```
/*Title :Write a program to solve the travelling salesman problem and to print the path and
the cost using LC Branch and Bound.*/
#include<iostream>
using namespace std;
int inf=9999;
int n;
int mat[50][50];
int rm[50][50];
int temp[50][50];
void make_inf(int arr[],int size,int inf,int col)
{
	for(int i=0; i<size; i++)
	{
		int r=arr[i];
		for(int p=0; p<n; p++)
		{
			temp[r][p]=inf;
		}
	}
	for(int i=0; i<n; i++)
	{
		temp[i][col]=inf;
	}
	int first=arr[0];
	temp[col][first]=inf;
	for(int i=1; i<size; i++)
	{
		int t=arr[i];
		temp[t][first]=inf;
	}
	cout<<"------------------------\n";
	for(int i=0;i<n;i++)
	{
		for(int j=0;j<n;j++)
		{
			cout<<"\t"<<temp[i][j];
		}
		cout<<"\n";
	}
}
int check(int arr[],int size,int ch)
{
	for(int i=0; i<size; i++)
	{
		if(arr[i]==ch)
		{
			return 1;
```

```
                }
        }
        return 0;
}
int minimization(int inf,int c)
{
        int cost=c;
        for(int i=0; i<n; i++)
        {
                int min=temp[i][0];
                for(int j=1; j<n; j++)
                {
                        if(min > temp[i][j])
                        {
                                min=temp[i][j];
                        }
                }
                if(min != inf)
                {
                        cost=cost+min;
                        for(int k=0; k<n; k++)
                        {
                                if(temp[i][k] != inf)
                                {
                                        temp[i][k]=temp[i][k]-min;
                                }
                                else
                                {
                                        temp[i][k]=inf;
                                }
                        }
                }
        }
        for(int i=0; i<n; i++)
        {
                int min=temp[0][i];
                for(int j=1; j<n; j++)
                {
                        if(min > temp[j][i])
                        {
                                min=temp[j][i];
                        }
                }
                if(min != inf)
                {
                        cost=cost+min;
                        for(int k=0; k<n; k++)
                        {
```

```
                                        if(temp[k][i] != inf)
                                        {
                                                temp[k][i]=temp[k][i]-min;
                                        }
                                        else
                                        {
                                                temp[k][i]=inf;
                                        }
                                }
                        }
                }
                return cost;
}
int main()
{
        cout<<"\nEnter Number of Vertices :";
        cin>>n;
        for(int i=0; i<n; i++)
        {
                for(int j=0; j<n; j++)
                {
                        mat[i][j]=inf;
                }
        }
        int e;
        cout<<"\nEnter Number of edges :";
        cin>>e;
        for(int i=0; i<e; i++)
        {
                int u,v,wt;
                cout<<"\nEnter Source Vertex :";
                cin>>u;
                cout<<"\nEnter Destination Vertex :";
                cin>>v;
                cout<<"\nEnter Weight of this edge :";
                cin>>wt;
                mat[u][v]=wt;
        }
        cout<<"------Distance matrix------\n";
        for(int i=0;i<n;i++)
        {
                for(int j=0;j<n;j++)
                {
                        cout<<"\t"<<mat[i][j];
                }
                cout<<"\n";
        }
        int cost=0;
```

```
for(int i=0; i<n; i++)
{
	int min=mat[i][0];
	for(int j=1; j<n; j++)
	{
		if(min > mat[i][j])
		{
			min=mat[i][j];
		}
	}
	cost=cost+min;
	for(int k=0; k<n; k++)
	{
		if(mat[i][k] != inf)
		{
			rm[i][k]=mat[i][k]-min;
		}
		else
		{
			rm[i][k]=inf;
		}
	}
}
cout<<"------Row Minimization------\n";
for(int i=0;i<n;i++)
{
	for(int j=0;j<n;j++)
	{
		cout<<"\t"<<rm[i][j];
	}
	cout<<"\n";
}
for(int i=0; i<n; i++)
{
	int min=rm[0][i];
	for(int j=1; j<n; j++)
	{
		if(min > rm[j][i])
		{
			min=rm[j][i];
		}
	}
	cost=cost+min;
	for(int k=0; k<n; k++)
	{
		if(rm[k][i] != inf)
		{
			rm[k][i]=rm[k][i]-min;
```

```
                }
                else
                {
                        rm[k][i]=inf;
                }
        }
}
cout<<"------Column Minimization------\n";
for(int i=0;i<n;i++)
{
        for(int j=0;j<n;j++)
        {
                cout<<"\t"<<rm[i][j];
        }
        cout<<"\n";
}
int visited[n];
int size=0;
visited[size]=0;
size++;
int ans=cost;
while(true)
{
        int min=INT_MAX;
        int min_ind=0;
        for(int j=1; j<n; j++)
        {
                int ch=check(visited,size,j);
                if(ch==0)
                {
                        int c1=0;
                        for(int p=0; p<n; p++)
                        {
                                for(int q=0; q<n; q++)
                                {
                                        temp[p][q]=rm[p][q];
                                }
                        }
                        make_inf(visited,size,inf,j);
                        c1=minimization(inf,cost);
                        c1=c1+rm[visited[size-1]][j];
                        if(c1 < min)
                        {
                                min=c1;
                                min_ind=j;
                        }
                }
        }
```

```
                for(int p=0; p<n; p++)
                {
                        for(int q=0; q<n; q++)
                        {
                                temp[p][q]=rm[p][q];
                        }
                }
                make_inf(visited,size,inf,min_ind);
                int tpp=minimization(inf,cost);
                for(int p=0; p<n; p++)
                {
                        for(int q=0; q<n; q++)
                        {
                                rm[p][q]=temp[p][q];
                        }
                }
                visited[size]=min_ind;
                size++;
                cost=min;
                if(size==n)
                {
                        break;
                }
        }
        cout<<"\nPath :- "<<endl;
        for(int k=0; k<size; k++)
        {
                cout<<visited[k]<<" --> ";
        }
        cout<<visited[0];
        cout<<"\nMinimum cost "<<cost;
        return 0;
}
```