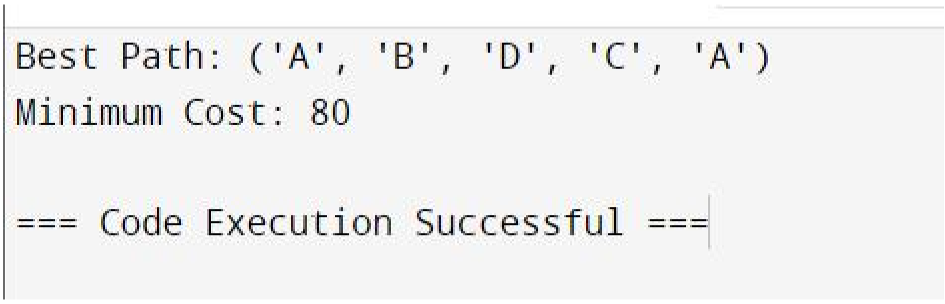
100. Travelling salesman problem

PROGRAM:

```
import itertools

def tsp_brute_force(graph, start):
    all_nodes = set(graph.keys())
    all_nodes.remove(start)
    min_cost = float('inf')
    best_path = None

    for path in itertools.permutations(all_nodes):
        path = (start,) + path + (start,)
        cost = sum(graph[path[i]][path[i + 1]] for i in range(len(path) - 1))

        if cost < min_cost:
            min_cost = cost
            best_path = path

    return best_path, min_cost

# Example graph with distances between cities
graph = {
    'A': {'A': 0, 'B': 10, 'C': 15, 'D': 20},
    'B': {'A': 10, 'B': 0, 'C': 35, 'D': 25},
    'C': {'A': 15, 'B': 35, 'C': 0, 'D': 30},
    'D': {'A': 20, 'B': 25, 'C': 30, 'D': 0}
}

start_node = 'A'
best_path, min_cost = tsp_brute_force(graph, start_node)

print(f"Best Path: {best_path}")
print(f"Minimum Cost: {min_cost}")
```

OUTPUT:



```
Best Path: ('A', 'B', 'D', 'C', 'A')
Minimum Cost: 80

=== Code Execution Successful ===
```

TIME COMPLEXITY:O((N-1))!