Aman Kalla
RA1911003010640

Artificial Intelligence Lab

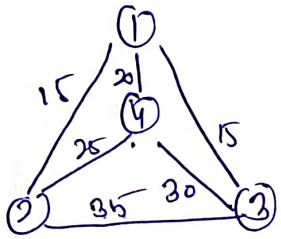
Lab-2

Aim:- Developing Agent programs for Real World Problems – Travelling Salesman Problem (TSP)

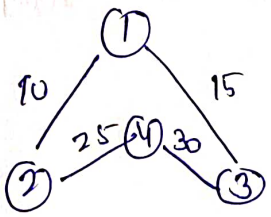
Problem Formulation

For a given complete graph with n vertices and weight function defined on the edges, the objective is to constructs a tour i.e., a circuit that passes through each vertex only one of maximum total weight.

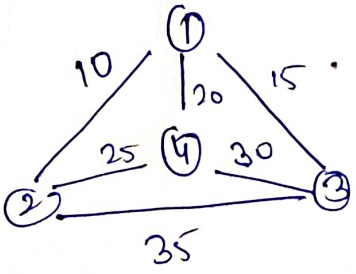
initial State:-



final state:-

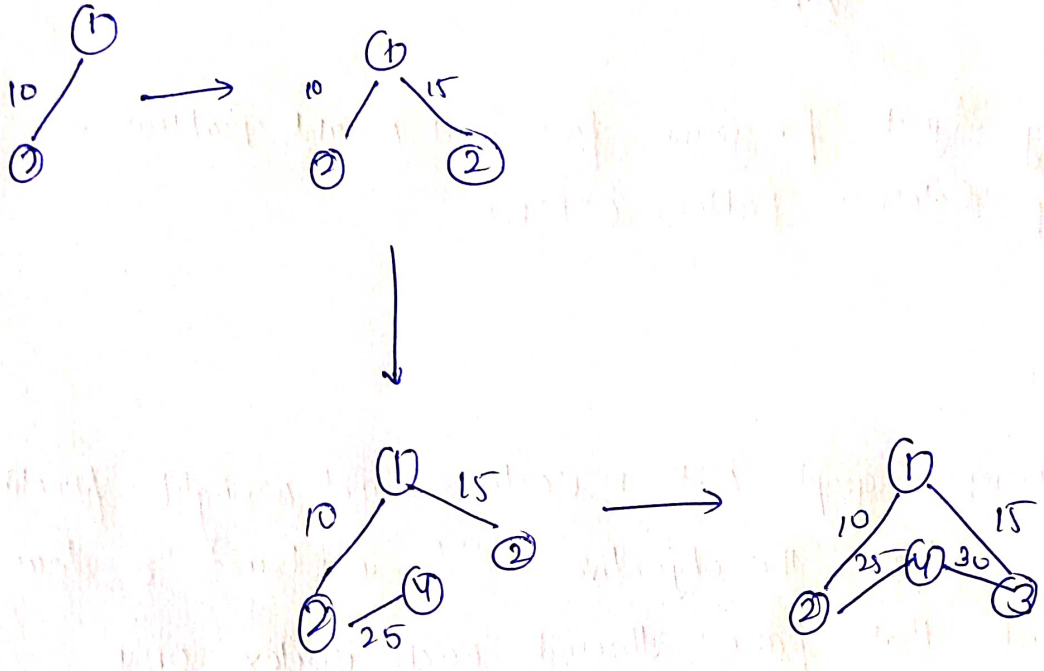


Problem Solving.



We start at vertex 1 and find the min. cost path with 1 as starting point, i as ending point and all vertices appearing exactly once.

For path 1→2, the minimum cost would be through direct path



It tries for various other permutations as well and 1-2-4-3-1 permutation works perfect as it provides minimum cost.

ARTIFICIAL INTELLIGENCE

LAB 2

Travelling Salesman Problem

Aman Kalla

RA1911003010640

Algorithm:-

Step 1: Consider city 1 as the starting and ending point.

Step 2: Generate all (n-1)! Permutations of cities.

Step 3: Calculate cost of every permutation and keep track of minimum cost permutation.

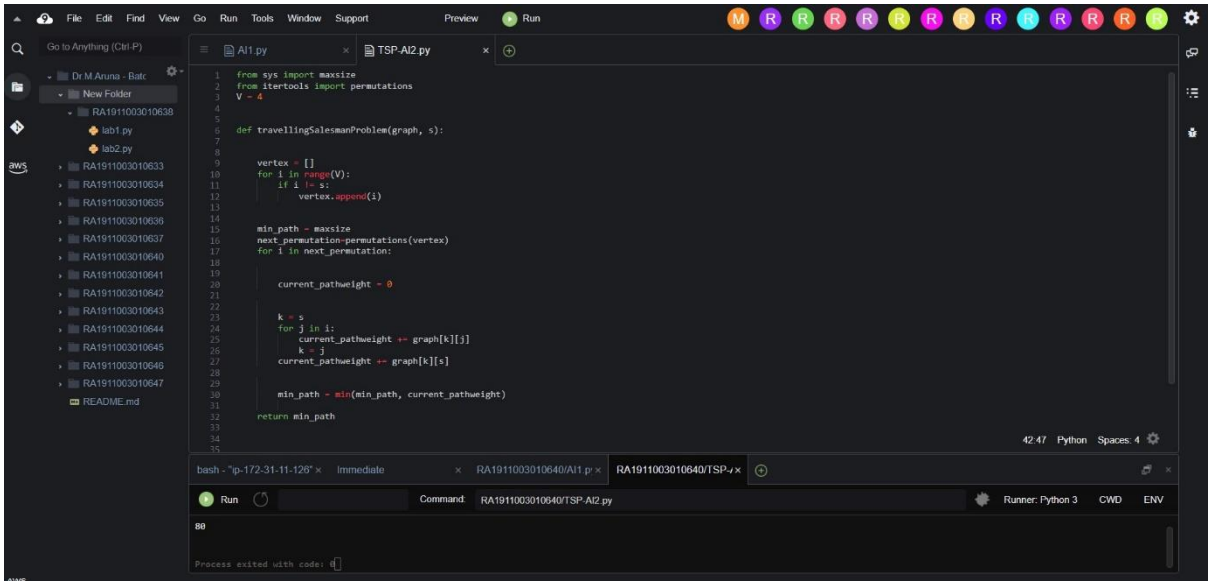
Step 4: Return the permutation with minimum cost.

Code:-

```
from sys import maxsize
from itertools import permutations
V = 4


def travellingSalesmanProblem(graph, s):


    vertex = []
    for i in range(V):
        if i != s:
            vertex.append(i)
```

```
    min_path = maxsize
    next_permutation=permutations(vertex)
    for i in next_permutation:


        current_pathweight = 0


        k = s
        for j in i:
            current_pathweight += graph[k][j]
            k = j
        current_pathweight += graph[k][s]


        min_path = min(min_path, current_pathweight)

    return min_path



if __name__ == "__main__":


    graph = [[0, 10, 15, 20], [10, 0, 35, 25],
            [15, 35, 0, 30], [20, 25, 30, 0]]
    s = 0
    print(travellingSalesmanProblem(graph, s))
```

Output:-



Result:-

Hence, the implementation of Travelling Salesman Problem is done successfully.