H. Case Study 8: Dynamic Programming - Inventory Management

6. Write a Python program to compute the optimal reorder points using dynamic programming.

```
import numpy as np


def inventory_management_dp(horizon, demand, holding_cost,
                            shortage_cost, ordering_cost, max_inventory):
    # Initialize the DP table with infinity
    dp = np.full((horizon + 1, max_inventory + 1), float('inf'))
    dp[horizon, :] = 0  # Base case: cost is 0 at the end of the horizon


    # Fill the DP table backwards from the horizon
    for t in range(horizon - 1, -1, -1):
        for s in range(max_inventory + 1):
            for q in range(max_inventory - s + 1):
                new_inventory = s + q - demand[t]
                if new_inventory < 0:
                    cost = ordering_cost * (q > 0) + abs(new_inventory) * shortage_cost + dp[t + 1, 0]
                else:
                    cost = ordering_cost * (q > 0) + new_inventory * holding_cost + dp[t + 1,
new_inventory]
                dp[t, s] = min(dp[t, s], cost)


    return dp
```

```
# Input parameters
horizon = 5
demand = [2, 3, 4, 1, 2]
holding_cost = 1
shortage_cost = 5
ordering_cost = 10
max_inventory = 10

# Calculate the DP table
dp_table = inventory_management_dp(horizon, demand, holding_cost,
                                   shortage_cost, ordering_cost, max_inventory)

# Print the optimal cost table
print("Optimal Cost Table:")
print(dp_table)
```