**Batch: A1 Roll No.: 1911004**

**Experiment No 6**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **Title:** Implementation of The Travelling Salesman Problem using Dynamic Programming Approach |

**Objective:** To learn the Dynamic Programming Approach of solving the problems for different types of problems

**CO to be achieved:**

|  |  |
| --- | --- |
| Sr. No | Objective |
| CO 1 | Analyze the asymptotic running time and space complexity of algorithms. |
| CO 2 | Describe various algorithm design strategies to solve different problems and analyze Complexity. |
| CO 3 | Develop string matching techniques |
| CO 4 | Describe the classes P, NP, and NP-Complete |

**Books/ Journals/ Websites referred:**

1. **Ellis horowitz, Sarataj Sahni, S.Rajsekaran,” Fundamentals of computer algorithm”, University Press**
2. **T.H.Cormen ,C.E.Leiserson,R.L.Rivest and C.Stein,” Introduction to algortihtms”,2nd Edition ,MIT press/McGraw Hill,2001**
3. **http://lcm.csa.iisc.ernet.in/dsa/node184.htm**
4. **http://lcm.csa.iisc.ernet.in/dsa/node183.html**
5. **http://students.ceid.upatras.gr/~papagel/project/prim.htm**
6. **http://www.cse.ust.hk/~dekai/271/notes/L07/L07.pdf**

**Pre Lab/ Prior Concepts:**

Data structures, Concepts of algorithm analysis

**Historical Profile:**

Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point.

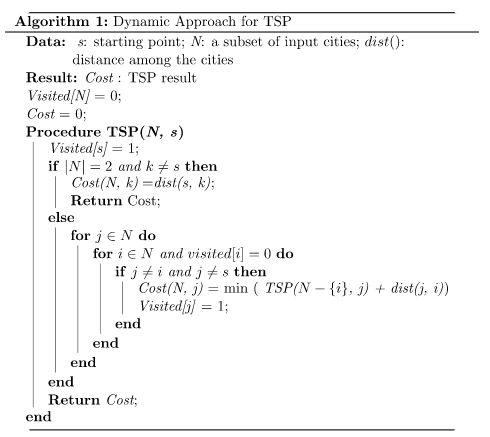
In the traveling salesman Problem, a salesman must visits n cities. We can say that salesman wishes to make a tour or Hamiltonian cycle, visiting each city exactly once and finishing at the city he starts from. There is a non-negative cost c (i, j) to travel from the city i to city j. The goal is to find a tour of minimum cost. We assume that every two cities are connected. Such problems are called Traveling-salesman problem (TSP).

