单纯形

```
#include <bits/stdc++.h>
using namespace std;
typedef long double LD;
const int N=200;
int n,m,type,q[N],id[N*2];
double a[N][N],ans[N];

namespace LP {
    const double eps=1e-8, INF=1e15;
    inline void pivot(int l,int e) {
        swap(id[l+n],id[e]);
        double t=a[l][e]; a[l][e]=1;
        for(int i=0;i<=n;i++) a[l][i]/=t;
        int p=0;
        for(int i=0;i<=n;i++) if(fabs(a[l][i])>eps) q[++p]=i;
        for(int i=0;i<=m;i++) if(i!=l && fabs(a[i][e])>eps) {
            double t=a[i][e]; a[i][e]=0;
            for(int j=1;j<=p;j++) a[i][q[j]]-=t*a[l][q[j]];
        }
    }
    inline bool init() {
        while(true) {
            int l=0,e=0;
            for(int i=1;i<=m;i++)
                if(a[i][0]<-eps && (!l || (rand()%2))) l=i;
            if(!l) break;
            for(int j=1;j<=n;j++)
                if(a[l][j]<-eps && (!e || (rand()%2))) e=j;
            if(!e) return puts("Infeasible"), false;
            pivot(l,e);
        } return true;
    }
    inline bool simplex() {
        while(true) {
            int l=0,e=0; double mn=INF;
            for(int j=1;j<=n;j++)
                if(a[0][j]>eps) {e=j; break;}
            if(!e) break;
            for(int i=1;i<=m;i++)
                if(a[i][e]>eps && mn>a[i][0]/a[i][e])
                    mn=a[i][0]/a[i][e], l=i;
```

```
            if(!l) return puts("Unbounded"), false;
            pivot(l,e);
        } return true;
    }
    inline void solve() {
        for(int i=1;i<=n;i++) id[i]=i;
        if(init() && simplex()) {
            printf("%.8lf\n",-a[0][0]);
            for(int i=1;i<=m;i++) ans[id[n+i]]=a[i][0];
            if(type) for(int i=1;i<=n;i++) printf("%.8lf ",ans[i]);
        }
    }
}

int main() {
    freopen("2.in","r",stdin);
    freopen("2.out","w",stdout);
    cin>>n>>m>>type;
    for(int i=1;i<=n;i++) cin>>a[0][i];
    for(int i=1;i<=m;i++) {
        for(int j=1;j<=n;j++) cin>>a[i][j];
        cin>>a[i][0];
    }
    LP::solve();
}
```