101. Travelling salesman problem

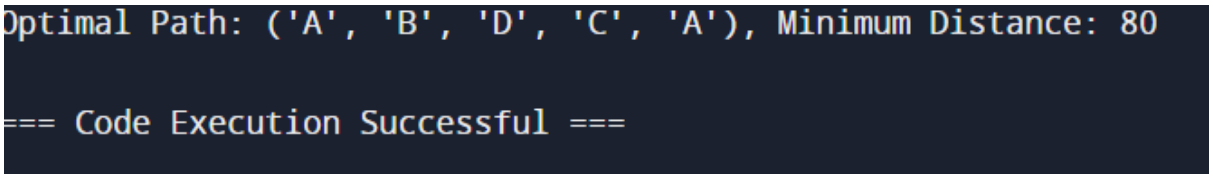
AIM: to find the shortest distance.

Program:

```
from itertools import permutations

def tsp_brute_force(graph, start):
    all_cities = set(graph.keys())
    all_cities.remove(start)
    min_distance = float('inf')
    optimal_path = None

    for path in permutations(all_cities):
        path = (start,) + path + (start,)
        distance = sum(graph[path[i]][path[i + 1]] for i in range(len(path) - 1))

        if distance < min_distance:
            min_distance = distance
            optimal_path = path

    return optimal_path, min_distance

# Example Usage
graph = {
    'A': {'B': 10, 'C': 15, 'D': 20},
    'B': {'A': 10, 'C': 35, 'D': 25},
    'C': {'A': 15, 'B': 35, 'D': 30},
```

```
    'D': {'A': 20, 'B': 25, 'C': 30}
}
start_city = 'A'
optimal_path, min_distance = tsp_brute_force(graph, start_city)
print(f"Optimal Path: {optimal_path}, Minimum Distance: {min_distance}")
```

Output:



Time complexity:

O(n^2*2^n)