Instructors: Tunca Doğan, Fuat Akal, Aydın Kaya
TA: Hayriye Çelikbilek
Course: BBM103 Fall 2022
Subject: File I/O, Exceptions
Given: 08.12.2022
Due: 29.12.2022 23:00, to classroom.github.com, 1 day late, 2 days late, 3 days late.

# Assignment 4: Battle of Ships

## 1. Introduction

Battleship (also known as Battleships or Sea Battle) is a strategy-type guessing game for two players. It is played on ruled grids (paper or board) on which each player's fleet of warships is marked. The locations of the fleets are concealed from the other player. Players alternate turns calling "shots" at the other player's ships, and the game's objective is to destroy the opposing player's fleet.

The game is played in four grids of squares 10x10, two for each player. The individual squares in the grids are identified by letter and number. On one grid, the player arranges ships and records the shots by the opponent. On the other grid, the player records their shots. Before play begins, each player secretly arranges the ships on their hidden grid. Each ship occupies several consecutive squares on the grid, arranged either horizontally or vertically. The type of ship determines the number of squares for each ship. The ships cannot overlap (i.e., only one can occupy any given square in the grid). The types and numbers of ships allowed are the same for each player. The ships should be hidden from the players' sight, and it is not allowed to see each other's pieces. The game is a discovery game in which players must discover their opponents' positions.

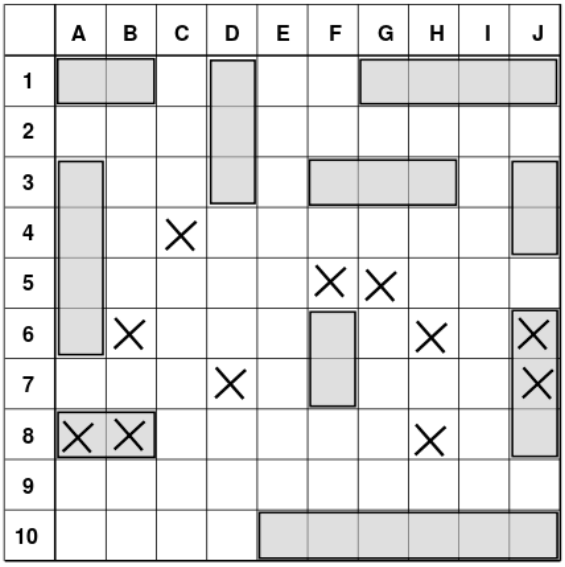


**Figure 1.** A map of one player's ships and the hits against them from a game in progress. The grey boxes are the ships placed by the player, and the cross marks show the squares their opponent fired upon.

Rules specify the following ships:

| No. | Class of ship | Size | Count | Label |
|---|---|---|---|---|
| 1 | Carrier | 5 | 1 | CCCCC |
| 2 | Battleship | 4 | 2 | BBBB |
| 3 | Destroyer | 3 | 1 | DDD |
| 4 | Submarine | 3 | 1 | SSS |
| 5 | Patrol Boat | 2 | 4 | PP |

After the ships have been positioned, the game proceeds in a series of rounds. In each round, the current player announces a target square in the opponent's grid to shoot. The computer announces whether or not the square is occupied by a ship, then marks the hit or miss on the grid.

When all of the squares of a ship have been hit, the computer announces the sinking of the Carrier, Submarine, Destroyer, Patrol Boat, or titular Battleship. If all a player's ships have been sunk, the game is over, and their opponent wins. If all ships of both players are sunk by the end of the round, the game is a draw. This means if Player1 bombed all ships and Player1 has only 1 ship left you need to ask 1 more move, before prompting the final info.

## 2. Problem

Using a **multi-dimensional built-in python list** as the data structure for the grid in the game is **optional**. For example, example_grid = [["-","-","-"], ["-","-","-"], ["-","-","-"]] represents a 3x3 initial board for the Battleship game. Necessary printing operations can be held besides this multi-dimensional list.

