0;
    for(int i=1; i<MAXK; ++i) fc[0][i]=0;
    fc[0][0]=1;
    fill(&uc[0][0],&uc[0][0]+MAXN*MAXK,-1);
    for(int i=0; i<MAXN; ++i) uc[i][0]=0;
    for(int i=1; i<MAXK; ++i) uc[0][i]=1;


    int N, K; in >> N >> K;
    out << gfc(N-1,K-1) << endl;
}




/****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: ttwo
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <vector>
#include <string>
#include <set>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())


const int MAPSZ=10;


struct State {
    int y, x, d;
    State(int y=0, int x=0, int d=0): y(y), x(x), d(d) {}
    bool operator<(const State &ot) const {
        return (y < ot.y) ||
               (y == ot.y && x < ot.x) ||
               (y == ot.y && x == ot.x && d < ot.d);
    }
};


State next_state(const State &c, const vector<string> &m) {
    int dy[4]={-1,0,1,0}, dx[4]={0,1,0,-1};
    int ny = c.y + dy[c.d], nx = c.x + dx[c.d];
    if(ny < 0 || ny >= 10 || nx < 0 || nx >= 10 || m[ny][nx]=='*')
        return State(c.y, c.x, (c.d+1)%4);
    else
        return State(ny, nx, c.d);
}


typedef pair<State, State> pss;
```

```
int main() {
    ifstream in("ttwo.in");
    ofstream out("ttwo.out");


    vector<string> map;
    for(int i=0; i<MAPSZ; ++i) {
        string t; in >> t;
        map.push_back(t);
    }


    State F, C;
    for(int i=0; i<MAPSZ; ++i)
        for(int j=0; j<MAPSZ; ++j) {
            if(map[i][j] == 'F') {
                F = State(i,j,0);
                map[i][j] = '.';
            }
            if(map[i][j] == 'C') {
                C = State(i,j,0);
                map[i][j] = '.';
            }
        }


    set<pss> visited;
    int sol=0;
    bool valid=true;
    while(valid) {
        if(F.y == C.y && F.x == C.x) break;
        pss cs = make_pair(F,C);
        if(visited.count(cs)) { valid=false; break; }
        visited.insert(cs);


        F = next_state(F, map);
        C = next_state(C, map);
        ++sol;
    }


    out << (valid ? sol : 0) << endl;
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: maze1
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <string>
#include <vector>
#include <queue>
#include <set>
```

```cpp
#include <climits>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair


const int MAXH=101;
const int MAXW=39;
const int dy[4]={-1,0,1,0};
const int dx[4]={0,1,0,-1};
typedef pair<int, int> pii;


vector<string> map;


inline bool is_connected(int sy, int sx, int ey, int ex) {
    return map[(2*sy+2*ey+2)/2][(2*sx+2*ex+2)/2] == ' ';
}


inline bool is_exit(int y, int x, int H, int W) {
    if(y==0 && map[y][2*x+1]==' ') return true;
    if(y==H-1 && map[2*H][2*x+1]==' ') return true;
    if(x==0 && map[2*y+1][x]==' ') return true;
    if(x==W-1 && map[2*y+1][2*W]==' ') return true;
    return false;
}


int di[2][MAXH][MAXW];


int main() {
    ifstream in("maze1.in");
    ofstream out("maze1.out");
    fill(&di[0][0][0],&di[0][0][0]+2*MAXH*MAXW,-1);


    int W, H; in >> W >> H; in.ignore();
    for(int i=0; i<2*H+1; ++i) {
        string t; getline(in, t);
        map.push_back(t);
    }


    // do BFS for both exits
    int ndi=0;
    for(int y=0; y<H && ndi<2; ++y)
        for(int x=0; x<W && ndi<2; ++x)
            if(is_exit(y,x,H,W)) {
                deque<int> dp;
                deque<pii> np;
                set<pii> vp;
                dp.push_back(1);
                np.push_back(MP(y,x));
                vp.insert(MP(y,x));
                while(!np.empty()) {
                    int cd=dp.front(); dp.pop_front();
                    pii cp=np.front(); np.pop_front();
                    int cy=cp.first, cx=cp.second;


                    di[ndi][cy][cx]=cd;
                    for(int i=0; i<4; ++i) {
                        int nd=cd+1;
                        int ny=cy+dy[i];
                        int nx=cx+dx[i];
                        pii pp=MP(ny,nx);
                        if(!vp.count(pp) && ny>=0 && ny<H && nx>=0 && nx<W &&
is_connected(cy,cx,ny,nx)) {
                            dp.push_back(nd);
                            np.push_back(pp);
                            vp.insert(pp);
                        }
                    }
                }


                ++ndi;
            }


    int maxmin=0;
    for(int y=0; y<H; ++y)
        for(int x=0; x<W; ++x) {
            int mm=INT_MAX;
            for(int w=0; w<2; ++w)
                if(di[w][y][x]!=-1 && di[w][y][x]<mm)
                    mm = di[w][y][x];
            if(mm > maxmin) maxmin = mm;
        }


    out << maxmin << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: comehome
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define SS stringstream
```

```
const int MAXC=52;


int c2i(char c) {
    if('A'<=c && c<='Z') return c-'A';
    if('a'<=c && c<='z') return c-'a'+26;
    assert(0);
}


char i2c(int i) {
    if(0<=i && i<=25) return 'A'+i;
    if(26<=i && i<=51) return 'a'+i-26;
    assert(0);
}


bool visited[MAXC];
int di[MAXC];
int pa[MAXC][MAXC];


int main() {
    ifstream in("comehome.in");
    ofstream out("comehome.out");


    fill(visited,visited+MAXC,false);
    fill(di,di+MAXC,-1);
    fill(pa[0],pa[0]+MAXC*MAXC,-1);
    for(int i=0; i<MAXC; ++i) pa[i][i]=0;


    int P; in >> P;
    for(int i=0; i<P; ++i) {
        char s,e; in >> s >> e;
        int d; in >> d;


        int &cpa1=pa[c2i(s)][c2i(e)];
        int &cpa2=pa[c2i(e)][c2i(s)];
        if(cpa1==-1 || cpa1>d) cpa1=cpa2=d;
    }


    int nv=c2i('Z');
    visited[nv]=true;
    di[nv]=0;
    for(;;) {
        if('A'<=i2c(nv) && i2c(nv)<'Z') break;
        for(int i=0; i<MAXC; ++i)
            if(!visited[i] && pa[nv][i]!=-1 && (di[i]==-1 || di[nv]+pa[nv][i]<di[i]))
                di[i]=di[nv]+pa[nv][i];
        nv=-1;
        for(int i=0; i<MAXC; ++i)
            if(!visited[i] && di[i]!=-1 && (nv==-1 || di[i]<di[nv]))
                nv=i;
        visited[nv]=true;
        assert(nv!=-1);
    }
    out << i2c(nv) << " " << di[nv] << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: cowtour
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <vector>
#include <string>
#include <limits>
#include <cmath>
#include <queue>
#include <set>
#include <iomanip>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXN=151;


int sy[MAXN],sx[MAXN];
vector<string> am;
double di[MAXN][MAXN];
double mdi[MAXN];
double tdi[MAXN];


int main() {
  ifstream in("cowtour.in");
  ofstream out("cowtour.out");


  int N; in >> N;
  for(int i=0; i<N; ++i) in >> sx[i] >> sy[i];
  for(int i=0; i<N; ++i) {
    string ts; in >> ts;
    am.PB(ts);
  }


  fill(di[0],di[0]+MAXN*MAXN,INF);
  for(int i=0; i<N; ++i) di[i][i]=0;
  for(int i=0; i<N; ++i)
    for(int j=i+1; j<N; ++j) {
      if(am[i][j]=='1') {
        double dx=sx[i]-sx[j];
        double dy=sy[i]-sy[j];
        di[i][j]=di[j][i]=sqrt(dx*dx + dy*dy);
      }
    }


  for(int k=0; k<N; ++k)
    for(int i=0; i<N; ++i)
      for(int j=0; j<N; ++j)
        if(di[i][j] > di[i][k]+di[k][j])
          di[i][j] = di[i][k]+di[k][j];


  for(int i=0; i<N; ++i) {
    mdi[i]=0;
    for(int j=0; j<N; ++j)
      if(di[i][j]<INF && di[i][j]>mdi[i])
        mdi[i] = di[i][j];
  }


  for(int i=0; i<N; ++i) {
    tdi[i]=0;
    for(int j=0; j<N; ++j)
      if(di[i][j]<INF && tdi[i]<mdi[j])
        tdi[i]=mdi[j];
  }


  double sol=INF;
  for(int i=0; i<N; ++i)
    for(int j=0; j<N; ++j)
      if(!(di[i][j] < INF)) {
        double dx=sx[i]-sx[j];
        double dy=sy[i]-sy[j];
        double cdi=sqrt(dx*dx + dy*dy);
        double cb=max(tdi[i],tdi[j]);
        double ca=max(cb,cdi+mdi[i]+mdi[j]);
        if(sol > ca) sol=ca;
      }


  out << fixed << setprecision(6) << sol << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: fracdec
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <string>
using namespace std;
```

```
#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXN=100010;


int rv[MAXN];


int main() {
  ifstream in("fracdec.in");
  ofstream out("fracdec.out");


  string N; in >> N;
  int D; in >> D;




  int R=0;
  string so="";
  for(int i=0; i<SZ(N); ++i) {
    R=10*R + N[i] - '0';
    if(R>=D) so.PB(R/D+'0');
    else if(SZ(so)) so.PB('0');
    R%=D;
  }
  //cerr << so << " " << R << endl;


  if(!SZ(so)) so="0";
  so += ".";


  string fr="";
  int j=1;
  if(R==0) so.PB('0');
  else {
    fill(rv,rv+D,0);
    while(R && !rv[R]) {
      rv[R]=j++;
      R=10*R;
      if(R>=D) fr.PB(R/D+'0');
      else fr.PB('0');
      R%=D;
    }


    so += fr.substr(0,rv[R]-1);
    if(R) {
      so += "(";
      so += fr.substr(rv[R]-1);
      so += ")";
    }
  }
  //cerr << fr << endl;
  for(int k=0; k<SZ(so); k+=76)
    out << so.substr(k,76) << endl;
}



