Title: Battleship Game Algorithm

Author: Shaun Clarke

Goal: Mimics the game battleship, where the users plays until they sink all ships.

Steps:

1. Import *random* module.
2. Instantiate *grid_size* global variable to determine the board size.
3. Instantiate *grid* global variable to hold list of lists which represents myboard.
4. Instantiate *num_of_ships* global variable for ships on the board.
5. Define a function *drawBoard(myboard):*
    a. This function prints the battlefield grid to the console.
    b. This function takes *myboard* as a parameter.
    c. This function returns nothing.
6. Define a function *generate_ship_coordinates():*
    a. This function generates 5 random ship coordinates to later add to the board.
    b. This function accepts no parameters.
    c. It returns a list of 5 randomly generated ship coordinates.
7. Define a function *setupBoard(myboard, ship_locations):*
    a. This function set's up the board with the ship icons and dots for the water.
    b. This function accepts 2 parameters myboard and ship_locations.
    c. This function returns nothing.
8. Define a function *get_user_coordinates(location):*
    a. This function gets the user inputs for the grid coordinates.
    b. This function takes one parameter, *location*
    c. This function returns:
        i. The user input if it meets criteria
        ii. None if the input is not an int
        iii. 'out of range' if the int is not between 0 and 9
9. Define a function *hitorMiss(myboard, row, col):*
    a. This function checks if the users input was a hit or a miss and updates the board.
    b. This function accepts 3 parameters:
        i. *myboard*, which is the *grid* variable
        ii. *row*, which is the output from *get_user_coordinates("row")*
        iii. *column*, which is the output from *get_user_coordinates("column")*
    c. This function returns True or False:
        i. True if the combination of row and column coordinates was a hit.

1. Update board to reflect hit

ii. False if the combination of row and column coordinates was a miss.

1. Update board to reflect miss

10. Define a function *main(myboard):*
    a. This function ties everything together and calls the program.
    b. This function accepts one input, *myboard*.
    c. Call *generate_ship_coordinates()* and save output
    d. Call *setupboard(grid, ship_locations)*
    e. Instantiate *ship_sunk_counter* with zero
    f. While loop that runs as long as *ship_sunk_counter is less than zero*.
        i. Call *drawBoard(myboard)*.
        ii. *Call get_user_coordinates("row") and save the output*.
        iii. *Call get_user_coordinates("column") and save the output*.
        iv. *Call hitormiss(myboard, row, col) and save output*
        v. *If a ship was hit and ship_sunk_counter is equal to 4*
            1. *Increment ship_sunk_counter by 1*
            2. Call drawboard(myboard).
            3. print game over
            4. Break loop to end the game.
        vi. If a ship was hit
            1. *Increment ship_sunk_counter by 1*
            2. *Calculate how many ships remaining on board*
            3. *Print status message that includes the number of remaining ships*.
            4. *Continue loop*
        vii. *If no ship was hit*
            1. *Print status message*
            2. *Continue loop*
        viii. *When all ships have been hit, GAME OVER!*
11. *Call the main function to run the program if the script is executed directly.*