Game steps:

- When the game starts, the program reads from two files (Player1.txt, Player2.txt) which have each player's initial selection of ship positions. For example:

**Player 1's Board**

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | C | | | |
| 2 | | | | | B | | C | | | |
| 3 | | P | | | B | | C | P | P | |
| 4 | | P | | | B | | C | | | |
| 5 | | | | | B | | C | | | |
| 6 | | B | B | B | B | | | | | |
| 7 | | | | | | S | S | S | | |
| 8 | | | | | | | | | | D |
| 9 | | | | | P | P | | | | D |
| 10 | | P | P | | | | | | | D |

**Player 2's Board**

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | | | | S |
| 2 | D | | C | C | C | C | C | | | S |
| 3 | D | | | | | | | | | S |
| 4 | D | | P | P | | | | | | |
| 5 | | | | | | | B | B | B | B |
| 6 | | | | | | | | | | |
| 7 | | B | | | P | P | | | | |
| 8 | | B | | | | | | P | | |
| 9 | | B | | P | P | | | P | | |
| 10 | | B | | | | | | | | |

Player1.txt

```
;;;;;;C;;;
;;;;B;;C;;;
;P;;;B;;C;P;P;
;P;;;B;;C;;;
;;;;B;;C;;;
;B;B;B;B;;;;;
;;;;;S;S;S;;
;;;;;;;;;D
;;;;P;P;;;;D
;P;P;;;;;;;D
```

Player2.txt

```
;;;;;;;;;S
D;;C;C;C;C;C;;;S
D;;;;;;;;;S
D;;P;P;;;;;;
;;;;;;B;B;B;B
;;;;;;;;;
;B;;;P;P;;;;
;B;;;;;;P;;
;B;;P;P;;;P;;
;B;;;;;;;;
```

- For the current player's move; round count, grid size, and two grids for two of the players and ship statuses ("-": floating, "X": sunk) for each player are prompted. In the hidden boards "X" shows current hits by the opponent, "O" (big oh letter) shows the current misses by the opponent and surrounding empty or hidden cells will remain full of "-" characters. (In the final board, "C", "B", "D", "S", and "P" letters show still-hidden cells of unsunk ships of the winner.)
- Also, in rounds, Each user is asked for the coordinates "x, y" of the cell to bomb. In each round, both users enters two characters separated by a comma indicating the row (x) and column (y) indexes of the wanted cell respectively. For example: "10,D". To make the game easy to evaluate, current user inputs are read from the files Player1.in and Player2.in. E.g.:

Player1.in

```
5,E;10,G;8,I;4,C;8,F;4,F;7,A;4,A;9,C;5
,G;6,G;2,H;2,F;10,E;3,G;10,I;10,H;4,E;
8,G;2,I;4,B;5,F;2,G;10,C;10,B;2,C;3,J;
10,A;8,H;4,G;9,E;6,A;7,D;6,H;10,D;6,C;
2,J;9,B;3,E;8,E;9,I;3,F;7,F;9,D;10,J;3
,B;9,F;5,H;3,C;2,D;1,G;7,I;8,D;9,H;7,H
;5,J;6,B;4,J;4,I;3,D;8,A;2,E;4,H;1,F;1
0,F;7,B;6,I;1,I;1,E;7,G;7,J;5,C;9,G;6,
D;8,J;4,D;1,D;3,I;3,H;1,C;2,B;7,C;1,J;
```

Player2.in

```
1,J;6,E;8,I;6,I;8,F;7,J;10,E;1,I;4,A;1
,D;7,A;10,D;2,G;8,A;5,F;5,A;5,J;1,G;6,
B;1,A;8,E;6,D;4,G;7,B;2,I;5,B;6,G;2,C;
8,D;10,I;9,G;3,F;1,F;4,H;8,J;4,J;5,C;6
,C;6,J;5,E;4,D;1,B;2,F;10,A;7,I;2,D;10
,G;7,H;6,H;9,H;7,E;9,J;3,I;3,E;7,D;9,E
;3,H;8,G;9,F;5,H;4,B;4,E;2,H;3,G;7,G;1
0,C;1,C;8,B;5,D;10,B;9,C;4,F;2,B;3,D;5
,G;9,I;3,J;7,C;7,F;2,J;10,J;3,B;2,E;
```

- The game repeats the previous two steps until the game finishes. When all of the squares of a ship have been hit, the computer prints the sinking of the Carrier, Submarine, Destroyer, Patrol Boat, or titular Battleship. If all a player's ships have been sunk, the game is over, and their opponent wins. If all ships of both players are sunk by the end of the round, the game is a draw. (Outputs are: "Player1 Wins!", "Player2 Wins!", "It is a Draw!")
- Lastly, the computer prompts the final non-hidden form of the two players' boards.

