Not FreeCell Design

The not Free cell game has been developed using object-oriented programming in Python language. The classes involved are *Card*, *Deck*, *NotFreecell*, *Player*. *Card* and *Deck* classes are generalized class whereas the *NotFreeCell* and the *Player* classes are developed specifically for playing the game. A card in the class is defined as [Colour, Suit, Value]. The classes are described in detail in the next section.

Class: CARD

The Card class is a generalized class which works in conjunction with the Deck class. It holds the card attributes like color, face, suit and also few useful to methods that are used by Deck to enhance the visibility card definition in its stack. The attributes can hold ‘n’ values and mapping to dynamically define a deck

The card attributes are stored as follows in the __init__ method:

```
# defines the faces of the cards
self.cardFace = ['Jack', 'Queen', 'King']
# defines the suits of the cards
self.ColorMap = {'Black': ['Clubs', 'Spade'], 'Red': ['Heart', 'Diamond']}
self.CardSuits = list(self.ColorMap.values())
self.CardColor = list(self.ColorMap.keys())
```

cardFace holds the values of the Faces that will be used by the game developer while invoking the Deck class. Similarly, ColorMap represents a dictionary where the color and the suit mapping will be set by the game developer to be used in the game. CardSuits stores the list of available suits from the dictionary mapping whereas the CardColor stores the list of available colors.

The methods available are:

- *parseSuitIndex:* This method returns the color and the name of the suit using the indexes in the list of existing suits. For example, in this case, the list of available suits are Clubs, Spade, Heart, and Diamond. Using the method, list [0] would return ‘Clubs’ and ‘Black.’ This is called by enhanceCard for card manipulation

- *enhanceCard:* This method modifies the card definition via input array. A card with definition [1, 2] will be amended to ['Black', 'Club', 2] with the help of this function. This method provides more flexibility to the developer to perform card calculation based on suit, colour, and value, without the need to parse the number every time. It significantly improves the game performance

- *getCardFace:* This mutator is provided to access the available cards in the game

- *__str__:* This describes the instance of the card class with values: cardFace and CardSuits. It returns the following:

  ```
  '\nThe cards generated are [total = {0}]: \n{1}\n\nThe shuffled Deck generated is: \n{2}\n'\
      .format(len(self.currentSuitedList), self.currentSuitedList, self.getCurrentSuitedListShuffled())
  ```

- *__init__:* Initiates the card attributes as mentioned above

Class: DECK

The Deck class requires three inputs to instantiate its object. They are *value_start, value_end* and *number_of_suits*. The value_start and value_end define the range of cards that will be produced for each suit. The number of the suit is defined by number_of_suits. For example, Deck (1,14,2) generates 28 cards in total (14 for each suit)

The card attributes are stored as follows in the __init__ method:

```
# Object of the Card to use them in this instance of Deck.
self.theCard = Card()
# Initial value of the suit of a card
self.startIndex = value_start
# Final value of the suit of a card
self.endIndex = value_end
# Number of available suits in the card
self.variety = number_of_suits
# Available series for the card
self.currentList = []
# Contains the list of cards available to prepare the deck
self.currentSuitedList = []
# Refers the Deck with shuffled cards. May be accessed from different class
self.currentSuitedListShuffled = []
# Puts a drawn card from the deck using method: drawTheCard
self.drawnCard = None
```

The methods available are:

- *addToDeck*: This dynamically generates the face values of the card in accordance to the specified range. It then passes the series to the method moduloDeck to dynamically generate the cards for the 'n' suits specified. It stores the generated list in currentSuitedList. For testing, it contains a commented line which prints the built face value series.

- *moduloDeck*: For a given series and range of suits it creates the list by appending individual series value and suit value. It returns list of lists

- *shuffle*: This is used to shuffle the currentSuitedList and store in currentSuitedListShuffled

- *drawTheCard*: This method draws a single card from the shuffled deck. While the card is drawn it is modified on the fly using the card method enhanceCard

- *getCurrentSuitedListShuffled*: This mutator is provided to access the available shuffled cards in the game.

- *__str__:* This describes the instance of the Deck class with values: length of the Deck, raw list of cards and enhanced shuffled the list of cards. It returns the following:

```
\nThe cards generated are [total = {0}]: \n{1}\n\nThe shuffled Deck generated is: \n{2}\n'\
   .format(len(self.currentSuitedList), self.currentSuitedList, self.getCurrentSuitedListShuffled())
```

Class: NOTFREECELL

This is specifically developed to design the game rules and manage player interface. It implements the Deck class methods to create a deck, shuffle it and split the cards into the game component free cell. It implements Panda dataframe to display the result.

The game attributes are stored as follows in the __init__ method:

```
def __init__(self, name):
    # Set the dimension of the data frame to display the game in a terminal without word wraps
    desired_width = 320
    pd.set_option('display.width', desired_width)

    # Initialize the object of Deck for further use
    self.theDeck = Deck(1, 13, 4)
    # Add the cards to the decks
    self.theDeck.addToDeck()
    # Shuffle the deck
    self.theDeck.shuffle()

    # Initializing the game components viz: CASCADES, FOUNDATIONS and OPENCELLS
    self.listOfCascades = [[], [], [], [], [], [], [], []]
    self.listOfFoundations = [[], [], [], []]
    self.listOfCells = [[], [], [], []]

    # Instance variables
    self.peekedValue = None
    self.color = None
    self.suits = None
    self.values = None
    self.item = None

    # Game response when a move is made by the user
    self.flag = True
    self.msg = None

    # Visualisation
    self.df_cascade = None
    self.df_Foundation = None
    self.df_OpenCell = None

    #Player details
    self.name = name
```

The __init__ method takes the player name to create a game instance for the user. As soon as the instance is created, the __init__ creates an object of the class Deck. The object is then used to create a shuffled list of cards in the Deck which the class NotFreecell uses for further manipulation.

The methods available are:

- *fillCascades*: It uses the method of the Deck class called drawTheCard to fill all the eight cascades. First, the four cascades are filled with seven cards in each, and then the remaining cascades are filled with six cards in each.

- *cardParser*: It returns the value of a card for input Face. For example, for a list input of ['Black', 'Club', 'Jack'], the method will return three values Black, Club and 13. This is very much useful to compare 2 cards while moving from one component to another.

- *moveCard*: The method is used to move card from one component to another component. It takes four input fromArea, fromIndex, toArea, toIndex where the list of areas can be C, OC, F and index starts from 0. To move a card first cascade to the last opencell the method should be invoked as: moveCard(C, 0, OC, 1). It returns two values, an error response and an error flag.

  The algorithm for move is as below:
  a) Check the source is valid or not. If valid, pick the last element from the source
  b) Check the destination:
     a. If the destination is C [Cascade]: **the selected item is of different colour than the Last card in the cascade. It also satisfies that the value of the selected card is lesser than the last card in the cascade. Return true if successful**
     b. If the destination is F [foundation]: **the selected item is of different colour than the last card in the cascade. It also satisfies that the value of the selected card is lesser than the last card in the cascade. Return true if successful**
     c. If the destination is OC [Open Cell]: Push only if the cell is empty. Returns true if successful
  c) If the response is true, delete the item from the source
  d) Note: the source is defined by fromArea, fromIndex and the destination is defined by toArea, toIndex

- *showcaseCascade*: This returns the concatenation of all the cascades in a panda dataframe. The columns with variable length shows NAN which is replaced by the string '[EMPTY]'

- *autoMove:* Many freecell game has a feature that if there are any 'ACE' in the last position of any cascade, it is automatically moved to the empty Foundation. This method does the same thing when invoked. First, it checks if there are any ACE in the last position of any cascade. If true it holds the index of those cascades in to a list. Now it invokes move method where the arguments are (C,i,F,j) where i is the index of the cascade in previous step and J is the next empty foundation.

