176. Assume you are solving the Traveling Salesperson Problem for 4 cities (A, B, C, D) with known distances between each pair of cities. Now, you need to add a fifth city (E) to the problem.

Test Cases

1. Symmetric Distances

- **Description: All distances are symmetric (distance from A to B is the same as B to A).**

Distances:

A-B: 10, A-C: 15, A-D: 20, A-E: 25 B-C: 35, B-D: 25, B-E: 30 C-D: 30, C-E: 20 D-E: 15

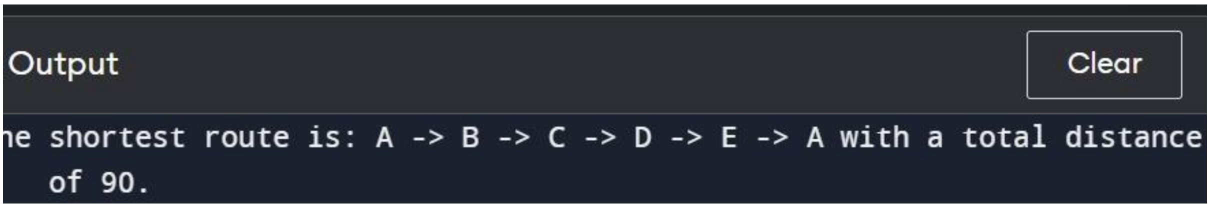
Expected Output: The shortest route and its total distance. For example, A -> B -> D -> E -> C -> A might be the shortest route depending on the given distances.

Program: from itertools import permutations

```
distances = {
    ('A', 'B'): 10, ('A', 'C'): 15, ('A', 'D'): 20, ('A', 'E'): 25,
    ('B', 'C'): 35, ('B', 'D'): 25, ('B', 'E'): 30,
    ('C', 'D'): 30, ('C', 'E'): 20,
    ('D', 'E'): 15
}

cities = ['A', 'B', 'C', 'D', 'E']

min_distance = float('inf')
best_route = None

for route in permutations(cities):
    route_distance = sum(distances.get((route[i], route[i + 1]), float('inf')) for i in range(len(route) - 1))
    if route_distance < min_distance:
        min_distance = route_distance
        best_route = route
```

print(f"The shortest route is: {' -> '.join(best_route)} -> {best_route[0]} with a total distance of {min_distance}.")

Output:



```
e shortest route is: A -> B -> C -> D -> E -> A with a total distance
   of 90.
```

Timecomplexity: Time Complexity: O(n^2 * 2^n)