/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: agrinet
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXN=101;


int hd[MAXN][MAXN];
int di[MAXN], ndi;


int main() {
  ifstream in("agrinet.in");
  ofstream out("agrinet.out");


  fill(di,di+MAXN,-1);


  int N; in >> N;
  for(int i=0; i<N; ++i)
    for(int j=0; j<N; ++j)
      in >> hd[i][j];


  di[0]=0; ndi=1;
  for(int i=1; i<N; ++i)
    di[i]=hd[0][i];


  int sol=0;
  while(ndi<N) {
    int mi=-1;
    for(int i=0; i<N; ++i)
      if(di[i]!=0 && (mi==-1 || di[mi]>di[i]))
        mi=i;
    sol+=di[mi];
    di[mi]=0; ++ndi;

    for(int i=0; i<N; ++i) {
      if(i==mi || di[i]==0) continue;
      if(di[i]>hd[mi][i]) di[i]=hd[mi][i];
    }
  }

  out << sol << endl;
}



/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: inflate
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;

#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXN=10010;

int dp[MAXN];

int main() {
  ifstream in("inflate.in");
  ofstream out("inflate.out");
  fill(dp,dp+MAXN,-1);

  int M, N; in >> M >> N;
  dp[0]=0;
  for(int i=0; i<N; ++i) {
    int p, m; in >> p >> m;
    for(int j=m; j<=M; ++j)
      if(dp[j-m]!=-1 && (dp[j]==-1 || dp[j]<dp[j-m]+p))
        dp[j] = dp[j-m]+p;
  }
  int sol=0;
  for(int j=0; j<=M; ++j)
    if(sol<dp[j]) sol=dp[j];
  out << sol << endl;
}




/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: humble
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <climits>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
typedef long long ll;
const double INF=numeric_limits<double>::infinity();
const int MAXK=101;
const int MAXN=100010;


int hn[MAXN], pi[MAXK], pn[MAXK];


int main() {
  ifstream in("humble.in");
  ofstream out("humble.out");


  int K, N; in >> K >> N;
  for(int i=0; i<K; ++i) in >> pn[i];


  hn[0]=1;
  fill(pi,pi+MAXK,0);


  for(int i=1; i<=N; ++i) {
    int hc=INT_MAX;


    for(int j=0; j<K; ++j) {
      ll tm;
      for(;;) {
        tm=ll(pn[j])*ll(hn[pi[j]]);
        if(tm > ll(hn[i-1])) break;
        ++pi[j];
      }
      if(tm < ll(hc)) hc=int(tm);
    }
    hn[i]=hc;
  }


  out << hn[N] << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: contact
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <vector>
#include <algorithm>
#include <bitset>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXAB=13;
const int MAXD=10000;


struct Pattern {
  int f, l, h;
  Pattern(int f, int l, int h): f(f), l(l), h(h) {}
  bool operator<(const Pattern &ot) const {
    return (f > ot.f) ||
           (f == ot.f && l < ot.l) ||
           (f == ot.f && l == ot.l && h < ot.h);
  }
};


inline string dec(int l, int h) {
  return bitset<MAXAB>(h).to_string().substr(MAXAB-l,l);
}


int cn[MAXAB][MAXD];


int main() {
  ifstream in("contact.in");
  ofstream out("contact.out");
  fill(&cn[0][0],&cn[0][0]+MAXAB*MAXD,0);


  int A, B, N; in >> A >> B >> N; in.ignore();
  string s;
  for(;;) {
    string t; getline(in, t); s+=t;
    if(SZ(t)<80) break;
  }
  int szs=SZ(s);
  //cerr << s << endl << SZ(s) << endl;


  for(int i=A; i<=min(B,szs); ++i) {
    int hs=0;
    for(int j=0; j<i; ++j) hs+=(s[j]-'0')<<(i-1-j);
    ++cn[i][hs];
    for(int j=i; j<szs ; ++j) {
      hs-=(s[j-i]-'0')<<(i-1);
      hs*=2;
      hs+=(s[j]-'0');
      ++cn[i][hs];
    }
  }


  vector<Pattern> fr;
  for(int i=A; i<=B; ++i)
    for(int j=0; j<MAXD; ++j)
      if(cn[i][j])
        fr.PB(Pattern(cn[i][j],i,j));
  sort(ALL(fr));
  //for(int i=0; i<SZ(fr); ++i) cerr << fr[i].f << " " << dec(fr[i].l, fr[i].h) << endl;


  int frsz=SZ(fr);
  int p=0;
  for(int i=0; p<frsz && i<N; ++i) {
    int cf=fr[p].f;
    int sp=p;


    out << cf << endl;
    for(; p<frsz && fr[p].f==cf; ++p) {
      if((p-sp)%6) out << " ";
      out << dec(fr[p].l, fr[p].h);
      if((p-sp)%6==5) out << endl;
    }
    if((p-sp)%6!=0) out << endl;
  }
}




/****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: stamps
*/
```

```
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXK=202;
const int MAXV=10010;
const int MAXN=55;


int dp[MAXK*MAXV];
int co[MAXN];


int main() {
  ifstream in("stamps.in");
  ofstream out("stamps.out");
  fill(dp,dp+MAXK*MAXV,-1);
  dp[0]=0;


  int N, K; in >> K >> N;
  for(int i=0; i<N; ++i) in >> co[i];


  int sol=1;
  for(;;) {
    dp[sol]=MAXK;
    for(int i=0; i<N; ++i)
      if(sol-co[i]>=0 && dp[sol]>dp[sol-co[i]]+1) {
        dp[sol] = dp[sol-co[i]] + 1;
      }
    if(dp[sol] > K) break;
    ++sol;
  }


  out << --sol << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: fact4
*/
#define NDEBUG
#include <cassert>
#include <limits>
```

```
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXL=20020;


int dg[MAXL], ps, pe;


int main() {
  ifstream in("fact4.in");
  ofstream out("fact4.out");


  int N; in >> N;
  ps=0; pe=1;
  dg[ps]=1;
  for(int i=2; i<=N; ++i) {
    int c=0;
    for(int p=ps; p<pe; ++p) {
      int m=dg[p]*i+c;
      dg[p]=m%10;
      c=m/10;
    }
    while(c) {
      dg[pe]=c%10;
      c/=10;
      ++pe;
    }
    while(!dg[ps]) ++ps;
  }
  out << dg[ps] << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: kimbits
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <bitset>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
```

```
#define PB push_back


const int MAXN=32;
const int MAXL=32;


unsigned int nc[MAXN][MAXN], ac[MAXN][MAXN];


int main() {
  ifstream cin("kimbits.in");
  ofstream cout("kimbits.out");


  for(int i=0; i<MAXN; ++i)
    nc[i][0] = nc[i][i] = 1;


  for(int i=1; i<MAXN; ++i)
    for(int j=1; j<MAXN; ++j)
      nc[i][j] = nc[i-1][j-1] + nc[i-1][j];


  for(int i=0; i<MAXN; ++i) {
    ac[i][0] = nc[i][0];
    for(int j=1; j<MAXN; ++j)
      ac[i][j] = ac[i][j-1] + nc[i][j];
  }


  unsigned int N, L, I; cin >> N >> L >> I;


  bitset<32> sol;
  unsigned int rc=I-1;
  unsigned int ro=L;
  for(int i=N-1; i>=0; --i) {
    if(rc >= ac[i][ro]) {
      sol[i]=1;
      rc-=ac[i][ro];
      --ro;
    } else {
      sol[i]=0;
    }
  }


  cout << sol.to_string().substr(32-N,N) << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: butter
*/
#define NDEBUG
```

```
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <algorithm>
#include <climits>
#include <vector>
#include <functional>
#include <queue>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
const int MAXN=505;
const int MAXP=808;
const int MAXC=1500;
typedef pair<int,int> pii;
typedef vector<pii> vpii;
typedef vector<vpii> vvpii;


int di[MAXP][MAXP];
int pc[MAXP];
bool visited[MAXP];


int main() {
  ifstream in("butter.in");
  ofstream out("butter.out");


  fill(&di[0][0],&di[0][0]+MAXP*MAXP,INT_MAX);
  fill(&pc[0],&pc[0]+MAXP,0);


  int N, P, C; in >> N >> P >> C;
  for(int i=0; i<N; ++i) {
    int t; in >> t;
    ++pc[t-1];
  }


  vvpii al(MAXP);
  for(int i=0; i<C; ++i) {
    int v1, v2, l; in >> v1 >> v2 >> l;
    al[v1-1].PB(MP(l,v2-1));
    al[v2-1].PB(MP(l,v1-1));
  }
  for(int i=0; i<P; ++i)
    sort(ALL(al[i]));


  for(int i=0; i<P; ++i) {
    priority_queue<pii, vector<pii>, greater<pii> > nv;
    fill(&visited[0],&visited[0]+MAXP,false);
    nv.push(MP(0,i));
    while(!nv.empty()) {
      pii cv=nv.top(); nv.pop();
      if(visited[cv.second]) continue;


      visited[cv.second] = true;
      di[i][cv.second] = cv.first;


      for(int j=0; j<SZ(al[cv.second]); ++j)
        if(!visited[al[cv.second][j].second])
          nv.push(MP(cv.first+al[cv.second][j].first, al[cv.second][j].second));
    }
  }


  int sol=INT_MAX;
  for(int i=0; i<P; ++i) {
    int csol=0;
    for(int j=0; j<P; ++j)
      csol += pc[j]*di[i][j];
    if(sol>csol) sol=csol;
  }


  out << sol << endl;
}




/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: msquare
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <set>
#include <queue>
#include <map>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define SS stringstream
const double INF=numeric_limits<double>::infinity();
typedef pair<int,int> pii;
const int P10[8] = {1, 10, 100, 1000, 10000, 100000, 1000000, 10000000 };
const int MAXL=40500;


