Milestone Project 1: Walkthrough Steps Workbook

Below is a set of steps for you to follow to try to create the Tic Tac Toe Milestone Project game!

Some suggested tools before you get started:

To take input from a user:

```
player1 = input("Please pick a marker 'X' or 'O'")
```

Note that input() takes in a string. If you need an integer value, use

```
position = int(input('Please enter a number'))
```

To clear the screen between moves:

```
from IPython.display import clear_output
clear_output()
```

Note that clear_output() will only work in jupyter. To clear the screen in other IDEs, consider:

```
print('\n'*100)
```

This scrolls the previous board up out of view. Now on to the program!

Milestone Project 1 Congratulations on making it to your first milestone! You've already learned a ton and are ready to work on a real project.

Your assignment: Create a Tic Tac Toe game. You are free to use any IDE you like.

Here are the requirements:

2 players should be able to play the game (both sitting at the same computer) The board should be printed out every time a player makes a move You should be able to accept input of the player position and then place a symbol on the board Feel free to use Google to help you figure anything out (but don't just Google "Tic Tac Toe in Python" otherwise you won't learn anything!) Keep in mind that this project can take anywhere between several hours to several days.

There are 4 Jupyter Notebooks related to this assignment:

This Assignment Notebook A "Walkthrough Steps Workbook" Notebook A "Complete Walkthrough Solution" Notebook An "Advanced Solution" Notebook I encourage you to just try to start the project on your own without referencing any of the notebooks. If you get stuck, check out the next lecture which is a text lecture with helpful hints and steps. If you're still stuck after that, then check out the Walkthrough Steps Workbook, which breaks up the project in steps for you to solve. Still stuck? Then check out the Complete Walkthrough Solution video for more help on approaching the project!

There are parts of this that will be a struggle...and that is good! I have complete faith that if you have made it this far through the course you have all the tools and knowledge to tackle this project. Remember, it's totally open book, so take your time, do a little research, and remember:

HAVE FUN!¶

Step 1: Write a function that can print out a board. Set up your board as a list, where each index 1-9 corresponds with a number on a number pad, so you get a 3 by 3 board representation.

In [6]:

```
from IPython.display import clear_output

def display_board(board):
    print("   |   |   ")
    print(" {a} | {b} | {c}  ".format(a=board[1],b=board[2],c=board[3]))
    print("   |   |   ")
    print("   |   |   ")
    print("-----------")
    print("   |   |   ")
    print(" {d} | {e} | {f}  ".format(d=board[4],e=board[5],f=board[6]))
    print("   |   |   ")
    print("-----------")
    print("   |   |   ")
    print(" {g} | {h} | {i}  ".format(g=board[7],h=board[8],i=board[9]))
    print("   |   |   ")
```

In [7]:

```
test_board = ['#','X','O','X','O','X','O','X','O','X']
board = ['#','1','2','3','4','5','6','7','8','9']
display_board(board)
```

```
   |   |
 1 | 2 | 3
   |   |
-----------
   |   |
 4 | 5 | 6
   |   |
-----------
   |   |
 7 | 8 | 9
   |   |
```

TEST Step 1: run your function on a test version of the board list, and make adjustments as necessary

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

Step 2: Write a function that can take in a player input and assign their marker as 'X' or 'O'. Think about using *while* loops to continually ask until you get a correct answer.

In [8]:

```
def player_input():
    marker = ''
    
    while not (marker == 'X' or marker == 'O'):
        marker = input('Player 1: Do you want to be X or O? ').upper()
    
    if marker == 'X':
        return ('X', 'O')
    else:
        return ('O', 'X')
```

TEST Step 2: run the function to make sure it returns the desired output

In [9]:

```
player_input()
```

```
Player 1: Do you want to be X or O? x
```

Out[9]:

```
('X', 'O')
```

Step 3: Write a function that takes in the board list object, a marker ('X' or 'O'), and a desired position (number 1-9) and assigns it to the board.

In [10]:

```
def place_marker(board, marker, position):
    #print(position)
    board[position] = marker
```

TEST Step 3: run the place marker function using test parameters and display the modified board

In [11]:

```
place_marker(test_board,'$',8)
display_board(test_board)
```

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

Step 4: Write a function that takes in a board and a mark (X or O) and then checks to see if that mark has won.

In [ ]:

```
def win_check(board,mark):

    return ((board[7] == mark and board[8] == mark and board[9] == mark) or # across the top
    (board[4] == mark and board[5] == mark and board[6] == mark) or # across the middle
    (board[1] == mark and board[2] == mark and board[3] == mark) or # across the bottom
    (board[7] == mark and board[4] == mark and board[1] == mark) or # down the middle
    (board[8] == mark and board[5] == mark and board[2] == mark) or # down the middle
    (board[9] == mark and board[6] == mark and board[3] == mark) or # down the right side
    (board[7] == mark and board[5] == mark and board[3] == mark) or # diagonal
    (board[9] == mark and board[5] == mark and board[1] == mark)) # diagonal
```

def win_check(board, mark): print(board) if board[1]== mark and board[2]== mark and board[3]== mark or board[4]== mark and board[5]== mark and board[6]== mark or board[7]== mark and board[8]== mark and board[9]== mark: print('You win') if board[1]== mark and board[4]== mark and board[7]== mark or board[2]== mark and board[5]== mark and board[8]== mark or board[3]== mark and board[6]== mark and board[9]== mark: print('You win') if board[1]== mark and board[5]== mark and board[9]== mark or board[3]== mark and board[5]== mark and board[7]== mark: print('You win')

TEST Step 4: run the win_check function against our test_board - it should return True

In [31]:

```
win_check(test_board,'X')
```

```
['#', 'X', 'O', 'X', 'O', 'X', 'O', 'X', '$', 'X']
You win
```

Step 5: Write a function that uses the random module to randomly decide which player goes first. You may want to lookup random.randint() Return a string of which player went first.

In [32]:

```
import random

def choose_first():
    print("Printing random number using random.random()")
    a = random.randint(1,2)
    if a == 1:
        #print("player1")
        return 'Player 1'
    if a == 2:
        #print("player2")
        return 'Player 2'
        return 'Player 2'
```

In [33]:

```
choose_first()
```

```
Printing random number using random.random()
```

Out[33]:

```
'Player 1'
```

Step 6: Write a function that returns a boolean indicating whether a space on the board is freely available.

def space_check(board, position): return board[position] == ' '

In [34]:

```
def space_check(board, position):
    print(board)
    character = ['x','X','o','O']
    #print(character[position])
    for ch in  character:
        #print(ch)
        if board[position] == ch:
            #print('yes')
            return False
            break
        else:
            #print('No')
            pass
    return True
```

Step 7: Write a function that checks if the board is full and returns a boolean value. True if full, False otherwise.

In [35]:

```
def full_board_check(board):
    for i in range(1,10):
        if space_check(board, i):
            return False
    return True
```

def full_board_check(board): true_full = True print(board)

```
#print(board[1].isdigit())
#print(len(board))
for ch in range(1,len(board)):
    #print(ch)
    #print(board[ch].isdigit())
    if board[ch].isdigit():
        #print('YES')
        true_full = False
return true_full
```

Step 8: Write a function that asks for a player's next position (as a number 1-9) and then uses the function from step 6 to check if it's a free position. If it is, then return the position for later use.

In [36]:

```
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 your next position: (1-9)  '))

    return position
```

Step 9: Write a function that asks the player if they want to play again and returns a boolean True if they do want to play again.

def replay(): return input('Do you want to play again? Enter Yes or No: ').lower().startswith('y')

In [37]:

```
def replay():
    cyle = True
    while cyle:
        print("Do you want to play again ?")
        re_game = input(" yes or no")
        if re_game == 'yes' or re_game == 'no':
            if re_game == 'yes':
                return True
            elif re_game == 'no':
                return False
            cyle = False
```

Step 10: Here comes the hard part! Use while loops and the functions you've made to run the game!

print('Welcome to Tic Tac Toe!') #while True: # Set the game up here #pass #while game_on: #Player 1 Turn # Player2's turn. #pass #if not replay(): #break

In [ ]:

```
print('Welcome to Tic Tac Toe!')

while True:
    # Reset the board
    theBoard = [' '] * 10
    player1_marker, player2_marker = player_input()
    turn = choose_first()
    print(turn + ' will go first.')

    play_game = input('Are you ready to play? Enter Yes or No.')

    if play_game.lower()[0] == 'y':
        game_on = True
    else:
        game_on = False

    while game_on:
        if turn == 'Player 1':
            # Player1's turn.

            display_board(theBoard)
            position = player_choice(theBoard)
            place_marker(theBoard, player1_marker, position)

            if win_check(theBoard, player1_marker):
                display_board(theBoard)
                print('Congratulations! You have won the game!')
                game_on = False
            else:
                if full_board_check(theBoard):
                    display_board(theBoard)
                    print('The game is a draw!')
                    break
                else:
                    turn = 'Player 2'

        else:
            # Player2's turn.

            display_board(theBoard)
            position = player_choice(theBoard)
            place_marker(theBoard, player2_marker, position)

            if win_check(theBoard, player2_marker):
                display_board(theBoard)
                print('Player 2 has won!')
                game_on = False
            else:
                if full_board_check(theBoard):
                    display_board(theBoard)
                    print('The game is a draw!')
                    break
                else:
                    turn = 'Player 1'

    if not replay():
        break
```

```
Welcome to Tic Tac Toe!
Player 1: Do you want to be X or O? x
Printing random number using random.random()
Player 1 will go first.
Are you ready to play? Enter Yes or No.yes
   |   |
   |   |
   |   |
-----------
   |   |
   |   |
   |   |
-----------
   |   |
   |   |
   |   |
Choose your next position: (1-9) 1
[' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
   |   |
 X |   |
   |   |
-----------
   |   |
   |   |
   |   |
-----------
   |   |
   |   |
   |   |
Choose your next position: (1-9) 2
[' ', 'X', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
   |   |
 X | O |
   |   |
-----------
   |   |
   |   |
   |   |
-----------
   |   |
   |   |
   |   |
Choose your next position: (1-9) 4
[' ', 'X', 'O', ' ', ' ', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', ' ', ' ', ' ', ' ', ' ']
   |   |
 X | O |
   |   |
-----------
   |   |
 X |   |
   |   |
-----------
   |   |
   |   |
   |   |
Choose your next position: (1-9) 5
[' ', 'X', 'O', ' ', 'X', ' ', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', ' ', ' ', ' ']
   |   |
 X | O |
   |   |
-----------
   |   |
 X | O |
   |   |
-----------
   |   |
   |   |
   |   |
Choose your next position: (1-9) 7
[' ', 'X', 'O', ' ', 'X', 'O', ' ', ' ', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', 'X', ' ', ' ']
You win
[' ', 'X', 'O', ' ', 'X', 'O', ' ', 'X', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', 'X', ' ', ' ']
[' ', 'X', 'O', ' ', 'X', 'O', ' ', 'X', ' ', ' ']
   |   |
 X | O |
   |   |
-----------
   |   |
 X | O |
   |   |
-----------
   |   |
 X |   |
   |   |
```

Good Job!

In [ ]: