CSE101-Term Project (2017 SPRING)

Simplified Uno

Rules of Simplified Uno

A Simplified Uno deck comes with 72 cards: Yellow, Red, Green and Blue are the four colors of cards and there are two copies of each number 1 through 9, for each color.

There are two players in the game.

The deck is shuffled and then each player is dealt 7 cards. The first player gets to discard any card they like into the discard pile. From that point on, the players alternate turns.

From this point on, a playable card is simply one that is either the same color or number as the top card (last card discarded) of the discard stack.

If the player has a "playable" card in his/her hand, then they are asked which card to play. If they choose one of the playable cards, it is removed from their hand and added to the top of the discard pile. If they choose an unplayable card and have a playable card, ask again until they choose a playable card. Otherwise, if they cannot play a card, they draw one card from the deck, and their turn is over.

The game continues until either player has discarded all of his/her cards or the initial deck is emptied. If a player gets rid of all their cards, they win at that point in time. If the initial deck is exhausted, the game finishes in a tie.

Listing of Methods for UnoCard

```
# Pre-conditions: c must be in between 0 and 3 inclusive,
#                 n must be in between 1 and 9, inclusive.
def __init__(self, c, n):

# Returns a String representation of a UnoCard. For example
# the green card with 7 on it should be represented as
# "Green 7".
def __str__(self):

# Returns true iff other can be played on this card or
# vice versa.
def canPlay(self, other):
```

Listing of Methods for CollectionOfUnoCards

```
# Creates an empty list of cards.
def  __init__(self):

# Adds c to the end of this collection.
def addCard(self, c):

# Clears out this collection and fills it with all of the cards
# in the deck.
def makeDeck(self):

# Shuffles the cards in this collection. (This can be achieved
# by repeatedly picking two cards at random and swapping them.
# A couple hundred times should suffice.
def shuffle(self):

# Returns a String representation of the collection, as shown
# in the example output.
def __str__(self):

# Returns the number of cards in the collection.
def getNumCards(self):
# Returns the top card in the collection without removing it.
def getTopCard(self):

# Returns true if there exists at least one card in this
# collection that can be played on the card c.
def canPlay(self, c):

# Returns the card in location index of the collection.
def getCard(self, index):
```

Instance Variables for Uno

```
deck (CollectionOfUnoCards)
lastPlayedCard (just store as a single UnoCard)
hand1 (CollectionOfUnoCards)
hand2 (CollectionOfUnoCards)
```

Listing of Methods for Uno

You'll create a real Uno object that contains the four collections listed above. My particular implementation only has the four following methods. It is quite possible you'll choose to have different methods than me. But, you are free to follow my design as well.

```
# Creates a new game instance as follows: Fills the deck and
# starts with two empty hands, and then deals each hand 7 cards
# from the deck, alternating hands.
def __init__(self):

# Plays the game once it has been set up.
def playGame(self):

# Executes playing one turn for the player designated by the
# input parameter (which must be either 1 or 2).
def playTurn(self, player):

# Prints the result after the game has finished.
def printResult(self):
```

Main Method in Uno class

```
def main():
     my_game = Uno()
     my_game.playGame()
```