Operation

Research

Laboratory

(Professor M.P. Biswal

&

Professor Geetanjali Panda)

**Keshav Agarwal**

**12MA20020**

**Code to solve a Linear Programming Problem**

/\*

Keshav Agarwal

12MA20020

\*/

#include<stdio.h>

#include<math.h>

int dif,m,n,q;

double a[100][100],b[100][100];

int c[100];

double fn[100];

double val[100];

int bfs;

double max;

//function to calculate the function value

double cal(double g[])

{

int i;

double total=fn[n];

for(i=0;i<n;i++)

{

total+=(g[i]\*fn[i]);

}

return total;

}

//function to exchange the required rows

void exc(int q,int w)

{

int i,t;

for(i=0;i<=m;i++)

{

t=b[q][i];

b[q][i]=b[w][i];

b[w][i]=t;

}

}

//function to convert into echolon form and solve the matrix

void find()

{

c[dif]=n;

int i,t=0,j,k;

int p[100];

for(k=0;k<m;k++)

{

int l=0;t=0;

for(i=0;i<=dif;i++)

{

for(j=t;j<c[i];j++)

{

b[k][l++]=a[k][j];

if(k==0)

p[l-1]=j;

}

t=c[i]+1;

}

b[k][m]=a[k][n];

}

for(i=1;i<m;i++)

{

for(j=0;j<i;j++)

{

double f=0.0;

if(b[i][j]==0)

continue;

if(b[j][j]==0)

exc(j,i);

f=b[i][j]/b[j][j];

for(k=j;k<=m;k++)

{

b[i][k]-=(f\*b[j][k]);

}

}

}

double g[100];

for(i=0;i<dif;i++)

{

g[c[i]]=0;

}

g[p[m-1]]=b[m-1][m]/b[m-1][m-1];

for(i=m-2;i>=0;i--)

{

double r=0.0;

for(j=i+1;j<m;j++)

{

r=r+b[i][j]\*g[p[j]];

}

r=b[i][m]-r;

g[p[i]]=r/b[i][i];

}

int ng=0,nb=0;

for(i=0;i<n;i++)

{

if(g[i]<0||g[i]!=g[i]||isinf(g[i]))

ng=1;

if(g[i]!=g[i]||isinf(g[i]))

nb=1;

}

printf("\n");

if(nb==0)

{

q++;

printf("\n");

for(i=0;i<n;i++)

{

printf("x%d= %lf ",(i+1),g[i]);

}

}

if(ng==0)

{bfs++;

printf("\nThis a BFS \n");

double y=cal(g);

if(y>max)

{

for(i=0;i<n;i++)

val[i]=g[i];

max=y;

}

}

}

//function to generate the permutations of required 0's

void recur(int pos,int var)

{

if(pos==dif)

{

find();

return;

}

int i;

for(i=var+1;i<=n-(dif-pos);i++)

{

c[pos]=i;

recur(pos+1,i);

}

}

//main function of the program

int main()

{

printf("Enter no. of equations and no. of variables respectively \n");

scanf("%d %d",&m,&n);

// m is the number of equations

// n is the number of variables

//dif is the no. of variable which must be set to 0

dif=n-m;

//i,j are the loop variables

int i,j,k,l;

printf("Enter the equation coefficients \n");

for(i=0;i<m;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j]);

}

bfs=0;q=0;

max=fn[n];

for(i=0;i<=m;i++)

{

c[0]=i;

recur(1,i);

}

printf("\nTotal Basic Solutions=%d\n",q);

printf("Total BFS = %d \n",bfs);

return 0;

}

**OUTPUT**

Case 1:

Enter no. of equations and no. of variables respectively

3 5

Enter the equation coefficients

x1: 1

x2: 1

x3: 1

x4: 0

x5: 0

constant: 10

x1: 1

x2: 2

x3: 0

x4: 1

x5: 0

constant: 11

x1: 1

x2: 4

x3: 0

x4: 0

x5: 1

constant: 16

x1= 0.000000 x2= 0.000000 x3= 10.000000 x4= 11.000000 x5= 16.000000

This a BFS

x1= 0.000000 x2= 10.000000 x3= 0.000000 x4= -9.000000 x5= -24.000000

x1= 0.000000 x2= 5.500000 x3= 4.500000 x4= 0.000000 x5= -6.000000

x1= 0.000000 x2= 4.000000 x3= 6.000000 x4= 3.000000 x5= 0.000000

This a BFS

x1= 10.000000 x2= 0.000000 x3= 0.000000 x4= 1.000000 x5= 6.000000

This a BFS

x1= 11.000000 x2= 0.000000 x3= -1.000000 x4= 0.000000 x5= 5.000000

x1= 16.000000 x2= 0.000000 x3= -6.000000 x4= -5.000000 x5= 0.000000

x1= 9.000000 x2= 1.000000 x3= 0.000000 x4= 0.000000 x5= 3.000000

This a BFS

x1= 8.000000 x2= 2.000000 x3= 0.000000 x4= -1.000000 x5= 0.000000

x1= 6.000000 x2= 2.500000 x3= 1.500000 x4= 0.000000 x5= 0.000000

This a BFS

Total Basic Solutions=10

Total BFS = 5

Case 2:

Enter no. of equations and no. of variables respectively

3 7

Enter the equation coefficients

x1: 1

x2: 1

x3: 1

x4: 0

x5: 1

x6: 0

x7: 0

constant: 10

x1: 1

x2: 0

x3: 1

x4: 1

x5: 0

x6: 1

x7: 0

constant: 12

x1: 1

x2: 1

x3: 0

x4: 1

x5: 0

x6: 0

x7: 1

constant: 15

x1= 0.000000 x2= 0.000000 x3= 0.000000 x4= 0.000000 x5= 10.000000 x6= 12.000000

x7= 15.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 0.000000 x4= 12.000000 x5= 10.000000 x6= 0.000000

x7= 3.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 0.000000 x4= 15.000000 x5= 10.000000 x6= -3.000000

x7= 0.000000

x1= 0.000000 x2= 0.000000 x3= 10.000000 x4= 0.000000 x5= 0.000000 x6= 2.000000 x

7= 15.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 12.000000 x4= 0.000000 x5= -2.000000 x6= 0.000000

x7= 15.000000

x1= 0.000000 x2= 0.000000 x3= 10.000000 x4= 2.000000 x5= 0.000000 x6= 0.000000 x

7= 13.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 10.000000 x4= 15.000000 x5= 0.000000 x6= -13.00000

