**//Gaussian Elimination. (UVa-10109)**

/\*

Eqn=number of equation, Ukn=Unknown variables.

mat[][] is the given matrix of co-efficient, res[] is the value of unknown variables.

If res[] does less than MOD, then infinite solution turns into MOD^(Ukn-rankAC) solutions.

\*/

int rankA, rankAC, Ukn, Eqn;

double mat[110][110], res[110];

void swapRows(int r1, int r2)

{

for(int i=0;i<=Ukn;i++) swap(mat[r1][i],mat[r2][i]);

return;

}

void Forward\_Elimination()

{

for(int p=0;p<Ukn;p++)

{

int maxR=p;

for(int i=p+1;i<Eqn;i++) if(mat[i][p]!=0.0) maxR=i;

swapRows(p,maxR);

if(mat[p][p]!=0.0)

for(int i=p+1;i<Eqn;i++)

{

double tmp=mat[i][p]/mat[p][p];

for(int j=0;j<=Ukn;j++)

{

mat[i][j]-=(tmp\*mat[p][j]);

if(fabs(mat[i][j])<0.00000001) mat[i][j]=0.0;

}

}

}

return;

}

void rank()

{

rankA=rankAC=Eqn;

bool allZeroes;

for(int i=0;i<Eqn;i++)

{

allZeroes=true;

for(int j=0;j<Ukn;++j)

{

if(mat[i][j]!=0)

{

allZeroes=false;

break;

}

}

if(allZeroes)

{

--rankA;

if(mat[i][Ukn]==0) --rankAC;

}

}

return;

}

void Back\_Substitution()

{

for(int p=Ukn-1;p>=0;p--)

{

double s=0.0;

for(int k=p+1;k<Ukn;k++) s+=(res[k]\*mat[p][k]);

res[p]=(mat[p][Ukn]-s)/mat[p][p];

}

return;

}

int main()

{

scanf("%d%d", &Ukn, &Eqn);

for(int i=0;i<Eqn;i++)

for(int j=0;j<=Ukn;j++) scanf("%lf", &mat[i][j]);

Forward\_Elimination();

rank();

if(rankA==rankAC && rankAC==Ukn) Back\_Substitution();

if(rankA!=rankAC) printf("No Solution.\n");

else if(rankAC<Ukn)

printf("Infinitely many solutions containing %d arbitrary constants.\n", Ukn-rankAC);

else

{

for(int i=0;i<Ukn;i++)

printf("%lf\n",res[i]);

}

return 0;

}