Phase 1 Group: Sapiens Members: Shreya, Supriya, Zeenat

Objective: An agent play trick taking games such as Hearts, Spades, and Bridge.

Approach:

Considering roughly two data structures, viz. Self (containing element the agent has) and ActiveList (containing elements that are on table for current round, which can contain from 0 to 3 elements for 4 player game.), the agent will play as follows:

Game moves:

1. If the agent is first player, i.e., the ActiveList has 0 elements and the agent has an ace of any suite, it will randomly select from that suite and play the ace - PlayfirstHighest()
2. If the agent doesn't have ace, it will play any random card from lowest to second highest card that it has - PlayRandom().
3. If there is 1 element in ActiveList, if agent don't have the largest card of played suite or larger than the elements in ActiveList then he plays the least card of the played suite.
4. If the agent has the larger card than the played card, then it compares its larger card with PlayedList and if no other element is larger than the agent's card, then it plays that card.
5. If the agent doesn't have a card of played suite, it plays the least card of random suite.- Playdifferentsuite()
6. PlayedList: agent maintains a list containing cards that has been already played. Every time a card is being played, the agent inserts the element in the playedList.
7. Record_OutOfCards(): If a player P plays a card of different suite S' from the ones which was on table S, then record player name and the suite he doesn't have (S, P).

Functions:

1. **PlayfirstHighest(self)**
   The method will take a list of two character string that the agent has and will search for the Ace if it is the first one to play. If it has ace of multiple suites, it will select any suite randomly and play the ace. The method will return a single two character string representing the card the agent has chosen to play.

2. **PlayRandomCard(self)**
   The method will take a list of two character string that the agent has and will search for the Ace if it is the first one to play. If the agent doesn't have ace, it will play any random card from lowest to second highest card that it has. The method will return a single two character string representing the card the agent has chosen to play.

3. **PlayLeastCard(self, trick)**
   The method will take a list of two character string that the agent has and a list of two character strings of the cards that have been played so far this trick and identifies the suite. if agent don't have the largest card of played suite or larger than the elements in ActiveList then he plays the least card of the played suite. The method will return a single two character string representing the card the agent has chosen to play.
4. **PlayGreaterCard(self, trick)**
   The method will take a list of two character string that the agent has and a list of two character strings of the cards that have been played so far this trick and identifies the suite.

If the agent has the larger card than the played card, then it compares its larger card with trick and if no other element is larger than the agent's card, then it plays that card. The method will return a single two character string representing the card the agent has chosen to play.

5. **Playdifferentsuite(self, trick)**
   The method will take a list of two character string that the agent has and a list of two character strings of the cards that have been played so far this trick and identifies the suite. If the agent doesn't have a card of played suite, it plays the least card of random suite. The method will return a single two character string representing the card the agent has chosen to play.

6. **RecordOutOfSuite(lead, trick)**
   The method will take the lead name who played a different suite other than trick and a list of two character strings of the cards that have been played so far this trick and identifies that which suite this lead does not have. It will return a two value tuple with player name and two character string of cards that the player does not have.