- *showcaseCell*: This returns the concatenation of all the opencells in a panda dataframe. The columns with NAN value is replaced by the string '[EMPTY]'

- *showcaseFoundation*: This returns the concatenation of all the foundations in a panda dataframe. The columns with NAN value is replaced by the string '[EMPTY]'

- *showBoard*: This is used for visualisation. It builds the entire board to display while the game is running. It uses showcaseCascade, showcaseCell, showcaseFoundation and string concatenation to print the output

- *__str__:* This describes the instance of the NotFreecell class with values: name of the player, welcome message, and formatting that is required before the game starts It returns the following:

```
def __str__(self):
    """ prints the name of the player """

            line_0 = ' \n\n'
             line_1 = ('.' * 145 + '\n')
            line_2 = ('.' + '\t' * 13 + 'Welcome to Not Free Cell game' + '\t' * 16 + '.'+'\n')
            line_3 = ('.' * 145 + '\n\n')
            line_4 = ('Player Name: ' + self.name + '\n')
    return line_0+line_1+line_2+line_3+line_4
```

Class: PLAYER

This class is used to run the Not Free Cell game for a specific player. It has one method which controls the game flow:

> To restart a game the user needs to key in 'R', to quit the user needs to key in 'Q', and to continue 'C' is required.

```
# Game is played until its finished
while not (len(thePlayerOne.listOfFoundations[0]) == 13 and
           len(thePlayerOne.listOfFoundations[1]) == 13 and
           len(thePlayerOne.listOfFoundations[2]) == 13 and
           len(thePlayerOne.listOfFoundations[3]) == 13):
```

The method mainly consists of a while loop which executes until all the foundations are full. The user can break the loop by typing 'Q' when prompted.

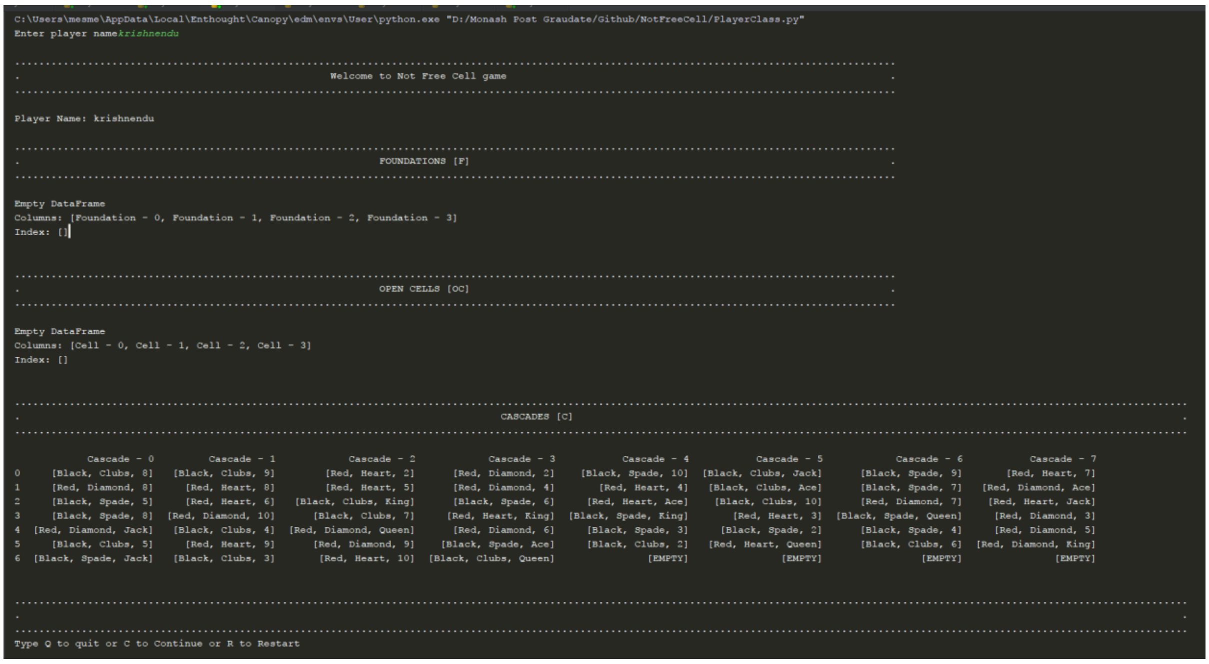
```
# Quit the game
if modifier == 'Q':
    text = 'Thanks for playing'
    break
# Restart the game
elif modifier == 'R':
    self.interface()
    break
# Continue the game
elif modifier == 'C':
    text = 'Thanks for playing'
    pass
# Invalid Input
else:
    text = 'Invalid choice, Quitting the game'
    break
```

To avoid moving the Ace automatically to the Foundation on the first move a counter check is implemented. The Automove method executes from the second move onwards.

```
if error is True:
    thePlayerOne.autoMove()
    thePlayerOne.showBoard()
```

The interface:

The following picture show the interface of the game:



```
C:\Users\mesme\AppData\Local\Enthought\Canopy\edm\envs\User\python.exe "D:/Monash Post Graudate/Github/NotFreeCell/PlayerClass.py"
Enter player namekrishnendu

.......................................................................................................................
.                                            Welcome to Not Free Cell game                                            .
.......................................................................................................................

Player Name: krishnendu

.......................................................................................................................
.                                                  FOUNDATIONS [F]                                                    .
.......................................................................................................................

Empty DataFrame
Columns: [Foundation - 0, Foundation - 1, Foundation - 2, Foundation - 3]
Index: []


.......................................................................................................................
.                                                  OPEN CELLS [OC]                                                    .
.......................................................................................................................

Empty DataFrame
Columns: [Cell - 0, Cell - 1, Cell - 2, Cell - 3]
Index: []


..........................................................................................................................................................
.                                                                   CASCADES [C]                                                                         .
..........................................................................................................................................................

          Cascade - 0         Cascade - 1            Cascade - 2            Cascade - 3            Cascade - 4          Cascade - 5            Cascade - 6          Cascade - 7
0     [Black, Clubs, 8]   [Black, Clubs, 9]         [Red, Heart, 2]      [Red, Diamond, 2]    [Black, Spade, 10]  [Black, Clubs, Jack]      [Black, Spade, 9]      [Red, Heart, 7]
1     [Red, Diamond, 8]     [Red, Heart, 8]         [Red, Heart, 5]      [Red, Diamond, 4]       [Red, Heart, 4]   [Black, Clubs, Ace]      [Black, Spade, 7]  [Red, Diamond, Ace]
2     [Black, Spade, 5]     [Red, Heart, 6]    [Black, Clubs, King]      [Black, Spade, 6]     [Red, Heart, Ace]    [Black, Clubs, 10]      [Red, Diamond, 7]    [Red, Heart, Jack]
3     [Black, Spade, 8]  [Red, Diamond, 10]       [Black, Clubs, 7]     [Red, Heart, King]  [Black, Spade, King]       [Red, Heart, 3]  [Black, Spade, Queen]    [Red, Diamond, 3]
4  [Red, Diamond, Jack]   [Black, Clubs, 4]  [Red, Diamond, Queen]      [Red, Diamond, 6]     [Black, Spade, 3]     [Black, Spade, 2]      [Black, Spade, 4]    [Red, Diamond, 5]
5     [Black, Clubs, 5]     [Red, Heart, 9]       [Red, Diamond, 9]    [Black, Spade, Ace]     [Black, Clubs, 2]   [Red, Heart, Queen]      [Black, Clubs, 6]  [Red, Diamond, King]
6  [Black, Spade, Jack]   [Black, Clubs, 3]        [Red, Heart, 10]  [Black, Clubs, Queen]               [EMPTY]               [EMPTY]                [EMPTY]              [EMPTY]


..........................................................................................................................................................
.                                                                                                                                                        .
..........................................................................................................................................................
Type Q to quit or C to Continue or R to Restart
```

The interface has 5 areas:

a. Welcome Message
b. Foundation display
c. Open Cell Display
d. Cascade Display
e. User input

Since the display is implemented using panda Dataframe, an empty dataframe will put the following message.