- You may **not** assume the "Player1.txt", "Player2.txt", "Player1.in", and "Player2.in" files' inputs are correctly given. You **do** need to check for input **mistakes and exceptions.**
- When the program is executed, all the program outputs will be provided in both the command line and the "Battleship.out" text file.

## 3. Exception Handling

You might have noticed that many things could go wrong with inputs and implementation for your Battleship game. In order to prevent inconvenient cases, you will at least handle those situations (exceptions) mentioned below.

a. **IOError:** A problem can occur if the input files are not reachable at the specified file path (if files do not exist, or if their names are misspelled, etc.). Assume that your code is executed while one or more of the input files ("Player1.txt", "Player2.txt", "Player1.in", and "Player2.in") do not exist, you will handle it by printing the warning **prompt to the output file** "IOError: input file(s) Player1.txt is/are not reachable." or "IOError: input file(s) Player1.txt, Player2.txt, Player1.in, Player2.in is/are not reachable.".

d. **IndexError:** For missing arguments, your program should throw an index error and handle it by giving an explanatory prompt to the user. For example, ",A;" "A;" ",;" ";" "1,;" "1;" are some of these situations. Each situation should have a distinct message prompt. "IndexError: <your message>". Later, you must keep reading the same input file until the current user can play his/her turn.

e. **ValueError:** If any of the first operands are given as non-numeric values, trying to convert them to integers will raise a ValueError exception. Also, if operands are given but you can not interpret them, you will raise the ValueError exception. For example, "A,1;" "1,1;" "A,A;" "5,E10,G;" are some of these situations. Each situation should have a distinct message prompt. Your code must handle this problem by printing the error message "ValueError: <your message>". Later, you must keep reading the same input file until this user can play his/her turn.

b. **AssertionError:** If everything goes well and you get rounds for a set of user inputs, you must assert that your result matches the game rules. If they do not match, your code should raise an AssertionError exception and handle it by printing the error message" AssertionError: Invalid Operation." into the output file. Later, you must keep reading the same input file until this current user can play his/her turn. For example: "11,A;" "5,K;"

c. **Any Other Exception:** You should be prepared for any other possible problem when your code is run, as we cannot always anticipate all future problems. In case of any other error not mentioned above, your code should print the error message" kaBOOM: run for your life!" into the output file.

**Note:** Please be consider to **prompt** your errors **into** the created **Battleship.out** always.

## 6. What to include in the report:

- Cover (course name, assignment number, title, student number and name-surname, delivery date)
- Analysis (describe and explain the problem in your own words,
- Design (Write your thoughts on the solution to the problem,
    - for each solution/sub-solution/subroutine/function;
        - write the prototype of that subroutine and explain what it does,
        - write the data structures (arrays) used in that subprogram, explain their purpose,
        - write the algorithm (pseudocode) of that program/subroutine.)
- Programmer's Catalogue (write down the time you spent analyzing, designing, implementing, testing, and reporting; attach the code and explain the reusability of your program/subprogram for other programmers, show description of possible function calls.)
- User Catalogue (program's user manual/tutorial, restrictions on the program)

Some example reports are provided as attachments.

## 7. Grading

In evaluating the assignment, efficiency and compliance with the structural programming principles will be considered in addition to the correct and complete operation of the program. It is essential to use several necessary functions because, in computer science, it is vital to break down a problem into considerable sub-problems.

Why Do We Write Functions?

1. They allow us to conceive of our program as a bunch of sub-steps. (Each sub-step can be its function. When any program seems too hard, break the overall program into sub-steps!)
2. They allow us to reuse code instead of rewriting it.
3. Functions allow us to keep our variable namespace clean (local variables only "live" as long as the function does). In other words, function_1 can use a variable called I, and function_2 can also use a variable called I, and there is no confusion. Each variable I only exists when the computer is executing the given function.
4. Functions allow us to test small parts of our program in isolation from the rest. This aid is especially true in interpreted languages, such as Python and Matlab, but it can be helpful in C, Java, and ActionScript.

Accordingly, the scoring is as follows:

| Evaluation | Points | Evaluate Yourself / Guess Grading |
|---|---|---|
| **Readable Codes and Meaningful Naming** | 5 | … |

| Evaluation | Points | Evaluate Yourself / Guess Grading |
|---|---|---|
| **Using Explanatory Comments** | 5 | … |
| **Efficiency (avoiding unnecessary actions)** | 5 | … |
| **Function Usage** | 15 | … |
| **Correctness, File I/O** | 30 | … |
| **Exceptions** | 20 | |
| **Report** | 20 | … |
| **There are several negative evaluations** | … | … |

## 5. Example Program Input/Output Scenario:

```
Battle of Ships Game

Player1’s Move

Round : 1                       Grid Size: 10x10

Player1’s Hidden Board          Player2’s Hidden Board
  A B C D E F G H I J             A B C D E F G H I J
1 - - - - - - - - - -           1 - - - - - - - - - -
2 - - - - - - - - - -           2 - - - - - - - - - -
3 - - - - - - - - - -           3 - - - - - - - - - -
4 - - - - - - - - - -           4 - - - - - - - - - -
5 - - - - - - - - - -           5 - - - - - - - - - -
6 - - - - - - - - - -           6 - - - - - - - - - -
7 - - - - - - - - - -           7 - - - - - - - - - -
8 - - - - - - - - - -           8 - - - - - - - - - -
9 - - - - - - - - - -           9 - - - - - - - - - -
10- - - - - - - - - -           10- - - - - - - - - -

Carrier       -                 Carrier       -
Battleship    - -               Battleship    - -
Destroyer     -                 Destroyer     -
Submarine     -                 Submarine     -
Patrol Boat   - - - -           Patrol Boat   - - - -

Enter your move: 5,E

Player2’s Move
```

```
Round : 1                       Grid Size: 10x10

Player1's Hidden Board          Player2's Hidden Board
  A B C D E F G H I J             A B C D E F G H I J
1 - - - - - - - - - -           1 - - - - - - - - - -
2 - - - - - - - - - -           2 - - - - - - - - - -
3 - - - - - - - - - -           3 - - - - - - - - - -
4 - - - - - - - - - -           4 - - - - - - - - - -
5 - - - - - - - - - -           5 - - - - O - - - - -
6 - - - - - - - - - -           6 - - - - - - - - - -
7 - - - - - - - - - -           7 - - - - - - - - - -
8 - - - - - - - - - -           8 - - - - - - - - - -
9 - - - - - - - - - -           9 - - - - - - - - - -
10- - - - - - - - - -           10- - - - - - - - - -

Carrier       -                 Carrier       -
Battleship    - -               Battleship    - -
Destroyer     -                 Destroyer     -
Submarine     -                 Submarine     -
Patrol Boat   - - - -           Patrol Boat   - - - -

Enter your move: 1,J

Player1's Move

Round : 2                       Grid Size: 10x10

Player1's Hidden Board          Player2's Hidden Board
  A B C D E F G H I J             A B C D E F G H I J
1 - - - - - - - - - O           1 - - - - - - - - - -
2 - - - - - - - - - -           2 - - - - - - - - - -
3 - - - - - - - - - -           3 - - - - - - - - - -
4 - - - - - - - - - -           4 - - - - - - - - - -
5 - - - - - - - - - -           5 - - - - O - - - - -
6 - - - - - - - - - -           6 - - - - - - - - - -
7 - - - - - - - - - -           7 - - - - - - - - - -
8 - - - - - - - - - -           8 - - - - - - - - - -
9 - - - - - - - - - -           9 - - - - - - - - - -
10- - - - - - - - - -           10- - - - - - - - - -

Carrier       -                 Carrier       -
Battleship    - -               Battleship    - -
Destroyer     -                 Destroyer     -
```

```
Submarine     -              Submarine     -
Patrol Boat   - - - -        Patrol Boat   - - - -

Enter your move: 10,G

{…}
{More rounds continues… The program will print them…}
{…}

Player1’s Move

Round : 38                   Grid Size: 10x10

Player1’s Hidden Board       Player2’s Hidden Board
  A B C D E F G H I J          A B C D E F G H I J
1 O - - O - O X - O O        1 - - - - - - - - - -
2 - - O - - - X - O -        2 - - X - - X X O O X
3 - - - - - - O - - - -      3 - - - - - - - O - - X
4 O - - - - - - X O - O      4 X O X - O O O - - -
5 O O O - - O - - - O        5 - - - - - O O X - - -
6 - X - X X - O - O -        6 O - O - - - O O - -
7 O O - - - - - - - - O      7 O - - O - - - - - -
8 O - - O O O - - O X        8 - - - - - - O O X O -
