Traveling selesman

```
#include<bits/stdc++.h>

#include <vector>

using namespace std;

int tsp(const vector<vector<int>>& cities, int pos, int visited, vector<vector<int>>&
state)
{
   if(visited == ((1 << cities.size()) - 1))
      return cities[pos][0];

   if(state[pos][visited] != INT_MAX)
      return state[pos][visited];

   for(int i = 0; i < cities.size(); ++i)
   {
      if(i == pos || (visited & (1 << i)))
         continue;

      int distance = cities[pos][i] + tsp(cities, i, visited | (1 << i), state);
      if(distance < state[pos][visited])
         state[pos][visited] = distance;
   }

   return state[pos][visited];
}

int main()
{
   vector<vector<int>> cities = {
{ 0, 13, 11, 17, 2, 8, 16, 2 , 3 , 1},
{ 10, 0, 35, 25, 3, 10, 15, 20, 6 , 3 },
{ 15, 35, 6, 30, 4, 11, 5, 1, 23, 23},
{ 20, 25, 30, 0, 15,35, 6, 32 ,23, 23},
{ 5, 10, 15, 20, 0, 17, 34, 1 ,23, 23 },
{ 10, 7, 25, 25, 8, 0, 15, 20, 12, 2},
{ 15, 35, 2, 30, 4, 19, 0, 15, 23, 23},
{ 20, 25, 30, 4, 15,35, 6, 0 ,43, 2},
{ 10, 1, 35, 25, 3, 20, 15, 20, 0 , 3 },
{ 15, 35, 5, 30, 4, 14, 5, 1, 23, 0} };

   vector<vector<int>> state(cities.size());
   for(auto& neighbors : state)
      neighbors = vector<int>((1 << cities.size()) - 1, INT_MAX);

   cout << "minimum: " << tsp(cities, 0, 1, state) << endl;

   return 0;
}
```

Output:
Minimum : 38

Multi stage

```
 #include<iostream>

#include<limits.h>
#include<vector>
using namespace std;
int main()
{
   int stages,n;
   cout<<"\tEnter no of stages:=";
   cin>>stages;
   cout<<"\nEnter no of vertex:=";
   cin>>n;
   int wt[n+1][n+1];
   cout<<"\nEnter weight matrix for the graph of "<<stages<<endl;
   for(int i=1;i<=n;i++)
```

```
    {
        for(int j=1;j<=n;j++)
        {
                    cin>>wt[i][j];
        }
    }
    int cost[n],d[n],path[stages];
    cost[n]=0;
    for(int i=n-1;i>=1;i--)
    {
        int min=INT_MAX;
        for(int k=i+1;k<=n;k++)
        {
            if(wt[i][k]!=0 && wt[i][k]+cost[k]<min)
            {
                min=wt[i][k]+cost[k];
                d[i]=k;
            }
        }
        cost[i]=min;
    }
    cout<<endl<<endl;
    cout<<"\nPath from starting vertex to ending vertex:===\n";
    path[1]=1,path[stages]=n;
    cout<<path[1]<<"-->";
    int total=0;
    for(int i=2;i<stages;i++)
    {
        path[i]=d[path[i-1]];
        total+=path[i];
        cout<<path[i]<<"-->";
    }
    cout<<path[stages];
    cout<<"\nTotal="<<total;
    return 0;
}
```

Input & output:

```
            Enter no of stages:=5

Enter no of vertex:=12

Enter weight matrix for the graph of 5


0  9  7  3  2  0  0  0  0  0  0  0
0  0  0  0  0  4  0  1  0  0  0  0
0  0  0  0  0  2  7  0  0  0  0  0
0  0  0  0  0  0  0  11  0  0  0  0
0  0  0  0  0  0  11  8  0  0  0  0
0  0  0  0  0  0  0  0  6  5  0  0
0  0  0  0  0  0  0  0  4  3  0  0
0  0  0  0  0  0  0  0  0  5  6  0
0  0  0  0  0  0  0  0  0  0  0  4
0  0  0  0  0  0  0  0  0  0  0  2
0  0  0  0  0  0  0  0  0  0  0  5
0  0  0  0  0  0  0  0  0  0  0  0



Path from starting vertex to ending vertex:===
1-->3-->6-->10-->12
Total=19
Exit code: 0 (normal program termination)
```