Empty DataFrame Columns: [Foundation - 0, Foundation - 1, Foundation - 2, Foundation - 3] Index: []
Empty DataFrame Columns: [Cell - 0, Cell - 1, Cell - 2, Cell - 3] Index: []

Assumption:

a) The Deck is a general class and implements Card
b) The Card is a general class having the card attributes
c) Deck is able to generate ‘n’ number of cards dynamically based on the range provided along with ‘m’ number of suits having ‘z’ faces
d) Card class enhances the Deck class with proper attribute values like colour and suit names
e) For ‘m’ suits the colour mapping and the suit name needs to be defined by the developer in the card class
f) Face values are dynamically appended to the last of the series after 10th value card
g) For any Deck, Ace is always the first face value
h) Card and Deck class is accessible to the Developer
i) Ace is moved automictically to foundation if it’s the last card in cascade

Improvements:

a) Design can be significantly improved if the Deck class is allowed to take more than 3 parameters. This will allow the developer to name the faces and suit-colour mapping from the game itself, without modifying the card class
b) The face values can be mapped to specific positions to improve visualisation
c) Add pygame library to display the game in a graphical user interface
d) The current scope of the game excludes user score, number of moves, time taken. This can be amended to improve the game play experience

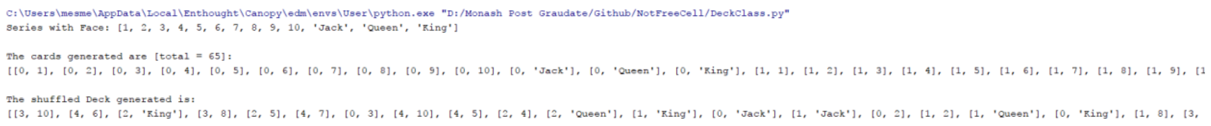
Testing:

<u>Dynamic card generation:</u>

For 5 faces, 6 suits and range of 13, the number of cards generated are by the Deck `(1, 13, 5)`:

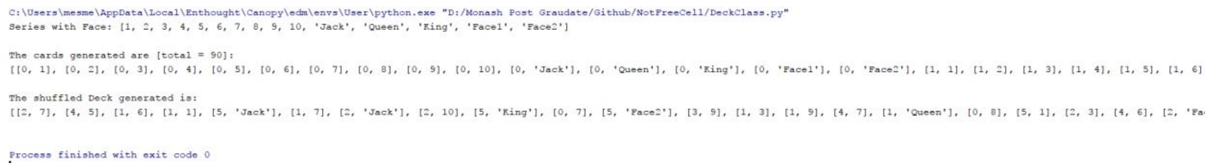
```
self.cardFace = ['Jack', 'Queen', 'King', 'Face1', 'Face2']
# defines the suits of the cards
self.ColorMap = {'Black': ['Clubs', 'Spade'], 'Red': ['Heart', 'Diamond'],
'Violet': ['Suits5', 'Suits6']}
```

For 5 Faces, 6 suits and range of 13, the number of cards generated by the Deck `(1, 1, 5)` is:



```
C:\Users\mesme\AppData\Local\Enthought\Canopy\edm\envs\User\python.exe "D:/Monash Post Graudate/Github/NotFreeCell/DeckClass.py"
Series with Face: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 'Jack', 'Queen', 'King']

The cards generated are [total = 65]:
[[0, 1], [0, 2], [0, 3], [0, 4], [0, 5], [0, 6], [0, 7], [0, 8], [0, 9], [0, 10], [0, 'Jack'], [0, 'Queen'], [0, 'King'], [1, 1], [1, 2], [1, 3], [1, 4], [1, 5], [1, 6], [1, 7], [1, 8], [1, 9], [1

The shuffled Deck generated is:
[[3, 10], [4, 6], [2, 'King'], [3, 8], [2, 5], [4, 7], [0, 3], [4, 10], [4, 5], [2, 4], [2, 'Queen'], [1, 'King'], [0, 'Jack'], [1, 'Jack'], [0, 2], [1, 2], [1, 'Queen'], [0, 'King'], [1, 8], [3,
```

Similarly, for same configuration, Deck (1, 15, 6) generates full combination of all the card attributes.



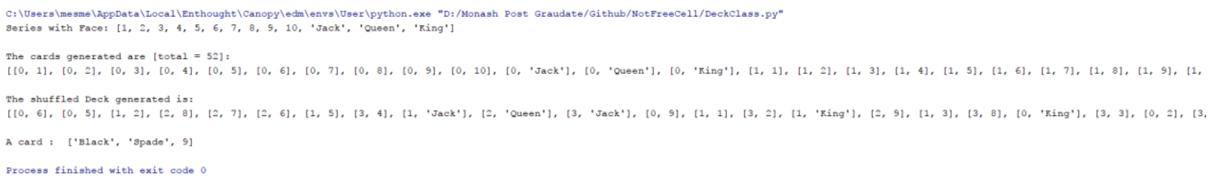
```
C:\Users\mesme\AppData\Local\Enthought\Canopy\edm\envs\User\python.exe "D:/Monash Post Graudate/Github/NotFreeCell/DeckClass.py"
Series with Face: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 'Jack', 'Queen', 'King', 'Face1', 'Face2']

The cards generated are [total = 90]:
[[0, 1], [0, 2], [0, 3], [0, 4], [0, 5], [0, 6], [0, 7], [0, 8], [0, 9], [0, 10], [0, 'Jack'], [0, 'Queen'], [0, 'King'], [0, 'Face1'], [0, 'Face2'], [1, 1], [1, 2], [1, 3], [1, 4], [1, 5], [1, 6]

The shuffled Deck generated is:
[[2, 7], [4, 5], [1, 6], [1, 1], [5, 'Jack'], [1, 7], [2, 'Jack'], [2, 10], [5, 'King'], [0, 7], [5, 'Face2'], [3, 9], [1, 3], [1, 9], [4, 7], [1, 'Queen'], [0, 8], [5, 1], [2, 3], [4, 6], [2, 'Fa


Process finished with exit code 0
```

The following setup is placed for FreeCell game:

self.cardFace = ['Jack', 'Queen', 'King']

```python
# defines the suits of the cards
self.ColorMap = {'Black': ['Clubs', 'Spade'], 'Red': ['Heart', 'Diamond']}
```

The number of cards generated by the Deck (1,13,4) is as follows:

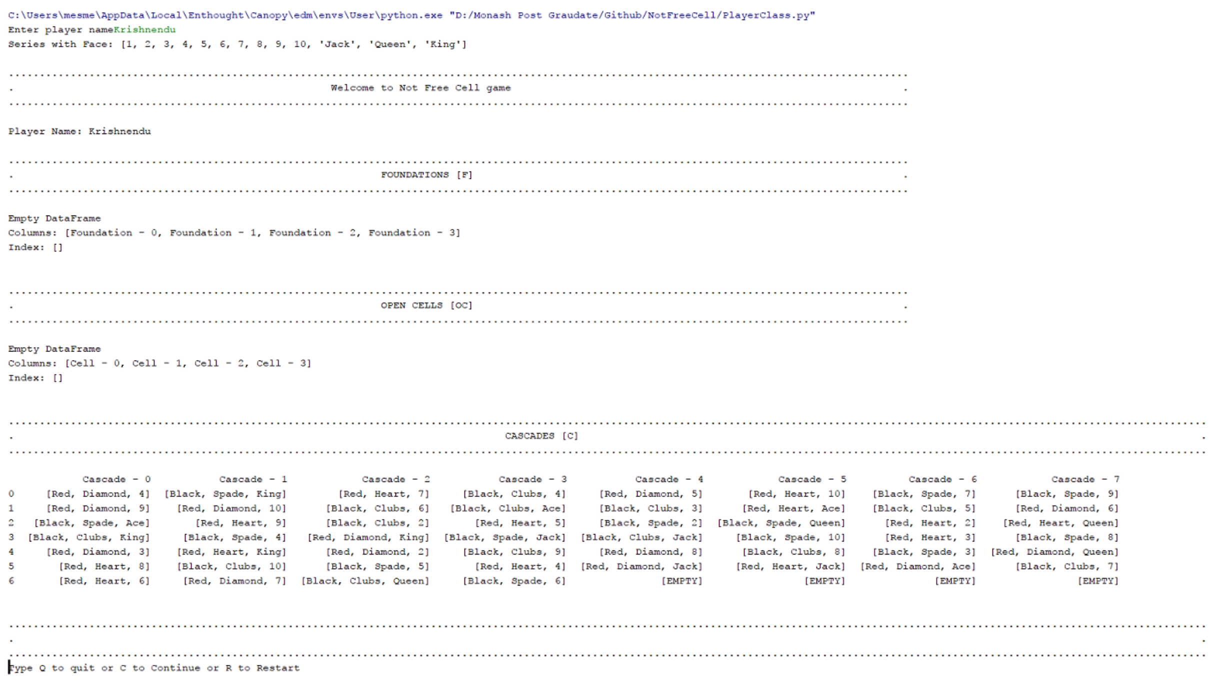


```
C:\Users\mesme\AppData\Local\Enthought\Canopy\edm\envs\User\python.exe "D:/Monash Post Graudate/Github/NotFreeCell/DeckClass.py"
Series with Face: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 'Jack', 'Queen', 'King']

The cards generated are [total = 52]:
[[0, 1], [0, 2], [0, 3], [0, 4], [0, 5], [0, 6], [0, 7], [0, 8], [0, 9], [0, 10], [0, 'Jack'], [0, 'Queen'], [0, 'King'], [1, 1], [1, 2], [1, 3], [1, 4], [1, 5], [1, 6], [1, 7], [1, 8], [1, 9], [1,

The shuffled Deck generated is:
[[0, 6], [0, 5], [1, 2], [2, 8], [2, 7], [2, 6], [1, 5], [3, 4], [1, 'Jack'], [2, 'Queen'], [3, 'Jack'], [0, 9], [1, 1], [3, 2], [1, 'King'], [2, 9], [1, 3], [3, 8], [0, 'King'], [3, 3], [0, 2], [3,

A card :  ['Black', 'Spade', 9]

Process finished with exit code 0
```

The sample of an enhanced card using Card class is provided in the screenshot above.

