Tictactoe game

January 18, 2022

[1]:
```
def display(row1,row2,row3):
    print(row1)
    print(row2)
    print(row3)
```

[2]:
```
row1=[' ',' ',' ']
row2=[' ',' ',' ']
row3=[' ',' ',' ']

display(row1,row2,row3)
```

```
[' ', ' ', ' ']
[' ', ' ', ' ']
[' ', ' ', ' ']
```

[3]:
```
row2[1]= 'X'
```

[4]:
```
display(row1,row2,row3)
```

```
[' ', ' ', ' ']
[' ', 'X', ' ']
[' ', ' ', ' ']
```

[5]:
```
pos_in = int(input("Enter an index position: "))
#pos_in is typecasted because it usually returns a string  but we need an␣
 ↪integer value since we are dealing with position indices.
```

```
Enter an index position: 2
```

[6]:
```
row1[pos_in]= 'O'
```

[7]:
```
row1[pos_in]
```

[7]:
```
'O'
```

[8]:
```
display(row1,row2,row3)
```

```
[' ', ' ', 'O']
[' ', 'X', ' ']
[' ', ' ', ' ']
```

[11]:
```
def user_choice():

    choice = 'Wrong'

    while choice.isdigit() == False:

        choice= input("Enter a number between 0 and 10: ")
        #you can check if choice is a digit instead of typecasting choice.
        if choice.isdigit() == False:
            print('sorry,That is not a digit!')

    return int(choice)
```

[12]:
```
user_choice()
```

```
Enter a number between 0 and 10: 3
```

[12]:
```
3
```

[13]:
```
def user_choice():

    #two variables
    choice = 'Wrong'
    acceptable_range= range(0,10)
    within_range = False

    #Two conditions to be checked one digit and then range
    while choice.isdigit() == False or within_range == False:

        choice= input("Enter a number between 0 and 10: ")


        #CHECKING IF INPUT IS DIGIT
        if choice.isdigit() == False:
            print('sorry,That is not a digit!')

             #RANGE CHECK

        if choice.isdigit()== True:
                if int(choice) in acceptable_range:
                    within_range = True
                else:
                    within_range= False
                    print('The number is not in range')


    return int(choice)
```

[14]:
```
user_choice()
```

```
Enter a number between 0 and 10: save
sorry,That is not a digit!
Enter a number between 0 and 10: 2345
The number is not in range
Enter a number between 0 and 10: 5
```

[14]: 5

[15]:
```
game_list= [0,1,2]
```

[16]:
```
def display_game(game_list):
    print("The current list is")
    print(game_list)
```

[17]:
```
display_game(game_list)
```

```
The current list is
[0, 1, 2]
```

[18]:
```
def placement_choice():

    choice = 'Wrong'

    while choice not in ['0','1','2']:
        choice = input('Pick an index position')
        if choice not in ['0','1','2']:
            print('Sorry,You have chosen and invalid index position.')

    return int(choice)
```

[19]:
```
placement_choice()
```

```
Pick an index position1
```

[19]: 1

[20]:
```
def replacement_choice(game_list,pos):

    user_choice= input("Replace string at picked index position with a string␣
 ↪of your choice!: ")
    game_list[pos]= user_choice


    return game_list
```

[21]:
```
replacement_choice(['0','1','2'],1)
```

```
Replace string at picked index position with a string of your choice!: new
```

[21]:
```
['0', 'new', '2']
```

[22]:
```
def end_gamechoice():

    choice= 'Wrong'

    while choice not in ['Y','N']:

        choice= input("Do you want to keep playing (Y or N): ")

        if choice not in ['Y','N']:
            print('Sorry! I do not understand Please choose either Y or N')

    if choice=='Y':
                return True
    else:
                return False
```

[23]:
```
end_gamechoice()
```

```
Do you want to keep playing (Y or N): y
Sorry! I do not understand Please choose either Y or N
Do you want to keep playing (Y or N):
Sorry! I do not understand Please choose either Y or N
Do you want to keep playing (Y or N): Y
```

[23]:
```
True
```

[24]:
```
game_on = True
game_list=['0','1','2']

while game_on:
           display_game(game_list)

           placement= placement_choice()

           game_list= replacement_choice(game_list,placement)

           display_game(game_list)

           game_on = end_gamechoice()
```

```
The current list is
['0', '1', '2']
Pick an index positiontw9
Sorry,You have chosen and invalid index position.
Pick an index position1
Replace string at picked index position with a string of your choice!: maanav
The current list is
```

```
['0', 'maanav', '2']
Do you want to keep playing (Y or N): y
Sorry! I do not understand Please choose either Y or N
Do you want to keep playing (Y or N): Y
The current list is
['0', 'maanav', '2']
Pick an index position2
Replace string at picked index position with a string of your choice!: new
The current list is
['0', 'maanav', 'new']
Do you want to keep playing (Y or N): N
```

[25]:
```
from IPython.display import clear_output
def display_board(board):
    clear_output()
    print(board[7]+'|'+board[8]+'|'+board[9])
    print('-+-+-')
    print(board[4]+'|'+board[5]+'|'+board[6])
    print('-+-+-')
    print(board[1]+'|'+board[2]+'|'+board[3])
```

