Extra Credit – Portfolio Project Idea
Casey Bellew

Idea – Text Based Backgammon

For this project you will write a class called Backgammon that allows two people to play text-based Backgammon (https://en.wikipedia.org/wiki/Backgammon). Backgammon is a two-player game of contrary movement in which each player has fifteen pieces that move along twenty-four 'points' according to the roll of two dice. The objective of the game is to move the fifteen pieces around the board and be first to bear off, i.e., remove them from the board. Once a player removes all their pieces from the board a winner is determined.

**Rules: **

* The game is played on an 5x25 board.
* Players take turns rolling two 6-sided dice and choosing to move an existing piece or a new piece currently in the “bank.”
* Each player will get 2 randomly generated numbers (1-6) to move their pieces, where they must choose which piece to move. They must choose an existing piece on the board or a new piece not yet on the board.
* Each player starts with 15 pieces in their counter that is moved throughout the 24 “columns” to be put into the 25th column to complete removing of the piece off the board.
* Each time a player puts a new piece on the board it reduces the player’s “start” counter. Each time a player successfully removes a piece from the board it increases the player’s “end” counter.
* Each column can only contain 5 “pieces” in it, so there must be a check to verify that the move the player is wanting to make based on the two 6-sided being rolled is a valid move. If the move is valid, it will make the move followed by a game check.
* The game check will verify the number of pieces within the player’s end counter. The first player to get an end counter to equal 15 is the winner.

**Game Board: **
The game board is represented by a 7x27 grid:
(for display purposes and ease of movement the board can be split in half, board_1/board_2; this would modify each board to be 7x15 so that there is the outline edge. The general rules for backgammon has the player go in a horseshoe pattern which would make the board_1 to go in the clockwise direction that would then continue into board_2 starting at the “end” and progressing to the left. This however I believe would make the programming significantly harder for this level of experience.)
* Edge: * (star)
* Black piece: X
* Red piece: O
* Empty space: “. “ (dot)

**Player: **
The Player class represents a player in the game. It contains the following information:
* Player name (string)
* Piece color (string): Either "black" or "red"

**Backgammon: **
The Backgammon object represents the game as played. It contains information about the players and the board. Name the board as "board", so we could access each position value on the board by **self._board[row][column]**. It must include those methods (but may have more):

* print_board(self): print out the current board, including the boundaries
* create_player(self, player_name, color): creates a player object with the given name and color ("black" or "red") and adds it to the player list
* return_winner(self): returns "Winner is red player: player’s name" when red player wins the game, and returns “Winner is black player: player’s name" when black player wins the game
* return_player_roll(self, color): returns a list of the two randomly generated numbers representing the two 6-sided dice (dice_roll1, dice_roll2).
* make_player_move_one(self, color, piece_position, dice_roll1): moves a piece of the specified color from piece position to the new column based on the dice roll they are choosing.
* make_player_move_two(self, color, piece_position, dice_roll2): moves a piece of the specified color from piece position to the new column based on the dice roll they are choosing.
* play_game(self, color, move_piece_one, move_piece_two): attempts to make a move for the player with the given color with the two moves. The player can move the same piece with both dice rolls but needs to be declared as a second move.
* return "Invalid move” and print out this message "Here are the movable pieces:" followed by a list of possible pieces that can be moved including the number of starting pieces still left for the player to use. A check includes that the columns do not contain more than 5 pieces at a time. If no valid moves exist, then the returned list is empty. If the position is valid, the function should make that move and update the board. If the game is ended at that point, the function should print "Game is ended red piece: number black piece: number" and call the return_winner method.

Your python file must be named **Backgammon.py**