13.	15/10/25	~~TJ game~~ Finding the winning strategy in a cardgame. ✓	15	R. Dony

Use Case: Finding the winning strategy In A Card Game

Aim:- To implement a Python program that simulates a card game between two players and determine the winning strategy using drawn cards

Algorithm:-

1. Start the program.
2. Create a deck of cards.
3. Shuffle the deck randomly.
4. Each player draws a fixed number of cards.
5. Define the strategy
6. Compare the chosen cards of both players.
   - If player 1's card > player 2's card → player 1 scores.
   - Else if player 2's card > player 1's card → player 2 scores.
   - Else → Draw
7. Repeat for all rounds.
8. The player with the highest score wins the game
9. Display input, output, and final result.
10. End the program.

Program:-

```
import random
suits = ["Hearts", "Diamonds", "Clubs", "Spades"]
values = list (range [1,14])
deck = [(value, suit) for suit in suits for value in values]
random. shuffle (deck)
player 1 - hand = deck [:5]
player 2 - hand = deck [5:10]
print ("player 1 Hand: ", player 1 - hand)
print ("player 2 Hand: ", player 2 - hand)
def play - highest - card (hand):
```

Sample input:

Player 1 Hand: [(13, 'Hearts'), (2, 'Clubs'), (10, 'Spades'), (5, 'Diamonds'),
(7, clubs")]

Player 2 Hand: [(9, "Hearts"), (12, 'Diamonds"), (3, "Clubs"),
(11, "Spades"), (6, "Hearts")]

Sample Output:

Round 1: player 1 → (13, 'Hearts'), player 2 → (12, 'Diamonds')

Winner: player 1

Round 2: player 2 → (10, 'Spades'), player 2 → (11, 'Spades')

Winner: player 2

Round 3: player 1 → (7, 'Clubs'), player 2 → (9, 'Hearts')

Winner: Player 2

Round 4: Player 1 → (5, 'Diamonds'), player 2 → (6, 'Hearts')

Winner: Player 2

Round 5: player 1 → (2, 'Clubs'), player 2 → (3, 'Clubs')

Winner: player 2

```
    highest = max(hand, key = lambda x: x[1])
    hand.remove(highest)
    return highest
p1_score, p2_score = 0, 0
print("\n--- Game Rounds ---")
for i in range(5):
    p1_card = play_highest_card(player1_hand)
    p2_card = play_highest_card(player2_hand)
    print(f"Round {i+1}: player 1 -> {p1_card},
                player 2 -> {p2_card}"),
    if p1_card[0] > p2_card[0]:
        print("Winner: player 1")
        p1_score += 1
    elif p2_card[0] > p1_card[0]:
        print("Winner: player 2")
        p2_score += 1
    else:
        print("Result: Draw")
print("\n--- Final Result ---")
print("player 1 score:", p1_score)
print("player 2 score:", p2_score)
if p1_score > p2_score:
    print("player 1 wins the game with winning strategy!")
elif p2_score > p1_score:
    print("player 2 wins the game with winning strategy!")
else:
    print("The game is a Draw!")
```

VEL TECH - CSE	VC
EX NO.	
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	
TOTAL (20)	15
SIGNATURE WITH DATE	

R. Jony 15/10/25

R. Jony 15/10/25

Result:-

Thus, the finding winning strategy in a card game is executed and verified successfully.