9 - - - - - - - O - - -      9 - - O - X - - - - -
10- - - O O - - - O -        10O X O O O - O O O -

Carrier       -              Carrier       -
Battleship    - -            Battleship    - -
Destroyer     -              Destroyer     -
Submarine     -              Submarine     -
Patrol Boat   - - - -        Patrol Boat   - - - -

Enter your move: 9,B

Player2’s Move

Round : 38                   Grid Size: 10x10

Player1’s Hidden Board       Player2’s Hidden Board
  A B C D E F G H I J          A B C D E F G H I J
1 O - - O - O X - O O        1 - - - - - - - - - -
2 - - O - - - X - O -        2 - - X - - X X O O X
3 - - - - - - O - - - -      3 - - - - - - - O - - X
```

```
4 O - - - - - - X O - O      4 X O X - O O O - - -
5 O O O - - O - - - O        5 - - - - O O X - - -
6 - X - X X - O - O -        6 O - O - - - O O - -
7 O O - - - - - - - O        7 O - - O - - - - - -
8 O - - O O O - - O X        8 - - - - - O O X O -
9 - - - - - - - O - - -      9 - X O - X - - - - -
10- - - O O - - - O -        10O X O O O - O O O -

Carrier       -              Carrier       -
Battleship    - -            Battleship    - -
Destroyer     -              Destroyer     -
Submarine     -              Submarine     -
Patrol Boat   - - - -        Patrol Boat   - - - -

Enter your move: 6,C

Player1's Move

Round : 39                   Grid Size: 10x10

Player1's Hidden Board       Player2's Hidden Board
  A B C D E F G H I J          A B C D E F G H I J
1 O - - O - O X - O O        1 - - - - - - - - - -
2 - - O - - - X - O -        2 - - X - - X X O O X
3 - - - - - - O - - - -      3 - - - - - - - O - - X
4 O - - - - - - X O - O      4 X O X - O O O - - -
5 O O O - - O - - - O        5 - - - - O O X - - -
6 - X X X X - O - O -        6 O - O - - - O O - -
7 O O - - - - - - - O        7 O - - O - - - - - -
8 O - - O O O - - O X        8 - - - - - O O X O -
9 - - - - - - - O - - -      9 - - O - X - - - - -
10- - - O O - - - O -        10O - O O O - O O O -

Carrier       -              Carrier       -
Battleship    X -            Battleship    - -
Destroyer     -              Destroyer     -
Submarine     -              Submarine     -
Patrol Boat   - - - -        Patrol Boat   - - - -

Enter your move: 3,E

{…}
{More rounds continues… The program will print them…}
{…}
```

```
Player2’s Move

Round : 83                    Grid Size: 10x10

Player1’s Hidden Board        Player2’s Hidden Board
  A B C D E F G H I J           A B C D E F G H I J
1 O O O O - O X - O O         1 - - O O O O O - O X
2 - O O O - O X O O O         2 - O X X X X X O O X
3 - X - O X O X X X O         3 - O O O O O O O O X
4 O X - O X O X O - O         4 X O X - O O O O O O
5 O O O O X O X O - O         5 - - O - O O X X - X
6 - X X X X - O O O O         6 O O O O - - O O O -
7 O O O O O X X X O O         7 O X O O - X O O O O
8 O O - O O O O - O X         8 O - - O O O O X O O
9 - - O - X X O O O X         9 - X O X X O O X O -
10O X X O O - O - O X         10O X O O O O O O O O

Carrier       X               Carrier       X
Battleship    X -             Battleship    - -
Destroyer     X               Destroyer     -
Submarine     X               Submarine     X
Patrol Boat   X X X X         Patrol Boat   X X - -

Enter your move: 2,E

Player2 Wins!

Final Information

Player1’s Board               Player2’s Board
  A B C D E F G H I J           A B C D E F G H I J
1 O O O O - O X - O O         1 - - O O O O O - O X
2 - O O O X O X O O O         2 D O X X X X X O O X
3 - X - O X O X X X O         3 D O O O O O O O O X
4 O X - O X O X O - O         4 X O X P O O O O O O
5 O O O O X O X O - O         5 - - O - O O X X B X
6 - X X X X - O O O O         6 O O O O - - O O O -
7 O O O O O X X X O O         7 O X O O P X O O O O
8 O O - O O O O - O X         8 O B - O O O O X O O
9 - - O - X X O O O X         9 - X O X X O O X O -
10O X X O O - O - O X         10O X O O O O O O O O
```

```
Carrier       X              Carrier       X
Battleship    X X            Battleship    - -
Destroyer     X              Destroyer     -
Submarine     X              Submarine     X
Patrol Boat   X X X X        Patrol Boat   X X - -
```

