Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai

Artificial Intelligence and Data Science Department

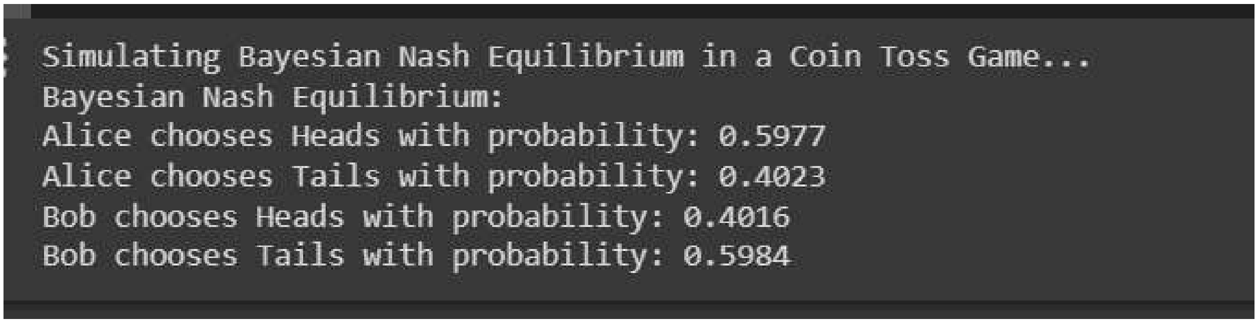
Subject/Odd Sem 2023-23/Experiment 7

Program :

1.

```python
import random

# Function to simulate a coin toss
def coin_toss():
    return random.choice(["Heads", "Tails"])

# Function to calculate the utility of a player given their choice and the coin toss result
def calculate_utility(player_choice, coin_result):
    if player_choice == coin_result:
        return 1
    else:
        return 0

# Main function to simulate the Bayesian Nash equilibrium
def simulate_bayesian_nash_equilibrium():
    num_simulations = 10000
    player_Alice_heads_count = 0
    player_Alice_tails_count = 0
    player_Bob_heads_count = 0
    player_Bob_tails_count = 0

    for _ in range(num_simulations):
        coin_result = coin_toss()

        # Alice's strategy (Heads or Tails)
        p_Alice_Heads = 0.6  # Adjust this probability as desired
        p_Alice_Tails = 1 - p_Alice_Heads

        # Bob's strategy (Heads or Tails)
        p_Bob_Heads = 0.4  # Adjust this probability as desired
        p_Bob_Tails = 1 - p_Bob_Heads

        # Choose based on probabilities
        alice_choice = "Heads" if random.random() < p_Alice_Heads else "Tails"
        bob_choice = "Heads" if random.random() < p_Bob_Heads else "Tails"

        # Calculate utilities
        alice_utility = calculate_utility(alice_choice, coin_result)
        bob_utility = calculate_utility(bob_choice, coin_result)

        # Update counts
        if alice_choice == "Heads":
            player_Alice_heads_count += 1
        else:
            player_Alice_tails_count += 1

        if bob_choice == "Heads":
            player_Bob_heads_count += 1
        else:
            player_Bob_tails_count += 1

    # Calculate probabilities from counts
    p_Alice_Heads_eq = player_Alice_heads_count / num_simulations
    p_Alice_Tails_eq = player_Alice_tails_count / num_simulations
    p_Bob_Heads_eq = player_Bob_heads_count / num_simulations
    p_Bob_Tails_eq = player_Bob_tails_count / num_simulations

    print("Bayesian Nash Equilibrium:")
    print(f"Alice chooses Heads with probability: {p_Alice_Heads_eq}")
    print(f"Alice chooses Tails with probability: {p_Alice_Tails_eq}")
    print(f"Bob chooses Heads with probability: {p_Bob_Heads_eq}")
    print(f"Bob chooses Tails with probability: {p_Bob_Tails_eq}")

if __name__ == "__main__":
    print("Simulating Bayesian Nash Equilibrium in a Coin Toss Game...")
    simulate_bayesian_nash_equilibrium()
```

Output



```
Simulating Bayesian Nash Equilibrium in a Coin Toss Game...
Bayesian Nash Equilibrium:
Alice chooses Heads with probability: 0.5977
Alice chooses Tails with probability: 0.4023
Bob chooses Heads with probability: 0.4016
Bob chooses Tails with probability: 0.5984
```