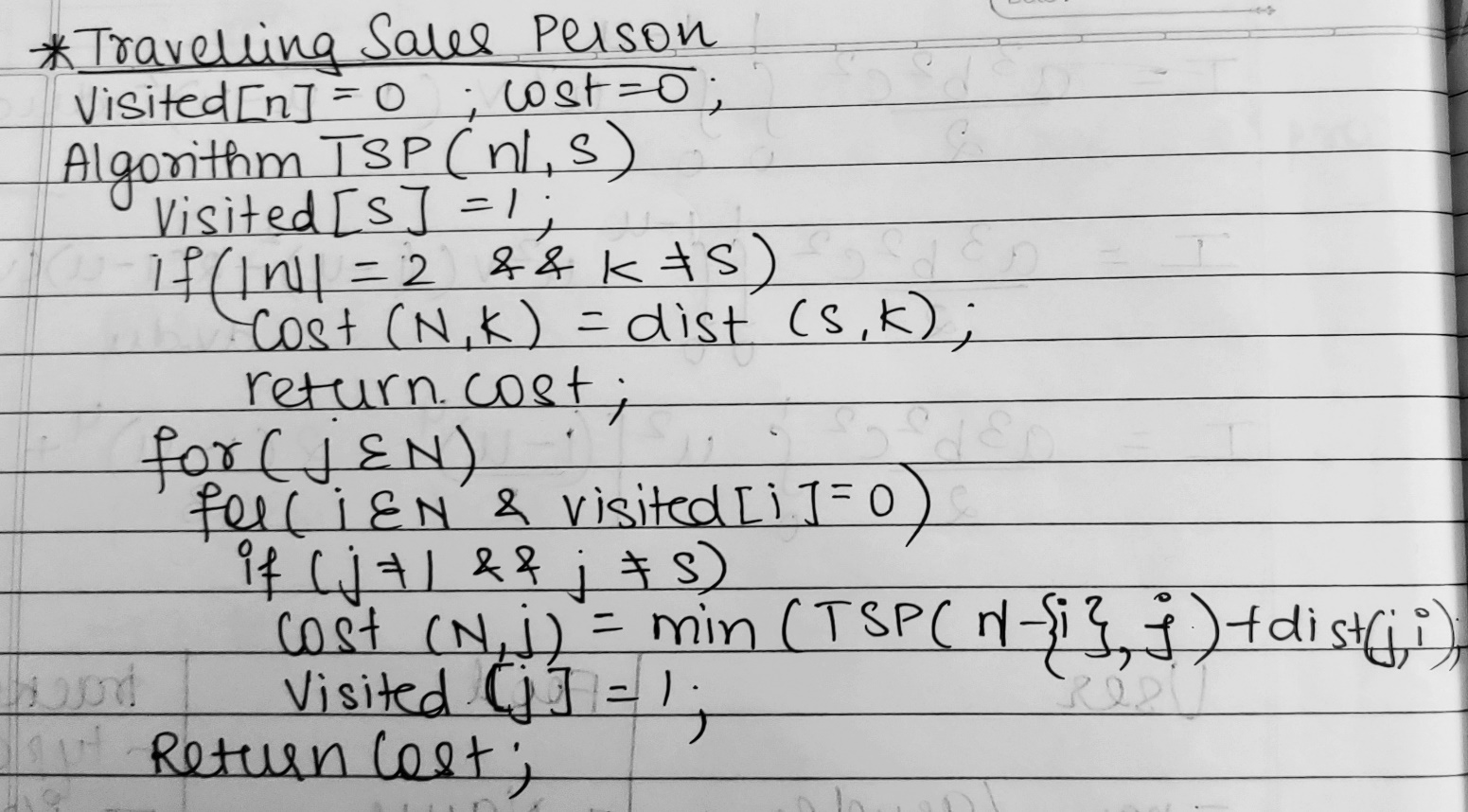
We can model the cities as a complete graph of n vertices, where each vertex represents a city.

**New Concepts to be learned:**

Application of algorithmic design strategy to any problem, Dynamic programming Approach of problem-solving Vs other methods of problem solving, optimality of the solution, The Travelling Salesman Problem, and their applications

**The Travelling Salesman Problem Algorithm:**





**Code:**

import java.util.\*;

class TSP{

public static int tCost,path[],graph[][];public static boolean v[];

public static void main(String[] args) {

System.out.println("Travelling Salesman");

Scanner ob = new Scanner(System.in);

System.out.print("Enter no of Cities ");

int i,j,k,n= ob.nextInt();tCost = Integer.MAX\_VALUE;

graph = new int[n][n];path = new int[n+1];

v = new boolean[n];

System.out.print("Enter the Cost Matrix\n");

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

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

System.out.print("Cost of M["+(i+1)+"]["+(j+1)+"] = ");

graph[i][j]=ob.nextInt();

}

}

path[0]=1;path[n]=1;

if (n<=1)

tCost = 0;

else{

v[0] = true;

tCost = tsp(graph,v,0,n,1,0,tCost);

minCost(0,n,0);

}

System.out.println("Min Tour costs "+tCost);

System.out.print("Min Path = ");

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

System.out.print(path[j]+"->");

System.out.print(path[n]+"\n\n");

}

static int leastPath(int C,int n){

int i,\_c=Integer.MAX\_VALUE,min=Integer.MAX\_VALUE,cMin=0;

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

if(graph[C][i]!=0&&(v[i]==false)){

if(graph[C][i]+graph[i][C]<min){

min=graph[i][0]+graph[C][i];

cMin=graph[C][i];

\_c=i;

}

}

}

return \_c;

}

static void minCost(int C,int n,int j){

int i,\_C;

v[C]=true;

path[j]=C+1;

\_C=leastPath(C,n);

if(\_C==Integer.MAX\_VALUE){

\_C=0;return;

}

minCost(\_C,n,j+1);

}

static int tsp(int graph[][], boolean v[],int cPos, int n,int count, int cost, int tCost){

if (count == n && graph[cPos][0] > 0) {

tCost = Math.min(tCost, cost + graph[cPos][0]);

return tCost;

}

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

if (v[i] == false && graph[cPos][i] > 0) {

v[i] = true;

tCost = tsp(graph, v, i, n, count+1,cost+graph[cPos][i], tCost);

v[i] = false;

