TIC TAC TOE

def tic\_tac\_toe():

board = [1, 2, 3, 4, 5, 6, 7, 8, 9]

end = False

win\_commbinations = ((0, 1, 2), (3, 4, 5), (6, 7, 8), (0, 3, 6), (1, 4, 7), (2, 5, 8), (0, 4, 8), (2, 4, 6))

def draw():

print(board[0], board[1], board[2])

print(board[3], board[4], board[5])

print(board[6], board[7], board[8])

print()

def p1():

n = choose\_number()

if board[n] == "X" or board[n] == "O":

print("\nYou can't go there. Try again")

p1()

else:

board[n] = "X"

def p2():

n = choose\_number()

if board[n] == "X" or board[n] == "O":

print("\nYou can't go there. Try again")

p2()

else:

board[n] = "O"

def choose\_number():

while True:

while True:

a = input()

try:

a = int(a)

a -= 1

if a in range(0, 9):

return a

else:

print("\nThat's not on the board. Try again")

continue

except ValueError:

print("\nThat's not a number. Try again")

continue

def check\_board():

count = 0

for a in win\_commbinations:

if board[a[0]] == board[a[1]] == board[a[2]] == "X":

print("Gurnoor Wins!\n")

print("Congratulations!\n")

return True

if board[a[0]] == board[a[1]] == board[a[2]] == "O":

print("Player 2 Wins!\n")

print("Congratulations!\n")

return True

for a in range(9):

if board[a] == "X" or board[a] == "O":

count += 1

if count == 9:

print("The game ends in a Tie\n")

return True

while not end:

draw()

end = check\_board()

if end == True:

break

print("Gurnoor choose where to place a X")

p1()

print()

draw()

end = check\_board()

if end == True:

break

print("Player 2 choose where to place a O")

p2()

print()

if input("Play again (y/n)\n") == "y":

print()

tic\_tac\_toe()

TIC TAC TOE AI

# Tic Tac Toe

import random

def drawBoard(board):

print(' | |')

print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])

print(' | |')

print('-----------')

print(' | |')

print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])

print(' | |')

print('-----------')

print(' | |')

print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])

print(' | |')

def inputPlayerLetter():

letter = ''

while not (letter == 'X' or letter == 'O'):

print('Do you want to be X or O?')

letter = input().upper()

if letter == 'X':

return ['X', 'O']

else:

return ['O', 'X']

def whoGoesFirst():

if random.randint(0, 1) == 0:

return 'computer'

else:

return 'player'

def playAgain():

print('Do you want to play again? (yes or no)')

return input().lower().startswith('y')

def makeMove(board, letter, move):

board[move] = letter

def isWinner(bo, le):

return ((bo[7] == le and bo[8] == le and bo[9] == le) or # across the top

(bo[4] == le and bo[5] == le and bo[6] == le) or # across the middle

(bo[1] == le and bo[2] == le and bo[3] == le) or # across the bottom

(bo[7] == le and bo[4] == le and bo[1] == le) or # down the left side

(bo[8] == le and bo[5] == le and bo[2] == le) or # down the middle

(bo[9] == le and bo[6] == le and bo[3] == le) or # down the right side

(bo[7] == le and bo[5] == le and bo[3] == le) or # diagonal

(bo[9] == le and bo[5] == le and bo[1] == le)) # diagonal

def getBoardCopy(board):

dupeBoard = []

for i in board:

dupeBoard.append(i)

return dupeBoard

def isSpaceFree(board, move):

return board[move] == ' '

def getPlayerMove(board):

move = ' '

while move not in '1 2 3 4 5 6 7 8 9'.split() or not isSpaceFree(board, int(move)):

print('What is your next move? (1-9)')

move = input()

return int(move)

def chooseRandomMoveFromList(board, movesList):

possibleMoves = []

for i in movesList:

if isSpaceFree(board, i):

possibleMoves.append(i)

if len(possibleMoves) != 0:

return random.choice(possibleMoves)

else:

return None

def getComputerMove(board, computerLetter):

# Given a board and the computer's letter, determine where to move and return that move.

if computerLetter == 'X':

playerLetter = 'O'

else:

playerLetter = 'X'

# Here is our algorithm for our Tic Tac Toe AI:

# First, check if we can win in the next move

for i in range(1, 10):

copy = getBoardCopy(board)

if isSpaceFree(copy, i):

makeMove(copy, computerLetter, i)

if isWinner(copy, computerLetter):

return i

# Check if the player could win on his next move, and block them.

for i in range(1, 10):

copy = getBoardCopy(board)

if isSpaceFree(copy, i):

makeMove(copy, playerLetter, i)

if isWinner(copy, playerLetter):

return i

# Try to take one of the corners, if they are free.

move = chooseRandomMoveFromList(board, [1, 3, 7, 9])

if move != None:

return move

# Try to take the center, if it is free.

if isSpaceFree(board, 5):

return 5

# Move on one of the sides.

return chooseRandomMoveFromList(board, [2, 4, 6, 8])

def isBoardFull(board):

# Return True if every space on the board has been taken. Otherwise return False.

for i in range(1, 10):

if isSpaceFree(board, i):

return False

return True

print('Welcome to Tic Tac Toe!')

while True:

# Reset the board

theBoard = [' '] \* 10

playerLetter, computerLetter = inputPlayerLetter()

turn = whoGoesFirst()

print('The ' + turn + ' will go first.')

gameIsPlaying = True

while gameIsPlaying:

if turn == 'player':

# Player's turn.

drawBoard(theBoard)

move = getPlayerMove(theBoard)

makeMove(theBoard, playerLetter, move)

if isWinner(theBoard, playerLetter):

drawBoard(theBoard)

print('Hooray! You have won the game!')

gameIsPlaying = False

else:

if isBoardFull(theBoard):

drawBoard(theBoard)

print('The game is a tie!')

break

else:

turn = 'computer'

else:

# Computer's turn.

move = getComputerMove(theBoard, computerLetter)

makeMove(theBoard, computerLetter, move)

if isWinner(theBoard, computerLetter):

drawBoard(theBoard)

print('The computer has beaten you! You lose.')

gameIsPlaying = False

else:

if isBoardFull(theBoard):

drawBoard(theBoard)

print('The game is a tie!')

break

else:

turn = 'player'

if not playAgain():

break