Subject: DAA LAB	Software: Ubuntu
	Hardware: Core i5
Branch: CSE / Semester: 4th	Page No. 35 / Prog No. 09

STATEMENT: Write a Program to solve Travelling Salas Person Problem using Dynamic Programming.

ALGORITHM & CODE:

```
#include <stdio.h>
#include <limits.h>
#include <stdbool.h>
#define MAX_CITIES 20
int n; // Number of cities
int cost [MAX_CITIES][MAX_CITIES]; // cost matrix
int memo [MAX_CITIES][1 << MAX_CITIES];
                        // memoization table.
int main (int a, int b)
 {return (a<b)? a:b;
 }
int tsp (int current, int visited)
 {
  //Base case: all cities visited, return to start
  if (visited == (1<<n)-1)
  {return cost [current][0];
  }
  if (memo [current][visited] != -1)
  {return memo [current) [visited];
  }
  int min_cost = INT_MAX;
 // Try visiting each unvisited city
 for (int next = 0; next < n; next++)
 {if (!(visited & (1 << next)))
```

INPUT GIVEN:

OUTPUT OBTAINED:

REMARKS:

GRADE:

Signature of Faculty
Date:

Signature of Student: Rudranarayan Sahoo
Date: 17/04/2025

Subject : DAA LAB	Software : Ubuntu
	Hardware : Core i5
Branch : CSE / Semester : 4th	Page No. 36 / Prog No. 09

STATEMENT

THM & CODE :

```
{int new_cost = cost[current][next] + tsp(next,
        visited | (1<<next));
    min_cost = min(min_cost, new_cost);
  }
 }
 return memo[current][visited] = min_cost;
}

void print_path()
{
 int path[MAX_CITIES];
 int visited = 1; // Start with city 0 visited
 path[0] = 0;
 for (int i = 1; i < n; i++)
 {
  int current = path[i-1];
  int next_city = -1;
  int min_next_cost = INT_MAX;
  for (int next = 0; next < n; next++)
  { if(!(visited & (1<<next)))
   {
    int potential_cost = memo[next][visited | (1<<next)]
+ cost[current][next];
    if (potential_cost < min_next_cost)
```

INPUT GIVEN

OUTPUT OBTAINED

REMARKS

GRADE :	Signature of Faculty Date :	Signature of Student Rudranarayan Sahoo Date : 17/04/2025

	Subject : DAA LAB		Software : Ubuntu
			Hardware : Core i5
	Branch : CSE	Semester : 4th	Page No. 37 / Prog No. 09

M STATEMENT

THM & CODE :

```
        {
            min_next_cost = Potential_cost;
            next_city = next;
        }
    }
}
path[i] = next_city;
visited |= (1 << next_city);
}
Printf("Optimal path:");
for(int i=0; i<n; i++)
{
printf("%d ->", path[i]);
}
Printf("0\n");
}
int main()
{
Printf("Enter number of cities (max %d):",
                    MAX_CITIES);
Scanf("%d", &n);
Printf("Enter cost matrix (%d for infinity):\n",
                    INT_MAX);
```

NPUT GIVEN

UTPUT OBTAINED

EMARKS

GRADE :	Signature of Faculty Date :	Signature of Student Rudranarayan Sahoo Date : 17/04/2025

Subject : DAA LAB	Software : Ubuntu
	Hardware : Core i5
Branch : CSE / Semester : 4th	Page No. 38 / Prog No. 09

PROBLEM STATEMENT

ALGORITHM & CODE :

```
for (int i = 0; i<n; i++)
{ for (int j =0; j<n; j++)
  {
    scanf("%d", &cost[i][j]);
    if (i == j) cost[i][j] = 0;
  }
}
// initialize memoization table
for (int i = 0; i <n; i++)
 { for (int j = 0; j< (1<<n) ; j++)
   { memo[i][j] = -1;
   }
}

int min_cost = tsp (0, 1); // start at a city 0 with the
city 0 visited

Printf ("\n minimum TSP cost : %d \n", min_cost);

Print_Path ();
return 0;
}
```

INPUT GIVEN	
OUTPUT OBTAINED	
REMARKS	
GRADE :	Signature of Faculty Date :
	Signature of Student Rudranarayan Sahoo Date : 17/04/2025

Subject : DAA LAB	Software : Ubuntu
	Hardware : Corce i5
Branch : CSE / Semester : 4th	Page No. 39 / Prog No. 09

PROBLEM STATEMENT

ALGORITHM & CODE :

OUTPUT

```
Enter number of cities (max 20): 4
Enter cost matrix (2147483647 for infinity):
0 16 11 6
8 0 13 6
4 7 0 9
5 12 20 0

Minimum TSP cost : 23
Optimal path : 0 -> 3 -> 2 -> 1 -> 0
```

INPUT GIVEN

OUTPUT OBTAINED

REMARKS