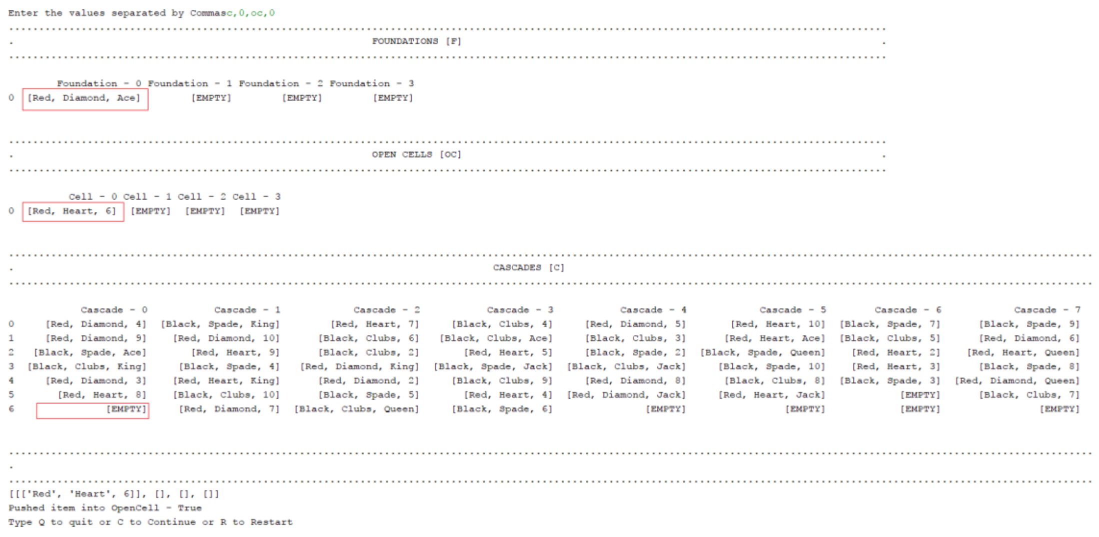
Game Play:

a) Initial Launch



```
C:\Users\mesme\AppData\Local\Enthought\Canopy\edm\envs\User\python.exe "D:/Monash Post Graudate/Github/NotFreeCell/PlayerClass.py"
Enter player nameKrishnendu
Series with Face: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 'Jack', 'Queen', 'King']

..........................................................................................
.                                Welcome to Not Free Cell game                           .
..........................................................................................

Player Name: Krishnendu

..........................................................................................
.                                     FOUNDATIONS [F]                                    .
..........................................................................................

Empty DataFrame
Columns: [Foundation - 0, Foundation - 1, Foundation - 2, Foundation - 3]
Index: []


..........................................................................................
.                                     OPEN CELLS [OC]                                    .
..........................................................................................

Empty DataFrame
Columns: [Cell - 0, Cell - 1, Cell - 2, Cell - 3]
Index: []


........................................................................................................................
.                                                    CASCADES [C]                                                      .
........................................................................................................................

          Cascade - 0         Cascade - 1          Cascade - 2         Cascade - 3           Cascade - 4          Cascade - 5         Cascade - 6           Cascade - 7
0    [Red, Diamond, 4]  [Black, Spade, King]      [Red, Heart, 7]    [Black, Clubs, 4]    [Red, Diamond, 5]     [Red, Heart, 10]   [Black, Spade, 7]    [Black, Spade, 9]
1    [Red, Diamond, 9]   [Red, Diamond, 10]     [Black, Clubs, 6]  [Black, Clubs, Ace]    [Black, Clubs, 3]    [Red, Heart, Ace]   [Black, Clubs, 5]    [Red, Diamond, 6]
2  [Black, Spade, Ace]      [Red, Heart, 9]     [Black, Clubs, 2]      [Red, Heart, 5]    [Black, Spade, 2] [Black, Spade, Queen]     [Red, Heart, 2]  [Red, Heart, Queen]
3 [Black, Clubs, King]    [Black, Spade, 4] [Red, Diamond, King] [Black, Spade, Jack] [Black, Clubs, Jack]   [Black, Spade, 10]     [Red, Heart, 3]    [Black, Spade, 8]
4    [Red, Diamond, 3]   [Red, Heart, King]     [Red, Diamond, 2]    [Black, Clubs, 9]    [Red, Diamond, 8]    [Black, Clubs, 8]   [Black, Spade, 3] [Red, Diamond, Queen]
5      [Red, Heart, 8]   [Black, Clubs, 10]     [Black, Spade, 5]      [Red, Heart, 4] [Red, Diamond, Jack]   [Red, Heart, Jack] [Red, Diamond, Ace]    [Black, Clubs, 7]
6      [Red, Heart, 6]    [Red, Diamond, 7] [Black, Clubs, Queen]    [Black, Spade, 6]              [EMPTY]              [EMPTY]             [EMPTY]              [EMPTY]


........................................................................................................................
.                                                                                                                      .
........................................................................................................................
Type Q to quit or C to Continue or R to Restart
```

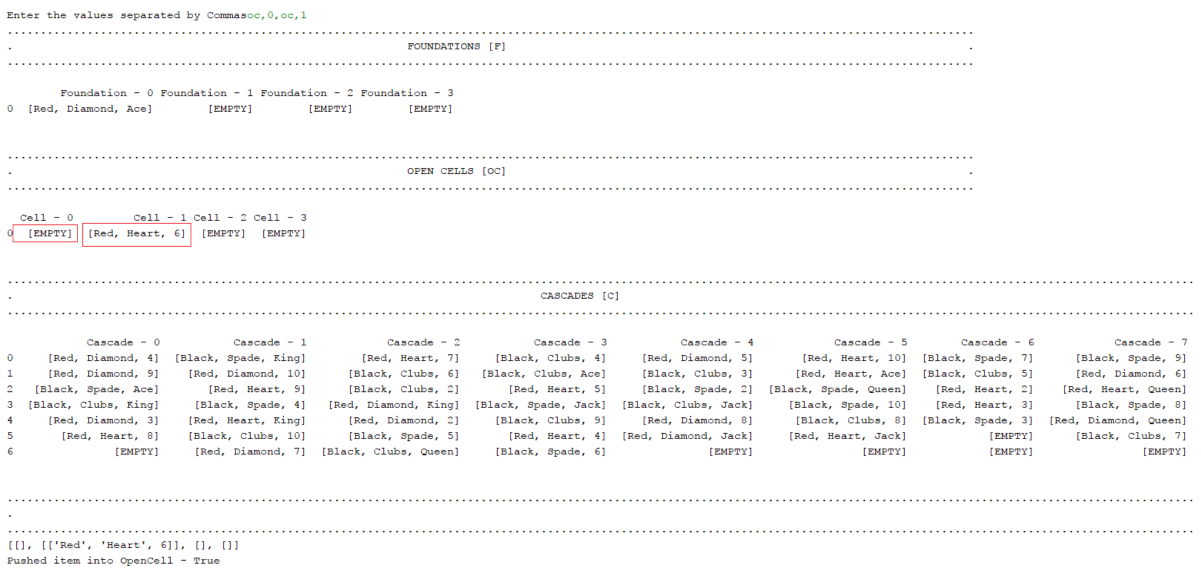
b) Move from first cascade to first open cell [C,0, OC,0]:



```
Enter the values separated by Commasc,0,oc,0
..........................................................................................
.                                     FOUNDATIONS [F]                                    .
..........................................................................................

       Foundation - 0 Foundation - 1 Foundation - 2 Foundation - 3
0  [Red, Diamond, Ace]        [EMPTY]        [EMPTY]        [EMPTY]


..........................................................................................
.                                     OPEN CELLS [OC]                                    .
..........................................................................................

          Cell - 0 Cell - 1 Cell - 2 Cell - 3
0  [Red, Heart, 6]  [EMPTY]  [EMPTY]  [EMPTY]


........................................................................................................................
.                                                    CASCADES [C]                                                      .
........................................................................................................................

          Cascade - 0         Cascade - 1          Cascade - 2         Cascade - 3           Cascade - 4          Cascade - 5         Cascade - 6           Cascade - 7
0    [Red, Diamond, 4]  [Black, Spade, King]      [Red, Heart, 7]    [Black, Clubs, 4]    [Red, Diamond, 5]     [Red, Heart, 10]   [Black, Spade, 7]    [Black, Spade, 9]
1    [Red, Diamond, 9]   [Red, Diamond, 10]     [Black, Clubs, 6]  [Black, Clubs, Ace]    [Black, Clubs, 3]    [Red, Heart, Ace]   [Black, Clubs, 5]    [Red, Diamond, 6]
2  [Black, Spade, Ace]      [Red, Heart, 9]     [Black, Clubs, 2]      [Red, Heart, 5]    [Black, Spade, 2] [Black, Spade, Queen]     [Red, Heart, 2]  [Red, Heart, Queen]
3 [Black, Clubs, King]    [Black, Spade, 4] [Red, Diamond, King] [Black, Spade, Jack] [Black, Clubs, Jack]   [Black, Spade, 10]     [Red, Heart, 3]    [Black, Spade, 8]
4    [Red, Diamond, 3]   [Red, Heart, King]     [Red, Diamond, 2]    [Black, Clubs, 9]    [Red, Diamond, 8]    [Black, Clubs, 8]   [Black, Spade, 3] [Red, Diamond, Queen]
5      [Red, Heart, 8]   [Black, Clubs, 10]     [Black, Spade, 5]      [Red, Heart, 4] [Red, Diamond, Jack]   [Red, Heart, Jack]             [EMPTY]    [Black, Clubs, 7]
6              [EMPTY]    [Red, Diamond, 7] [Black, Clubs, Queen]    [Black, Spade, 6]              [EMPTY]              [EMPTY]             [EMPTY]              [EMPTY]


........................................................................................................................
.
........................................................................................................................
[[['Red', 'Heart', 6]], [], [], []]
Pushed item into OpenCell - True
Type Q to quit or C to Continue or R to Restart
```

