## A. Case Study 1: Dynamic Programming - Supply Chain Optimization

6. Write a Python program to compute the minimum supply chain cost using dynamic programming

```
def min_supply_chain_cost(cost_matrix):
    n = len(cost_matrix)
    dp = [[float('inf')] * n for _ in range(n)]

    # Initialize the cost to reach each warehouse directly from the source
    for i in range(n):
        for j in range(n):
            if i == j:
                dp[i][j] = 0
            else:
                dp[i][j] = cost_matrix[i][j]

    # Apply the recurrence relation to fill the dp table
    for k in range(n):
        for i in range(n):
            for j in range(n):
                if dp[i][j] > dp[i][k] + dp[k][j]:
                    dp[i][j] = dp[i][k] + dp[k][j]
```

```
    return dp

# Example cost matrix (example values)
cost_matrix = [
    [0, 10, 15, 20],
    [10, 0, 35, 25],
    [15, 35, 0, 30],
    [20, 25, 30, 0]
]

min_cost = min_supply_chain_cost(cost_matrix)
print("Minimum Supply Chain Costs:")
for row in min_cost:
    print(row)
```