ASSIGNMENT NO 07

Name : Vishal Sule

Batch : D3

Roll no : 234

PRN No- 0120190064

Problem Statement :

Design & Implement Travelling salespersons Problem using Dynamic Programming. Also calculate the Time complexity for this algorithm.

CODE :

#include<bits/stdc++.h>

using namespace std;

using namespace std::chrono;

int tsp(int graph[][10], int s,int n)

{

vector<int> vertex;

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

if (i != s)

vertex.push\_back(i); //storing all vertex except the starting vertex

int minpath = INT\_MAX; //ensure minmium path is taken

do

{

int currentweight = 0;

int k = s;

for (int i = 0; i < vertex.size(); i++)

{

currentweight=currentweight+graph[k][vertex[i]]; //compute the current path

k = vertex[i];

}

currentweight=currentweight+graph[k][s];

minpath = min(minpath, currentweight);

} while(next\_permutation(vertex.begin(), vertex.end()));

return minpath;

}

int main()

{

int city[10][10],n;

cout<<"\nThe number of city salesperson has to visit:";

cin>>n;

cout<<"\nEnter the cost matrix:"<<endl;

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

{

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

cin>>city[i][j];

}

int s = 0; //starting from first node

steady\_clock::time\_point t1 = steady\_clock::now();

cout <<"\n The minimum cost to travel all cities is: "<<tsp(city, s,n) << endl;

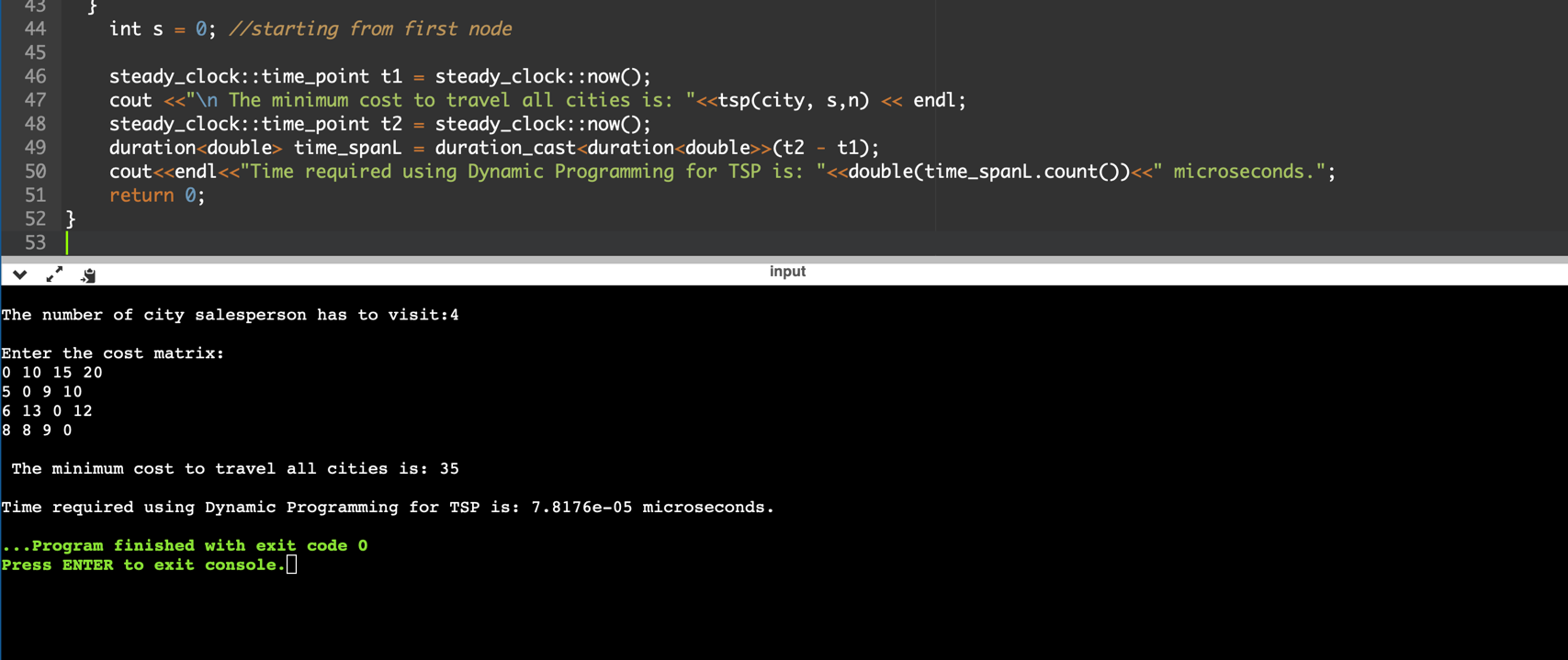
steady\_clock::time\_point t2 = steady\_clock::now();

duration<double> time\_spanL = duration\_cast<duration<double>>(t2 - t1);

cout<<endl<<"Time required using Dynamic Programming for TSP is: "<<double(time\_spanL.count())<<" microseconds.";

return 0;

}

OUTPUT :