For your final project you will be implementing a one player version of battleship that allows the user to play against an AI. Here is a link to a version of the game you can try playing online. The differences between this version and ours, is that in our version we will announce when a ship is destroyed, our board size is configurable, and the number and size of the ships is variable.

1. Setup
    1. Your program should ask the user for the following information in this order. You must make sure to validate that the input is correct before accepting it. If it isn't your program should request the user to enter the input again.
    2. The seed: An integer used to seed the random number generator.
        1. To seed the random number generator do `random.seed(seed)`
            1. Seed needs to be an integer
    3. The width of the board: An integer greater than 0
    4. The height of the board: An integer greater than 0
    5. The name of the file containing the user's ship placements
        1. The file is structured as follows
            1. Symbol_for_ship1 row1_index column1_index row2_index column2_index
            2. Symbol_for_ship2 row1_index column1_index row2_index column2_index
            3. Symbol_for_ship3 row1_index column1_index row2_index column2_index
            4. …
        2. All ship placement files will be structured correctly but you must verify that
            1. The user did not choose either x,X, o, O, or * as those are symbols reserved for hits, misses, and, unfired upon spaces
            2. The user did not choose the same symbol for multiple ships
            3. The user placed all of their ships on the board
            4. The user did not try to place their ships diagonally
            5. The user did not place their ships on top of each other
        3. If the user violated any of the above constraints, an appropriate message should be displayed to the screen and the program should terminate
            1. You will find `sys.exit` helpful for terminating the program. Make sure to put a 0 in it for the status though or else tester will think your program crashed.
    6. The number of the AI they would like to play against.
        1. 1 is for the Random AI
        2. 2 is for the "Smart" AI
        3. 3 is for the Cheating AI
        4. AI's will be explained in another section
    7. After receiving the setup information, your program should
        1. Seed the random number generator with the provided seed
        2. Construct the user's board
        3. Construct the AI's board
            1. The AI will have the same ships that the user has but will position its ships randomly throughout the board
            2. The AI should place ships in sorted order based on the symbol used for each ship
                1. The AI should first select a direction for the ship they wish to place, either vertical or horizontal.
                    1. Use the following code to do this: `random.choice(['vert', 'horz'])`
                2. Next select a valid starting point based on the direction that was previously

            selected
            1. If horz was selected we will be placing the ship from left to right
               1. For example, if the ship we want to place is length 3 and the the board is 10 wide then our ship could start in columns 0 – 7 and be placed in any row.
            2. If vert was selected we will be placing the ship from top to bottom
               1. For example, if the ship we want to place is length 5 and the board is 7 tall then our ship could start in rows 0,1, or 2 but be placed in any column
            3. When selecting the starting of the position first select the row and then select the column
               1. Use `random.randint` to generate the row and column values
               2. It is very important that you first generate then row and then the column or otherwise your answers will not match mine.
         3. If the ship does not overlap with any other ships it should be placed at the chosen location and the location should be printed out (we are only displaying the location so that you can more easily check to see if you are placing the ships at the same location my code does)
            1. If the ship cannot be placed at the location repeat steps 1 – 3 until it is successfully placed
      4. Randomly select a number between 0 and 1
         1. If 0 is selected the player goes first. If 1 is selected the AI goes first.
            1. Use `random.randint` to generate the turn
2. Gameplay
   1. Each turn either the player or the AI selects a location to fire upon
      1. If the location does not contain an opposing ship a miss is announced
         1. Misses are marked with O
      2. If the location does contain an opposing ship a hit is announced unless that is the last hit on the ship and then it is announced that you destroyed that ship
         1. Hits are marked with X
   2. The player and the AI alternate taking turns until one of them has destroyed all of their opponents ships
   3. The player will input their choice as a row column pair on one line separated by a space.
      1. Examples:
         1. 5 8
         2. 3 9
      2. Your program must validate that the user enters correct input and if they do not continues to ask them for more input until a valid location is chosen
         1. For input to be valid it must be a row column pair that is on the board and hasn't been fired upon before
3. AIs
   1. You will be developing three different Ais
   2. Random AI
      1. This AI randomly chooses a location to fire upon that it hasn't fired upon before
      2. In terms of implementing this you should have a list of locations the AI hasn't fired upon and then use `random.choice` to select one of them to shoot at. After selecting this location it should be removed from the list of unfired upon locations.
         1. This list of locations should be generated be ordered by row. For example if the board was a 2 X 2 the list of unfired upon locations would be [(0,0), (0,1), (1,0),

(1,1)] at the beginning of the game.

3. Smarter AI
   1. The Smarter AI has 2 modes: Hunt and Destroy
   2. In Hunt mode the AI behaves like the Random AI in that it randomly chooses a location to fire upon that it hasn't fired upon before. If the AI scores a hit it switches to Destroy Mode
   3. In Destroy mode the AI fires above, below, to the left of, then to the right of the location it hit, excluding spaces it has already checked. If any hits are scored while in Destroy mode the positions above, below, to the left and to the right of the hit are added to the list of spots to check on subsequent turns
      1. The AI returns to Hunt mode after all spaces added to check that were added by Destroy mode were exhausted.
   4. As an example consider the following board. (Player moves are not being shown.)

```
  0 1 2
0 * * *
1 * S *
2 * S *
3 * S *
4 * * *
```

```
The AI starts in Hunt mode. Let's say it chooses location 1,1 to fire
upon
```

```
  0 1 2
0 * * *
1 * X *
2 * S *
3 * S *
4 * * *
```

Since the AI scored a hit it will now switch to destroy mode. The next locations it should fire upon are (0,1), (2,1), (1,0), and (1,2)

```
  0 1 2
0 * O *
1 * X *
2 * S *
3 * S *
4 * * *
```

(0,1) was a miss so we then move on to (2,1).

```
  0 1 2
0 * O *
1 * X *
2 * X *
3 * S *
4 * * *
```

We have another hit so now our list of targets to fire upon is (1,0), (1,2) , (3,1), (2,0), (2,2). Notice we did not add (1,1) to the list as we have already fired upon this location.