Please note that,

- ✓ Example input and output files are provided in the attachment for the detailed examination. Note that **punctuation** and **white spaces** in both files are **fatally important**. You may lose points because of punctuation or white spaces.
- ✓ If your program **fails to read** some of the inputs, your whole assignment could crash at the beginning!: This **indicates no grading**.
- ✓ These long scenario round moves are randomly generated for testing purposes.

## 6. Notes

- **Do not use input() function.**
- Do not miss the submission deadline.
- Save all your work until the assignment is graded.
- The assignment must be original, individual work. Duplicates or very similar assignments are both going to be considered cheating.. Show any references or quotes for academic honesty.
- Write READABLE codes and their COMMENTS.
- You can ask your questions via Piazza, and you are supposed to be aware of everything discussed on Piazza.
- You will use the online Github Classroom system to submit your work.
- No other submission method (e-mail, etc.) will be accepted. Do not submit any file via e-mail related to this assignment.
- You need to name your assignment **"Assignment4.py"** and gather your work into a **single python file**.
- Do not forget to fill your grading table with your expected marks and attach the table to your report file. **The report is obligatory "Assignment4.pdf"**.
- Do not hesitate to ask questions to your TA. TA's office hours will be by appointment.
- Do not forget to **write down your name and number inside both the python (as a comment) and pdf documents.**
- Before uploading, you should write, run and **test** your assignment in your local environment and the **dev.cs.hacettepe.edu.tr** server.

- After **23:00(Yes, 23:01 included)**, we can not see your submissions; you have to submit to the next day's link.
- Please **do not submit large files bigger than 3MB**, especially for pdf sizes.
- Please **do not submit the "Battleship.out"** file. The output file will be created automatically when your code is run. It will be using **writing mode (not appending mode)**.
- Please **do not submit the "Player1.txt, Player2.txt, Player1.in, Player2.in"** files. I will provide the input files to your code within the same directory.
- Please mind that user can provide **different input file names** from bash.

- **File hierarchy**
  - <Inside your commit **without** a folder>
  - Assignment4.py
  - Assignment4.pdf


- **Sample Run**

**> python3 Assignment4.py "Player1.txt" "Player2.txt" "Player1.in" "Player2.in"**

```
[user@rdev ~]$ ls
Assignment4.py  Assignment4.pdf  Player1.txt  Player2.txt  Player1.in  Player2.in
[user@rdev ~]$ python3 Assignment4.py "Player1.txt" "Player2.txt" "Player1.in" "Player2.in"
[user@rdev ~]$ ls
Assignment4.py  Assignment4.pdf  Player1.txt  Player2.txt  Player1.in  Player2.in  Battleship.out
[user@rdev ~]$
```

# 7. Policy

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out abstractly. Discussions about a particular solution to a problem (either in actual code or pseudocode) will not be tolerated. In short, turning in someone else's work (from the internet), in whole or part, as your own will be considered a violation of academic integrity. Please note that the former condition also holds for the material found on the web, as everything on the web has been written by someone else.

References for the Academic Integrity (AI):

https://academicintegrity.ucsd.edu/AI-Handbook-for-UCSD_2019.pdf,
academicintegrity.org/resources/facts-and-statistics