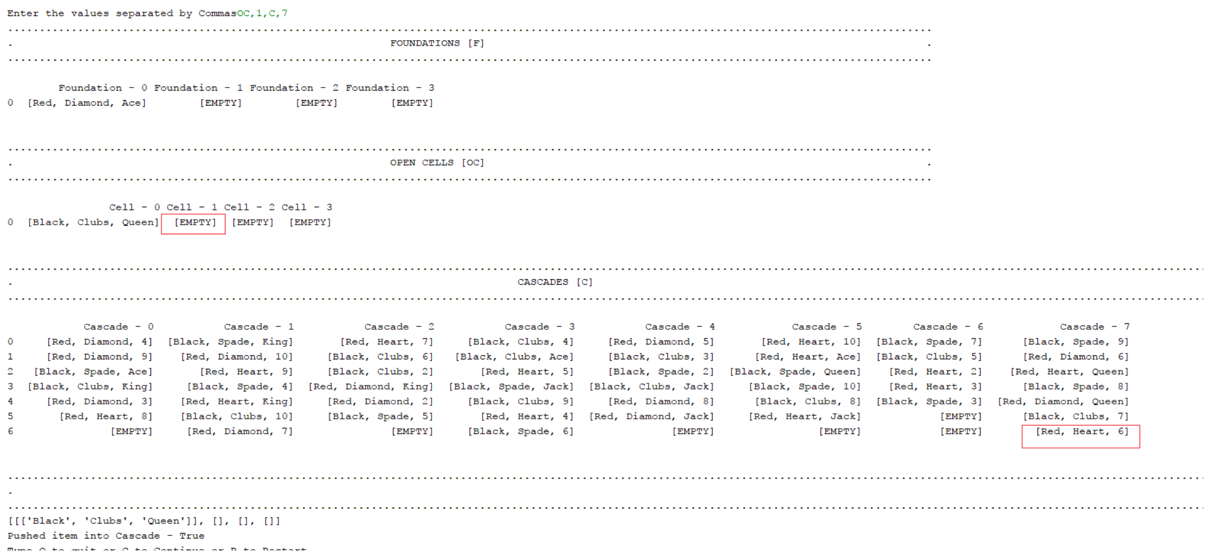
The item is successfully pushed into the open cell from the cascade. A success message id displayed to the user as well. It can be also seen that the Ace has been moved to the Foundation automatically.

c) Move from first open cell to second open cell [OC,0, OC,1]:



```
Enter the values separated by Commasoc,0,oc,1
.........................................................................................................................
.                                                  FOUNDATIONS [F]                                                      .
.........................................................................................................................

       Foundation - 0 Foundation - 1 Foundation - 2 Foundation - 3
0  [Red, Diamond, Ace]        [EMPTY]        [EMPTY]        [EMPTY]


.........................................................................................................................
.                                                  OPEN CELLS [OC]                                                      .
.........................................................................................................................

  Cell - 0          Cell - 1 Cell - 2 Cell - 3
0  [EMPTY]  [Red, Heart, 6]  [EMPTY]  [EMPTY]


...........................................................................................................................................................
.                                                                    CASCADES [C]
...........................................................................................................................................................

          Cascade - 0          Cascade - 1             Cascade - 2          Cascade - 3          Cascade - 4             Cascade - 5        Cascade - 6             Cascade - 7
0     [Red, Diamond, 4] [Black, Spade, King]        [Red, Heart, 7]     [Black, Clubs, 4]    [Red, Diamond, 5]       [Red, Heart, 10] [Black, Spade, 7]     [Black, Spade, 9]
1     [Red, Diamond, 9]    [Red, Diamond, 10]     [Black, Clubs, 6]   [Black, Clubs, Ace]    [Black, Clubs, 3]      [Red, Heart, Ace] [Black, Clubs, 5]     [Red, Diamond, 6]
2   [Black, Spade, Ace]       [Red, Heart, 9]     [Black, Clubs, 2]       [Red, Heart, 5]    [Black, Spade, 2] [Black, Spade, Queen]    [Red, Heart, 2]  [Red, Heart, Queen]
3  [Black, Clubs, King]     [Black, Spade, 4] [Red, Diamond, King] [Black, Spade, Jack]  [Black, Clubs, Jack]     [Black, Spade, 10]    [Red, Heart, 3]     [Black, Spade, 8]
4     [Red, Diamond, 3]   [Red, Heart, King]     [Red, Diamond, 2]     [Black, Clubs, 9]    [Red, Diamond, 8]      [Black, Clubs, 8] [Black, Spade, 3] [Red, Diamond, Queen]
5       [Red, Heart, 8]    [Black, Clubs, 10]     [Black, Spade, 5]       [Red, Heart, 4] [Red, Diamond, Jack]     [Red, Heart, Jack]           [EMPTY]     [Black, Clubs, 7]
6               [EMPTY]     [Red, Diamond, 7] [Black, Clubs, Queen]     [Black, Spade, 6]              [EMPTY]                [EMPTY]           [EMPTY]               [EMPTY]


...........................................................................................................................................................
.
...........................................................................................................................................................
[[], [['Red', 'Heart', 6]], [], []]
Pushed item into OpenCell - True
```