0 x7= 0.000000

x1= 0.000000 x2= 0.000000 x3= -3.000000 x4= 15.000000 x5= 13.000000 x6= 0.000000

x7= 0.000000

x1= 0.000000 x2= 10.000000 x3= 0.000000 x4= 0.000000 x5= 0.000000 x6= 12.000000

x7= 5.000000

This a BFS

x1= 0.000000 x2= 15.000000 x3= 0.000000 x4= 0.000000 x5= -5.000000 x6= 12.000000

x7= 0.000000

x1= 0.000000 x2= 10.000000 x3= 0.000000 x4= 12.000000 x5= 0.000000 x6= 0.000000

x7= -7.000000

x1= 0.000000 x2= 10.000000 x3= 0.000000 x4= 5.000000 x5= 0.000000 x6= 7.000000 x

7= 0.000000

This a BFS

x1= 0.000000 x2= 3.000000 x3= 0.000000 x4= 12.000000 x5= 7.000000 x6= 0.000000 x

7= 0.000000

This a BFS

x1= 0.000000 x2= -2.000000 x3= 12.000000 x4= 0.000000 x5= 0.000000 x6= 0.000000

x7= 17.000000

x1= 0.000000 x2= 15.000000 x3= -5.000000 x4= 0.000000 x5= 0.000000 x6= 17.000000

x7= 0.000000

x1= 0.000000 x2= 15.000000 x3= 12.000000 x4= 0.000000 x5= -17.000000 x6= 0.00000

0 x7= 0.000000

x1= 0.000000 x2= 6.500000 x3= 3.500000 x4= 8.500000 x5= 0.000000 x6= 0.000000 x7

= 0.000000

This a BFS

x1= 10.000000 x2= 0.000000 x3= 0.000000 x4= 0.000000 x5= 0.000000 x6= 2.000000 x

7= 5.000000

This a BFS

x1= 12.000000 x2= 0.000000 x3= 0.000000 x4= 0.000000 x5= -2.000000 x6= 0.000000

x7= 3.000000

x1= 15.000000 x2= 0.000000 x3= 0.000000 x4= 0.000000 x5= -5.000000 x6= -3.000000

x7= 0.000000

x1= 10.000000 x2= 0.000000 x3= 0.000000 x4= 2.000000 x5= 0.000000 x6= 0.000000 x

7= 3.000000

This a BFS

x1= 10.000000 x2= 0.000000 x3= 0.000000 x4= 5.000000 x5= 0.000000 x6= -3.000000

x7= 0.000000

x1= 15.000000 x2= 0.000000 x3= -5.000000 x4= 0.000000 x5= 0.000000 x6= 2.000000

x7= 0.000000

x1= 15.000000 x2= 0.000000 x3= -3.000000 x4= 0.000000 x5= -2.000000 x6= 0.000000

x7= 0.000000

x1= 13.000000 x2= 0.000000 x3= -3.000000 x4= 2.000000 x5= 0.000000 x6= 0.000000

x7= 0.000000

x1= 12.000000 x2= -2.000000 x3= 0.000000 x4= 0.000000 x5= 0.000000 x6= 0.000000

x7= 5.000000

x1= 12.000000 x2= 3.000000 x3= 0.000000 x4= 0.000000 x5= -5.000000 x6= 0.000000

x7= 0.000000

x1= 7.000000 x2= 3.000000 x3= 0.000000 x4= 5.000000 x5= 0.000000 x6= 0.000000 x7

= 0.000000

This a BFS

x1= 17.000000 x2= -2.000000 x3= -5.000000 x4= 0.000000 x5= 0.000000 x6= 0.000000

x7= 0.000000

Total Basic Solutions=29

Total BFS = 11

Case 3:

Enter no. of equations and no. of variables respectively

3 5

Enter the equation coefficients

x1: 1

x2: 2

x3: 1

x4: 2

x5: 1

constant: 22

x1: 2

x2: 1

x3: 1

x4: 2

x5: 1

constant: 20

x1: 2

x2: 2

x3: 1

x4: 2

x5: 1

constant: 24

x1= 2.000000 x2= 4.000000 x3= 0.000000 x4= 0.000000 x5= 12.000000

This a BFS

x1= 2.000000 x2= 4.000000 x3= 0.000000 x4= 6.000000 x5= 0.000000

This a BFS

x1= 2.000000 x2= 4.000000 x3= 12.000000 x4= 0.000000 x5= 0.000000

This a BFS

Total Basic Solutions=3

Total BFS = 3

Case 4:

Enter no. of equations and no. of variables respectively

2 6

Enter the equation coefficients

x1: 1

x2: 2

x3: 1

x4: 2

x5: 1

x6: 2

constant: 30

x1: 2

x2: 1

x3: 2

x4: 1

x5: 2

x6: 1

constant: 30

x1= 0.000000 x2= 0.000000 x3= 0.000000 x4= 0.000000 x5= 10.000000 x6= 10.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 0.000000 x4= 10.000000 x5= 10.000000 x6= 0.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 10.000000 x4= 0.000000 x5= 0.000000 x6= 10.000000

This a BFS

x1= 0.000000 x2= 0.000000 x3= 10.000000 x4= 10.000000 x5= 0.000000 x6= 0.000000

This a BFS

x1= 0.000000 x2= 10.000000 x3= 0.000000 x4= 0.000000 x5= 10.000000 x6= 0.000000

This a BFS

x1= 0.000000 x2= 10.000000 x3= 10.000000 x4= 0.000000 x5= 0.000000 x6= 0.000000

This a BFS

x1= 10.000000 x2= 0.000000 x3= 0.000000 x4= 0.000000 x5= 0.000000 x6= 10.000000

This a BFS

x1= 10.000000 x2= 0.000000 x3= 0.000000 x4= 10.000000 x5= 0.000000 x6= 0.000000

This a BFS

x1= 10.000000 x2= 10.000000 x3= 0.000000 x4= 0.000000 x5= 0.000000 x6= 0.000000

This a BFS

Total Basic Solutions=9

Total BFS = 9

**Code to solve LPP using Simplex Method**

#include<stdio.h>

#include<conio.h>

int m,n;

double a[100][100];

double fn[100];

int nbv[100];

int bv[100];

int dif;

void print()

{

int i,j;

for(i=0;i<n;i++)

{

printf("\t\t x%d",nbv[i]);

}

printf("\t\t 1\n");

for(i=0;i<m;i++)

{

printf("\t");

for(j=0;j<n;j++)

{

printf("\t %lf",a[i][j]);

}

printf("\t %lf\t x%d \n",a[i][j],bv[i]);

}

printf("\t");

for(i=0;i<=n;i++)

{

printf("\t %lf",fn[i]);

}

printf("\t z\n\n");

}

void change(int r,int c)

{

int t,i,j;

t=nbv[c];

nbv[c]=bv[r];

bv[r]=t;

double b;

double key=a[r][c];

for(i=0;i<m;i++)

{

if(i==r)

continue;

for(j=0;j<=n;j++)

{

if(j==c)

continue;

b=(a[i][j]\*key-a[i][c]\*a[r][j])/key;

a[i][j]=b;

}

}

for(i=0;i<m;i++)

{

if(i==r)

continue;

a[i][c]=(-1)\*a[i][c]/key;

}

for(i=0;i<=n;i++)

{

if(i==c)

continue;

b=(fn[i]\*key-a[r][i]\*fn[c])/key;

fn[i]=b;

a[r][i]=a[r][i]/key;

}

fn[c]=(-1)\*fn[c]/key;

a[r][c]=1/key;

}

int ratio(int p)

{

int i,l,c=0;

double r;

for(i=0;i<m;i++)

{

double k;

k=a[i][n]/a[i][p];

if(k>0&&c==0)

{

c=1;

r=k;

l=i;

continue;

}

if(k<r&&k>0)

{

r=k;

l=i;

}

}

return l;

}

int find\_min()

{

int i;

int p=0;

for(i=1;i<n;i++)

{

if(fn[i]<fn[p])

{

p=i;

}

}

return p;

}

int main()

{

int i,j;

printf("Enter m,n\n");

scanf("%d %d",&m,&n);

dif=n-m;

printf("Enter the equation coefficients \n");

for(i=0;i<m;i++)

{

bv[i]=n+i+1;

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j]);

}

int t;

printf("Enter function coefficients \n");

for(i=0;i<n;i++)

{

nbv[i]=i+1;

printf("x%d: ",(i+1));

scanf("%lf",&fn[i]);

fn[i]=0-fn[i];

}

printf("constant: ");

scanf("%lf",&fn[i]);

int p;

int x;

print();

p=find\_min();

while(fn[p]<0)

{

x=ratio(p);

change(x,p);

print();

p=find\_min();

}

getch();

return 0;

}

**Output**

Case 1:

Enter m,n

3 4

Enter the equation coefficients

x1: 2

x2: 1

x3: 2

x4: 1

constant: 20

x1: 1

x2: 1

x3: 1

x4: 1

constant: 12

x1: 1

x2: 2

x3: 3

x4: 4

constant: 36

Enter function coefficients

x1: 1

x2: 2

x3: 3

x4: 4

constant: 0

x1 x2 x3 x4 1

2.000000 1.000000 2.000000 1.000000 20.000000 x5

1.000000 1.000000 1.000000 1.000000 12.000000 x6

1.000000 2.000000 3.000000 4.000000 36.000000 x7

-1.000000 -2.000000 -3.000000 -4.000000 0.000000 z

x1 x2 x3 x7 1

1.750000 0.500000 1.250000 -0.250000 11.000000 x5

0.750000 0.500000 0.250000 -0.250000 3.000000 x6

0.250000 0.500000 0.750000 0.250000 9.000000 x4

0.000000 0.000000 0.000000 1.000000 36.000000 z

Case 2:

Enter m,n

2 3

Enter the equation coefficients

x1: 1

x2: 1

x3: 1

constant: 1

x1: 1

x2: 2

x3: 4

constant: 3

Enter function coefficients

x1: 100

x2: 110

x3: 160

constant: 0

x1 x2 x3 1

1.000000 1.000000 1.000000 1.000000 x4

1.000000 2.000000 4.000000 3.000000 x5

-100.000000 -110.000000 -160.000000 0.000000 z

x1 x2 x5 1

0.750000 0.500000 -0.250000 0.250000 x4

0.250000 0.500000 0.250000 0.750000 x3

-60.000000 -30.000000 40.000000 120.000000 z

x4 x2 x5 1

1.333333 0.666667 -0.333333 0.333333 x1

-0.333333 0.333333 0.333333 0.666667 x3

80.000000 10.000000 20.000000 140.000000 z

**Code to solve a LPP using Big-M Method**

#include<stdio.h>

#include<conio.h>

const int W = 10000;

int m,n,less,equal,great;

double a[100][100];

double fn[100];

int nbv[100];

int bv[100];

int dif;

void print()

