PERL PROJECT- OPEN SOURCE LAB- RIVA PARIKH- 13BIT045

Following is the code for tic tac toe game made in python.

import random  
  
def drawBoard(board):  
    # This function prints out the board that it was passed.  
  
    # "board" is a list of 10 strings representing the board (ignore index 0)  
    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():  
    # Lets the player type which letter they want to be.  
    # Returns a list with the player's letter as the first item, and the computer's letter as the second.  
    letter = ''  
    while not (letter == 'X' or letter == 'O'):  
        print('Do you want to be X or O?')  
        letter = input().upper()  
  
    # the first element in the tuple is the player's letter, the second is the computer's letter.  
    if letter == 'X':  
        return ['X', 'O']  
    else:  
        return ['O', 'X']  
  
def whoGoesFirst():  
    # Randomly choose the player who goes first.  
    if random.randint(0, 1) == 0:  
        return 'computer'  
    else:  
        return 'player'  
  
def playAgain():  
    # This function returns True if the player wants to play again, otherwise it returns False.  
    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):  
    # Given a board and a player's letter, this function returns True if that player has won.  
    # We use bo instead of board and le instead of letter so we don't have to type as much.  
    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):  
    # Make a duplicate of the board list and return it the duplicate.  
    dupeBoard = []  
  
    for i in board:  
        dupeBoard.append(i)  
  
    return dupeBoard  
  
def isSpaceFree(board, move):  
    # Return true if the passed move is free on the passed board.  
    return board[move] == ' '  
  
def getPlayerMove(board):  
    # Let the player type in his move.  
    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):  
    # Returns a valid move from the passed list on the passed board.  
    # Returns None if there is no valid move.  
    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