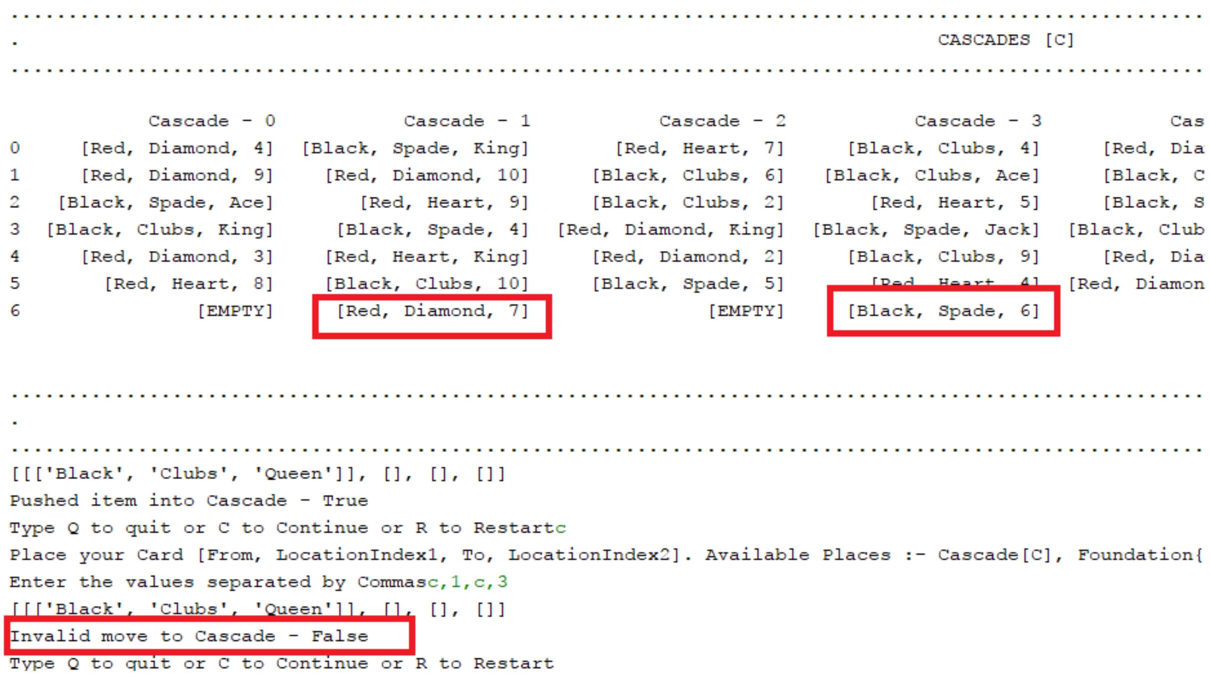
d) Move from second open cell to eight cascade [OC,1, C,7] {Successful move}:



```
Enter the values separated by CommasOC,1,C,7
.........................................................................................................................
.                                                  FOUNDATIONS [F]                                                      .
.........................................................................................................................

       Foundation - 0 Foundation - 1 Foundation - 2 Foundation - 3
0  [Red, Diamond, Ace]        [EMPTY]        [EMPTY]        [EMPTY]


.........................................................................................................................
.                                                  OPEN CELLS [OC]                                                      .
.........................................................................................................................

             Cell - 0 Cell - 1 Cell - 2 Cell - 3
0  [Black, Clubs, Queen]  [EMPTY]  [EMPTY]  [EMPTY]


...........................................................................................................................................................
.                                                                    CASCADES [C]
...........................................................................................................................................................

          Cascade - 0          Cascade - 1             Cascade - 2          Cascade - 3          Cascade - 4             Cascade - 5        Cascade - 6             Cascade - 7
0     [Red, Diamond, 4] [Black, Spade, King]        [Red, Heart, 7]     [Black, Clubs, 4]    [Red, Diamond, 5]       [Red, Heart, 10] [Black, Spade, 7]     [Black, Spade, 9]
1     [Red, Diamond, 9]    [Red, Diamond, 10]     [Black, Clubs, 6]   [Black, Clubs, Ace]    [Black, Clubs, 3]      [Red, Heart, Ace] [Black, Clubs, 5]     [Red, Diamond, 6]
2   [Black, Spade, Ace]       [Red, Heart, 9]     [Black, Clubs, 2]       [Red, Heart, 5]    [Black, Spade, 2] [Black, Spade, Queen]    [Red, Heart, 2]  [Red, Heart, Queen]
3  [Black, Clubs, King]     [Black, Spade, 4] [Red, Diamond, King] [Black, Spade, Jack]  [Black, Clubs, Jack]     [Black, Spade, 10]    [Red, Heart, 3]     [Black, Spade, 8]
4     [Red, Diamond, 3]   [Red, Heart, King]     [Red, Diamond, 2]     [Black, Clubs, 9]    [Red, Diamond, 8]      [Black, Clubs, 8] [Black, Spade, 3] [Red, Diamond, Queen]
5       [Red, Heart, 8]    [Black, Clubs, 10]     [Black, Spade, 5]       [Red, Heart, 4] [Red, Diamond, Jack]     [Red, Heart, Jack]           [EMPTY]     [Black, Clubs, 7]
6               [EMPTY]     [Red, Diamond, 7]               [EMPTY]     [Black, Spade, 6]              [EMPTY]                [EMPTY]           [EMPTY]       [Red, Heart, 6]


...........................................................................................................................................................
.
...........................................................................................................................................................
[[['Black', 'Clubs', 'Queen']], [], [], []]
Pushed item into Cascade - True
```

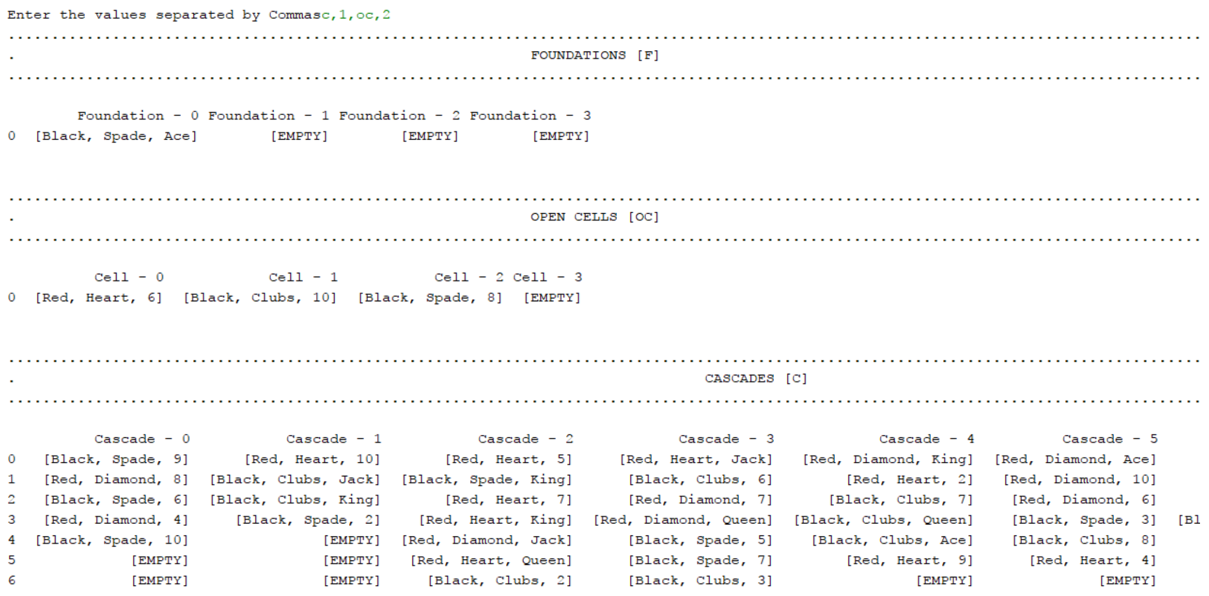
e) Move from second cascade to fourth cascade [OC,1, C,7] {Unsuccessful move} :

Next Page



```
.....................................................................................
.                                                                        CASCADES [C]
.....................................................................................

           Cascade - 0           Cascade - 1           Cascade - 2           Cascade - 3          Cas
0     [Red, Diamond, 4]  [Black, Spade, King]      [Red, Heart, 7]    [Black, Clubs, 4]    [Red, Dia
1     [Red, Diamond, 9]    [Red, Diamond, 10]    [Black, Clubs, 6]  [Black, Clubs, Ace]    [Black, C
2   [Black, Spade, Ace]       [Red, Heart, 9]    [Black, Clubs, 2]      [Red, Heart, 5]    [Black, S
3  [Black, Clubs, King]     [Black, Spade, 4]  [Red, Diamond, King] [Black, Spade, Jack] [Black, Club
4     [Red, Diamond, 3]    [Red, Heart, King]    [Red, Diamond, 2]    [Black, Clubs, 9]    [Red, Dia
5       [Red, Heart, 8]   [Black, Clubs, 10]    [Black, Spade, 5]      [Red, Heart, 4]  [Red, Diamon
6               [EMPTY]     [Red, Diamond, 7]              [EMPTY]    [Black, Spade, 6]


.....................................................................................
.
.....................................................................................
[[['Black', 'Clubs', 'Queen']], [], [], []]
Pushed item into Cascade - True
Type Q to quit or C to Continue or R to Restartc
Place your Card [From, LocationIndex1, To, LocationIndex2]. Available Places :- Cascade[C], Foundation(
Enter the values separated by Commasc,1,c,3
[[['Black', 'Clubs', 'Queen']], [], [], []]
Invalid move to Cascade - False
Type Q to quit or C to Continue or R to Restart
```

