4.4 MINIMUM PATH DISTANCE USING MATRIX FORM

Question:

Write a c program to find the minimum path distance by using matrix form.

AIM

To implement a C program that calculates the minimum path distance in a matrix using dynamic programming.

ALGORITHM

1. Input the number of rows m and columns n.
2. Input the cost matrix cost[m][n].
3. Initialize a DP matrix dp[m][n] where dp[i][j] stores the minimum cost to reach cell (i,j).
4. Set dp[0][0] = cost[0][0].
5. Fill the first row and first column using cumulative sums.
6. For each cell (i,j), compute dp[i][j] = cost[i][j] + min(dp[i-1][j], dp[i][j-1], dp[i-1][j-1]).
7. The final answer is dp[m-1][n-1].

PROGRAM

```
from itertools import permutations

def tsp_4_cities(matrix):
    n = len(matrix)
    min_cost = float('inf')
    best_path = []

    for perm in permutations(range(1, n)):
        path = [0] + list(perm) + [0]
        cost = sum(matrix[path[i]][path[i+1]] for i in range(n))
        if cost < min_cost:
            min_cost = cost
            best_path = path

    return min_cost, best_path


print("Enter 4x4 distance matrix row by row:")
matrix = []
for _ in range(4):
    row = list(map(int, input().split()))
    matrix.append(row)

cost, path = tsp_4_cities(matrix)
city_labels = ['A', 'B', 'C', 'D']
route = ' -> '.join(city_labels[i] for i in path)
print(f"Minimum path cost: {cost}")
print(f"Optimal route: {route}")
```

Input:

Enter the cost matrix:

0 10 15 20
10 0 35 25
15 35 0 30
20 20 30 0

Output:

```
Enter 4x4 distance matrix row by row:
0 10 15 20
10 0 35 25
15 35 0 30
20 25 30 0
Minimum path cost: 80
Optimal route: A -> B -> D -> C -> A
>>>
```

RESULT:

Thus the program is successfully executed, and the output is verified.

PERFORMANCE ANALYSIS:

- Time Complexity: O(m × n)
- Space Complexity: O(m × n)