{

printf("\n");

int i,j;

for(i=0;i<n;i++)

{

printf("\t\t x%d",nbv[i]);

}

printf("\t\t 1");

printf("\n\t\t");

for(i=0;i<n+1;i++)

{

printf("----------------");

}

printf("\n");

for(i=0;i<m;i++)

{

printf("\t");

for(j=0;j<n;j++)

{

printf("\t %lf",a[i][j]);

}

printf("\t %lf\t| x%d \n",a[i][j],bv[i]);

}

printf("\t");

for(i=0;i<=n;i++)

{

printf("\t %lf",fn[i]);

}

printf("\t| z\n\n");

}

void sol\_print()

{

int i;

printf("\n\t\t The solution is:\n\n");

for(i=0;i<m;i++)

{

printf("\t\t x%d = %lf\n",bv[i],a[i][n]);

}

printf("\n\t\t z = %lf \n",fn[n]);

printf("\n\t\t Other variables are 0\n\n");

}

void change(int r,int c)

{

int t,i,j;

t=nbv[c];

nbv[c]=bv[r];

bv[r]=t;

double b;

double key=a[r][c];

for(i=0;i<m;i++)

{

if(i==r)

continue;

for(j=0;j<=n;j++)

{

if(j==c)

continue;

b=(a[i][j]\*key-a[i][c]\*a[r][j])/key;

a[i][j]=b;

}

}

for(i=0;i<m;i++)

{

if(i==r)

continue;

a[i][c]=(-1)\*a[i][c]/key;

}

for(i=0;i<=n;i++)

{

if(i==c)

continue;

b=(fn[i]\*key-a[r][i]\*fn[c])/key;

fn[i]=b;

a[r][i]=a[r][i]/key;

}

fn[c]=(-1)\*fn[c]/key;

a[r][c]=1/key;

}

int ratio(int p)

{

int i,l=-1,c=0;

double r;

for(i=0;i<m;i++)

{

double k;

k=a[i][n]/a[i][p];

if(k>0&&c==0)

{

c=1;

r=k;

l=i;

continue;

}

if(k>0&&k<r)

{

r=k;

l=i;

}

}

return l;

}

int find\_min()

{

int i;

int p=0;

for(i=1;i<n;i++)

{

if(fn[i]<fn[p])

{

p=i;

}

}

return p;

}

int main()

{

int i,j;

printf("Enter m,n\n");

scanf("%d %d",&m,&n);

dif=n-m;

printf("Enter the no. of equations with <= sign:\n");

scanf("%d",&less);

printf("Enter the no. of equations with = sign:\n");

scanf("%d",&equal);

printf("Enter the no. of equations with >= sign:\n");

scanf("%d",&great);

if(less>0)

printf("Enter the equation coefficients for <= sign\n");

for(i=0;i<less;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j+great]);

}

if(equal>0)

printf("Enter the equation coefficients for = sign\n");

for(i=less;i<less+equal;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j+great]);

}

if(great>0)

printf("Enter the equation coefficients for >= sign\n");

for(i=0;i<great;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[less+equal+i][j]);

}

for(j=n;j<n+great;j++)

{

if(i==j-n)

a[less+equal+i][j] = -1;

else

a[less+equal+i][j] = 0;

}

printf("constant: ");

scanf("%lf",&a[less+equal+i][n+great]);

}

for(i=0;i<m;i++)

{

bv[i]=n+great+i+1;

}

int t;

double sum=0.0;

printf("Enter function coefficients \n");

for(i=0;i<n;i++)

{

nbv[i]=i+1;

printf("x%d: ",(i+1));

scanf("%lf",&fn[i]);

sum=0.0;

for(j=0;j<equal;j++)

{

sum+=a[j+less][i];

}

for(j=0;j<great;j++)

{

sum+=a[j+less+equal][i];

}

fn[i]=sum\*W+fn[i];

fn[i]=0-fn[i];

}

printf("constant: ");

scanf("%lf",&fn[n+great]);

for(i=n;i<n+great;i++)

{

nbv[i]=i+1;

fn[i]=W;

}

sum=0.0;

for(j=0;j<equal;j++)

{

sum+=a[j+less][n+great];

}

for(j=0;j<great;j++)

{

sum+=a[j+less+equal][n+great];

}

fn[i]=sum\*W+fn[i];

fn[i]=0-fn[i];

n=n+great;

int p;

int x;

print();

p=find\_min();

while(fn[p]<0)

{

x=ratio(p);

if(x==-1)

{

printf("\n\t\tNo solution\n");

break;

}

change(x,p);

print();

p=find\_min();

}

if(x!=-1)

sol\_print();

if(fn[p]==0)

{

printf("\n\t\tAlternate Solution:\n\n");

x=ratio(p);

if(x==-1)

{

printf("No solution\n");

}

else{

change(x,p);

print();

sol\_print();}

}

getch();

return 0;

}

**Output**

Case 1:

Enter no of equations and no of variables respectively.

3 2

Enter the no. of equations with <= sign:

0

Enter the no. of equations with = sign:

0

Enter the no. of equations with >= sign:

3

Enter the equation coefficients for >= sign

x1: 1

x2: 1

constant: 100

x1: 1

x2: 2

constant: 110

x1: 1

x2: 4

constant: 160

Enter function coefficients

x1: -1

x2: -3

constant: 0

Taking M to be 10000, we solve the problem

x1 x2 x3 x4 x5 1

---------------------------------------------------------------------------------------------------------------

1.000000 1.000000 -1.000000 0.000000 0.000000 100.000000 | x6

1.000000 2.000000 0.000000 -1.000000 0.000000 110.000000 | x7

1.000000 4.000000 0.000000 0.000000 -1.000000 160.000000 | x8

-29999.00 -69997.00 10000.00 10000.00 10000.00 -3700000.00 | z

x1 x8 x3 x4 x5 1

-------------------------------------------------------------------------------------------------------------

0.750000 -0.250000 -1.000000 0.000000 0.250000 60.000000 | x6

0.500000 -0.500000 0.000000 -1.000000 0.500000 30.000000 | x7

0.250000 0.250000 0.000000 0.000000 -0.250000 40.000000 | x2

-12499.75 17499.25 10000.00 10000.00 -7499.25 -900120.00 | z

x7 x8 x3 x4 x5 1

-------------------------------------------------------------------------------------------------------------

-1.500000 0.500000 -1.000000 1.500000 -0.500000 15.000000 | x6

2.000000 -1.000000 0.000000 -2.000000 1.000000 60.000000 | x1

-0.500000 0.500000 0.000000 0.500000 -0.500000 25.000000 | x2

24999.50 4999.50 10000.00 -14999.50 5000.50 -150135.00 | z

x7 x8 x3 x6 x5 1

-------------------------------------------------------------------------------------------------------------

-1.000000 0.333333 -0.666667 0.666667 -0.333333 10.000000 | x4

0.000000 -0.333333 -1.333333 1.333333 0.333333 80.000000 | x1

0.000000 0.333333 0.333333 -0.333333 -0.333333 20.000000 | x2

10000.00 9999.33 0.333333 9999.67 0.67 -140.00 | z

The solution is:

x4 = 10.000000

x1 = 80.000000

x2 = 20.000000

z = -140.000000

Other variables are 0

Case 2:

Enter no of equations and no of variables respectively.