char se[MAXL];
```

```
inline int trn(int n, char c) {
  int r=0;
  if(c=='A')
    while(n) {
      int t=n%10;
      n/=10;
      r = 10*r + t;
    }
  else if(c=='B') {
    int s1=n/P10[5];
    int s2=(n%P10[5])/P10[4];
    int s3=(n%P10[4])/P10[3];
    int s4=n%P10[3];
    r = s2*P10[7] + s1*P10[4] + s4*P10[1] + s3;
  }
  else if(c=='C') {
    int s1=n/P10[7];
    int s2=(n%P10[7])/P10[6];
    int s3=(n%P10[6])/P10[5];
    int s4=(n%P10[5])/P10[3];
    int s5=(n%P10[3])/P10[2];
    int s6=(n%P10[2])/P10[1];
    int s7=n%P10[1];
    r = s1*P10[7] + s6*P10[6] + s2*P10[5] + s4*P10[3] + s3*P10[2] + s5*P10[1] + s7;
  }


  return r;
}


int main() {
  ifstream in("msquare.in");
  ofstream out("msquare.out");


  int E=0;
  for(int i=0; i<8; ++i) {
    int t; in >> t;
    E = 10*E + t;
  }
  //cerr << trn(12345678, 'A') << endl;
  //cerr << trn(12345678, 'B') << endl;
  //cerr << trn(12345678, 'C') << endl;


  map<int,int> pr;
  map<int,char> ag;
  deque<int> cn;


  cn.PB(12345678); pr[12345678]=0; ag[12345678]='-';
  for(;;) {
    int cc=cn.front(); cn.pop_front();
    if(cc == E) break;


    for(char c='A'; c<='C'; ++c) {
      int nc=trn(cc, c);
      if(pr.count(nc)) continue;
      pr[nc] = cc;
      ag[nc] = c;
      cn.PB(nc);
    }
  }


  int le=0, nu=E;
  while(nu!=12345678) {
    se[le] = ag[nu];
    nu = pr[nu];
    ++le;
  }
  se[le] = 0;


  reverse(se,se+le);
  out << le << endl;
  out << se << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: ratios
*/
#define NDEBUG
#include <cassert>
#include <limits>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back


int d[3], m[3][3], t[3][3], s[3];


inline int det(int c[][3]) {
  int r=0;
  for(int i=0; i<3; ++i) {
    r += c[0][i]*c[1][(i+1)%3]*c[2][(i+2)%3];
    r -= c[0][i]*c[1][(i+3-1)%3]*c[2][(i+3-2)%3];
  }
  return r;
}


inline int gcd(int a, int b) {
  if(b==0) return a;
  return gcd(b,a%b);
}
```

```
int main() {
  ifstream in("ratios.in");
  ofstream out("ratios.out");


  for(int i=0; i<3; ++i) in >> d[i];
  for(int i=0; i<3; ++i)
    for(int j=0; j<3; ++j)
      in >> m[j][i];


  int dd=det(m);
  int sd=dd/abs(dd);
  dd*=sd;
  bool solvable=dd;


  int dt[3];
  int cg;
  if(solvable) {
    for(int i=0; i<3 && solvable; ++i) {
      copy(m[0],m[0]+3*3,t[0]);
      for(int j=0; j<3; ++j) t[j][i] = d[j];
      dt[i] = sd*det(t);
      solvable = dt[i]>=0;
    }
    cg=gcd(dt[0],dt[1]);
    cg=gcd(cg,dt[2]);
    cg=gcd(cg,dd);
  }


  dd /= cg;
  for(int i=0; i<3; ++i) {
    dt[i] /= cg;
  }


  fill(s,s+3,0);
  for (int i=0; i<3; ++i) {
    for (int j=0; j<3; ++j) {
      s[i] += dt[j] * m[i][j];
    }
    //cerr << s[i] << endl;
  }


  if(!solvable) {
    out << "NONE" << endl;
  } else {
    int f=1;
    for(int i=0; i<3; ++i) {
      if(s[i]) f=max(d[i]/s[i],1);
      out << (i?" ":"") << f*dt[i];
    }
    out << " " << f*dd << endl;
  }
}
```

```
/****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: spin
*/
//#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back


const int NUMW=5;


int as[NUMW], an[NUMW], aws[NUMW][NUMW], awe[NUMW][NUMW];


int main() {
  ifstream cin("spin.in");
  ofstream cout("spin.out");


  for(int i=0; i<NUMW; ++i) {
    cin >> as[i] >> an[i];
    for(int j=0; j<an[i]; ++j)
      cin >> aws[i][j] >> awe[i][j];
  }


  for(int i=0; i<360; ++i) {
    int sp[360] = {0};
    for(int j=0; j<NUMW; ++j) {
      for(int k=0; k<an[j]; ++k) {
        for(int l=aws[j][k]; l<=aws[j][k]+awe[j][k]; ++l) {
          int p=l%360;
          ++sp[p];
          if(sp[p]==NUMW) {
            cout << i << endl;
            return 0;
          }
        }


        aws[j][k] += as[j];
      }
    }
  }


  cout << "none" << endl;
}
```

```
/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: fence
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <algorithm>
#include <stack>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back


const int MAXN=505;


int ed[MAXN][MAXN];


int main() {
  ifstream cin("fence.in");
  ofstream cout("fence.out");


  fill(&ed[0][0],&ed[0][0]+MAXN*MAXN,0);


  int N; cin >> N;
  for(int i=0; i<N; ++i) {
    int x, y; cin >> x >> y;
    ++ed[x][y];
    ++ed[y][x];
  }


  int s=MAXN;
  for(int i=1; i<MAXN; ++i) {
    int c=0;
    for(int j=1; j<MAXN; ++j)
      c += ed[i][j];
    if(c>0 && i<s)
      s=i;
    if(c%2) {
      s=i;
      break;
    }
  }


  stack<int> np, tr;
  np.push(s);
  while(!np.empty()) {
    int n=np.top();
    int e=0;
    for(e=1; e<MAXN && !ed[n][e]; ++e);
    if(e<MAXN) {
      --ed[n][e];
      --ed[e][n];
      np.push(e);
    } else {
      tr.push(n);
      np.pop();
    }
  }


  while(!tr.empty()) {
    cout << tr.top() << endl;
    tr.pop();
  }
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: shopping
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back


const int MAXB=6;
const int MAXS=101;


int on[MAXS], oc[MAXS][MAXB], ok[MAXS][MAXB], op[MAXS];
int uc[MAXB], uk[MAXB], up[MAXB];
int dp[MAXB][MAXB][MAXB][MAXB][MAXB];


int main() {
  ifstream cin("shopping.in");
  ofstream cout("shopping.out");


  int S; cin >> S;
  for(int i=0; i<S; ++i) {
    cin >> on[i];
    for(int j=0; j<on[i]; ++j)
      cin >> oc[i][j] >> ok[i][j];
    cin >> op[i];
  }


  int B; cin >> B;
  for(int i=0; i<B; ++i)
    cin >> uc[i] >> uk[i] >> up[i];


  fill(&dp[0][0][0][0][0],&dp[0][0][0][0][0]+MAXB*MAXB*MAXB*MAXB*MAXB,-1);
  dp[0][0][0][0][0]=0;


  int p[5];
  for(p[0]=0; p[0]<MAXB; ++p[0])
    for(p[1]=0; p[1]<MAXB; ++p[1])
      for(p[2]=0; p[2]<MAXB; ++p[2])
        for(p[3]=0; p[3]<MAXB; ++p[3])
          for(p[4]=0; p[4]<MAXB; ++p[4])
            dp[p[0]][p[1]][p[2]][p[3]][p[4]] = p[0]*up[0] + p[1]*up[1] + p[2]*up[2] + p[3]*up[3] +
p[4]*up[4];


  for(int i=0; i<S; ++i) {
    int t[MAXB] = {0};
    for(int k=0; k<B; ++k)
      for(int l=0; l<on[i]; ++l)
        if(uc[k]==oc[i][l]) {
          t[k] = ok[i][l];
          break;
        }


    int q[5];
    for(q[0]=0; q[0]<=uk[0] || (B<1 && q[0]==0); ++q[0])
      for(q[1]=0; q[1]<=uk[1] || (B<2 && q[1]==0); ++q[1])
        for(q[2]=0; q[2]<=uk[2] || (B<3 && q[2]==0); ++q[2])
          for(q[3]=0; q[3]<=uk[3] || (B<4 && q[3]==0); ++q[3])
            for(q[4]=0; q[4]<=uk[4] || (B<5 && q[4]==0); ++q[4])
              if(q[0]-t[0]>=0 && q[1]-t[1]>=0 && q[2]-t[2]>=0 && q[3]-t[3]>=0 && q[4]-t[4]>=0) {
                int &dpp = dp[q[0]-t[0]][q[1]-t[1]][q[2]-t[2]][q[3]-t[3]][q[4]-t[4]];
                int &dpc = dp[q[0]][q[1]][q[2]][q[3]][q[4]];
                if(dpp!=-1 && (dpc==-1 || dpc > dpp+op[i]))
                  dpc = dpp+op[i];
              }
  }


  cout << dp[(B>0) ? uk[0] : 0][(B>1) ? uk[1] : 0][(B>2) ? uk[2] : 0][(B>3) ? uk[3] : 0][(B>4) ?
uk[4] : 0] << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: camelot
```

```
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <sstream>
#include <queue>
#include <climits>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define PF pop_front
#define SS stringstream


struct KT {
  int r, c, d;
  KT(int ri, int ci, int di): r(ri), c(ci), d(di) {}
};


const int MAXC=26;
const int MAXR=30;
const int KNMR[8]={-2,-2,-1, 1, 2, 2, 1,-1};
const int KNMC[8]={-1, 1, 2, 2, 1,-1,-2,-2};


int nn, nr[MAXR*MAXC], nc[MAXR*MAXC];
int kr, kc;
int di[MAXR][MAXC][MAXR][MAXC];


int main() {
  ifstream cin("camelot.in");
  ofstream cout("camelot.out");


  int R, C; cin >> R >> C; cin.ignore();
  string tms1; getline(cin, tms1);
  char tmc1; SS(tms1) >> tmc1 >> kr; --kr; kc=tmc1-'A';


  nn=0;
  for(;;) {
    string tms2; getline(cin,tms2);
    if(tms2.empty()) break;


    SS tmss(tms2); char tmc2;
    while(tmss >> tmc2 >> nr[nn]) {
      --nr[nn];
      nc[nn]=tmc2-'A';
      ++nn;
    }
  }


  fill(&di[0][0][0][0],&di[0][0][0][0]+MAXR*MAXC*MAXR*MAXC,-1);
```

```cpp
for(int r=0; r<R; ++r) {
  for(int c=0; c<C; ++c) {
    deque<KT> bfq; bfq.PB(KT(r,c,0));
    di[r][c][r][c]=0;
    while(!bfq.empty()) {
      KT nkt=bfq.front(); bfq.PF();
      for(int i=0; i<8; ++i) {
        int pr=nkt.r+KNMR[i];
        int pc=nkt.c+KNMC[i];
        int pd=nkt.d+1;
        //cerr << pd << endl;
        if(pr>=0 && pr<R && pc>=0 && pc<C && di[r][c][pr][pc]==-1) {
          bfq.PB(KT(pr,pc,pd));
          di[r][c][pr][pc]=pd;
        }
      }
    }
  }
}


int sol=INT_MAX;
for(int r=0; r<R; ++r) {
  for(int c=0; c<C; ++c) {
    int pnd=0;
    for(int n=0; n<nn; ++n) {
      int cdi = di[r][c][nr[n]][nc[n]];
      if(cdi==-1) {
        pnd=INT_MAX;
        break;
      }
      pnd += cdi;
    }
    if(pnd==INT_MAX) continue;


    int pkd1 = max( abs(r-kr), abs(c-kc) );


    if(sol>pnd+pkd1) sol=pnd+pkd1;


    for(int sr=max(kr-2,0); sr<=min(kr+2,R-1); ++sr)
      for(int sc=max(kc-2,0); sc<=min(kc+2,C-1); ++sc)
        for(int n=0; n<nn; ++n) {
          int di1 = di[r][c][nr[n]][nc[n]];
          int di2 = di[r][c][sr][sc];
          int di3 = di[sr][sc][nr[n]][nc[n]];
          if(di1==-1 || di2==-1 || di3==-1) continue;


          pnd -= di1;
          pnd += di2 + di3;
          int pkd2 = max( abs(sr-kr), abs(sc-kc) );
          if(sol>pnd+pkd2) sol=pnd+pkd2;
          pnd -= di[r][c][sr][sc] + di[sr][sc][nr[n]][nc[n]];
          pnd += di[r][c][nr[n]][nc[n]];
        }
  }
}
```

```
  cout << sol << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: range
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <climits>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define PF pop_front


const int MAXN=255;


int fi[MAXN][MAXN];


int main() {
  ifstream cin("range.in");
  ofstream cout("range.out");


  int N; cin >> N;
  for(int i=0; i<N; ++i) {
    string tmp; cin >> tmp;
    for(int j=0; j<N; ++j)
      fi[i][j] = tmp[j] - '0';
  }


  for(int s=2; s<=N; ++s)
    for(int i=0; i<N-s+1; ++i)
      for(int j=0; j<N-s+1; ++j) {
        int c=0;
        for(int di=0; di<2; ++di)
          for(int dj=0; dj<2; ++dj)
            c += (fi[i+di][j+dj] >= s-1);
        if(c==4) fi[i][j]=s;
      }


  for(int s=2; s<=N; ++s) {
    int r=0;
    for(int i=0; i<N-s+1; ++i)
      for(int j=0; j<N-s+1; ++j)
        r += (fi[i][j] >= s);
    if(!r) break;
    cout << s << ' ' << r << endl;
  }
}




/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: game1
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define PF pop_front


const int MAXN=101;


int dp[MAXN][MAXN][2];
int bd[MAXN];


int main() {
  ifstream cin("game1.in");
  ofstream cout("game1.out");


  int N; cin >> N;
  int il = (N+1)%2;
  int ip = N%2;
  for(int i=0; i<N; ++i) {
    cin >> bd[i];
    dp[1][i][il]=bd[i];
    dp[1][i][ip]=0;
  }


  for(int s=2; s<=N; ++s) {
    for(int i=0; i<N-s+1; ++i) {
      int ka=(N+s)%2;
      int kp=(N+s-1)%2;
      int t1 = dp[s-1][i][ka] + bd[i+s-1];
      int t2 = dp[s-1][i+1][ka] + bd[i];
      if(t1 > t2) {
        dp[s][i][ka] = t1;
        dp[s][i][kp] = dp[s-1][i][kp];
      } else {
        dp[s][i][ka] = t2;
        dp[s][i][kp] = dp[s-1][i+1][kp];
      }
    }
  }


  cout << dp[N][0][0] << ' ' << dp[N][0][1] << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: heritage
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define PF pop_front


int border[30];
int sz, cnt=0;
string opr, oin, opo(30,' ');


void walk_tree(int ip, int ii) {
  int dp=1;


  //haz-left?
  if(ip+dp<sz)
    for(int i=ii-1; i>=0; --i)
      if(opr[ip+dp]==oin[i]) {
        border[i]=ii;
        walk_tree(ip+dp,i);
        ++dp;
        break;
      }


  //haz-right?
  bool right_found=false;
  for(int p=ip+dp; p<sz && !right_found; ++p)
    for(int i=ii+1; i<border[ii]; ++i)
      if(opr[p]==oin[i]) {
        border[i]=border[ii];
        walk_tree(p,i);
        right_found=true;
        break;
      }


  opo[cnt++] = opr[ip];
}


int main() {
  ifstream cin("heritage.in");
  ofstream cout("heritage.out");


  cin >> oin >> opr;
  sz = SZ(oin);


  int ii=0;
  for(; ii<sz && oin[ii]!=opr[0]; ++ii);


  fill(border,border+30,-1);
  border[ii]=sz;


  walk_tree(0,ii);
  cout << opo.substr(0,SZ(oin)) << endl;
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: fence9
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define PF pop_front


int cnt_without(int L, int H) {
  int r=0;
  for(int i=1; i<L; ++i) {
    int t=H*i;
    int d=t/L;
    r += (t%L==0) ? d-1 : d;
  }
  return r;
}
```

```
int cnt_with(int L, int H) {
  int r=0;
  for(int i=1; i<L; ++i) {
    int t=H*i;
    r += t/L;
  }
  return r;
}


int main() {
  ifstream cin("fence9.in");
  ofstream cout("fence9.out");


  int n, m, p; cin >> n >> m >> p;
  int r = cnt_without(n,m);
  if(p<n) r -= cnt_with(n-p,m);
  if(p>n) r += cnt_without(p-n,m) + (n!=0 ? m-1 : 0);


  cout << r << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: rockers
*/
#define NDEBUG
#include <cassert>
#include <fstream>
#include <iostream>
#include <algorithm>
using namespace std;


#define ALL(x) (x).begin(), (x).end()
#define SZ(a) ((int) (a).size())
#define MP make_pair
#define PB push_back
#define PF pop_front


const int MAXN=21;


int aa[MAXN];
int dp[MAXN*MAXN];
int mem[MAXN][MAXN][MAXN];
int N, T, M;


int rec(int s, int e, int t) {
  int &r = mem[s][e][t];
  if(r!=-1) return r;


  r=0;
  if(t==1) {
    fill(dp,dp+MAXN*MAXN,-1);
    dp[0]=0;
    for(int i=s; i<e; ++i)
      for(int j=T; j>=aa[i]; --j)
        if(dp[j-aa[i]]!=-1 && (dp[j] < dp[j-aa[i]]+1)) {
          dp[j] = dp[j-aa[i]]+1;
          r=max(r,dp[j]);
        }
    return r;
  }
  for(int k=s; k<e; ++k)
    for(int g=1; g<t; ++g)
      r = max(r, rec(s,k,g) + rec(k,e,t-g));
  return r;
}


int main() {
  ifstream cin("rockers.in");
  ofstream cout("rockers.out");


  fill(&mem[0][0][0],&mem[0][0][0]+MAXN*MAXN*MAXN,-1);


  cin >> N >> T >> M;
  for(int i=0; i<N; ++i)
    cin >> aa[i];


  cout << rec(0,N,M) << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: nuggets
*/
// train.usaco.com - nuggets
// Restricted version of the Coin Ptroblem


// Observation:
// - for two packages (if coprime) n and m => n*m - n - m
// - for two packages (not coprime) => Infinity


// Solution:
// - check all numbers till 65024 -> DP approach


#include <iostream>
```

```cpp
#include <fstream>


const int MAXN = 10;
const int MAXP = 65024 + 1;


inline int gcd(int a, int b) {
  if (b == 0) return a;
  return gcd(b, a % b);
}


int main() {
  std::ifstream cin("nuggets.in");
  std::ofstream cout("nuggets.out");


  int N, a[MAXN];
  cin >> N;
  for (int i = 0; i < N; ++i) {
    cin >> a[i];
  }


  int gcd_all = a[0];
  for (int i = 1; i < N; ++i) {
    gcd_all = gcd(gcd_all, a[i]);
  }


  if (gcd_all > 1) {
    cout << 0 << std::endl;
    return 0;
  }


  bool possible[MAXP] = {true}; // {true, false, false, ...}
  int sol = 0;


  for (int i = 1; i < MAXP; ++i) {
    for (int j = 0; j < N; ++j) {
      if (i >= a[j] && possible[i - a[j]]) {
        possible[i] = true;
        break;
      }
    }
    if (!possible[i]) {
      sol = i;
    }
  }


  cout << sol << std::endl;
  return 0;
}
```

```
/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: fence6
*/
// usaco - fence6
// Dijkstra on every edge and find smallest cycle.


#include <fstream>
#include <algorithm>
#include <set>
#include <queue>
#include <stack>
#include <utility>
#include <functional>
#include <climits>
#include <cassert>


const int MAXN = 2 * 100 + 1;
using std::pair;


class Edge {
 public:
  int di;
  int la;
  int ne;
  Edge(): di(0), la(0), ne(0) {}
  Edge(int dii, int lai, int nei): di(dii), la(lai), ne(nei) {}
  bool operator<(const Edge &ot) const {
    return (di < ot.di) ||
      (di == ot.di && la < ot.la) ||
      (di == ot.di && la == ot.la && ne < ot.ne);
  }
  bool operator>(const Edge &ot) const {
    return (di > ot.di) ||
      (di == ot.di && la > ot.la) ||
      (di == ot.di && la == ot.la && ne > ot.ne);
  }
};


class AdjacencyList {
 private:
  int size_;
  Edge ve_[MAXN][MAXN];
  int num_ve_[MAXN];
  int id_[MAXN][MAXN];
  int num_id_[MAXN];


 public:
  const Edge &ve(int i, int j) { return ve_[i][j]; }
  int num_ve(int i) { return num_ve_[i]; }
  int size() { return this->size_; }


  AdjacencyList(): size_(0) {
    std::fill(&num_ve_[0], &num_ve_[0]+MAXN, 0);
    std::fill(&num_id_[0], &num_id_[0]+MAXN, 0);
  }


  void insert(int s1n_i, int *s1_i, int s2n_i, int *s2_i, int sl) {
    int sn[2] = { s1n_i, s2n_i };
    int *s[2] = { s1_i, s2_i };
    int sid[2] = {-1, -1};
    int sla = s[0][0];


    for (int i = 0; i < 2; ++i) {
      std::sort(&s[i][0], &s[i][0] + sn[i]);
      for (int j = 0; (j < this->size_) && (sid[i] == -1); ++j) {
        if (num_id_[j] != sn[i] || (i == 1 && j == sid[0])) {
          continue;
        }


        sid[i] = j;
        for (int k = 0; k < num_id_[j]; ++k) {
          if (id_[j][k] != s[i][k]) {
            sid[i] = -1;
            break;
          }
        }
      }
    }


    for (int i = 0; i < 2; ++i) {
      if (sid[i] == -1) {
        num_id_[this->size_] = sn[i];
        std::copy(&s[i][0], &s[i][0] + sn[i], &id_[this->size_][0]);
        sid[i] = this->size_;
        ++this->size_;
      }
    }


    for (int i = 0; i < 2; ++i) {
      ve_[sid[i]][num_ve_[sid[i]]] = Edge(sl, sla, sid[1-i]);
      ++num_ve_[sid[i]];
    }
  }


  void sort() {
    for (int i = 0; i < this->size_; ++i) {
      std::sort(&ve_[i][0], &ve_[i][0] + num_ve_[i]);
    }
  }
};


int main() {
  std::ifstream fin("fence6.in");
  std::ofstream fout("fence6.out");


  int N;
  fin >> N;
```

```
AdjacencyList graph;


for (int i = 0; i < N; ++i) {
  int sid, sl, sn[2];
  fin >> sid >> sl >> sn[0] >> sn[1];


  int vertid[2][MAXN] = { {sid}, {sid} };


  for (int j = 0; j < 2; ++j) {
    for (int k = 1; k <= sn[j]; ++k) {
      fin >> vertid[j][k];
    }
  }
  graph.insert(sn[0]+1, &vertid[0][0], sn[1]+1, &vertid[1][0], sl);
}


int min_cyc = INT_MAX;
for (int i = 0; i < graph.size(); ++i) { // Dijkstra on every vertex
  std::set<int> visited;
  std::set<int> used;
  std::priority_queue<Edge, std::vector<Edge>, std::greater<Edge>> pq;
  int pdi[MAXN] = {0};
  //std::stack<Edge> pq;


  for (int j = 0; j < graph.num_ve(i); ++j) {
    pq.push(graph.ve(i, j));
    visited.insert(i);
  }


  while (!pq.empty()) {
    int cdi = pq.top().di;
    int cla = pq.top().la;
    int cve = pq.top().ne;
    pq.pop();


    if (visited.count(cve)) {
      assert (cdi > 0);
      int tdi = pdi[cve] + cdi;
      if (tdi < min_cyc) {
        min_cyc = tdi;
      }
      break;
    }


    if (used.count(cla)) {
      continue;
    }
    visited.insert(cve);
    used.insert(cla);
    pdi[cve] = cdi;
```

```
      for (int j = 0; j < graph.num_ve(cve); ++j) {
        int ndi = graph.ve(cve, j).di;
        int nla = graph.ve(cve, j).la;
        int nve = graph.ve(cve, j).ne;
        if (!visited.count(nve) && !used.count(nla)) {
          pq.push(Edge(cdi+ndi, nla, nve));
        }
      }//for
    }//while


  }//for


  fout << min_cyc << std::endl;
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: ditch
*/
#include <fstream>
#include <climits>


const int MAXN = 220;


int adj[MAXN][MAXN] = {{0}};


int main() {
  std::ifstream fin("ditch.in");
  std::ofstream fout("ditch.out");


  int N, M;
  fin >> N >> M;
  for (int i = 0; i < N; ++i) {
    int sr, sn, cp;
    fin >> sr >> sn >> cp;
    adj[sr][sn] += cp;
  }


  int total_flow = 0;
  while (true) {
    int flow[MAXN] = {0, INT_MAX};
    int prev[MAXN] = {0};
    bool visited[MAXN] = {false};


    int cn;
    while (true) {
      cn = 0;
      for (int i = 1, cf = 0; i <= M; ++i) {
        if (flow[i] > cf && !visited[i]) {
          cf = flow[i];
          cn = i;
        }
      }

      if (cn == 0 || cn == M) {
        break;
      }

      visited[cn] = true;

      for (int i = 1; i <= M; ++i) {
        int nf = std::min(flow[cn], adj[cn][i]);
        if (nf > flow[i] && !visited[i]) {
          flow[i] = nf;
          prev[i] = cn;
        }
      }
    }

    if (cn == 0) {
      break;
    }

    total_flow += flow[M];
    while(prev[cn]) {
      adj[ prev[cn] ][ cn ] -= flow[M];
      adj[ cn ][ prev[cn] ] += flow[M];
      cn = prev[cn];
    }
  }
  fout << total_flow << std::endl;
  fin.close();
  fout.close();
}



/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: stall4
*/
#include <fstream>
#include <climits>

const int MAXN = 220;
const int SRC = 0;
const int DST = 2*MAXN;
int adj[2*MAXN+1][2*MAXN+1];
```

```
inline int st(int n) { return n + MAXN; }


int main() {
  std::ifstream fin("stall4.in");
  std::ofstream fout("stall4.out");


  int N, M;
  fin >> N >> M;
  for (int i = 1; i <= N; ++i) {
    int S; fin >> S;
    for (int j = 0; j < S; ++j) {
      int p; fin >> p;
      adj[ i ][ st(p) ] = 1;
    }
  }
  for (int i = 1; i <= N; ++i) {
    adj[ SRC ][ i ] = 1;
  }
  for (int i = 1; i <= M; ++i) {
    adj[ st(i) ][ DST ] = 1;
  }


  int total = 0;
  while (true) {
    int flow[2*MAXN+1] = { INT_MAX };
    int prev[2*MAXN+1] = { -1 };
    int visited[2*MAXN+1] = { false };


    int cn;
    while (true) {
      cn = -1;
      for (int i = 0, cf = 0; i <= DST; ++i) {
        if (flow[i] > cf && !visited[i]) {
          cf = flow[i];
          cn = i;
        }
      }


      if (cn == -1 || cn == DST) {
        break;
      }


      visited[cn] = true;
      for (int i = 0; i <= DST; ++i) {
        int nf = std::min(flow[cn], adj[cn][i]);
        if (nf > flow[i] && !visited[i]) {
          flow[i] = nf;
          prev[i] = cn;
        }
      }
    }


    if (cn == -1) {
      break;
    }
    total += flow[DST];
    while(prev[cn] != -1) {
      adj[ prev[cn] ][ cn ] -= flow[DST];
      adj[ cn ][ prev[cn] ] += flow[DST];
      cn = prev[cn];
    }
  }


  fout << total << std::endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: job
*/
#include <fstream>
#include <climits>


const int MAXT = 20020;
const int MAXAB = 33;


int ma[MAXAB], mb[MAXAB];
int ta[MAXAB], tb[MAXAB];
int ab[MAXT];


int main() {
  std::ifstream fin("job.in");
  std::ofstream fout("job.out");


  int N, M1, M2;
  fin >> N >> M1 >> M2;


  for (int i = 0; i < M1; ++i) {
    fin >> ma[i];
  }
  for (int i = 0; i < M2; ++i) {
    fin >> mb[i];
  }


  int Ta = 0;
  for (int i = 0; ; ++i) {
    int mw = -1;
    int tw = INT_MAX;
    for (int j = 0; j < M1; ++j) {
      int tj = ta[j] + ma[j];
      if (tj < tw) {
        tw = tj;
        mw = j;
      }
    }
    ta[mw] = tw;
    ab[tw] += 1;
    if (i == N-1) {
      Ta = tw;
      break;
    }
  }


  int mm = INT_MAX;
  for (int i = 0; i < M2; ++i) {
    if (mm > mb[i]) {
      mm = mb[i];
    }
  }
  int Tb = Ta + mm;
  for (int i = 0; i < M2; ++i) {
    tb[i] = Tb - mb[i];
  }


  for (int t = Ta; t >= 1; --t) {
    int cab = 0;
    while (cab < ab[t]) {
      int mw = -1;
      int tw = INT_MIN;
      for (int i = 0; i < M2; ++i) {
        if (tw < tb[i]) {
          tw = tb[i];
          mw = i;
        }
      }
      if (tw < t) {
        int td = t - tw;
        for (int i = 0; i < M2; ++i) {
          tb[i] += td;
        }
        Tb += td;
      }
      tb[mw] -= mb[mw];
      ++cab;
    }
  }


  fout << Ta << " " << Tb << std::endl;
  fin.close();
  fout.close();
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: buylow
*/
#include <fstream>
```

```cpp
#include <algorithm>
#include <sstream>
using std::string;


const int MAXN = 5050;
const int MAXD = 300;


class BigInt {
 private:
  int sz;
  int dig[MAXD];


 public:
  BigInt(): sz(1), dig{0} {}
  BigInt(int);
  string str() const;
  BigInt operator+(const BigInt &) const;
};


BigInt::BigInt(int n) {
  sz = 0;
  while(n) {
    dig[sz] = n % 10;
    n /= 10;
    ++sz;
  }
}


string BigInt::str() const {
  std::stringstream ss;
  for (int i = sz-1; i >= 0; --i) {
    ss << dig[i];
  }
  return ss.str();
}


std::ostream &operator<<(std::ostream &os, const BigInt &n) {
  return os << n.str();
}


BigInt BigInt::operator+(const BigInt &ot) const {
  BigInt r;
  int &ci = r.sz; ci = 0;
  int ca = 0;
  int le = std::max(this->sz, ot.sz);
  while (ci < le || ca > 0) {
    ca += this->dig[ci] + ot.dig[ci];
    r.dig[ci] = ca % 10;
    ca /= 10;
    ++ci;
  }
  return r;
}
```

```
int ms[MAXN];
BigInt ns[MAXN];
int aa[MAXN];
int pi[MAXN];


int main() {
  std::ifstream fin("buylow.in");
  std::ofstream fout("buylow.out");


  int N; fin >> N;
  for (int i = 0; i < N; ++i) {
    fin >> aa[i];
  }


  std::fill(&pi[0], &pi[0]+MAXN, -1);
  for (int i = 0; i < N; ++i) {
    if (pi[i] == -1) {
      int ca = aa[i];
      int ci = i;
      for (int j = i+1; j < N; ++j) {
        if (aa[j] == ca) {
          pi[j] = ci;
          ci = j;
        }
      }
    }
  }


  for (int i = N-1; i >= 0; --i) {
    ms[i] = 1;
    ns[i] = 1;
    for (int j = i+1; j < N; ++j) {
      if (aa[i] > aa[j] && pi[j] < i) {
        if (ms[i] < ms[j] + 1) {
          ms[i] = ms[j] + 1;
          ns[i] = ns[j];
        }
        else if (ms[i] == ms[j] + 1) {
          ns[i] = ns[i] + ns[j];
        }
      }
    }
  }


  int msol = 0;
  BigInt nsol = 0;
  for (int i = N-1; i >= 0; --i) {
    if (pi[i] == -1) {
      if (ms[i] > msol) {
        msol = ms[i];
        nsol = ns[i];
      }
      else if (ms[i] == msol) {
        nsol = nsol + ns[i];
      }
    }
  }
  fout << msol << ' ' << nsol << std::endl;
}



/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: race3
*/
#include <fstream>
#include <stack>
#include <algorithm>


const int MAXN = 55;


int al[MAXN] = {0}, aa[MAXN][MAXN];
bool visited1[MAXN], visited2[MAXN];
int s1l = 0, s1[MAXN];
int s2l = 0, s2[MAXN];


void traverse(int s, bool *v) {
  std::stack<int> next;
  next.push(s);
  while (!next.empty()) {
    int cn = next.top(); next.pop();
    for (int i = 0; i < al[cn]; ++i) {
      if (!v[aa[cn][i]]) {
        next.push(aa[cn][i]);
        v[aa[cn][i]] = true;
      }
    }
  }
}


int main() {
  std::ifstream fin("race3.in");
  std::ofstream fout("race3.out");


  int N = 0;
  while (true) {
    int tmp; fin >> tmp;
    if (tmp == -1) {
      break;
    }
    else if (tmp == -2) {
      ++N;
    }
    else {
      aa[N][ al[N] ] = tmp;
      ++al[N];
    }
  }
  for (int i = 1; i < N-1; ++i) {
    std::fill(&visited1[0], &visited1[0]+MAXN, false);
    visited1[0] = true;
    visited1[i] = true;
    traverse(0, visited1);


    if (!visited1[N-1]) {
      s1[s1l] = i;
      ++s1l;


      std::fill(&visited2[0], &visited2[0]+MAXN, false);
      visited2[i] = true;
      traverse(i, visited2);


      bool valid = true;
      for (int j = 0; j < N && valid; ++j) {
        valid = (j == i || !visited1[j] || !visited2[j]);
      }


      if (valid) {
        s2[s2l] = i;
        ++s2l;
      }
    }
  }


  fout << s1l;
  for (int i = 0; i < s1l; ++i) {
    fout << ' ' << s1[i];
  }
  fout << std::endl << s2l;
  for (int i = 0; i < s2l; ++i) {
    fout << ' ' << s2[i];
  }
  fout << std::endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: lgame
*/
#include <fstream>
#include <vector>
#include <utility>
#include <set>
#include <algorithm>
using namespace std;


const int MAXN = 40040;
const int MAXL = 26;
const int enc_table[MAXL] = { 2, //a
                              5, //b
                              4, //c
                              4, //d
                              1, //e
                              6, //f
                              5, //g
                              5, //h
                              1, //i
                              7, //j
                              6, //k
                              3, //l
                              5, //m
                              2, //n
                              3, //o
                              5, //p
                              7, //q
                              2, //r
                              1, //s
                              2, //t
                              4, //u
                              6, //v
                              6, //w
                              7, //x
                              5, //y
                              7, //z
};


struct Validator {
  int cnt[MAXL];
  int sz;


  Validator(const string &s): cnt{0} {
    sz = int(s.size());
    for (int i = 0; i < sz; ++i) {
      ++cnt[s[i] - 'a'];
    }
  }
  bool is_valid(const string &s) {
    int tmp[MAXL] = {0};
    int sl = int(s.size());
    for (int i = 0; i < sl; ++i) {
      if (!isspace(s[i])) {
        int ind = s[i] - 'a';
        ++tmp[ind];
        if (tmp[ind] > cnt[ind]) {
          return false;
        }
      }
    }
    return true;
  }
};


inline int enc(const string &s) {
  int sl = int(s.size());
  int ret = 0;
  for (int i = 0; i < sl; ++i) {
    if (!isspace(s[i])) {
      ret += enc_table[s[i]-'a'];
    }
  }
  return ret;
}


bool cmp(const pair<int,string> &left, const pair<int,string> &right) {
  return left.first > right.first;
}


int main() {
  ifstream fin("lgame.in");
  ifstream fdict("lgame.dict");
  ofstream fout("lgame.out");


  string tmp;
  fin >> tmp;
  Validator vdtr(tmp);


  vector<pair<int,string>> wrds;
  wrds.reserve(MAXN);
  while (true) {
    fdict >> tmp;
    if (tmp == ".") {
      break;
    }
    if (vdtr.is_valid(tmp)) {
      wrds.push_back(pair<int,string>(int(tmp.size()), tmp));
    }
  }


  sort(wrds.begin(), wrds.end(), greater<pair<int,string>>());
  set<string> sol_str;
  int sol_val = 0;


  auto lower_1 = lower_bound(wrds.begin(), wrds.end(), pair<int,string>(vdtr.sz, ""), cmp);
  for (auto it1 = lower_1; it1 != wrds.end(); ++it1) {
    int cenc1 = enc(it1->second);
    if (sol_val < cenc1) {
      sol_str.clear();
      sol_val = cenc1;
    }
    if (sol_val == cenc1) {
      sol_str.insert(it1->second);
    }
    auto lower_2 = lower_bound(wrds.begin(), wrds.end(), pair<int,string>(vdtr.sz - it1->first,
""), cmp);
    for (auto it2 = lower_2; it2 != wrds.end(); ++it2) {
      string cstr2;
      if (it1->second < it2->second) {
        cstr2 = it1->second + ' ' + it2->second;
      }
      else {
        cstr2 = it2->second + ' ' + it1->second;
      }
      if (vdtr.is_valid(cstr2)) {
        int cenc2 = enc(cstr2);
        if (sol_val < cenc2) {
          sol_str.clear();
          sol_val = cenc2;
        }
        if (sol_val == cenc2) {
          sol_str.insert(cstr2);
        }
      }
    }
  }


  fout << sol_val << endl;
  for (auto it = sol_str.begin(); it != sol_str.end(); ++it) {
    fout << *it << endl;
  }
}




/*****************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: shuttle
*/
#include <fstream>
#include <queue>
#include <functional>
using namespace std;


string BEGIN;
string END;


bool check(const string &s, int h, int c) {
  int sl = int(s.size());
  if (c == 0) {
    int i = h + 2;
    return (i < sl) && (s[h+1] == 'w') && (s[i] == 'b');
  }
  else if (c == 1) {
    int i = h - 2;
    return (i >= 0) && (s[h-1] == 'b') && (s[i] == 'w');
  }
  else if (c == 2) {
    int i = h - 1;
    return (i >= 0) && (s[i] == 'w');
  }
  else {
    //c == 3
    int i = h + 1;
    return (i < sl) && (s[i] == 'b');
  }
}


void change(string &s, int &h, int c) {
  int i = int(s.size());
  if (c == 0) {
    i = h + 2;
  }
  else if (c == 1) {
    i = h - 2;
  }
  else if (c == 2) {
    i = h - 1;
  }
  else {
    i = h + 1;
  }


  swap(s[h], s[i]);
  h = i;
}


int main() {
  ifstream fin("shuttle.in");
  ofstream fout("shuttle.out");


  int N;
  fin >> N;
  for (int i = 0; i < N; ++i) {
    BEGIN += 'w';
    END += 'b';
  }
  BEGIN += ' ';
  END += ' ';
  for (int i = 0; i < N; ++i) {
    BEGIN += 'b';
    END += 'w';
  }


  string state = BEGIN;
  int hole = N;
  int io = 0;
  bool is_2nd = true;
  while (true) {
    priority_queue<int, vector<int>, greater<int>> pq;
    for (int c = 0; c < 4; ++c) {
      if (check(state, hole, c)) {
        pq.push(c);
      }
    }


    if (int(pq.size()) >= 2 && pq.top() == 2) {
      if (is_2nd) {
        change(state, hole, 2);
        is_2nd = false;
      }
      else {
        change(state, hole, 3);
        is_2nd = true;
      }
    }
    else {
      change(state, hole, pq.top());
    }
    fout << (io == 0 ? "" : " ") << hole+1;
    ++io;
    if (io == 20) {
      fout << endl;
      io = 0;
    }


    if (hole == N && state == END) {
      if (io != 0) {
        fout << endl;
      }
      break;
    }
  }
}




/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: milk6
*/
#include <fstream>
#include <vector>
#include <map>
#include <algorithm>
#include <queue>
#include <utility>
#include <climits>
using namespace std;


const int MAXN = 35;
const int MAXM = 1010;


struct Edge {
  Edge(): src(0), dst(0), cos(0) {}
  Edge(int s, int d, int c): src(s), dst(d), cos(c) {}
  int src, dst, cos;
};


Edge ed[MAXM];
int amo[MAXN][MAXN];
int amc[MAXN][MAXN];
map<vector<int>, int> cache;
int N, M;


bool cmp(const vector<int> &left, const vector<int> &right) {
  if (left.size() < right.size()) {
    return true;
  }
  else if (left.size() > right.size()) {
    return false;
  }
  int sz = int(left.size());
  for (int i = 0; i < sz; ++i) {
    if (left[i] != right[i]) {
      return left[i] < right[i];
    }
  }
  return false;
}


int max_flow(const vector<int> &c) {
  if (cache.count(c)) {
    return cache[c];
  }


  copy(&amo[0][0], &amo[0][0] + MAXN*MAXN, &amc[0][0]);
  for (const auto &i : c) {
    amc[ ed[i].src ][ ed[i].dst ] -= ed[i].cos;
  }


  int total_flow = 0;
  const int S = 1;
  const int D = N;


  while (true) {
    bool visited[MAXN] = {false};
    int flow[MAXN]   = {0};
    int prev[MAXN] = {0};
    priority_queue<pair<int,int>> next;


    flow[S] = INT_MAX;
    prev[S] = 0;
    next.push(make_pair(INT_MAX, S));


    int path_flow = 0;
    while (true) {
      if (next.empty()) {
        break;
      }


      int cn = next.top().second;
      next.pop();


      if (cn == D) {
        path_flow = flow[D];
        break;
      }


      if (visited[cn]) {
        continue;
      }
      visited[cn] = true;
      for (int i = 1; i <= D; ++i) {
        int nf = min(flow[cn], amc[cn][i]);
        if (!visited[i] && nf > flow[i]) {
          flow[i] = nf;
          prev[i] = cn;
          next.push(make_pair(nf, i));
        }
      }
    }


    if (path_flow == 0) {
      break;
    }


    total_flow += path_flow;
    int nn = D;
    while (prev[nn] != 0) {
      amc[ prev[nn] ][ nn ] -= path_flow;
      amc[ nn ][ prev[nn] ] += path_flow;
      nn = prev[nn];
    }
  }
  return cache[c] = total_flow;
}


int main() {
  ifstream fin("milk6.in");
  ofstream fout("milk6.out");


  fin >> N >> M;
  for (int i = 1; i <= M; ++i) {
    fin >> ed[i].src >> ed[i].dst >> ed[i].cos;
    amo[ ed[i].src ][ ed[i].dst ] += ed[i].cos;
  }


  vector<int> pos;
  pos.reserve(MAXM);
  vector<bool> pos_used;
  pos_used.reserve(MAXM);


  int max_flow_default = max_flow(vector<int>(0));
  for (int i = 1; i <= M; ++i) {
    int nf = max_flow(vector<int>{i});
    if (ed[i].cos == (max_flow_default - nf)) {
      pos.push_back(i);
      pos_used.push_back(false);
    }
  }


  vector<vector<int>> sol;
  int np = 0;
  int psz = int(pos.size());
  while (np < psz) {
    if (!pos_used[np]) {
      vector<int> csol { pos[np] };


      for (int i = np+1; i < psz; ++i) {
        if (!pos_used[i]) {
          int eval = ed[pos[i]].cos;
          int cmf = max_flow( csol );
          csol.push_back(pos[i]);
          int nmf = max_flow( csol );
          if (eval > cmf - nmf) {
            csol.pop_back();
          }
        }
      }


      if (max_flow(csol) == 0) {
        for (int i = 0; i < int(csol.size()); ++i) {
          pos_used[i] = true;
        }
        sort(csol.begin(), csol.end());
        sol.push_back(csol);
      }
    }
    ++np;
  }


  sort(sol.begin(), sol.end(), cmp);
  fout << max_flow_default;
  if (!sol.empty()) {
    fout << ' ' << sol[0].size() << endl;
    for (const auto &e : sol[0]) {
      fout << e << endl;
    }
  }
  else {
    fout << ' ' << 0 << endl;
  }
}






/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: frameup
*/
#include <fstream>
#include <vector>
#include <algorithm>
using namespace std;


const int MAXHW = 33;
const int MAXL=26;
```

```
vector<string> fr(MAXHW, string(MAXHW, ' '));
int ub[MAXL], rb[MAXL], db[MAXL], lb[MAXL];


vector<string> solve(const vector<string> &frm, const vector<bool> &used) {
  // find all possible
  string let = "";
  for (int i = 0; i < MAXL; ++i) {
    if (!used[i] && lb[i] != -1) {
      char cc = char('A' + i);
      bool pos = true;
      for (int y = ub[i]; y <= db[i] && pos; ++y) {
        pos = pos && (frm[y][ lb[i] ] == cc || frm[y][ lb[i] ] == '.');
        pos = pos && (frm[y][ rb[i] ] == cc || frm[y][ rb[i] ] == '.');
      }
      for (int x = lb[i]+1; x < rb[i] && pos; ++x) {
        pos = pos && (frm[ ub[i] ][x] == cc || frm[ ub[i] ][x] == '.');
        pos = pos && (frm[ db[i] ][x] == cc || frm[ db[i] ][x] == '.');
      }
      if (pos) {
        let += cc;
      }
    }
  }


  // return if no letters
  if (let.empty()) {
    return vector<string>(1, "");
  }


  // solve recursively
  vector<string> sol;
  int letsz = int(let.size());
  for (int i = 0; i < letsz; ++i) {
    // modify image
    vector<string> nfrm = frm;
    char cc = let[i];
    int ci = cc - 'A';
    for (int y = ub[ci]; y <= db[ci]; ++y) {
      nfrm[y][ lb[ci] ] = '.';
      nfrm[y][ rb[ci] ] = '.';
    }
    for (int x = lb[ci]+1; x < rb[ci]; ++x) {
      nfrm[ ub[ci] ][x] = '.';
      nfrm[ db[ci] ][x] = '.';
    }
    // modify used
    vector<bool> nused = used;
    nused[ci] = true;


    vector<string> tmp = solve(nfrm, nused);
    for (const auto &s : tmp) {
      sol.push_back(s + cc);
    }
  }


  return sol;
}
```

```
int main() {
  ifstream fin("frameup.in");
  ofstream fout("frameup.out");


  int H, W;
  fin >> H >> W;
  for (int y = 0; y < H; ++y) {
    for (int x = 0; x < W; ++x) {
      fin >> fr[y][x];
    }
  }


  for (int i = 0; i < MAXL; ++i) {
    char cc = char('A' + i);
    ub[i] = -1;
    for (int y = 0; y < H && ub[i] == -1; ++y) {
      for (int x = 0; x < W && ub[i] == -1; ++x) {
        if (fr[y][x] == cc) {
          ub[i] = y;
        }
      }
    }
    rb[i] = -1;
    for (int x = W-1; x >= 0 && rb[i] == -1; --x) {
      for (int y = 0; y < H && rb[i] == -1; ++y) {
        if (fr[y][x] == cc) {
          rb[i] = x;
        }
      }
    }
    db[i] = -1;
    for (int y = H-1; y >= 0 && db[i] == -1; --y) {
      for (int x = 0; x < W && db[i] == -1; ++x) {
        if (fr[y][x] == cc) {
          db[i] = y;
        }
      }
    }
    lb[i] = -1;
    for (int x = 0; x < W && lb[i] == -1; ++x) {
      for (int y = 0; y < H && lb[i] == -1; ++y) {
        if (fr[y][x] == cc) {
          lb[i] = x;
        }
      }
    }
  }


  vector<string> sol = solve(fr, vector<bool>(MAXL, false));
  sort(sol.begin(), sol.end());
  for (const auto &s : sol) {
    fout << s << endl;
  }
}
```

```
/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: fc
*/
#include <fstream>
#include <algorithm>
#include <iomanip>
#include <cmath>
using namespace std;


const int MAXN = 10100;


struct Point {
  double x, y;
  Point operator-(const Point &ot) const {
    Point res = *this;
    res.x -= ot.x;
    res.y -= ot.y;
    return res;
  }
};


double inner_prod(const Point &s, const Point &p1, const Point &p2) {
  return (p1.x - s.x) * (p2.x - s.x) + (p1.y - s.y) * (p2.y - s.y);
}


double cross_prod_z(const Point &p1, const Point &p2) {
  return p1.x * p2.y - p2.x * p1.y;
}


double distance(const Point &p1, const Point &p2) {
  double dx = p1.x - p2.x;
  double dy = p1.y - p2.y;
  return sqrt(dx*dx + dy*dy);
}


double angle(const Point &s, const Point &p1, const Point &p2) {
  return acos( inner_prod(s, p1, p2) / (distance(s, p1) * distance(s, p2)) );
}


struct Comparator {
  Point ref;
  Point ref2;
  bool operator() (const Point &p1, const Point &p2) {
    double ang1 = angle(ref, ref2, p1);
    double ang2 = angle(ref, ref2, p2);
    return (ang1 < ang2);
  }
};
```

