**SARDAR PATEL INSTITUTE OF TECHNOLOGY**

Name: Darshit Bhagtani

2021700006

CSE DS D1

Exp. 9: Travelling Salesman Problem

**AIM:** To implement Approximation algorithms (Travelling Salesman Problem)

**THEORY:**

The Traveling Salesman Problem (TSP) is a classic optimization problem in computer science, which involves finding the shortest possible route that visits a set of cities and returns to the starting city. In more detail, given a set of n cities and the distances between each pair of cities, the goal of the TSP is to find a Hamiltonian cycle (a cycle that visits each city exactly once) with minimum total length. The TSP is an NP-hard problem, which means that there is no known algorithm that can solve all instances of the problem in polynomial time, and it is an active research area in both computer science and mathematics.

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

#define N 100

// Global variables

int n=5;

int matrix[5][5] = { { 0, 10, 18, 40, 20 },

{ 10, 0, 35, 15, 12 },

{ 18, 35, 0, 25, 25 },

{ 40, 15, 25, 0, 30 },

{ 20, 13, 25, 30, 0 } };

int visited[N];

// Function prototypes

int nearest\_neighbor(int start);

void tsp();

int main() {

// Solve TSP using nearest neighbor algorithm

tsp();

return 0;

}

// Find the nearest unvisited city to a given city

int nearest\_neighbor(int city) {

int min\_distance = INT\_MAX;

int nearest\_city = -1;

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

if (!visited[i] && matrix[city][i] < min\_distance) {

min\_distance = matrix[city][i];

nearest\_city = i;

}

}

return nearest\_city;

}

// Solve TSP using nearest neighbor algorithm

void tsp() {

int start = 0;

visited[start] = 1;

printf("Path: %d ", start);

int total\_distance = 0;

for (int i = 0; i < n - 1; i++) {

int next\_city = nearest\_neighbor(start);

visited[next\_city] = 1;

printf("%d ", next\_city);

total\_distance += matrix[start][next\_city];

start = next\_city;

}

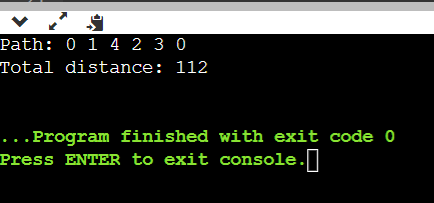
printf("%d\n", 0);

total\_distance += matrix[start][0];

printf("Total distance: %d\n", total\_distance);

}

**OUTPUT:**

****

**CONCLUSION:**

In this experiment, I implemented the travelling salesman problem.