[26]:
```
test_board= ['#','X','O','X','O','X','O','X','O','X']
display_board(test_board)
```

```
X|O|X
-+-+-
O|X|O
-+-+-
X|O|X
```

[35]:
```
def marker_choice():



    marker = ''

    #Condition runs untill either X or O are chosen
    while marker!= 'X' and marker!= 'O':

        #Asking player1 to make a choice
        marker= input('Player1 please choose either X or O: ')

        player1= marker
        #Checking player1's choice to assign the oter to player2
    if player1 == 'X':
                    return('X','O')
    else:
                    return('O','X')
```

[36]:
```
player1, player2= marker_choice()
```

```
Player1 please choose either X or O: X
```

[37]:
```
player2
```

[37]:
```
'O'
```

[38]:
```
def game_play(board,marker,position):
    board[position]=marker
```

[40]:
```
game_play(test_board,'$',8)
display_board(test_board)
```

```
X|$|X
-+-+-
O|X|O
-+-+-
X|O|X
```

[41]:
```
def win_check(board,mark):
    return((board[7]==board[8]==board[9]==mark)or #row1
    (board[4]==board[5]==board[6]==mark)or #row2
    (board[1]==board[2]==board[3]==mark)or #row3
    (board[7]==board[4]==board[1]==mark)or #column1
    (board[8]==board[5]==board[2]==mark)or #column2
    (board[9]==board[6]==board[3]==mark)or #column3
    (board[7]==board[5]==board[3]==mark)or #diag1
    (board[1]==board[5]==board[9]==mark))  #diag2
```

[42]:
```
display_board(test_board)
win_check(test_board,'X')
```

```
X|$|X
-+-+-
O|X|O
-+-+-
X|O|X
```

[42]:
```
True
```

[43]:
```
import random

def first_move():

    flip= random.randint(0,1)
    if flip == '0':
        return 'Player1'
    else:
```

```
        return 'Player2'
```

[44]:
```
def space_check(board,position):

    return board[position]== ' '
```

[45]:
```
def full_board_check(board):
    #CHECKING IF BOARD HAS EMPTY SPACES

    for i in range(1,10):
        if space_check(board,i):
            #SINCE SPACE CHECK CHECKS FOR SPACES IF IT TURNS OUT TO BE TRUE␣
 ↪THEN THE BOARD IS NOT FULL HENCE WE RETURN FALSE
            return False

    return True
    #BUT AFTER GOING THROUGH THE LOOP BLANK POS ISNT ENCOUNTERED THEN WE RETURN␣
 ↪TRUE.
```

[46]:
```
display_board(test_board)
full_board_check(test_board)
#returns true because there are no blank spaces.
```

```
X|$|X
-+-+-
O|X|O
-+-+-
X|O|X
```

[46]:
```
True
```

[47]:
```
def player_choice(board):

    position= 0

    while (position not in [1,2,3,4,5,6,7,8,9] or not␣
 ↪space_check(board,position)):
        position= int(input("Choose a position: (1-9)"))

    return position
```

[48]:
```
def keep_playing():

    choice= input('Do you want to play again? Yes or No: ')

    return choice =='Yes'
```

[49]:
```
#WHILE LOOP TO KEEP THE GAME RUNNING
print('WELCOME TO TIC TAC TOE')
while True:
        #PLAY GAME

        ## SET EVERYTHING UP I.E. BOARD, PLAYER MARKER AND WHO GOES FIRST
        the_board=[' ']*10

        player1_marker,player2_marker= marker_choice()

        turn= first_move()

        print(turn +'will go first')

        play_game= input('Ready to play? y or n?')

        if play_game== 'y':
            game_on= True
        else:
            game_on= False

        while game_on:


            if turn== 'Player1':

                #display board
                display_board(the_board)

                #Choose a position
                position= player_choice(the_board)

                #Place marker in that position
                game_play(the_board,player1_marker,position)

                #check if player won
                if win_check(the_board,player1_marker):
                    display_board(the_board)
                    print('Player1 has won')
                    game_on = False

                #check if it is a tie
                else:
                    if full_board_check(the_board):
                            display_board(the_baord)
                            print('TIE game')
                            game_on = False
```

```
                    else:
                            turn = 'Player2'


            else:

                 #display board
                display_board(the_board)

                #Choose a position
                position= player_choice(the_board)

                #Place marker in that position
                game_play(the_board,player2_marker,position)

                #check if player won
                if win_check(the_board,player2_marker):
                        display_board(the_board)
                        print('Player2 has won')
                        game_on = False

                #check if it is a tie
                else:
                        if full_board_check(the_board):
                            display_board(the_baord)
                            print('TIE game')
                            game_on = False

                        else:
                            turn = 'Player1'



        if not keep_playing():
                break
#BREAK OUT OF THE GAME BASED ON replay()
```

```
X| |X
-+-+-
O| |X
-+-+-
O|O|O
Player2 has won
Do you want to play again? Yes or No: No
```

[ ]: