```
using CPLEX;

//Initialize data:
int n=...;
range I=1..n;
range J=1..n;
int c[I][J]=...;

//Define variables
dvar boolean  y1[I];
dvar boolean  y2[I];
dvar boolean  W1[I][J];
dvar boolean  W2[I][J];
dvar float+ z;

//Model
maximize sum(i in I , j in J)
  c[i][j]*(W1[i][j]+W2[i][j])-z;

//constraints
subject to{
forall(i in I , j in J )
  W1[i][j]<=y1[i];

 forall(i in I , j in J )
   W1[i][j]<=y1[j];
```

```
 forall(i in I , j in J )
   W1[i][j]>=y1[i]+y1[j]-1;

  forall(i in I , j in J )
  W2[i][j]<=y2[i];

 forall(i in I , j in J )
  W2[i][j]<=y2[j];

 forall(i in I , j in J )
   W2[i][j]>=y2[i]+y2[j]-1;

 forall(i in I )
  y1[i]+y2[i]==1;

 sum(i in I)
   (y1[i]-y2[i])-z==0;

 sum(i in I)
   (y1[i]-y2[i])+z==0;
};
```

The optimal answer for y1 and y2:

Y1=[1 1 1 0 0 0]

Y2=[0 0 0 1 1 1]