3 2

Enter the no. of equations with <= sign:

1

Enter the no. of equations with = sign:

1

Enter the no. of equations with >= sign:

1

Enter the equation coefficients for <= sign

x1: 1

x2: 4

constant: 200

Enter the equation coefficients for = sign

x1: 1

x2: 2

constant: 140

Enter the equation coefficients for >= sign

x1: 1

x2: 1

constant: 100

Enter function coefficients

x1:-5

x2: -1

constant: 0

Taking M to be 10000, we solve the problem

x1 x2 x3 1

-------------------------------------------------------------------------

1.000000 4.000000 0.000000 200.000000 | x4

1.000000 2.000000 0.000000 140.000000 | x5

1.000000 1.000000 -1.000000 100.000000 | x6

-19995.00 -29999.00 10000.00 -2400000.00 | z

x1 x4 x3 1

----------------------------------------------------------------

0.250000 0.250000 0.000000 50.000000 | x2

0.500000 -0.500000 0.000000 40.000000 | x5

0.750000 -0.250000 -1.000000 50.000000 | x6

-12495.25 7499.75 10000.00 -900050.00 | z

x6 x4 x3 1

------------------------------------------------------------------------

-0.333333 0.333333 0.333333 33.333333 | x2

-0.666667 -0.333333 0.666667 6.666667 | x5

1.333333 -0.333333 -1.333333 66.666667 | x1

16660.33 3334.67 -6660.33 -67033.33 | z

x6 x4 x5 1

------------------------------------------------------------------------

0.000000 0.500000 -0.500000 30.000000 | x2

-1.000000 -0.500000 1.500000 10.000000 | x3

0.000000 -1.000000 2.000000 80.000000 | x1

10000.00 4.50 9990.50 -430.00 | z

The solution is:

x2 = 30.000000

x3 = 10.000000

x1 = 80.000000

z = -430.000000

Other variables are 0

Case 3:

Enter no of equations and no of variables respectively.

3 4

Enter the no. of equations with <= sign:

0

Enter the no. of equations with = sign:

0

Enter the no. of equations with >= sign:

3

Enter the equation coefficients for >= sign

x1: 1

x2: 2

x3: 1

x4: 1

constant: 100

x1: 1

x2: 1

x3: 2

x4: 1

constant: 120

x1: 2

x2: 1

x3: 1

x4: 2

constant: 200

Enter function coefficients

x1: -1

x2: -1

x3: -1

x4: -1

constant: 0

Taking M to be 10000, we solve the problem

x1 x2 x3 x4 x5 x6 x7 1

--------------------------------------------------------------------------------------------------------------------------------------------------

1.000000 2.000000 1.000000 1.000000 -1.000000 0.000000 0.000000 100.000000 | x8

1.000000 1.000000 2.000000 1.000000 0.000000 -1.000000 0.000000 120.000000 | x9

2.000000 1.000000 1.000000 2.000000 0.000000 0.000000 -1.000000 200.000000 | x10

-39999.00 -39999.00 -39999.00 -39999.00 10000.00 10000.00 10000.00 -4200000.00 | z

x8 x2 x3 x4 x5 x6 x7 1

---------------------------------------------------------------------------------------------------------------------------------------------------

1.000000 2.000000 1.000000 1.000000 -1.000000 0.000000 0.000000 100.000000 | x1

-1.000000 -1.000000 1.000000 0.000000 1.000000 -1.000000 0.000000 20.000000 | x9

-2.000000 -3.000000 -1.000000 0.000000 2.000000 0.000000 -1.000000 0.000000 | x10

39999.00 39999.00 0.00 0.00 -29999.00 10000.00 10000.00 -200100.00 | z

x8 x2 x3 x4 x9 x6 x7 1

---------------------------------------------------------------------------------------------------------------------------------------------------

0.000000 1.000000 2.000000 1.000000 1.000000 -1.000000 0.000000 120.000000 | x1

-1.000000 -1.000000 1.000000 0.000000 1.000000 -1.000000 0.000000 20.000000 | x5

0.000000 -1.000000 -3.000000 0.000000 -2.000000 2.000000 -1.000000 -40.000000 | x10

10000.00 10000.00 29999.00 0.00 29999.00 -19999.00 10000.00 399880.00 | z

No solution

**Code to solve a method using Two Phase Method**

#include<stdio.h>

#include<conio.h>

const int W = 10000;

int m,n,less,equal,great;

double a[100][100];

int nbv[100];

int bv[100];

int dif;

void print(double fn[])

{

printf("\n");

int i,j;

for(i=0;i<n;i++)

{

printf("\t\t x%d",nbv[i]);

}

printf("\t\t 1");

printf("\n\t\t");

for(i=0;i<n+1;i++)

{

printf("----------------");

}

printf("\n");

for(i=0;i<m;i++)

{

printf("\t");

for(j=0;j<n;j++)

{

printf("\t %lf",a[i][j]);

}

printf("\t %lf\t| x%d \n",a[i][j],bv[i]);

}

printf("\t");

for(i=0;i<=n;i++)

{

printf("\t %lf",fn[i]);

}

printf("\t| z\n\n");

}

void sol\_print(double fn[])

{

int i;

printf("\n\t\t The solution is:\n\n");

for(i=0;i<m;i++)

{

printf("\t\t x%d = %lf\n",bv[i],a[i][n]);

}

printf("\n\t\t z = %lf \n",fn[n]);

printf("\n\t\t Other variables are 0\n\n");

}

void change(int r,int c,double fn[])

{

int t,i,j;

t=nbv[c];

nbv[c]=bv[r];

bv[r]=t;

double b;

double key=a[r][c];

for(i=0;i<m;i++)

{

if(i==r)

continue;

for(j=0;j<=n;j++)

{

if(j==c)

continue;

b=(a[i][j]\*key-a[i][c]\*a[r][j])/key;

a[i][j]=b;

}

}

for(i=0;i<m;i++)

{

if(i==r)

continue;

a[i][c]=(-1)\*a[i][c]/key;

}

for(i=0;i<=n;i++)

{

if(i==c)

continue;

b=(fn[i]\*key-a[r][i]\*fn[c])/key;

fn[i]=b;

a[r][i]=a[r][i]/key;

}

fn[c]=(-1)\*fn[c]/key;

a[r][c]=1/key;

}

int ratio(int p,double fn[])

{

int i,l=-1,c=0;

double r;

for(i=0;i<m;i++)

{

double k;

k=a[i][n]/a[i][p];

if(k>0&&c==0)

{

c=1;

r=k;

l=i;

continue;

}

if(k>0&&k<r)

{

r=k;

l=i;

}

}

return l;

}

int find\_min(double fn[])

{

int i;

int p=0;

for(i=1;i<n;i++)

{

if(fn[i]<fn[p])

{

p=i;

}

}

return p;

}

int main()

{

int i,j;

double fn[100],fn1[100],fn2[100];

printf("Enter no of equations and no of variables respectively.\n");

scanf("%d %d",&m,&n);

dif=n-m;

printf("Enter the no. of equations with <= sign:\n");

scanf("%d",&less);

printf("Enter the no. of equations with = sign:\n");

scanf("%d",&equal);

printf("Enter the no. of equations with >= sign:\n");

scanf("%d",&great);

if(less>0)

printf("Enter the equation coefficients for <= sign\n");

for(i=0;i<=n+great;i++)

{

fn1[i]=0;

fn2[i]=0;

}

for(i=0;i<less;i++)

{

for(j=0;j<n;j++)

{

fn1[j]=0;

fn2[j]=0;

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j+great]);

}

fn1[n+great]=0;

fn2[n+great]=0;

if(equal>0)

printf("Enter the equation coefficients for = sign\n");

for(i=less;i<less+equal;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

fn1[j]-=a[i][j];

}

printf("constant: ");

scanf("%lf",&a[i][j+great]);

fn1[n+great]-=a[i][j+great];

}

if(great>0)

printf("Enter the equation coefficients for >= sign\n");

for(i=0;i<great;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[less+equal+i][j]);

fn1[j]-=a[less+equal+i][j];

//printf("%lf\n",fn1[0]);

}

for(j=n;j<n+great;j++)

{

fn2[j]=0;

if(i==j-n)

a[less+equal+i][j] = -1;

else

a[less+equal+i][j] = 0;

fn1[j]-=a[less+equal+i][j];

}

printf("constant: ");

scanf("%lf",&a[less+equal+i][n+great]);

fn1[n+great]-=a[less+equal+i][n+great];

}

//printf("%lf\n",fn1[0]);

for(i=0;i<m;i++)

{

bv[i]=n+great+i+1;

}

int t;

double sum=0.0;

printf("Enter function coefficients \n");

for(i=0;i<n;i++)

{

nbv[i]=i+1;

printf("x%d: ",(i+1));

scanf("%lf",&fn[i]);

fn[i]=0-fn[i];

fn2[i]=fn[i];

}

for(i=n;i<n+great;i++)

{

fn[i]=0;

fn2[i]=0;

nbv[i]=i+1;

