Alternate Project 4: Tic-Tac-Toe

Modified by Brian Weinfeld

Using loops, variables and functions in Python, you will create a Tic-Tac-Toe game for two human players.

Overview

Tic-Tac-Toe is a game in which one player draws X's and another player draws O's inside a set of nine squares and each player tries to be the first to fill a row, column, or diagonal of squares with either X's or O's.

Behavior

```
Welcome to Tic-Tac-Toe!
'X' what is your name: Alice
'O' what is your name: Bob
Game Start
 | |
 | |
 | |

Alice what row is your move: 1
Alice what col is your move: 1
 | |
 |X|
 | |

Bob what row is your move: 0
Bob what col is your move: 0
O| |
 |X|
 | |

Alice what row is your move: 2
Alice what col is your move: 0
O| |
 |X|
X| |

Bob what row is your move: 2
Bob what col is your move: 0
Invalid move. Try again.

Bob what row is your move: 4
Bob what col is your move: 4
Invalid move. Try again.

Bob what row is your move: 0
Bob what col is your move: 1
O|O|
```

```
 |X|
X| |

Alice what row is your move: 0
Alice what col is your move: 2
O|O|X
 |X|
X| |

X wins!

Would you like to play again? no
Goodbye
```

Implementation Details

The gameboard should be stored in a list of lists. Each of the internal lists represents one row of the gameboard. If the element in the list is the empty string (the empty string is ''), then the spot is unoccupied. If the element in the list is an 'X' or an 'O', then it is occupied by that player.

Begin by completing the below functions. Do your best to ensure that these functions work properly before moving onto the rest of the project.

```
def print_gameboard(gameboard):
  # finish
```

- print_gameboard takes the gameboard as its only parameter. This function nicely prints the gameboard.

```
def check_winner(gameboard):
  # finish
```

- check_winner uses the gameboard to check whether or not a player has won. Return 'X' or 'O' if either of those players have won the game and None if neither player has won.

```
def check_tied(gameboard):
  # finish
```

- check_tied is used to determine whether the game is tied (all 9 spaces are filled but no one has won). Return True if the game is tied and False if it has not.

```
def make_move(gameboard, name, symbol):
  # finish
```

- make a move is used to filled the gameboard with the current player's move. The name parameter is a string representing the current player's name and symbol is a string representing their game symbol ('X' or 'O'). Prompt the user by their name for a row and column and place their symbol in the indicated spot. If they select an illegal spot, prompt them for a new selection.

After completing and testing the above functions, complete the following steps to create the main game loop.

- Prompt the 'X' player and 'O' player for their names. You can use these names to prompt each player for their move.
- The players will take turns making moves utilizing the make_move function
- At the end of each player's turn the program will:
    - check if that player has won using the check_winner function.
    - print the updated game board using the print_gameboard function.
- If there are no more spots open and nobody has won the game, the program will print 'Tie game!'. Check this using the check_tied function.
- After the game is over, prompt the players if they would like to play again or quit.

Challenge

- Going first is a big advantage in this game. Modify your game loop so that if the users indicate they want to play again, the players swap who goes first. The players should keep their symbol ('X' or 'O') but the symbol that goes first should swap. Every time the players replay the game, the player who goes first should swap.

- Instead of a single game, some players may want to play a best-of-5 or best-of-7 series. In such an event, players repeatedly play the game until one player has won a majority of the games in the series. Before the first game is played, prompt the players for the best-of series they want to play. Remember you don't need to play unnecessary games. If 'X' wins the first 3 games in a best-of-5, the last two games don't need to be played. You should still prompt the player's to see if they want to play again (that is, play a new series of games).

Super Challenge

The super challenge will require knowledge that has not been taught yet. You will need to do additional research on your own. Good luck!

- Modify the gameloop so that instead of two human players, either (or both) of the players can be computers. The computer's names can be some variation on "Computer 1". Modify the make_move function so that computer can make an automatic selection. To begin, simply have the computer randomly select an available empty space for their move.

- Continue to build on the above challenge by giving the computer some basic game logic. If the other player is about to win (ie: they have two in a row), then the computer should always select the blocking move. What kind of logic can you give the computer for its turn if it doesn't have to make a blocking move?