```cpp
Point pi[MAXN];
Point ph[MAXN];


int main() {
  std::ifstream fin("fc.in");
  std::ofstream fout("fc.out");


  int N;
  fin >> N;
  for (int i = 0; i < N; ++i) {
    fin >> pi[i].x >> pi[i].y;
  }


  // lowest point (most to the left)
  for (int i = 1; i < N; ++i) {
    if (pi[i].y < pi[0].y || (pi[i].y == pi[0].y && pi[i].x < pi[0].x)) {
      swap(pi[i], pi[0]);
    }
  }


  Comparator comp;
  comp.ref = pi[0];
  comp.ref2 = pi[0];
  ++comp.ref2.x;
  sort(&pi[0]+1, &pi[0]+N, comp);


  ph[0] = pi[0];
  ph[1] = pi[1];
  int hsz = 2;
  for (int i = 2; i < N; ++i) {
    while (hsz >= 2 && cross_prod_z(ph[hsz-1] - ph[hsz-2],pi[i] - ph[hsz-1]) < 0){
      --hsz;
    }
    ph[hsz] = pi[i];
    ++hsz;
  }


  double circ = distance(ph[hsz-1], ph[0]);
  for (int i = 0; i < hsz-1; ++i) {
    circ += distance(ph[i], ph[i+1]);
  }


  fout << fixed << setprecision(2) << circ << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: starry
*/
#include <fstream>
```

```cpp
#include <vector>
#include <map>
#include <utility>
using namespace std;


const int MAXWH = 101;


struct Cluster {
  int wi;
  int he;
  vector<string> st;
  Cluster(const vector<string> &v): wi(!v.empty() ? int(v[0].size()) : 0), he(int(v.size())),
st(v) {}


  Cluster rotate_cw() const;
  Cluster mirror_h() const;
  Cluster normalize() const;
  bool operator<(const Cluster &) const;
};


Cluster Cluster::rotate_cw() const {
  vector<string> rs (this->wi, string(this->he, '0'));
  for (int i = 0; i < this->he; ++i) {
    for (int j = 0; j < this->wi; ++j) {
      rs[j][this->he-i-1] = this->st[i][j];
    }
  }
  return Cluster(rs);
}


Cluster Cluster::mirror_h() const {
  vector<string> rs(this->he, string(this->wi, '0'));
  for (int i = 0; i < this->he; ++i) {
    for (int j = 0; j < this->wi; ++j) {
      rs[i][this->wi-j-1] = this->st[i][j];
    }
  }
  return Cluster(rs);
}


Cluster Cluster::normalize() const {
  Cluster minc = *this;
  Cluster curc = *this;
  for (int i = 1; i < 8; ++i) {
    curc = curc.rotate_cw();
    if (i == 4) {
      curc = curc.mirror_h();
    }
    if (curc < minc) {
      minc = curc;
    }
  }
  return minc;
}
```

```
bool Cluster::operator<(const Cluster &ot) const {
  int th = int(this->st.size());
  int oh = int(ot.st.size());
  int tw = !this->st.empty() ? int(this->st[0].size()) : 0;
  int ow = !ot.st.empty() ? int(ot.st[0].size()) : 0;
  if (th != oh) {
    return th < oh;
  }
  if (tw != ow) {
    return tw < ow;
  }
  for (int i = 0; i < th; ++i) {
    for (int j = 0; j < tw; ++j) {
      if (this->st[i][j] != ot.st[i][j]) {
        return this->st[i][j] < ot.st[i][j];
      }
    }
  }
  return false;
}


vector<string> sky;
map<Cluster, char> clusters;


int main() {
  std::ifstream fin("starry.in");
  std::ofstream fout("starry.out");


  int W, H;
  fin >> W >> H;
  sky.reserve(H);


  for (int i = 0; i < H; ++i) {
    string tmp;
    fin >> tmp;
    sky.push_back(tmp);
  }


  for (int i = 0; i < H; ++i) {
    for (int j = 0; j < W; ++j) {
      if (sky[i][j] == '1') {
        int mini = i, maxi = i;
        int minj = j, maxj = j;
        vector<pair<int,int>> scoo(1, make_pair(i, j));


        for(int ind = 0; ind < int(scoo.size()); ++ind) {
          int ci = scoo[ind].first;
          int cj = scoo[ind].second;
          mini = min(mini, ci);
          maxi = max(maxi, ci);
          minj = min(minj, cj);
          maxj = max(maxj, cj);
          for (int di = -1; di <= 1; ++di) {
            for (int dj = -1; dj <= 1; ++dj) {
              if ((di != 0 || dj != 0) && ci+di >= 0 && ci+di < H && cj+dj >= 0 && cj+dj < W &&
sky[ci+di][cj+dj] == '1') {
```

```
                sky[ci+di][cj+dj] = '0';
                scoo.push_back(make_pair(ci+di, cj+dj));
              }
            }
          }
        }
        vector<string> st(maxi-mini+1, string(maxj-minj+1, '0'));
        for(int ind = 0; ind < int(scoo.size()); ++ind) {
          int ci = scoo[ind].first;
          int cj = scoo[ind].second;
          st[ci-mini][cj-minj] = '1';
        }
        Cluster cnormal = Cluster(st).normalize();
        char fillc = char('a' + int(clusters.size()));
        if (clusters.count(cnormal)) {
          fillc = clusters[cnormal];
        }
        else {
          clusters.insert(make_pair(cnormal, fillc));
        }
        for(int ind = 0; ind < int(scoo.size()); ++ind) {
          int ci = scoo[ind].first;
          int cj = scoo[ind].second;
          sky[ci][cj] = fillc;
        }
      }
    }
  }


  for (int i = 0; i < H; ++i) {
    fout << sky[i] << endl;
  }
}




/*******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: theme
*/
#include <fstream>
using namespace std;


const int MAXN = 5050;
int num[MAXN];


int main() {
  std::ifstream fin("theme.in");
  std::ofstream fout("theme.out");


  int N;
  fin >> N;
  for (int i = 0; i < N; ++i) {
    fin >> num[i];
  }
  int sol = 0;
  for (int di = 5; di < N; ++di) {
    int st = di;
    for (int j = di+1; j < N; ++j) {
      if (num[j] - num[j-di] == num[j-1] - num[j-di-1] && j - di < st) {
        sol = max(sol, j - st + 1);
      }
      else {
        st = j;
      }
    }
  }


  fout << (sol < 5 ? 0 : sol) << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: snail
*/
#include <fstream>
using namespace std;


// d: 0-up, 1-right, 2-down, 3-left
struct State {
  int y, x, d, s;
};


const int MAXN = 130;
const int dy[4] = {-1, 0, 1, 0};
const int dx[4] = {0, 1, 0, -1};


int wall[MAXN][MAXN] = {0};
int N;


int dfs(int y, int x) {
  int di = 0;
  int re = 0;
  for (int i = 0; i < 4; ++i) {
    int ny = y + dy[i];
    int nx = x + dx[i];
    if (0 <= ny && ny < N && 0 <= nx && nx < N && !wall[ny][nx]) {
      int ly = ny;
      int lx = nx;
      while (0 <= ly && ly < N && 0 <= lx && lx < N && !wall[ly][lx]) {
        ++di;
        wall[ly][lx] = 2;
        ly += dy[i];
        lx += dx[i];
      }
      int rec = 0;
      if (ly < 0 || N <= ly || lx < 0 || N <= lx || wall[ly][lx] == 1) {
        rec = dfs(ly-dy[i], lx-dx[i]);
      }
      re = max(re, di + rec);
      while (ly != ny || lx != nx) {
        wall[ly-dy[i]][lx-dx[i]] = 0;
        lx -= dx[i];
        ly -= dy[i];
        --di;
      }
    }
  }
  return re;
}


int main() {
  std::ifstream fin("snail.in");
  std::ofstream fout("snail.out");


  int B;
  fin >> N >> B;
  for (int i = 0; i < B; ++i) {
    char c; int n;
    fin >> c >> n;
    wall[n-1][c-'A'] = 1;
  }
  wall[0][0] = 2;


  fout << 1 + dfs(0, 0) << endl;
}




/******************************************************************************/
/*
ID: dnkihot1
LANG: C++11
TASK: milk4
*/
#include <fstream>
#include <algorithm>
using namespace std;


const int MAXQ = 20020;
const int MAXP = 101;


int hn[MAXP];
int hs[MAXP];
bool qq[MAXQ];


int main() {
  std::ifstream fin("milk4.in");
  std::ofstream fout("milk4.out");
  int Q, P;
  fin >> Q >> P;
  for (int i = 0; i < P; ++i) {
    fin >> hn[i];
  }
  sort(&hn[0], &hn[0] + P);


  hs[0] = 0;
  int hsz = 1;
  int L = 1;


  while (true) {
    //check solution
    fill(&qq[0], &qq[0] + Q, false);
    for (int i = 0; i <= Q; i += hn[hs[0]]) {
      qq[i] = true;
    }
    for (int i = 1; i < hsz; ++i) {
      for (int j = hn[hs[i]]; j <= Q; ++j) {
        qq[j] |= qq[j - hn[hs[i]]];
      }
    }
    if (qq[Q]) {
      fout << hsz;
      for (int j = 0; j < hsz; ++j) {
        fout << ' ' << hn[hs[j]];
      }
      fout << endl;
      break;
    }


    int next;
    do {
      next = hs[hsz-1] + 1;
      --hsz;
    } while (hsz && P - next < L - hsz);


    if (!hsz && next == P - L + 1) {
      ++L;
      next = 0;
    }


    if (L > P) {
      break;
    }


    while (hsz < L) {
      hs[hsz] = next;
      ++hsz;
      ++next;
    }
  }
}
```