}

printf("constant: ");

scanf("%lf",&fn[n+great]);

n=n+great;

int p;

int x;

print(fn1);

p=find\_min(fn1);

while(fn1[p]<0)

{

x=ratio(p,fn1);

if(x==-1)

{

printf("\n\t\tNo solution\n");

break;

}

change(x,p,fn1);

print(fn1);

p=find\_min(fn1);

}

if(x!=-1)

{

printf("Solution after phase 1:\n");

sol\_print(fn1);

}

if(fn1[p]==0)

{

for(i=0;i<n;i++)

{

if(nbv[i]>n)

{

for(j=0;j<m;j++)

{

a[j][i]=0;

}

fn2[i]=0;

}

}

int ch=0;

for(i=0;i<m;i++)

{

if(bv[i]<=n-great)

{

ch++;

for(j=0;j<n;j++)

{

if(nbv[j]<=n)

{

if(ch==1)

fn2[j]=fn[j];

fn2[j]-=(fn[bv[i]-1]\*a[i][j]);

}

}

fn2[j]-=(fn[bv[i]-1]\*a[i][j]);

}

}

print(fn2);

p=find\_min(fn2);

while(fn2[p]<0)

{

x=ratio(p,fn2);

if(x==-1)

{

printf("\n\t\tNo solution\n");

break;

}

change(x,p,fn2);

print(fn2);

p=find\_min(fn2);

}

if(x!=-1)

{

printf("Solution after phase 2:\n");

sol\_print(fn2);

}

/\*if(fn2[p]==0)

{

printf("\n\t\tAlternate Solution:\n\n");

x=ratio(p,fn2);

if(x==-1)

{

printf("No solution\n");

}

else{

change(x,p,fn2);

print(fn2);

sol\_print(fn2);}

}\*/

}

else

printf("No soln");

getch();

return 0;

}

Output:

Enter no of equations and no of variables respectively.

3

2

Enter the no. of equations with <= sign:

0

Enter the no. of equations with = sign:

0

Enter the no. of equations with >= sign:

3

Enter the equation coefficients for >= sign

x1: 1

x2: 1

constant: 10

x1: 1

x2: 2

constant: 11

x1: 1

x2: 4

constant: 16

Enter function coefficients

x1: -1

x2: -3

constant: 0

x1 x2 x3 x4 x5 1

------------------------------------------------------------------------------------------------

1.000000 1.000000 -1.000000 0.000000 0.000000 10.000000 | x6

1.000000 2.000000 0.000000 -1.000000 0.000000 11.000000 | x7

1.000000 4.000000 0.000000 0.000000 -1.000000 16.000000 | x8

-3.000000 -7.000000 1.000000 1.000000 1.000000 -37.000000 | z

x1 x8 x3 x4 x5 1

------------------------------------------------------------------------------------------------

0.750000 -0.250000 -1.000000 0.000000 0.250000 6.000000 | x6

0.500000 -0.500000 0.000000 -1.000000 0.500000 3.000000 | x7

0.250000 0.250000 0.000000 0.000000 -0.250000 4.000000 | x2

-1.250000 1.750000 1.000000 1.000000 -0.750000 -9.000000 | z

x7 x8 x3 x4 x5 1

------------------------------------------------------------------------------------------------

-1.500000 0.500000 -1.000000 1.500000 -0.500000 1.500000 | x6

2.000000 -1.000000 0.000000 -2.000000 1.000000 6.000000 | x1

-0.500000 0.500000 0.000000 0.500000 -0.500000 2.500000 | x2

2.500000 0.500000 1.000000 -1.500000 0.500000 -1.500000 | z

x7 x8 x3 x6 x5 1

------------------------------------------------------------------------------------------------

-1.000000 0.333333 -0.666667 0.666667 -0.333333 1.000000 | x4

0.000000 -0.333333 -1.333333 1.333333 0.333333 8.000000 | x1

0.000000 0.333333 0.333333 -0.333333 -0.333333 2.000000 | x2

1.000000 1.000000 0.000000 1.000000 0.000000 0.000000 | z

Solution after phase 1:

The solution is:

x4 = 1.000000

x1 = 8.000000

x2 = 2.000000

z = 0.000000

Other variables are 0

x7 x8 x3 x6 x5 1

------------------------------------------------------------------------------------------------

0.000000 0.000000 -0.666667 0.000000 -0.333333 1.000000 | x4

0.000000 0.000000 -1.333333 0.000000 0.333333 8.000000 | x1

0.000000 0.000000 0.333333 0.000000 -0.333333 2.000000 | x2

0.000000 0.000000 0.333333 0.000000 0.666667 -14.000000 | z

Solution after phase 2:

The solution is:

x4 = 1.000000

x1 = 8.000000

x2 = 2.000000

z = -14.000000

Other variables are 0

**Code to solve a LPP using Dual Simplex Method**

#include<stdio.h>

#include<conio.h>

int m,n;

double a[100][100];

double fn[100];

int nbv[100];

int bv[100];

int dif;

void print()

{

int i,j;

for(i=0;i<n;i++)

{

printf("\t\t x%d",nbv[i]);

}

printf("\t\t 1\n");

for(i=0;i<m;i++)

{

printf("\t");

for(j=0;j<n;j++)

{

printf("\t %lf",a[i][j]);

}

printf("\t %lf\t x%d \n",a[i][j],bv[i]);

}

printf("\t");

for(i=0;i<=n;i++)

{

printf("\t %lf",fn[i]);

}

printf("\t z\n\n");

}

void sol\_print()

{

int i;

printf("\n\t\t The solution is:\n\n");

for(i=0;i<m;i++)

{

printf("\t\t x%d = %lf\n",bv[i],a[i][n]);

}

printf("\n\t\t z = %lf \n",fn[n]);

printf("\n\t\t Other variables are 0\n\n");

}

void change(int c,int r)

{

int t,i,j;

t=nbv[c];

nbv[c]=bv[r];

bv[r]=t;

double b;

double key=a[r][c];

for(i=0;i<m;i++)

{

if(i==r)

continue;

for(j=0;j<=n;j++)

{

if(j==c)

continue;

b=(a[i][j]\*key-a[i][c]\*a[r][j])/key;

a[i][j]=b;

}

}

for(i=0;i<m;i++)

{

if(i==r)

continue;

a[i][c]=(-1)\*a[i][c]/key;

}

for(i=0;i<=n;i++)

{

if(i==c)

continue;

b=(fn[i]\*key-a[r][i]\*fn[c])/key;

fn[i]=b;

a[r][i]=a[r][i]/key;

}

fn[c]=(-1)\*fn[c]/key;

a[r][c]=1/key;

}

int ratio(int p)

{

int i,l=-1,c=0;

double r;

for(i=0;i<n;i++)

{

double k;

k=fn[i]/a[p][i];

if(a[i][n]>=0)

return -1;

if(k<0&&c==0)

{

c=1;

r=k;

l=i;

continue;

}

if(k>r&&k<0)

{

r=k;

l=i;

}

}

return l;

}

int find\_min()

{

int i;

int p=0;

for(i=1;i<m;i++)

{

if(a[i][n]<a[p][n])

{

p=i;

}

}

return p;

}

int main()

{

int i,j;

printf("Enter no. of equations and no. of variables respectively\n");

scanf("%d %d",&m,&n);

dif=n-m;

printf("Enter the equation coefficients \n");

for(i=0;i<m;i++)

{

bv[i]=n+i+1;

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j]);

}

int t;

printf("Enter function coefficients \n");

for(i=0;i<n;i++)

{

nbv[i]=i+1;

printf("x%d: ",(i+1));

scanf("%lf",&fn[i]);

fn[i]=0-fn[i];

}

printf("constant: ");

scanf("%lf",&fn[i]);

int p;

int x;

print();

p=find\_min();

while(a[p][n]<0)

{

x=ratio(p);

if(x==-1)

{

printf("\n\t\tUnbounded solution\n");

break;

}

change(x,p);

print();

p=find\_min();

}

if(x!=-1)

{

sol\_print();

}

if(a[p][n]==0)

{

printf("\n\t\tAlternate Solution:\n\n");

x=ratio(p);

if(x==-1)

{

printf("No solution\n");

}

else{

change(x,p);

print();

sol\_print();}

}

getch();

return 0;

}

Output:

Enter no. of equations and no. of variables respectively

2 3

Enter the equation coefficients

x1: -1

x2: -2

x3: 1

constant: -5

x1: -2

x2: -1

x3: -1

constant: -4

Enter function coefficients

x1: -5

x2: -2

x3: -3

constant: 0

x1 x2 x3 1

----------------------------------------------------------------

-1.000000 -2.000000 1.000000 -5.000000 | x4

-2.000000 -1.000000 -1.000000 -4.000000 | x5

5.000000 2.000000 3.000000 0.000000 | z

x1 x4 x3 1

----------------------------------------------------------------

0.500000 -0.500000 -0.500000 2.500000 | x2

-1.500000 -0.500000 -1.500000 -1.500000 | x5

4.000000 1.000000 4.000000 -5.000000 | z

x1 x5 x3 1

----------------------------------------------------------------

2.000000 -1.000000 1.000000 4.000000 | x2

3.000000 -2.000000 3.000000 3.000000 | x4

1.000000 2.000000 1.000000 -8.000000 | z

The solution is:

x2 = 4.000000

x4 = 3.000000

z = -8.000000

Other variables are 0

**Code to solve a LPP using Revised Simplex Method**

#include<stdio.h>

#define exc(a,b,c) (a)=(b),(b)=(c),(c)=(a)

const int W = 10000;

int m,n,less,equal,great;

double a[100][100];

double b[100][100]={0};

double d[100][100]={0};

double cb[100];

double y[100];

double xb[100];

double fn[100];

double cnt[100];

int nbv[100];

int bv[100];

int dif;

double calz()

{

int i;

double s=0.0;

for(i=0;i<m;i++)

{

s+=(cb[i]\*xb[i]);

}

return s;

}

void sol\_print()

{

int i;

printf("\n\t\t The solution is:\n\n");

for(i=0;i<m;i++)

{

printf("\t\t x%d = %lf\n",bv[i],xb[i]);

}

printf("\n\t\t z = %lf \n",calz());

printf("\n\t\t Other variables are 0\n\n");

}

void change(int r,double k[])

{

int i,j,h;

double e[100][100]={0.0};

double l[100][100]={0.0};

for(i=0;i<m;i++)

{

e[i][i]=1;

e[i][r]=k[i];

}

for(i=0;i<m;i++)

{

for(j=0;j<m;j++)

{

for(h=0;h<m;h++)

{

l[i][j]+=(e[i][h]\*d[h][j]);

}

}

}

for(i=0;i<m;i++){

for(j=0;j<m;j++)

{

d[i][j]=l[i][j];

}

}

}

int ratio(int p)

{

double k[100],s;

int i,c=0,j,q=-1;

double t=0;

for(i=0;i<m;i++)

{

s=0.0;

for(j=0;j<m;j++)

{

s+=d[i][j]\*a[j][p];

}

k[i]=s;

if(s>0)

{

double r=xb[i]/s;

if(c==0)

{

c++;

t=r;

q=i;

}

if(t>r)

{

t=r;

q=i;

}

}

}

if(q!=-1)

{

for(i=0;i<m;i++)

{

exc(t,b[i][q],a[i][p]);

if(i!=q)

k[i]=-k[i]/k[q];

}

k[q]=1.0/k[q];

change(q,k);

exc(t,fn[p],cb[q]);

exc(i,bv[q],nbv[p]);

return 1;

}

return 0;

}

int find\_min()

{

int i,pos=0,j;

double p=0,min=0;

for(i=0;i<n;i++)

{

p=0.0;

for(j=0;j<m;j++)

{

p+=y[j]\*a[j][i];

}

p=p-fn[i];

if(i==0)

{

min=p;

}

if(p<min)

{

min=p;

pos=i;

}

}

if(min>=0)

return -1;

return pos;

}

void caly()

{

int i,j;

double s;

for(i=0;i<m;i++)

{

s=0.0;

for(j=0;j<m;j++)

{

s+=cb[j]\*d[j][i];

}

y[i]=s;

}

}

void calxb()

{

int i,j;

double s;

for(i=0;i<m;i++)

{

s=0.0;

for(j=0;j<m;j++)

{

s+=d[i][j]\*cnt[j];

}

xb[i]=s;

}

}

void prit(double v[][100])

{

int i,j;

for(i=0;i<m;i++)

{

for(j=0;j<m;j++)

printf(" %lf ",v[i][j]);

printf("\n");

}

printf("\n");

}

void print(double v[])

{

int i,j;

for(i=0;i<m;i++)

{

printf(" %lf ",v[i]);

}

printf("\n\n");

}

int main()

{

freopen("input.txt","r",stdin);

int i,j;

printf("Enter no of equations and no of variables respectively.\n");

scanf("%d %d",&m,&n);

dif=n-m;

printf("Enter the no. of equations with <= sign:\n");

scanf("%d",&less);

printf("Enter the no. of equations with = sign:\n");

scanf("%d",&equal);

printf("Enter the no. of equations with >= sign:\n");

scanf("%d",&great);

if(less>0)

printf("Enter the equation coefficients for <= sign\n");

for(i=0;i<less;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&cnt[i]);

}

if(equal>0)

printf("Enter the equation coefficients for = sign\n");

for(i=less;i<less+equal;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

}

printf("constant: ");

scanf("%lf",&cnt[i]);

}

if(great>0)

printf("Enter the equation coefficients for >= sign\n");

for(i=0;i<great;i++)

{

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[less+equal+i][j]);

}

for(j=n;j<n+great;j++)

{

if(i==j-n)

a[less+equal+i][j] = -1;

else

a[less+equal+i][j] = 0;

}

printf("constant: ");

scanf("%lf",&cnt[less+equal+i]);

}

for(i=0;i<m;i++)

{

bv[i]=n+great+i+1;

}

int t;

double sum=0.0;

printf("Enter function coefficients \n");

for(i=0;i<n;i++)

{

nbv[i]=i+1;

printf("x%d: ",(i+1));

scanf("%lf",&fn[i]);

}

printf("constant: ");

scanf("%lf",&fn[n+m+great]);

for(i=n;i<n+great;i++)

{

nbv[i]=i+1;

fn[i]=0;

}

for(i=n+less+great;i<n+m+great;i++)

fn[i]=-W;

n=n+great;

printf("\n\nTaking M to be %d, we solve the problem\n\n",W);

int p,q;

int x;

for(i=0;i<m;i++)

{

b[i][i]=1;

d[i][i]=1;

cb[i]=fn[n+i];

}

do{

printf("\nB:\n");

prit(b);

printf("B inverse:\n");

prit(d);

caly();

printf("Y:\n");

print(y);

calxb();

p=find\_min();

if(p>=0)

q=ratio(p);

else

q=0;

}while(q==1);

sol\_print();

return 0;

}

**Output:**

Case 1:

Enter no of equations and no of variables respectively.

3

2

Enter the no. of equations with <= sign:

0

Enter the no. of equations with = sign:

0

Enter the no. of equations with >= sign:

3

Enter the equation coefficients for >= sign

x1: 1

x2: 1

constant: 10

x1: 1

x2: 2

constant: 11

x1: 1

x2: 4

constant: 16

Enter function coefficients

x1: -1

x2: -3

constant: 0

B:

1.000000 0.000000 0.000000

0.000000 1.000000 0.000000

0.000000 0.000000 1.000000

B inverse:

1.000000 0.000000 0.000000

0.000000 1.000000 0.000000

0.000000 0.000000 1.000000

Y:

0.000000 0.000000 0.000000

B:

1.000000 0.000000 1.000000

0.000000 1.000000 2.000000

0.000000 0.000000 4.000000

B inverse:

1.000000 0.000000 -0.250000

0.000000 1.000000 -0.500000

0.000000 0.000000 0.250000

Y:

0.000000 0.000000 0.750000

B:

1.000000 1.000000 1.000000

0.000000 1.000000 2.000000

0.000000 1.000000 4.000000

B inverse:

1.000000 -1.500000 0.500000

0.000000 2.000000 -1.000000

0.000000 -0.500000 0.500000

Y:

0.000000 0.500000 0.500000

The solution is:

x3 = 1.500000

x1 = 6.000000

x2 = 2.500000

z = 13.500000

Other variables are 0

Case 2:

Enter no of equations and no of variables respectively.

