# 

# COMP 5070 Statistical Programming for Data Science

# Test 1: Battleship Board Game

* This exercise is a part of the continuous assessment that is worth 25% of your overall grade.
* Your code should be submitted as a single .py file using LearnOnline. Do not hardcode any paths on your computer in the code, as **I should be able to load and run your code without any changes**.
* This assessment is to test students’ knowledge in Python fundamentals programming, that is, first three weeks of the course. No use of OOP is allowed. Submissions with custom classes/objects will get zero mark.
* The exercise is out of 100 marks. To obtain the maximum available marks you should aim to:

1. Code the requested program (70%).
2. Use a clear coding style (10%). Code clarity is an important part of your submission. Thus, you should choose meaningful variable names and adopt the use of comments - you don't need to comment every single line, as this will affect readability - however you should aim to comment at least each section of code.
3. Have the code run successfully (10%).
4. Document code limitations including, but not limited to, the requested functionalities (10%).

This assessable exercise can be openly discussed within the group online and you are welcome to share tips and tricks (not entire programs, however).

Having said that, the ground rules are:

* If you use another person’s code in your file, please note the source and how much of the code is not yours.
* If you submit a program cobbled together by other peoples’ code with no, or little, original input from yourself, you will automatically receive zero mark and will be reported to Academic Integrity office. The idea is to develop your own programming style with (or without) the help of others. Any code used should support your approach to how you write the program, not replace your own efforts.

If you’re unsure at any point, you’re welcomed to check with me.

Late submission will be penalized by 10-point deduction for each day or part of it after the due date.

**If you have any questions – feel free to ask them on the forum or by email.**

# Battleship Board Game

You should write a simulator of the board game of Battleship. You can read full rules of the game on the internet. Just google for “Battleship rules” and you will find many sources. However, you don’t really need as rules are presented below. Typically, this game is for two players, but you make a version for one player only.

Board preparation:

* The board is grid of 10 by 10 cells.
* There are 5 ships: Carrier (occupies 5 cells), Battleship (4), Cruiser (3), Submarine (3), and Destroyer (2).
* All ships can be placed vertically or horizontally. No diagonal positioning.
* Ships can touch each other but two ships cannot occupy the same cell. All ships should be completely within the playing board.

Game:

* Player cannot see the board with the ships and tries to guess/hit their locations by providing row and column of the cell to hit.
* Player gets a response from the game on the result of the guess: hit or miss.
* When the player hits all parts/cells of all ships, the game is over.

Your task is to code the game and the player(!). It is not for you or me to play this game, but computer will play. Your code will have several main components – all of them prepared as functions:

1. Prepare the board with ships. This function takes the board size and the list of ship sizes as parameters and returns a prepared board.
2. Player function might take the board size and its own version of the board with the history of past hits and misses as parameters and returns coordinates of the next guess.
3. Check function takes the board and player’s guess and returns the result: hit or miss.
4. Last two function would run in a loop until the player hits all ships and the game is over.
5. Run a simulation: repeat the game 1000 (or more) times and report statistics of how long it takes for the player to finish the game. You can start Player function as a random guess but then try to make it smart to achieve the shortest average game possible.