**Title:** Battleship Game Algorithm

**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):*
   1. This function prints the battlefield grid to the console.
   2. This function takes *myboard* as a parameter.
   3. This function returns nothing.
6. Define a function *generate\_ship\_coordinates():*
   1. This function generates 5 random ship coordinates to later add to the board.
   2. This function accepts no parameters.
   3. It returns a list of 5 randomly generated ship coordinates.
7. Define a function *setupBoard(myboard, ship\_locations):*
   1. This function set's up the board with the ship icons and dots for the water.
   2. This function accepts 2 parameters myboard and ship\_locations.
   3. This function returns nothing.
8. Define a function *get\_user\_coordinates(location):*
   1. This function gets the user inputs for the grid coordinates.
   2. This function takes one parameter, *location*
   3. This function returns:
      1. The user input if it meets criteria
      2. None if the input is not an int
      3. ‘out of range’ if the int is not between 0 and 9
9. Define a function *hitorMiss(myboard, row, col):*
   1. This function checks if the users input was a hit or a miss and updates the board.
   2. This function accepts 3 parameters:
      1. *myboard*, which is the *grid* variable
      2. *row*, which is the output from *get\_user\_coordinates(“row”)*
      3. *column*, which is the output from *get\_user\_coordinates(“column”)*
   3. This function returns True or False:
      1. True if the combination of row and column coordinates was a hit.
         1. Update board to reflect hit
      2. False if the combination of row and column coordinates was a miss.
         1. Update board to reflect miss
10. Define a function *main(myboard):*
    1. This function ties everything together and calls the program.
    2. This function accepts one input, *myboard.*
    3. Call *generate\_ship\_coordinates()* and save output
    4. Call *setupboard(grid, ship\_locations)*
    5. Instantiate *ship\_sunk\_counter* with zero
    6. While loop that runs as long as *ship\_sunk\_counter is less than zero.*
       1. Call *drawBoard(myboard).*
       2. *Call get\_user\_coordinates(“row”) and save the output.*
       3. *Call get\_user\_coordinates(“column”) and save the output.*
       4. *Call hitormiss(myboard, row, col) and save output*
       5. *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.
       6. 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*
       7. *If no ship was hit*
          1. *Print status message*
          2. *Continue loop*
       8. *When all ships have been hit, GAME OVER!*
11. *Call the main function to run the program if the script is executed directly.*