import random

import os

# for printing the grid after each move as well as before the game

def draw\_grid(grid):

print(" || || ")

print(" " + grid[1] + " || " + grid[2] + " || " + grid[3])

print(" || || ")

print("=============")

print(" || || ")

print(" " + grid[4] + " || " + grid[5] + " || " + grid[6])

print(" || || ")

print("=============")

print(" || || ")

print(" " + grid[7] + " || " + grid[8] + " || " + grid[9])

print(" || || ")

# for receiving the input for the grid

def input\_grid():

letter = ''

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

letter = input("What Would You Like To Play With 'X' or 'O' : ").upper()

if letter == 'X':

return ['X', 'O']

else:

return ['O', 'X']

# randomly deciding who plays first the player or the computer

def who\_plays\_first():

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

return 'computer'

else:

return 'player'

# yes or no function asking the user whether he wants to play again or not

def yes\_or\_no():

return input("Do You Want To Play Again? (yes or no) : ").lower().startswith('y')

# assigning the letter to the grid

def m\_move(grid, letter, move):

grid[move] = letter

# returning true for all the winning cases

def check\_winner(grid, l):

return ((grid[1] == l and grid[2] == l and grid[3] == l) or # horizontal top

(grid[4] == l and grid[5] == l and grid[6] == l) or # horizontal middle

(grid[7] == l and grid[8] == l and grid[9] == l) or # horizontal bottom

(grid[1] == l and grid[4] == l and grid[7] == l) or # vertical left

(grid[2] == l and grid[5] == l and grid[8] == l) or # vertical middle

(grid[3] == l and grid[6] == l and grid[9] == l) or # vertical right

(grid[1] == l and grid[5] == l and grid[9] == l) or # diagonal 1-5-9

(grid[3] == l and grid[5] == l and grid[7] == l)) # diagonal 3-5-7

# duplicating the grid list and returning the duplicate

def grid\_copy(grid):

d\_grid = []

for i in grid:

d\_grid.append(i)

return d\_grid

# return true if the slot in the grid is empty

def check\_space\_free(grid, move):

return grid[move] == ' '

# the user entering his move

def player\_move(grid):

move = ' '

while move not in '1 2 3 4 5 6 7 8 9'.split() or not check\_space\_free(grid, int(move)):

move = input("Next Move Please (1 - 9) : ")

return int(move)

# to return a valid move from the grid list if not None

def selecting\_random\_moves\_from\_list(grid, move\_list):

possible\_move = []

for i in move\_list:

if check\_space\_free(grid, i):

possible\_move.append(i)

if len(possible\_move) != 0:

return random.choice(possible\_move)

else:

return None

def computer\_move(grid, computer\_letter):

if computer\_letter == 'X':

player\_letter = 'O'

else:

player\_letter = 'X'

# checking if we can win in the next move

for i in range(1, 10):

copy = grid\_copy(grid)

if check\_space\_free(copy, i):

m\_move(copy, computer\_letter, i)

if check\_winner(copy, computer\_letter):

return i

# checking if the player can win in the next move

for i in range(1, 10):

copy = grid\_copy(grid)

if check\_space\_free(copy, i):

m\_move(copy, player\_letter, i)

if check\_winner(copy, player\_letter):

return i

# next try to take one of the corners if available

move = selecting\_random\_moves\_from\_list(grid, [1, 3, 7, 9])

if move != None:

return move

# try to take the center if available

if check\_space\_free(grid, 5):

return 5

# next is moving on one of the sides that are available

return selecting\_random\_moves\_from\_list(grid, [2, 4, 6, 8])

# if board is full

def check\_board\_full(grid):

for i in range(1, 10):

if check\_space\_free(grid, i):

return False

return True

while True:

os.system('cls')

print("\t\tTicTacToe.exe")

the\_grid = [' '] \* 10

player\_letter, computer\_letter = input\_grid()

turn = who\_plays\_first()

print(turn + " Plays First")

continue\_game = True

while continue\_game:

# players turn

if turn == 'player':

draw\_grid(the\_grid)

move = player\_move(the\_grid)

m\_move(the\_grid, player\_letter, move)

if check\_winner(the\_grid, player\_letter):

draw\_grid(the\_grid)

print(" The Player (You) Have Won The Game")

continue\_game = False

else:

if check\_board\_full(the\_grid):

draw\_grid(the\_grid)

print("The Game Ends In A Tie")

break

else:

turn = 'computer'

else:

# computers turn

move = computer\_move(the\_grid, computer\_letter)

m\_move(the\_grid, computer\_letter, move)

if check\_winner(the\_grid, computer\_letter):

draw\_grid(the\_grid)

print(" The Computer Have Won The Game")

continue\_game = False

else:

if check\_board\_full(the\_grid):

draw\_grid(the\_grid)

print("The Game Ends In A Tie")

break

else:

turn = 'player'

if not yes\_or\_no():

break