}

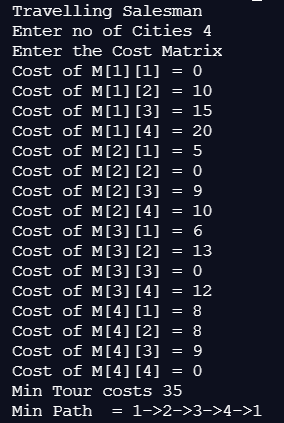
}

return tCost;

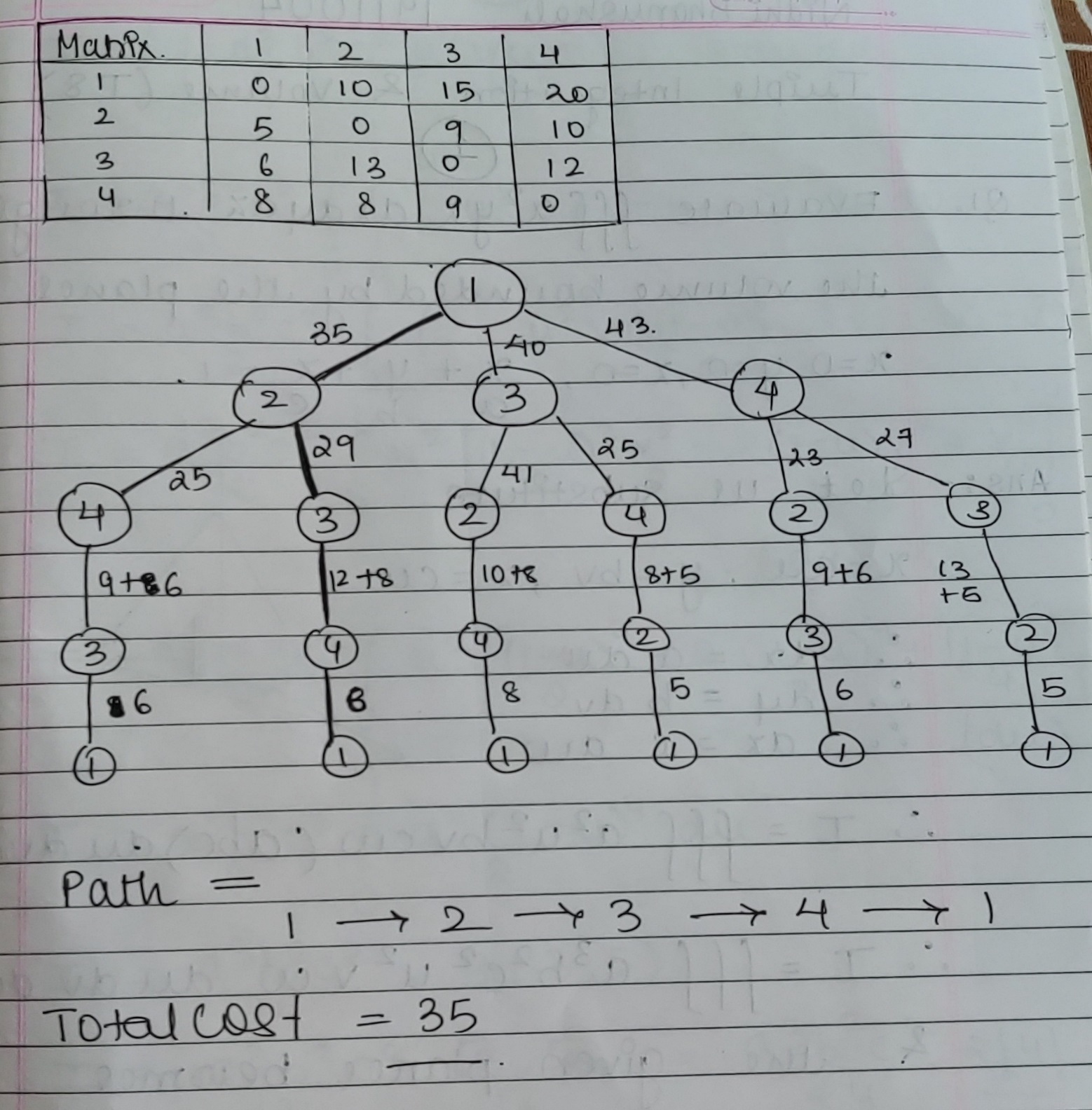
}

}

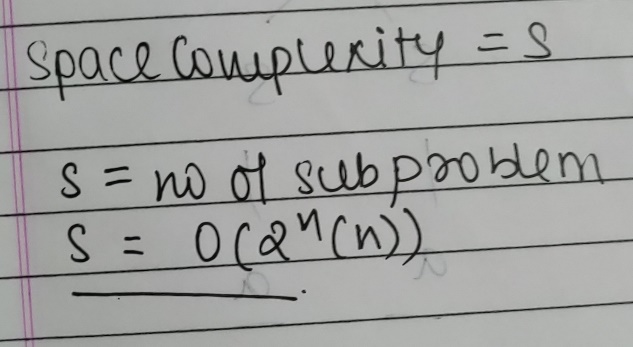
**Output:**

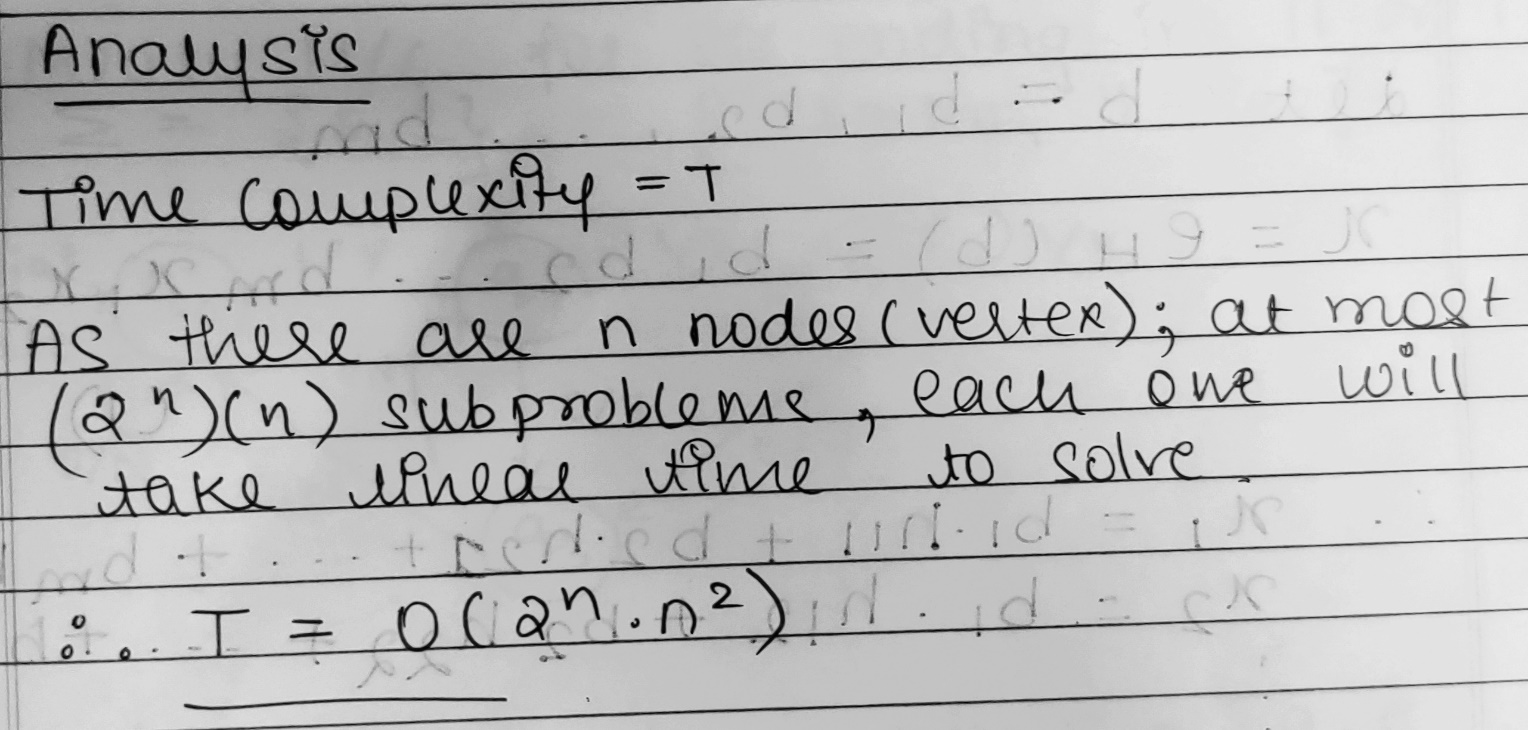


**Example: The Travelling Salesman Problem** (One solved example)



**Analysis of Travelling Salesman Problem algorithm:**





**CONCLUSION:**

We successfully understood & implemented Travelling Sales Problem with Dynamic Programming approach; thus obtained correct minimum cost & path implementing it in java.