Chapter 1: tic tac toe

Description:

Tic tac toe is a simple game between two people. It's played on a 3x3 grid and each player takes turns to move. The first player to move is X and then O. Once one of the players gets a line vertically, horizontally, or diagonally then they are the winner. In the case that all the spaces are filled and no lines are made then it's a draw.

Why?: This chapter will teach you the basic structures used in programming. If statements, while loops, dictionaries (or maps). Defining variables and functions and how to put it all together.

Resources:

- Python 3 (This is a coding language) https://www.python.org/downloads/
- A Code editor (I like visual studio code, you can download it for free) https://code.visualstudio.com/download
- Git (https://gitforwindows.org/) or GitHub desktop (https://desktop.github.com/)

To-do:
- Tic tac toe has two players (x starts first)
- Player needs to choose where on the board to put their next move
- Need to be able to check if there is a winner
- Need to be able to check if there is a draw
- Make a grid and print

Extra:
- Give player instructions
- Let users enter names

Starting steps:
- Download python 3

Note: you may need to fix some spacing or quotations. OR you could scroll to the bottom and take my prewritten code and just take this as a lesson.

Code:

Step 1: (mini step starting a function and a variable)
```
# In order to keep track of players we will need a variable to hold that info


def game():
    player = ‘X’
```

English:
Define function game
	Set variable ‘player’ to character ‘X’ to show who’s turn it will be

* # and // means comments in different coding languages, these are just notes for programmers who will be reading this afterward. It’s optional but kind of useful for WHEN you mess up. It’s not you but I’m assuming you’re human so that’ll happen

* The first line is creating a function, a function is a group of logic, it’s meant to do things related to one subject. So if you were boxing you would have multiple functions: defend, punch, fight(mixing both defend and punch)

* The next line is assigning a variable, it just puts information into a spot in your computer, right now we are saving a character but anything can be saved

* Tab spaces show where things below if something is behind something else horizontally it means that it belongs to the group of code above it. Here the player variable belongs to the game function. In other languages, they use brackets to show where the code goes

Ex. (Java - another coding language, you don’t need to think about this too hard just see how the brackets act like our tabs here)

```
Public void game(){
        Player =’X’
}
```

Step 2: Getting user input

```
# dictionary for board information

board = {1: ' ' , 2: ' ' , 3: ' ' ,
        4: ' ' , 5: ' ' , 6: ' ' ,
        7: ' ' , 8: ' ' , 9: ' ' }

# In order to keep track of players we will need a variable to hold that info

def game():
   player = 'X'
   count = 0

   while count < 10:
       print(“Player “+player+” it’s your turn”)
       move = input()
       if(board[move] == ' '):
           board[move] = player

           if(player=='X'):
               player = 'O'
           else:
               player = 'X'

       else:
           print("Spot taken, pick another")
           continue
       count = count + 1

game()
```

English:

```
Create a map with key values of numbers to represent locations on the board and empty strings to show untaken positions

Define function game
    Set variable 'player' to character 'X' to show who's turn it will be
    Set variable count to zero to show how many turns have passed

    While the count is less than 10 moves continue the game
        Print a sentence to show which player's turn it is
        Set variable move with user's movement on the board

        If the position the user input is not taken on the board
            Then set that spot on the board as X or O for the player

            If a player is x
                Then set player as O to let the next player go
            Else if the player is not x
                Then set player as X to let the next player go
        Else if spot is already taken
            Send a message that spot is taken
            Continue to the next cycle and don't count this one to repeat turn
        Increase turn count by 1

Run game function
```

* At the top is a dictionary (sometimes called a map) is a key-value grouping of info. Just like keys, different numbers here return different information. Currently, all the info is blank but once the user starts playing there will be X's or O's in those positions so using a number from 1-9 will

return a value of some kind. It’s just like a group of variables. Or A better example the numbers 1-9 are the houses to friends and you call one of your friends based on which number you use

* While means until the condition is met the actions below will loop continuously. This structure can run forever so make sure you make the loop close. ‘<’ this symbol means less so the whole thing says “while the count is less than 10 do something”

* continue is a special word that allows you to leave the loop for a moment to skip over a segment of code.
* The line "move = input()” means we are going to wait for user input and put it into a variable

* if/else statements, these are exactly what they sound like if something is true then do something or ELSE do another action

* At the very bottom we have “game()”, this is to call our function. This allows us to run our stuff now. Try putting this code in a file called “tictactoe.py” then open a terminal (A place where you can directly talk to your computer. Since I don’t know your computer you might need to do some googling here. Then you run “python tictactoe.py” which should start your stuff up for you

Step 3:

```
# dictionary for board information
board = {1: ' ' , 2: ' ' , 3: ' ' ,
       4: ' ' , 5: ' ' , 6: ' ' ,
       7: ' ' , 8: ' ' , 9: ' ' }

def instructions():
    print('Hello this is tic tac toe, the game\n')
    print('two players will take turns placing their pieces down\n')
    print('Player 1 is x and goes first\n')
    print('First to get a line wins\n')
    printInstructionBoard()

def printInstructionBoard():
    print('Here is the board\n')
    print('1|2|3')
    print('-+-+-')
    print('4|5|6')
    print('-+-+-')
    print('7|8|9')

# In order to keep track of players we will need a variable to hold that
info
```

```python
def game(player1,player2):
    player = 'X'
    count = 0
    try:
        while count <= 9:
            print(f"Player {player} it’s your turn")
            move = int(input())
            if(move >9 or move <1):
                print("Invalid number")
                continue
            if(board[move] == ' '):
                board[move] = player
                if(player=='X'):
                    player = 'O'
                else:
                    player = 'X'
            else:
                print("Spot taken, pick another")
                continue
            count = count + 1

            # the minimum count
            if(count >= 5):
                # win scenarios (check all values around current move
                # horizontal
                # last value in row
                if(move%3 ==0):
                    if(board[move]==board[move-1]==board[move-2]!=' '):
                        winner(board[move])
                        break

                #middle
                if( move%3==1):
                    if(board[move]==board[move+1]==board[move+2]!=' '):
                        winner(board[move])
                        break

                #first
                if(move%3 == 2):
                    if(board[move]==board[move+1]==board[move-1]!=' '):
```

```
                winner(board[move])
                break

        # vertical
        # last column
        if(int((move-1)/3)==2):
            if(board[move]==board[move-3]==board[move-6]!=' '):
                winner(board[move])
                break

        #middle column
        if(int((move-1)/3)==1):
            if(board[move]==board[move+3]==board[move-3]!=' '):
                winner(board[move])
                break

        #first column
        if(int((move-1)/3)==0):
            if(board[move]==board[move+3]==board[move+6]!=' '):
                winner(board[move])
                break

        # diagonals
        if(move == 1 or move == 5 or move == 9):
            if(board[1]==board[5]==board[9]!=' '):
                winner(board[move])
                break

        if(move ==3 or move == 5 or move == 7):
            if(board[3]==board[5]==board[7]!=' '):
                winner(board[move])
                break

    # tie scenario
    if (count ==9):
        printBoard()
        print("TIE")
        break

    printBoard()
    except:
        print("An exception occurred")

# save results and print
def winner(player):
    print("The winner is "+player)

def printBoard():
    print(board[1]+"|"+board[2]+"|"+board[3])
    print("-----")
    print(board[4]+"|"+board[5]+"|"+board[6])
    print("-----")
    print(board[7]+"|"+board[8]+"|"+board[9])

# input player
def inputPlayers():
    print("Enter username of player 1: ")
    player1 = input()

    print("Enter username of player 2: ")
    player2 = input()

    return (player1,player2)

instructions()
a,b = inputPlayers()
game(a,b)
```

English:

Create a map with key values of numbers to represent locations on the board and empty strings to show untaken positions

Define instructions to describe how the game works

Define a function to print board and show locations on the grid

```
Define function game
        Set variable ‘player’ to character ‘X’ to show who’s turn it will be
        Set variable count to zero to show how many turns have passed
```

```
While count is less than 10 moves continue the game
    Print a sentence to show which player's turn it is
    Set variable move with user's move on the board

    If the position the user input is not taken on the board
        Then set that spot on the board as X or O for the player

        If player is x
            Then set player as O to let the next player go
        Else if player is not x
            Then set player as X to let the next player go
    Else if spot is already taken
        Send message that spot is taken
        Continue to next cycle and don't count this one to repeat turn
    Increase turn count by 1
    If count is great or equal to five
        If the move modulus 3 is 0
            If board values at position move, move-1, move-2 all equal and have been filled
                Winner
        If move modulus 3 is 1
            If board values at position move, move+1, move+2 all equals and have been filled
                Winner
        If move modulus 3 is 2
            If board values at position move, move+1, move-1 all equals and have been filled
                Winner
        If move minus 1 divided by 3 (rounded by turning it to an integer) equals 2
            If board values at position move, move-3, move-6 all equals and have been filled
                Winner
        If move minus 1 divided by 3 (rounded by turning it to an integer) equals 1
            If board values at position move, move-3, move+3 all equals and have been filled
                Winner
        If move minus 1 divided by 3 (rounded by turning it to an integer) equals 0
            If board values at position move, move+3, move+6 all equals and have been filled
                Winner
        If move equals 1, 5 or 9
            If board values at position 1, 5, 9 all equals and have been filled
                Winner
        If move equals 3, 5, or 7
            If board values at position 3, 5, 7 all equals and have been filled
                Winner

    If count equals 9
        Print board
        Print message tie
        End loop
```

                    Print message

Run game function

* A lot more stuff to look at now but don't worry it's all normal stuff. At the top you can see some instructions that can be called for our user. All of the stuff added should look familiar ish, I wanted to rip the bandaid off so you can run this and start playing with the code right away

* At the bottom of our code, we are now adding situations where the player will win or tie

* Why count 6? That's the minimum amount of turns needed for someone to win 3 Xs and 2 Os

* ELIF means else if, this adds an extra if in a situation. For instance, Do you want an apple, peach, or blueberry pie? We can describe this as If(apple), Elif(peach) else: blueberry goes here

* The horizontal check you might notice a symbol "%" there, this means if we kept subtracting the value to the right what number would be leftover? So in the case of 2 %3 (2- 3(0)), it's 2 and in the case of 8%3 it is also 2. (8 -3 (2)= 2)

* The horizontal check is done this way to see which column we have to view the rest of the rows.

* The vertical check removes a value to make sure the line goes beneath the threshold. So when we use "/" this means to divide, in python when you divide a number it rounds down to the next whole number so 2/3 becomes 0 and 9/3 becomes 3. We do this to find out which row it is in to check the following columns.

* The final possible question: why didn't I just specify all the positions instead of what I did? Just in case you wanted to make this bigger it'd be easier to add the logic to it.

* At count ==10 then all moves have been made and there was no winner. It's a tie

Extra:
For extra stuff look here:
    https://github.com/DownRamp/Game/blob/master/tictactoe.py

* If git is already setup just run:

```
git clone https://github.com/DownRamp/TicTacToe.git
```

****

THIS IS THE IMPORTANT PART PLEASE DON'T SKIP

****

Next steps:
    Make it your own:

- Add a scoreboard
- Make a visual with pygame (You will learn this later)
- Big tic tac toe? 9x9 grid
- Allow overlapping on the grid (If there’s an X there you can put an O over it and take it)
- Invent a new rule