

Tic-Tac-Toe game without using OOP's

In []:

```
1 from termcolor import colored
2
3 # showing the game board after plotting any "X/O"
4 def game_board(x):
5     try:
6         print()
7         print (f" {x[1]} | {x[2]} | {x[3]} ")
8         print (f"---|---|---")
9         print (f" {x[4]} | {x[5]} | {x[6]} ")
10        print (f"---|---|---")
11        print (f" {x[7]} | {x[8]} | {x[9]} ")
12        print
13    except Exception as e:
14        pass
15
16 # Generating possible outcome from user inputted options
17 def possibilities(x):
18     poss_user = []
19     for i in range(0,len(x)-2):           # pos 1
20         for j in range(1,len(x)-1):       # pos 2
21             for k in range(1,len(x)):     # pos 3
22                 total=f"{x[i]}{x[j]}{x[k]}" # joining
23                 poss_user.append(int(total)) # adding to list of possi
24     return poss_user
25
26
27 # Checking tthe final score if any user has made XXX/000 in a row
28 def check_score(x):
29     fp = open("final_possible_scores.txt","r")
30     data = eval(fp.read())                # All possibilities
31     final_possible_scores = set(data)      # reading from a fi
32
33
34     user_score = set(possibilities(x))     # possiblities of
35
36     if (final_possible_scores & user_score): # checking user input
37         return True
38     else:
39         return False
40
41
42
43 # Places Reaminging on the board for the players
44 dict = {1: '1', 2: '2', 3: '3', 4: '4', 5: '5', 6: '6', 7: '7', 8: '8', 9: '9'}
45 choice = [1,2,3,4,5,6,7,8,9]
46
47 # User selected Locations
48 p1 = []
49 p2 = []
50
51 #Inital Board Situation
52 game_board(dict)
53
54 # will run until all spaces on board are occupied
55 while(choice!=[]):
56
57     # Odd number means Player 1 will be playing
58     if(len(choice)%2 != 0):
59         try:
```

```

60     pos1 = int(input(f"\nChoices Remaining : {len(choice)} || Player 1 Posit
61 except Exception as e:
62     print ("Please select an option from a range of 1-9")
63 else:
64     if pos1 in choice:                # Checking availability of usr entered l
65         dict[pos1]= "X"              # updating Dictionary for the player by
66         p1.append(pos1)              # updating p1 ie. user selced loations
67         choice.remove(pos1)          # removing the choice so other player ca
68         game_board(dict)             # showing the O/P
69
70     if(len(p1) >= 3):                 # usr should have played atleast 3 times
71         if(check_score(p1)):         # checking score and printing result
72             text = colored('PLAYER 1 IS WINNER', 'red', attrs=['reverse', 'b
73             print("\n|-----|")
74             print(f"|--- {text} ---|")
75             print("|-----|")
76             break
77
78 # Even number means Player 2 will be playing
79 else:
80     try:
81         pos2 = int(input(f"\nChoices Remaining : {len(choice)} || Player 2 Posit
82 except Exception as e:
83     print ("Please select an option from a range of 1-9")
84 else:
85     if pos2 in choice:
86         dict[pos2]= "0"
87         p2.append(pos2)
88         choice.remove(pos2)
89         game_board(dict)
90
91     if(len(p2) >= 3):
92         if(check_score(p2)):
93             text = colored('PLAYER 2 IS WINNER', 'red', attrs=['reverse', 'b
94             print("\n|-----|")
95             print(f"|--- {text} ---|")
96             print("|-----|")
97             break
98
99 # Match Draw
100 # If P1 & P2 both are not matching any of the winning possibilities
101 if( (check_score(p1)==False) and (check_score(p2)==False) ):
102     text = colored('MATCH DRAW', 'red', attrs=['reverse', 'blink'])
103     print("\n|-----|")
104     print(f"|----- {text} -----|")
105     print("|-----|")
106

```

Tic-Tac-Toe using OOP's

In []:

```
1 from termcolor import colored
2
3 # showing the game board after plotting any "X/O"
4 class Game():
5     def __init__(self):
6         self.player_list = []
7
8     def game_board(x):
9         try:
10             print()
11             print (f" {x[1]} | {x[2]} | {x[3]} ")
12             print (f"---|---|---")
13             print (f" {x[4]} | {x[5]} | {x[6]} ")
14             print (f"---|---|---")
15             print (f" {x[7]} | {x[8]} | {x[9]} ")
16             print
17         except Exception as e:
18             pass
19
20 # Generating possible outcome from user inputted options
21 def possibilities(x):
22     poss_user = []
23     for i in range(0,len(x)-2):           # pos 1
24         for j in range(1,len(x)-1):       # pos 2
25             for k in range(1,len(x)):      # pos 3
26                 total=f"{x[i]}{x[j]}{x[k]}" # joining
27                 poss_user.append(int(total)) # adding to list of p
28     return poss_user
29
30
31 # Checking tthe final score if any user has made XXX/000 in a row
32 def check_score(x):
33     fp = open("final_possible_scores.txt","r")
34     data = eval(fp.read())                # All possibili
35     final_possible_scores = set(data)      # reading from
36
37
38     user_score = set(Game.possibilities(x)) # possibl
39
40     if (final_possible_scores & user_score): # checking user i
41         return True
42     else:
43         return False
44
45
46 # Places Reaminging on the board for the players
47 dict = {1: '1', 2: '2', 3: '3', 4: '4', 5: '5', 6: '6', 7: '7', 8: '8', 9: '9'}
48 choice = [1,2,3,4,5,6,7,8,9]
49
50
51 # User selected Locations
52 p1 = Game()
53 p2 = Game()
54
55 #Inital Board Situation
56 Game.game_board(dict)
57
58 # will run until all spaces on board are occupied
59 while(choice!=[]):
```

```

60
61 # Odd number means Player 1 will be playing
62 if(len(choice)%2 != 0):
63     try:
64         pos1 = int(input(f"\nChoices Remaining : {len(choice)} || Player 1 Posit
65     except Exception as e:
66         print ("Please select an option from a range of 1-9")
67     else:
68         if pos1 in choice:                # Checking availability of usr entered l
69             dict[pos1]= "X"                # updating Dictionary for the player by
70             p1.player_list.append(pos1)    # updating p1 ie. user selce
71             choice.remove(pos1)            # removing the choice so other player ca
72             Game.game_board(dict)          # showing the O/P
73
74         if(len(p1.player_list) >= 3):      # usr should have played atl
75             if(Game.check_score(p1.player_list)): # checking score and pr
76                 text = colored('PLAYER 1 IS WINNER', 'red', attrs=['reverse', 'b
77                 print("\n|-----|")
78                 print(f"|--- {text} ---|")
79                 print("|-----|")
80                 break
81
82 # Even number means Player 2 will be playing
83 else:
84     try:
85         pos2 = int(input(f"\nChoices Remaining : {len(choice)} || Player 2 Posit
86     except Exception as e:
87         print ("Please select an option from a range of 1-9")
88     else:
89         if pos2 in choice:
90             dict[pos2]= "0"
91             p2.player_list.append(pos2)
92             choice.remove(pos2)
93             Game.game_board(dict)
94
95         if(len(p2.player_list) >= 3):
96             if(Game.check_score(p2.player_list)):
97                 text = colored('PLAYER 2 IS WINNER', 'red', attrs=['reverse', 'b
98                 print("\n|-----|")
99                 print(f"|--- {text} ---|")
100                 print("|-----|")
101                 break
102
103 # Match Draw
104 # If P1 & P2 both are not matching any of the winning possibilities
105 if( (Game.check_score(p1.player_list)==False) and (Game.check_score(p2.player_list)=
106     text = colored('MATCH DRAW', 'red', attrs=['reverse', 'blink'])
107     print("\n|-----|")
108     print(f"|----- {text} -----|")
109     print("|-----|")
110

```

In []:

```
1 # final_possible_scores = { 123,132,147,174,159,195,  
2 # 213,231,258,285,  
3 # 312,321,369,396,357,375,  
4 # 417,471,456,465, # ALL poss  
5 # 519,591,537,573,528,582,546,564,  
6 # 639,693,654,645,  
7 # 714,741,789,798,753,735,  
8 # 879,897,852,825,  
9 # 915,951,936,963,978,987 }
```