Use case: 1

Finding the winning strategy in a card game in python

problem description:-

Image a card game where each players receives a hand of cards with values. the objective is to find the best way to maximize the score for a players assuming the players take turns drawing cards. each player can either pick the first or last card from the remaining pile.

plan:-

we can slove this problem using dynamic programming by calculating the optimal score for every possible scenario, taking into account the best choice for both players.

steps:-

1. Define the base:- Represent the pile of cards as a list of integers.
2. Recursive strategy:- A function will recursively determine the best score a players can achieve.
3. dynamic programming:- Store intermediate result to avoid recalling them.
4. Base cases:- when only one card is left, the current players takes it

program:-

```
def find_optimal_strategy(cards):
    n = len(cards)
# create a memorization table to store sub problem results.
    dp = [[0] * n for _ in range(n)]
#fill the table for subproblems of increasing size
for length in range(1, n+1):
    for i in range(n - length + 1):
```

```
        J = i + length - 1
# Base case: only one card is left, the player takes it if
  i == j.
    dp[i][j] = cards[i]
else:
# choose the best of two choices:
# Take the left card and the opponent plays optimally
on the remaining (i+1, j)
maximum possible score
# Example case
cards = [3, 9, 1, 2]
print("First player optimal score:", find_optimal
strategy(cards))
```

Explanation:—

→ Dynamic programming table (dp):— Each cell dp[i][j] represents the difference in score between the first player and the opponent if the game is played between cards from index i to index j.

→ Two choices:— For each move, the player can either
1. pick the left most card cards[i] leaving the opponent to play optimally on the remaining cards

→ Two choice :— For each move, the player can either
1. pick the left most cards. [i], leaving the opponent to play optimally on the remaining cards
2. pick the right cards[j], leaving the opponent the rest of the cards

→ Recursive Relation:— The value of each sub problem is determined by maximizing the score difference between the current player and the opponent.

Example walk through:—
Consider the array of cards: [3, 9, 1, 2]
1. First player (you) can choose between
   2) Taking the left most card (3)

=) leaving the cards (9,1,2)

=) Taking the Right most card (2)

leaving the cards (3,9,1)

2. The opponent will then take their turn, playing optimally. to minimize the first players score

This program computes the best possible outcomes for the first player.

First player's optimal score: 5

optimizing strategy:-

By using. Dynamic programming, we ensure that the solution is computed efficiently, avoiding redundant calculations this approach ensures. both players . play optimally and the first player gets the highest score possible gives the opponents best move.

VEL TECH	
EX No.	13
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	5
TOTAL (20)	
SIGN WITH DATE	

completed