CPSC-442X Python Programming

Assignment 2: Tic-Tac-Toe

Due: March 4th, 2016 at 11:59 PM

Tic-tac-toe is a two players' game: player X and player O. Both players take turns placing their mark (X or O) on the spaces in a 3×3 grid. The first player to place three of their marks in a horizontal, vertical, or diagonal row wins the game.

For this assignment you need to implement the Tic-tac-toe game in Python as console application under the following guidelines:
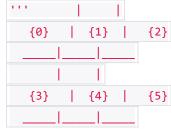
1. Create three classes `Player`, `Deck`, and `TicTacToe`.
2. For the class `Player`, it must have the following attributes:
   - `Name` (e.g. Alice, X, O …etc.)
   - `PlayingMark` (X or O),
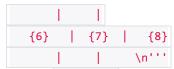   - `Statistics` (won, drawn, lost).

   It should implement the methods:

   - `__init__`: the class constructor
   - `get_score()` which should return `((won * 2) + draw - lost)`.
   - `__str__` method that would print the player name, mark, and score in the format: "Player: X, Mark: X, Score: 120"
   - And for comparison, it should implement the `__lt__` operator to compare players score.
3. For the class `Deck`, it must have the following data attributes:
   - `Board`: a 3×3 board implemented as a list or any other data structure.
   - `Player1Choices`: a list or any other data structure that contains the indexes of cells that player 1 choose.
   - `Player2Choices`: a list or any other data structure that contains the indexes of cells that player 2 choose.

   It should also implement the following methods:

   - `__init__`: the class constructor
   - `__str__`: To print the current board status as:
     ```
     '''        |       |
         {0}    |  {1}  |   {2}
        ____|____|____
                |       |
         {3}    |  {4}  |   {5}
        ____|____|____
     ```

```
        |       |
  {6}   |  {7}  |   {8}
        |       |    \n'''
```

4. The class `TicTacToe` must have the following attributes:
   - `DeckList` : a list of objects of the Deck class, where the current Deck is the last item.
   - `Player1` : an instance of the class Player representing player 1.
   - `Player2` : an instance of the class Player representing player 2.

   Additionally it should implement the following methods:

   - `__init__` : the class constructor
   - `validate_user_input()` : validate if the user input is an int between 0 - 8 and it was not played previously.
   - `is_game_over()` : check if the game is over by finding if a user won or if the board is full, if true then append the Deck lists to a file with the name "TicTacToe.txt" and return true, else return false.
   - `get_user_input()` : a method to get user input, it should display a message like "enter player {name} choice:". and then it should call the `validate_user_input()` to validate the user input, if user's input is valid, then store the value in the Board and the check if the game is over by calling `is_game_over()`
   - `start_game()` : the main game logic should go here in this class, it should add a list item to DeckList, and while the game is not over, keep calling `get_user_input()` for each user. Once a game is over, display each user data by calling the print as passing the player object to invoke the `__str__` function, then ask if the user wants to play again and then start a new game.

Notes:

- Use the function `os.system('cls')` to clear the screen after each move.
- Make sure that your code is well documented by using line comments and docStrings.

```
C:\Windows\system32\cmd.exe

Tic-Tac-Toe game number: 1
Enter player X move:
```

```
C:\Windows\system32\cmd.exe

  X  |     |
_____|_____|_____
     |     |
_____|_____|_____
     |     |
     |     |

Enter player O move:
```

```
C:\Windows\system32\cmd.exe                                          [_][□][X]

   X    |     |
_____|_____|_____
        |     |
        |     |
_____|_____|_____
        |     |
        |     |

Enter player O move: 0
The cell at index 0 was already taken!
--------------------------------------------------------------
Enter player O move: asd
invalid move, move should be an integer
--------------------------------------------------------------
Enter player O move: 15
invalid move, move should be between 0 and 8
--------------------------------------------------------------
Enter player O move:
```

```
C:\Windows\system32\cmd.exe                                          [_][□][X]

   X    |     |
_____|_____|_____
        |     |
        |  O  |
_____|_____|_____
        |     |
        |     |

Enter player X move:
```

```
C:\Windows\system32\cmd.exe

  X  |  X  |  O
_____|_____|_____
  X  |  O  |
_____|_____|_____
  O  |     |
     |     |

Player O won!
Game over!
Player: X, mark: X, score: -1
Player: O, mark: O, score: 2
start a new game? (Y\N)
```

```
C:\Windows\system32\cmd.exe

  O  |  O  |  X
_____|_____|_____
  X  |  X  |  O
_____|_____|_____
  O  |  X  |  X
     |     |

Draw, no more moves!
Game over!
Player: X, mark: X, score: 0
Player: O, mark: O, score: 3
start a new game? (Y\N)
```