13. USECASE :

FINDING THE WINNING STRATEGY IN A CARD GAME IN PYTHON

Plan: we can solve problem using Dynamic programming by calculating the optimal score for every possible scenario, taking into account the best choices for both players.

steps :

1. Define the game : Represent the pile of cards as a list of integers
2. Recursive strategy : A function will recursively determine the best score a player can acheive.
3. Dynamic programming : store intermediate results to avoid recalculating them.
4. Base cases : when only one card left, current player takes it,

PROGRAM :

```
def find_optimal_strategy(cards):
    n = len(cards)
# Create a memoization table to store subproblem results
    dp = [[0]*n for _ in range(n)]
# Fill the table for subproblem of increasing sizes
    for length in range(1, n+1):
        for i in range(n-length+1):
            j = i+length-1
# If only one card left, the player takes it
            if i == j:
                dp[i][j] = cards[i]
            else: # Choose the best of two choices:
```

```
        take_left = cards[i] - dp[i+1][j]
        take_right = cards[j] - dp[i][j-1]
    dp[i][j] = max(take_left, take_right)
# dp[0][n-1] will have optimal score
return (dp[0][n-1] + sum(cards)) // 2
# First player's max possible score
# Example case
cards = [3,9,1,2]
print("First player's optimal score:", find_optimal_
                                    strategy(cards))
```

Explanation:

- Dynamic programming Table (dp): Each cell dp[i][j] represents the difference in score blw the first player & opponent
- Two choices: For each move
  1. Pick leftmost card [i], leaving the opponent to play optimally on remaining cards
  2. Pick right card [j], leaving opponent the rest of cards.
- Recursive relation: The value of each problem determined by max the score difference blw current player & opponent

eg walkthrough: consider array of cards: [3,9,1,2].

1. First player (you) can choose blw:
   - Taking leftmost card (3) leaving [9,1,2].
   - Taking right most card [2], leaving [3,9,1].
2. The opponent will taken their turn

This program computes the best possible outcome for first player

First player optimal score : 5

First player, if playing optimally, can guarantee a score of 5 regardless of how the opponent plays.

Optimizing strategy: Using Dynamic programming we ensure that the soln is computed efficiently avoiding redundant calculations. This approach ensures both player play optimally & the first player gets the highest score possible given opponent's best move.

RESULT: Thus the use case Finding the winning strategy in a card game in python executed & verified sucessfully.

VELTECH	
EX No.	
PERFORMANCE (5)	
RESULT AND ANALYSIS (5)	
VIVA VOCE (5)	
RECORD (5)	
TOTAL (20)	
SIGN WITH DATE	