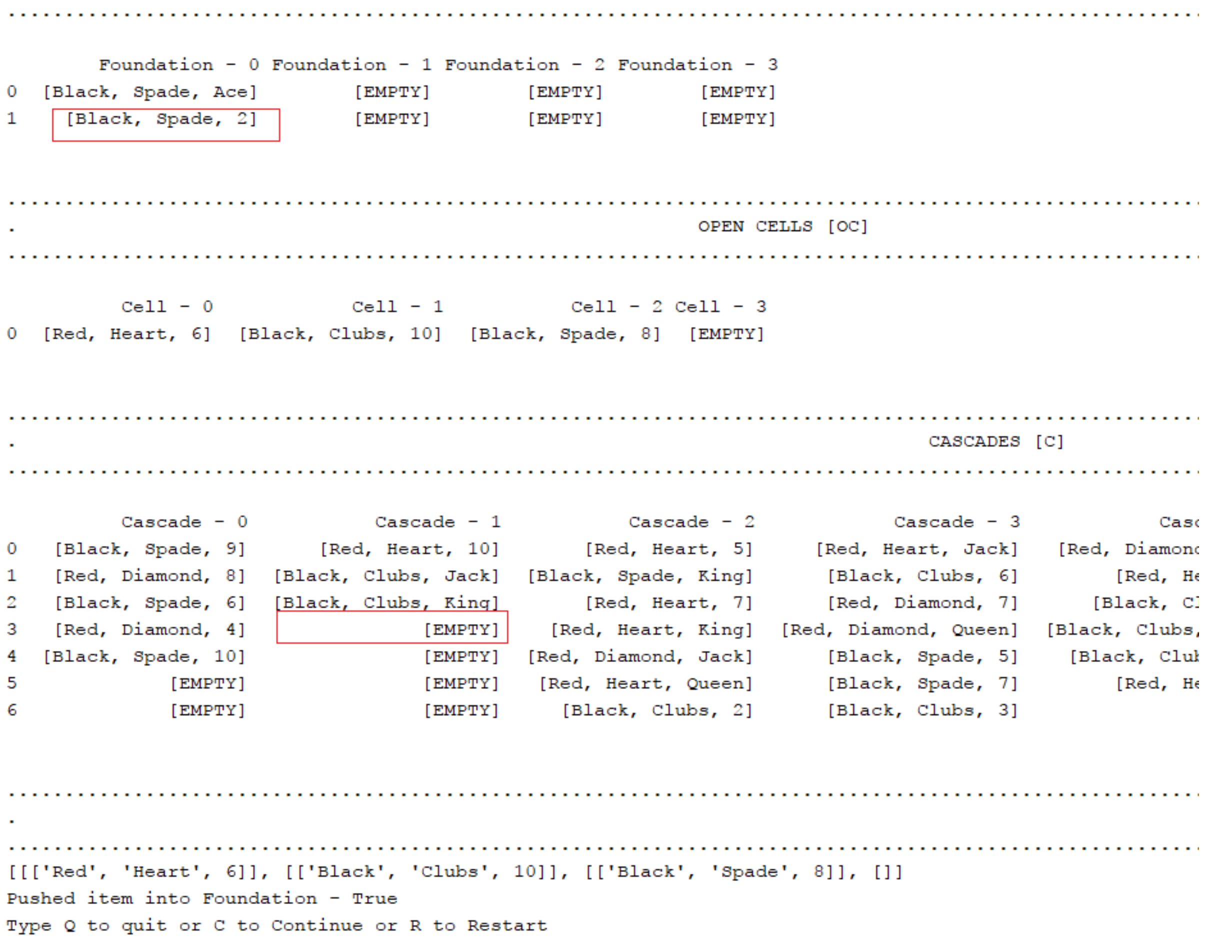
f) Move from Cascade to Foundation: *Initial*



```
Enter the values separated by Commasc,1,oc,2
..........................................................................................................
.                                                        FOUNDATIONS [F]
..........................................................................................................

      Foundation - 0 Foundation - 1 Foundation - 2 Foundation - 3
0  [Black, Spade, Ace]       [EMPTY]        [EMPTY]        [EMPTY]


..........................................................................................................
.                                                        OPEN CELLS [OC]
..........................................................................................................

        Cell - 0          Cell - 1          Cell - 2 Cell - 3
0  [Red, Heart, 6]  [Black, Clubs, 10]  [Black, Spade, 8]  [EMPTY]


..........................................................................................................
.                                                                          CASCADES [C]
..........................................................................................................

        Cascade - 0          Cascade - 1          Cascade - 2           Cascade - 3            Cascade - 4          Cascade - 5
0   [Black, Spade, 9]      [Red, Heart, 10]      [Red, Heart, 5]    [Red, Heart, Jack]  [Red, Diamond, King]  [Red, Diamond, Ace]
1   [Red, Diamond, 8]  [Black, Clubs, Jack] [Black, Spade, King]     [Black, Clubs, 6]       [Red, Heart, 2]   [Red, Diamond, 10]
2   [Black, Spade, 6]  [Black, Clubs, King]      [Red, Heart, 7]     [Red, Diamond, 7]     [Black, Clubs, 7]    [Red, Diamond, 6]
3   [Red, Diamond, 4]     [Black, Spade, 2]   [Red, Heart, King] [Red, Diamond, Queen] [Black, Clubs, Queen]    [Black, Spade, 3]  [Bl
4  [Black, Spade, 10]               [EMPTY] [Red, Diamond, Jack]     [Black, Spade, 5]   [Black, Clubs, Ace]    [Black, Clubs, 8]
5             [EMPTY]               [EMPTY]  [Red, Heart, Queen]     [Black, Spade, 7]       [Red, Heart, 9]      [Red, Heart, 4]
6             [EMPTY]               [EMPTY]    [Black, Clubs, 2]     [Black, Clubs, 3]               [EMPTY]              [EMPTY]
```

The move is now made from first Cascade to the foundation {Success} :

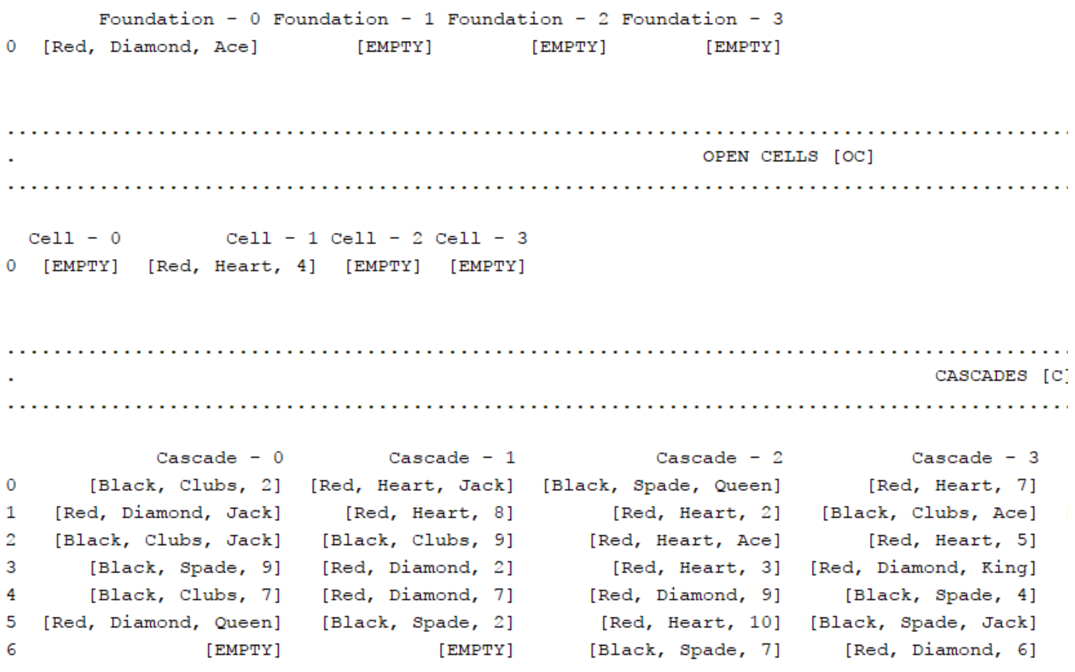
Next Page



```
......................................................................................................

        Foundation - 0 Foundation - 1 Foundation - 2 Foundation - 3
0  [Black, Spade, Ace]         [EMPTY]        [EMPTY]        [EMPTY]
1    [Black, Spade, 2]         [EMPTY]        [EMPTY]        [EMPTY]


......................................................................................................
.                                                            OPEN CELLS [OC]
......................................................................................................

          Cell - 0            Cell - 1           Cell - 2 Cell - 3
0  [Red, Heart, 6]  [Black, Clubs, 10]  [Black, Spade, 8]  [EMPTY]


......................................................................................................
.                                                                                 CASCADES [C]
......................................................................................................

          Cascade - 0          Cascade - 1          Cascade - 2            Cascade - 3           Casc
0   [Black, Spade, 9]       [Red, Heart, 10]       [Red, Heart, 5]     [Red, Heart, Jack]   [Red, Diamonc
1   [Red, Diamond, 8]   [Black, Clubs, Jack]   [Black, Spade, King]     [Black, Clubs, 6]        [Red, He
2   [Black, Spade, 6]   [Black, Clubs, King]       [Red, Heart, 7]     [Red, Diamond, 7]      [Black, Cl
3   [Red, Diamond, 4]                [EMPTY]    [Red, Heart, King]  [Red, Diamond, Queen]  [Black, Clubs,
4  [Black, Spade, 10]                [EMPTY]   [Red, Diamond, Jack]     [Black, Spade, 5]    [Black, Clut
5             [EMPTY]                [EMPTY]   [Red, Heart, Queen]      [Black, Spade, 7]        [Red, He
6             [EMPTY]                [EMPTY]     [Black, Clubs, 2]      [Black, Clubs, 3]


......................................................................................................
.
......................................................................................................
[[['Red', 'Heart', 6]], [['Black', 'Clubs', 10]], [['Black', 'Spade', 8]], []]
Pushed item into Foundation - True
Type Q to quit or C to Continue or R to Restart
```

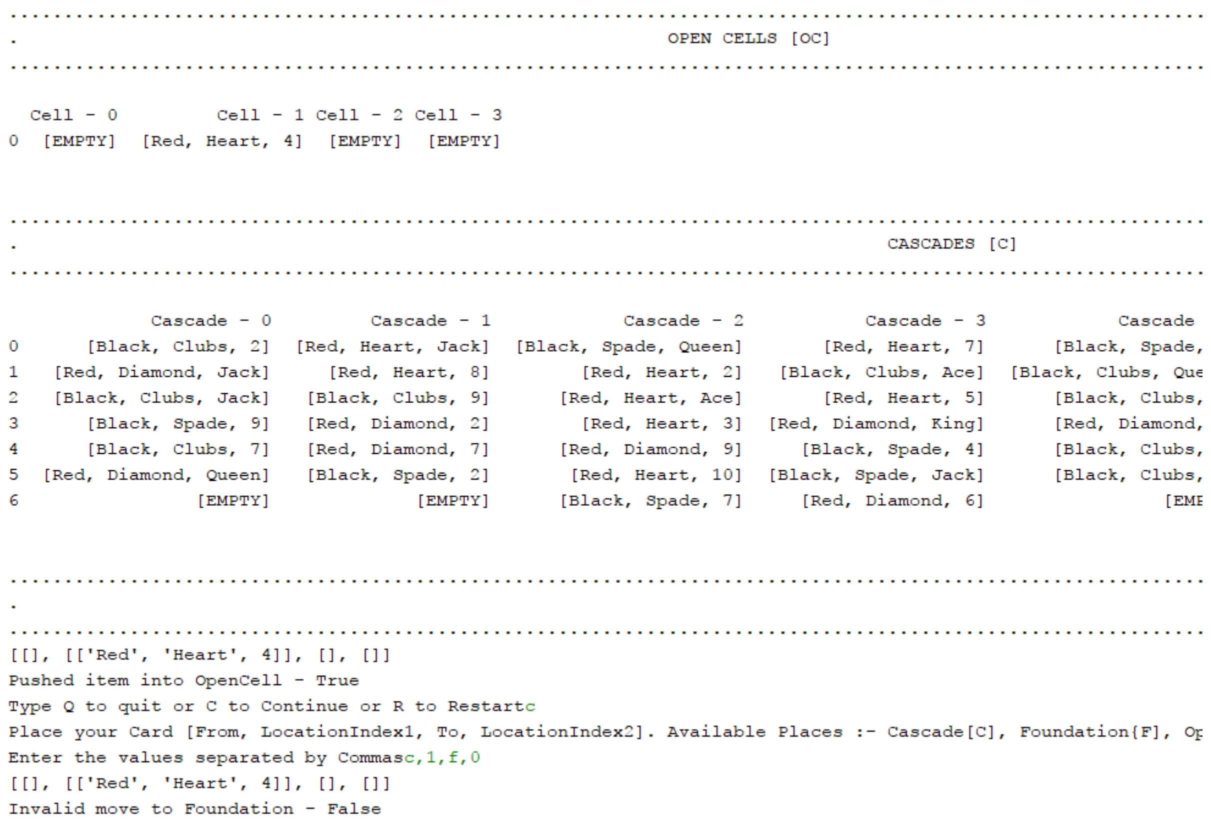
The move is now made from first Cascade to the foundation by any random card{Unsuccessful}:

Initial:



```
        Foundation - 0 Foundation - 1 Foundation - 2 Foundation - 3
0  [Red, Diamond, Ace]         [EMPTY]        [EMPTY]        [EMPTY]


.............................................................................................
.                                                            OPEN CELLS [OC]
.............................................................................................

  Cell - 0          Cell - 1 Cell - 2 Cell - 3
0  [EMPTY]  [Red, Heart, 4]  [EMPTY]  [EMPTY]


.............................................................................................
.                                                                                 CASCADES [C]
.............................................................................................

             Cascade - 0          Cascade - 1            Cascade - 2            Cascade - 3
0      [Black, Clubs, 2]  [Red, Heart, Jack]  [Black, Spade, Queen]       [Red, Heart, 7]
1   [Red, Diamond, Jack]     [Red, Heart, 8]        [Red, Heart, 2]   [Black, Clubs, Ace]
2   [Black, Clubs, Jack]   [Black, Clubs, 9]      [Red, Heart, Ace]       [Red, Heart, 5]
3      [Black, Spade, 9]   [Red, Diamond, 2]        [Red, Heart, 3]  [Red, Diamond, King]
4      [Black, Clubs, 7]   [Red, Diamond, 7]      [Red, Diamond, 9]     [Black, Spade, 4]
5  [Red, Diamond, Queen]   [Black, Spade, 2]       [Red, Heart, 10]  [Black, Spade, Jack]
6                [EMPTY]             [EMPTY]      [Black, Spade, 7]     [Red, Diamond, 6]
```

Invalidated:



```
.....................................................................................................
.                                                OPEN CELLS [OC]
.....................................................................................................

  Cell - 0        Cell - 1 Cell - 2 Cell - 3
0  [EMPTY]  [Red, Heart, 4]  [EMPTY]  [EMPTY]


.....................................................................................................
.                                                                CASCADES [C]
.....................................................................................................

           Cascade - 0         Cascade - 1          Cascade - 2          Cascade - 3          Cascade
0      [Black, Clubs, 2]  [Red, Heart, Jack]  [Black, Spade, Queen]      [Red, Heart, 7]      [Black, Spade,
1   [Red, Diamond, Jack]     [Red, Heart, 8]        [Red, Heart, 2]  [Black, Clubs, Ace]  [Black, Clubs, Que
2   [Black, Clubs, Jack]   [Black, Clubs, 9]      [Red, Heart, Ace]      [Red, Heart, 5]      [Black, Clubs,
3      [Black, Spade, 9]   [Red, Diamond, 2]        [Red, Heart, 3]  [Red, Diamond, King]     [Red, Diamond,
4      [Black, Clubs, 7]   [Red, Diamond, 7]      [Red, Diamond, 9]    [Black, Spade, 4]      [Black, Clubs,
5  [Red, Diamond, Queen]   [Black, Spade, 2]       [Red, Heart, 10]  [Black, Spade, Jack]     [Black, Clubs,
6                [EMPTY]             [EMPTY]      [Black, Spade, 7]    [Red, Diamond, 6]               [EMI


.....................................................................................................
.
.....................................................................................................
[[], [['Red', 'Heart', 4]], [], []]
Pushed item into OpenCell - True
Type Q to quit or C to Continue or R to Restartc
Place your Card [From, LocationIndex1, To, LocationIndex2]. Available Places :- Cascade[C], Foundation{F], Op
Enter the values separated by Commasc,1,f,0
[[], [['Red', 'Heart', 4]], [], []]
Invalid move to Foundation - False
```

References:

1) https://www.alexandriarepository.org/
2) https://stackoverflow.com/
3) https://www.learnpython.org/en/Pandas_Basics