Firing on (1,0). A miss so no new locations are added.

```
  0 1 2
0 * O *
1 O X *
2 * X *
3 * S *
4 * * *
```

Firing on (1,2). Another miss so no new locations are added.

```
  0 1 2
0 * O *
1 O X O
2 * X *
3 * S *
4 * * *
```

Firing on (3,1)

```
  0 1 2
0 * O *
1 O X O
2 * X *
3 * X *
4 * * *
```

This is a hit so now we add (4,1), (3,0), and (3,2) to our list of targets to get: (2,0), (2,2), (4,1), (3,0), (3,2). Even though the ship is destroyed we still stay in Destroy mode and continue firing through the rest of the locations.

Firing on (2,0)

```
  0 1 2
0 * O *
1 O X O
2 O X *
3 * X *
4 * * *
```

Firing on (2,2)

```
  0 1 2
0 * O *
1 O X O
2 O X O
3 * X *
4 * * *
```

Firing on (4,1)

```
  0 1 2
```

```
0 * O *
1 O X O
2 O X O
3 * X *
4 * O *
```

Firing on (3,0)

```
  0 1 2
0 * O *
1 O X O
2 O X O
3 O X *
4 * O *
```

Firing on (3,2)

```
  0 1 2
0 * O *
1 O X O
2 O X O
3 O X O
4 * O *
```

Now that the list of elements has been exhausted, the AI returns to Hunt mode.

4. Cheating AI
    1. This AI cheats and uses the location of the player ships to play a perfect game.
        1. The AI should go through the board row by row firing on all ships in one row before preceding to the next row that contains a ship
4. Some Notes on Random
    1. It is super important that you follow the same procedures I did when using the random module or else your answers won't match mine. You might logically be doing the correct thing but if you don't make calls to the random module in the same order I did our results will differ so be careful.
    2. The only functions you will need from the random module are seed, randint, and choice.
        1. Make sure that you call `random.seed` before you make any other calls to functions in the random module
        2. `Random.randint` is useful for placing the ships and choosing the starting turn
        3. `Random.choice` is useful for the random and smart ai (in hunt mode) to choose a location to fire on.
5. Structuring your Solution
    1. Battleship is another game, just like tic-tac-toe and hangman, and follows the same pattern of
        1. Setup
        2. Taking turns until the game is over
            1. Each turn involves
                1. Displaying the state of the game
                2. Getting a move
                3. Updating the state
        3. Declaring a winner after the game is over
    2. The difference between battleship and the other games we have done so far is in the details

but looking at those games will help guide you to your answer for battleship.

6. Examples: For these examples the file 2-ship.txt contains the following: P 0 0 1 0

   1.
      ```
      Enter the seed: 1
      Enter the width of the board: 2
      Enter the height of the board: 2
      Enter the name of the file containing your ship placements:
      2-ship.txt
      Choose your AI.
      1. Random
      2. Smart
      3. Cheater
       Your choice: 3
      Placing ship from 0,1 to 1,1.
      Scanning Board
        0 1
      0 * *
      1 * *

      My Board
        0 1
      0 P *
      1 P *

      Enter row and column to fire on separated by a space: 0 0
      Miss!
      The AI fires at location (0, 0)
      Hit!
      Scanning Board
        0 1
      0 O *
      1 * *

      My Board
        0 1
      0 X *
      1 P *

      Enter row and column to fire on separated by a space: 1 1
      Hit!
      The AI fires at location (1, 0)
      You sunk my P
      Scanning Board
        0 1
      0 O *
      1 * X

      My Board
        0 1
      0 X *
      ```

```
   1 X *

   The AI wins.
2. Enter the seed: 3
   Enter the width of the board: 2
   Enter the height of the board: 3
   Enter the name of the file containing your ship placements:
   2-ship.txt
   Choose your AI.
   1. Random
   2. Smart
   3. Cheater
    Your choice: 1
   Placing ship from 0,1 to 1,1.
   The AI fires at location (2, 1)
   Miss!
   Scanning Board
     0 1
   0 * *
   1 * *
   2 * *

   My Board
     0 1
   0 P *
   1 P *
   2 * O

   Enter row and column to fire on separated by a space: bob
   Enter row and column to fire on separated by a space: 0 1 3
   Enter row and column to fire on separated by a space: 0 0
   Miss!
   The AI fires at location (2, 0)
   Miss!
   Scanning Board
     0 1
   0 O *
   1 * *
   2 * *

   My Board
     0 1
   0 P *
   1 P *
   2 O O

   Enter row and column to fire on separated by a space: 0 1
   Hit!
   The AI fires at location (0, 0)
```

```
Hit!
Scanning Board
  0 1
0 O X
1 * *
2 * *

My Board
  0 1
0 X *
1 P *
2 O O

Enter row and column to fire on separated by a space: 1 1
You sunk my P
Scanning Board
  0 1
0 O X
1 * X
2 * *

My Board
  0 1
0 X *
1 P *
2 O O

You win!
```