3 2

Enter the no. of equations with <= sign:

0

Enter the no. of equations with = sign:

0

Enter the no. of equations with >= sign:

3

Enter the equation coefficients for >= sign

x1: 1

x2: 1

constant: 100

x1: 1

x2: 2

constant: 110

x1: 1

x2: 4

constant: 160

Enter function coefficients

x1: -1

x2: -3

constant: 0

Taking M to be 10000, we solve the problem

B:

1.000000 0.000000 0.000000

0.000000 1.000000 0.000000

0.000000 0.000000 1.000000

B inverse:

1.000000 0.000000 0.000000

0.000000 1.000000 0.000000

0.000000 0.000000 1.000000

Y:

-10000.000000 -10000.000000 -10000.000000

B:

1.000000 0.000000 1.000000

0.000000 1.000000 2.000000

0.000000 0.000000 4.000000

B inverse:

1.000000 0.000000 -0.250000

0.000000 1.000000 -0.500000

0.000000 0.000000 0.250000

Y:

-10000.000000 -10000.000000 7499.250000

B:

1.000000 1.000000 1.000000

0.000000 1.000000 2.000000

0.000000 1.000000 4.000000

B inverse:

1.000000 -1.500000 0.500000

0.000000 2.000000 -1.000000

0.000000 -0.500000 0.500000

Y:

-10000.000000 14999.500000 -5000.500000

B:

0.000000 1.000000 1.000000

-1.000000 1.000000 2.000000

0.000000 1.000000 4.000000

B inverse:

0.666667 -1.000000 0.333333

1.333333 0.000000 -0.333333

-0.333333 0.000000 0.333333

Y:

-0.333333 0.000000 -0.666667

The solution is:

x4 = 10.000000

x1 = 80.000000

x2 = 20.000000

z = -140.000000

Other variables are 0

**Code to solve IPP using Gomory’s Cutting Plane Method**

#include<stdio.h>

#include<conio.h>

#include<math.h>

int m,n;

double a[100][100];

double fn[100];

int nbv[100];

int bv[100];

int dif;

void print()

{

printf("\n");

int i,j;

for(i=0;i<n;i++)

{

printf("\t\t x%d",nbv[i]);

}

printf("\t\t 1");

printf("\n\t\t");

for(i=0;i<n+1;i++)

{

printf("----------------");

}

printf("\n");

for(i=0;i<m;i++)

{

printf("\t");

for(j=0;j<n;j++)

{

printf("\t %lf",a[i][j]);

}

printf("\t %lf\t| x%d \n",a[i][j],bv[i]);

}

printf("\t");

for(i=0;i<=n;i++)

{

printf("\t %lf",fn[i]);

}

printf("\t| z\n\n");

}

void change(int r,int c)

{

int t,i,j;

t=nbv[c];

nbv[c]=bv[r];

bv[r]=t;

double b;

double key=a[r][c];

for(i=0;i<m;i++)

{

if(i==r)

continue;

for(j=0;j<=n;j++)

{

if(j==c)

continue;

b=(a[i][j]\*key-a[i][c]\*a[r][j])/key;

a[i][j]=b;

}

}

for(i=0;i<m;i++)

{

if(i==r)

continue;

a[i][c]=(-1)\*a[i][c]/key;

}

for(i=0;i<=n;i++)

{

if(i==c)

continue;

b=(fn[i]\*key-a[r][i]\*fn[c])/key;

fn[i]=b;

a[r][i]=a[r][i]/key;

}

fn[c]=(-1)\*fn[c]/key;

a[r][c]=1/key;

}

int ratio(int p)

{

int i,l=-1,c=0,d=0;

double r;

//printf("%d\n",m);

for(i=0;i<m;i++)

{

double k;

k=a[i][n]/a[i][p];

if(a[i][n]<0)

{

d= -1;

}

if(k>=0&&c==0&&a[i][p]>0)

{

c=1;

r=k;

l=i;

continue;

}

if(k<r&&k>=0&&a[i][p]>0)

{

r=k;

l=i;

}

}

if(l!=-1&&d==-1)

return -2;

return l;

}

int find\_min()

{

int i;

int p=0;

for(i=1;i<n;i++)

{

if(fn[i]<fn[p])

{

p=i;

}

}

return p;

}

void sol\_print()

{

int i;

printf("\n\t\t The solution is:\n\n");

for(i=0;i<m;i++)

{

printf("\t\t x%d = %lf\n",bv[i],a[i][n]);

}

printf("\n\t\t z = %lf \n",fn[n]);

printf("\n\t\t Other variables are 0\n\n");

}

int max\_frac()

{

int i,pos=0,j;

double mx=a[0][n]-(int)a[0][n];

double sum=0.0,mxr;

sum=0.0;

for(j=0;j<n;j++)

{

sum+=(a[0][j]-floor(a[0][j]));

}

mxr=mx/sum;

for(i=1;i<m;i++)

{

double v=a[i][n]-floor(a[i][n]);

if(v==mx)

{

sum=0.0;

for(j=0;j<n;j++)

{

sum+=(a[i][j]-floor(a[i][j]));

}

double h=v/sum;

if(h>mxr)

{

mxr=h;

pos=i;

}

}

if(v>mx)

{

mx=v;

pos=i;

}

}

if(mx<=0.0000001)

return -1;

return pos;

}

int ratio\_dual(int p)

{

int i,l=-1,c=0;

double r;

for(i=0;i<n;i++)

{

double k;

k=fn[i]/a[p][i];

if(fn[i]<=0)

return -1;

if(k<0&&c==0)

{

c=1;

r=k;

l=i;

continue;

}

if(k>r&&k<0)

{

r=k;

l=i;

}

}

return l;

}

int find\_min\_dual()

{

int i;

int p=0;

for(i=1;i<m;i++)

{

if(a[i][n]<a[p][n])

{

p=i;

}

}

return p;

}

void change\_dual(int c,int r)

{

int t,i,j;

t=nbv[c];

nbv[c]=bv[r];

bv[r]=t;

double b;

double key=a[r][c];

for(i=0;i<m;i++)

{

if(i==r)

continue;

for(j=0;j<=n;j++)

{

if(j==c)

continue;

b=(a[i][j]\*key-a[i][c]\*a[r][j])/key;

a[i][j]=b;

}

}

for(i=0;i<m;i++)

{

if(i==r)

continue;

a[i][c]=(-1)\*a[i][c]/key;

}

for(i=0;i<=n;i++)

{

if(i==c)

continue;

b=(fn[i]\*key-a[r][i]\*fn[c])/key;

fn[i]=b;

a[r][i]=a[r][i]/key;

}

fn[c]=(-1)\*fn[c]/key;

a[r][c]=1/key;

}

int main()

{

freopen("input.txt","r",stdin);

int i,j;

printf("Enter no. of equations and no. of variables respectively\n");

scanf("%d %d",&m,&n);

printf("%d %d\n",m,n);

dif=n-m;

printf("Enter the equation coefficients \n");

for(i=0;i<m;i++)

{

bv[i]=n+i+1;

for(j=0;j<n;j++)

{

printf("x%d: ",(j+1));

scanf("%lf",&a[i][j]);

printf("%.2lf\n",a[i][j]);

}

printf("constant: ");

scanf("%lf",&a[i][j]);

printf("%.2lf\n",a[i][j]);

}

int t;

printf("Enter function coefficients \n");

for(i=0;i<n;i++)

{

nbv[i]=i+1;

printf("x%d: ",(i+1));

scanf("%lf",&fn[i]);

printf("%.2lf\n",fn[i]);

fn[i]=0-fn[i];

}

printf("constant: ");

scanf("%lf",&fn[i]);

printf("%.2lf\n",fn[i]);

int p;

int x;

print();

p=find\_min();

while(fn[p]<0)

{

x=ratio(p);

if(x==-2)

{

printf("\n\t\tInfeasible solution\n");

break;

}

if(x==-1)

{

printf("\n\t\tUnbounded solution\n");

break;

}

change(x,p);

print();

p=find\_min();

}

if(x!=-1&&x!=-2)

{

int frw=max\_frac();

for(i=0;i<=n;i++)

a[m][i]=floor(a[frw][i])-a[frw][i];

m++;

print();

p=m-1;

while(a[p][n]<0)

{

x=ratio\_dual(p);

if(x==-1)

{

printf("\n\t\tUnbounded solution\n");

break;

}

change\_dual(x,p);

print();

frw=max\_frac();

if(frw==-1)

break;

for(i=0;i<=n;i++)

a[m][i]=floor(a[frw][i])-a[frw][i];

if(a[m][n]==0.0||a[m][n]==-0.0)

break;

m++;

print();

p=m-1;

}

print();

sol\_print();

}

getch();

return 0;

}

**Output:**

Case1

Enter no. of equations and no. of variables respectively

5 4

Enter the equation coefficients

x1: 1.00

x2: 1.00

x3: 1.00

x4: 1.00

constant: 2.00

x1: 1.00

x2: 1.00

x3: 1.00

x4: -1.00

constant: 1.00

x1: 1.00

x2: 1.00

x3: -1.00

x4: 1.00

constant: 1.00

x1: 1.00

x2: -1.00

x3: 1.00

x4: 1.00

constant: 1.00

x1: -1.00

x2: 1.00

x3: 1.00

x4: 1.00

constant: 1.00

Enter function coefficients

x1: 2.00

x2: 3.00

x3: 4.00

x4: 5.00

constant: 0.00

x1 x2 x3 x4 1

--------------------------------------------------------------------------------

1.000000 1.000000 1.000000 1.000000 2.000000 | x5

1.000000 1.000000 1.000000 -1.000000 1.000000 | x6

1.000000 1.000000 -1.000000 1.000000 1.000000 | x7

1.000000 -1.000000 1.000000 1.000000 1.000000 | x8

-1.000000 1.000000 1.000000 1.000000 1.000000 | x9

-2.000000 -3.000000 -4.000000 -5.000000 0.000000 | z

x1 x2 x3 x7 1

--------------------------------------------------------------------------------

0.000000 0.000000 2.000000 -1.000000 1.000000 | x5

2.000000 2.000000 0.000000 1.000000 2.000000 | x6

1.000000 1.000000 -1.000000 1.000000 1.000000 | x4

0.000000 -2.000000 2.000000 -1.000000 0.000000 | x8

-2.000000 0.000000 2.000000 -1.000000 0.000000 | x9

3.000000 2.000000 -9.000000 5.000000 5.000000 | z

x1 x2 x8 x7 1

--------------------------------------------------------------------------------

0.000000 2.000000 -1.000000 0.000000 1.000000 | x5

2.000000 2.000000 -0.000000 1.000000 2.000000 | x6

1.000000 0.000000 0.500000 0.500000 1.000000 | x4

0.000000 -1.000000 0.500000 -0.500000 0.000000 | x3

-2.000000 2.000000 -1.000000 0.000000 0.000000 | x9

3.000000 -7.000000 4.500000 0.500000 5.000000 | z

x1 x9 x8 x7 1

--------------------------------------------------------------------------------

2.000000 -1.000000 0.000000 0.000000 1.000000 | x5

4.000000 -1.000000 1.000000 1.000000 2.000000 | x6

1.000000 -0.000000 0.500000 0.500000 1.000000 | x4

-1.000000 0.500000 0.000000 -0.500000 0.000000 | x3

-1.000000 0.500000 -0.500000 0.000000 0.000000 | x2

-4.000000 3.500000 1.000000 0.500000 5.000000 | z

x5 x9 x8 x7 1

--------------------------------------------------------------------------------

0.500000 -0.500000 0.000000 0.000000 0.500000 | x1

-2.000000 1.000000 1.000000 1.000000 0.000000 | x6

-0.500000 0.500000 0.500000 0.500000 0.500000 | x4

0.500000 0.000000 0.000000 -0.500000 0.500000 | x3

0.500000 0.000000 -0.500000 0.000000 0.500000 | x2

2.000000 1.500000 1.000000 0.500000 7.000000 | z

x5 x9 x8 x7 1

--------------------------------------------------------------------------------

0.500000 -0.500000 0.000000 0.000000 0.500000 | x1

-2.000000 1.000000 1.000000 1.000000 0.000000 | x6

-0.500000 0.500000 0.500000 0.500000 0.500000 | x4

0.500000 0.000000 0.000000 -0.500000 0.500000 | x3

0.500000 0.000000 -0.500000 0.000000 0.500000 | x2

-0.500000 -0.500000 0.000000 0.000000 -0.500000 | x10

2.000000 1.500000 1.000000 0.500000 7.000000 | z

x5 x10 x8 x7 1

--------------------------------------------------------------------------------

1.000000 -1.000000 -0.000000 -0.000000 1.000000 | x1

-3.000000 2.000000 1.000000 1.000000 -1.000000 | x6

-1.000000 1.000000 0.500000 0.500000 -0.000000 | x4

0.500000 0.000000 0.000000 -0.500000 0.500000 | x3

0.500000 0.000000 -0.500000 0.000000 0.500000 | x2

1.000000 -2.000000 -0.000000 -0.000000 1.000000 | x9

0.500000 3.000000 1.000000 0.500000 5.500000 | z

x5 x10 x8 x7 1

--------------------------------------------------------------------------------

1.000000 -1.000000 -0.000000 -0.000000 1.000000 | x1

-3.000000 2.000000 1.000000 1.000000 -1.000000 | x6

-1.000000 1.000000 0.500000 0.500000 -0.000000 | x4

0.500000 0.000000 0.000000 -0.500000 0.500000 | x3

0.500000 0.000000 -0.500000 0.000000 0.500000 | x2

1.000000 -2.000000 -0.000000 -0.000000 1.000000 | x9

-0.500000 0.000000 0.000000 -0.500000 -0.500000 | x11

0.500000 3.000000 1.000000 0.500000 5.500000 | z

x11 x10 x8 x7 1

--------------------------------------------------------------------------------

2.000000 -1.000000 -0.000000 -1.000000 -0.000000 | x1

-6.000000 2.000000 1.000000 4.000000 2.000000 | x6

-2.000000 1.000000 0.500000 1.500000 1.000000 | x4

1.000000 0.000000 0.000000 -1.000000 -0.000000 | x3

1.000000 0.000000 -0.500000 -0.500000 -0.000000 | x2

2.000000 -2.000000 -0.000000 -1.000000 -0.000000 | x9

-2.000000 -0.000000 -0.000000 1.000000 1.000000 | x5

1.000000 3.000000 1.000000 -0.000000 5.000000 | z

x0 x0 x8 x7 1

--------------------------------------------------------------------------------

2.000000 -1.000000 -0.000000 -1.000000 -0.000000 | x1

-6.000000 2.000000 1.000000 4.000000 2.000000 | x6

-2.000000 1.000000 0.500000 1.500000 1.000000 | x4

1.000000 0.000000 0.000000 -1.000000 -0.000000 | x3

1.000000 0.000000 -0.500000 -0.500000 -0.000000 | x2

2.000000 -2.000000 -0.000000 -1.000000 -0.000000 | x9

-2.000000 -0.000000 -0.000000 1.000000 1.000000 | x5

1.000000 3.000000 1.000000 -0.000000 5.000000 | z

The solution is:

x1 = -0.000000

x6 = 2.000000

x4 = 1.000000

x3 = -0.000000

x2 = -0.000000

x9 = -0.000000

x5 = 1.000000

z = 5.000000

Other variables are 0

Case 2

Enter no. of equations and no. of variables respectively

2 2

Enter the equation coefficients

x1: 1.00

x2: 1.00

constant: 3.00

x1: 4.00

x2: 1.00

constant: 8.00

Enter function coefficients

x1: 5.00

x2: 4.00

constant: 0.00

x1 x2 1

------------------------------------------------

1.000000 1.000000 3.000000 | x3

4.000000 1.000000 8.000000 | x4

-5.000000 -4.000000 0.000000 | z

x4 x2 1

------------------------------------------------

-0.250000 0.750000 1.000000 | x3

0.250000 0.250000 2.000000 | x1

1.250000 -2.750000 10.000000 | z

x4 x3 1

------------------------------------------------

-0.333333 1.333333 1.333333 | x2

0.333333 -0.333333 1.666667 | x1

0.333333 3.666667 13.666667 | z

x5 x3 1

------------------------------------------------

-0.333333 1.333333 1.333333 | x2

0.333333 -0.333333 1.666667 | x1

-0.333333 -0.666667 -0.666667 | x5

0.333333 3.666667 13.666667 | z

x5 x3 1

------------------------------------------------

-1.000000 2.000000 2.000000 | x2

1.000000 -1.000000 1.000000 | x1

-3.000000 2.000000 2.000000 | x4

1.000000 3.000000 13.000000 | z

x5 x3 1

------------------------------------------------

-1.000000 2.000000 2.000000 | x2

1.000000 -1.000000 1.000000 | x1

-3.000000 2.000000 2.000000 | x4

1.000000 3.000000 13.000000 | z

The solution is:

x2 = 2.000000

x1 = 1.000000

x4 = 2.000000

z = 13.000000

Other variables are 0