CMSC 671 Project
Fall 2019

In this project you will be programming an agent to play trick taking games such as Hearts, Spades, and Bridge.

Simple Trick Taking Game

This game is played with 4 players and a standard 52 card deck. There are four *suits*, Spades ♠, Hearts ♥, Diamonds ♦, and Clubs ♣. Each suit has 13 cards with *ranks* 2, 3, 4, 5, 6, 7, 8, 9, 10, J (Jack), Q (Queen), K (King), and A (Ace), with 2 being the lowest rank and A being the highest.

To begin, deal 13 cards to each player. The player to the left of the dealer will place any card from their hand face up on the table. This player is the *lead*.

Play continues clockwise with each player selecting a card from their hand to play. If a player can play a card that matches the suit of the lead, they must. Otherwise, they may play any card from their hand that they wish. This is called *discarding*. Play stops when all four players have played a card, completing the *trick.*

The winner of the trick is the player who played the highest ranked card that matches the lead suit. They collect the trick and keep the cards face down in front of them. This player then becomes the new lead and selects a card to begin the next trick.

Once all 13 tricks have been played. Players count one point for each trick they won to get their score. These scores carry over to the next hand and the players keep playing hands until one player scores 100 points. At this point the player with the most points wins.

Agent API

In this part of the project you will code an agent to play this game. We will provide the game environment for your agents to compete against other teams' agents. Cards will be represented by two character strings with 'S', 'H', 'D', and 'C' representing the suits and '2', '3', '4', '5', '6', '7', '8', '9', 'T', 'J', 'Q', 'K', 'A' representing the ranks. Thus A♠ would be represented as the string 'AS' and 5♦ would be represented as the string '5D'.

__init__(self)
An __init__ method that takes no arguments to instantiate the class.

get_name(self)
A getter method that takes no arguments and returns a string of your agent's name.

get_hand(self)
A getter method that takes no arguments and returns a list of two character strings representing all the cards in your agent's hand.

new_hand(self, names)
A method that takes a single argument, a list of strings of all the players' names in playing order, and returns nothing. This method is also responsible for clearing any data necessary for your agent to start a new hand.

add_cards_to_hand(self, cards)
A method that takes a single argument, a list of two character strings representing cards to be added to the agent's hand. This is used for dealing and passing and so should be able to take a list of any length. This method should return nothing.

play_card(self, lead, trick)
This will likely be the bulk of your code. This method takes two arguments, the name of the player who is leading the trick, and a list of two character strings of the cards that have been played so far this trick. This method should return a single two character string representing the card your agent has chosen to play. From this method your agent should also track which cards are left in its hand.

collect_trick(self, lead, winner, trick)
Takes 3 arguments. A string of the name of the player who led the trick, a string of the name of the player who won the trick, and a list of two character strings representing the cards in the trick in the order they were played, starting with lead.

score(self)
Takes no arguments and returns an integer of the number of points your agent scored on this hand.

Phase 1 Turn In

You will turn in a single .pdf file named **<project team name>.pdf**. This file is a single page write up (no code) explaining what strategy your agent uses to play the game. You should include the name of each method in the API and a short description of what the agent does and returns.