# CSCI 2312-001: Object Oriented Programming

# Assignment 1

# Assigned: January 17, 2018

# Due: January 29, 2018 at 11:59 PM

**Instructor: Marlu Oswald**

**Email: marlu.oswald@ucdenver.edu**

**Assignment 1**

Read the following description of the Battleship game and do a ROC(k)ET decomposition. For the Test phase, describe what testing you would need to do on each function to feel like the major threads of the function were tested properly.

**Background**

For your CSCI 2312 Project, you will develop a simple battleship game. Battleship is a [guessing game](https://en.wikipedia.org/wiki/Guessing_game) for two players. It is played on four grids. Two grids (one for each player) are used to mark each players' fleets of ships (including battleships). The locations of the fleet (these first two grids) are concealed from the other player so that they do not know the locations of the opponent’s ships. Players alternate turns by ‘firing torpedoes’ at the other player's ships. The objective of the game is to destroy the opposing player's entire fleet. In our game, ‘firing a torpedo’ will be allowing the player to take a guess at where on the grid their opponent may have placed a ship. For the Project, you will develop a single player Battleship game where the human player will play against the computer.

**Simplified Requirements for this assignment (The Project will have more detailed requirements)**

For this assignment, we will only be doing a ROC(K)ET analysis of the game setup portion of Battleship which includes setting up the game board (2 grids must be created to place ships on) and allowing the players to place their ships.

The game setup uses one grid for each player to ‘secretly’ place their ships. The grids are 10 by 10. The individual squares in the grid are identified by the x coordinate (indicated by a letter) followed by the y coordinate (indicated by a number).

Before play begins, each player secretly arranges their ships on their grid. This will be done by prompting the player for the location and the orientation of each type of ship.

Each ship occupies a certain number of consecutive squares on the grid (sizes of ships are in the following table), arranged either horizontally or vertically. The number of squares for each ship is determined by the type of the ship. The ships cannot overlap so only one ship can occupy any given square in the grid. The types and numbers of ships allowed are the same for each player and consist of 1 of each of the following:

|  |  |
| --- | --- |
| Ship Type | Number of Grid Squares |
| Carrier | 5 |
| Battleship | 4 |
| Cruiser | 3 |
| Submarine | 3 |
| Destroyer | 2 |

The following is an example of a game setup for one of the players with a Carrier placed at location B3 in the horizontal direction, a Battleship placed at J5 in the vertical direction, a Cruiser placed at A10 in the horizontal direction, a Submarine at location C6 in the vertical direction, and a Destroyer placed at location I4 with a horizontal direction.

A B C D E F G H I J

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  | C | C | C | C | C |  |  |  |  |
|  |  |  |  |  |  |  |  | D | D |
|  |  |  |  |  |  |  |  |  | B |
|  |  | S |  |  |  |  |  |  | B |
|  |  | S |  |  |  |  |  |  | B |
|  |  | S |  |  |  |  |  |  | B |
|  |  |  |  |  |  |  |  |  |  |
| Cr | Cr | Cr |  |  |  |  |  |  |  |

1

2

3

4

5

6

7

8

9

10

**ROCkET Analysis**

(R)ead the Entire Problem

(O)utline

(C)ode a small functionality

(E)valuate small functionality

(T)est entire program

Remember that this is an iterative process with a focus on breaking into small manageable components. For this assignment, your outline will be placed in //line comments. Think of individual functions for your high-level outline. Instead of Code, you will place pseudocode (in line or block comments) tabbed a level in from the outline. Describe your parameters (inputs) and expected output. Give enough information to describe the functionality of each outlined functionality. For the Evaluate portion, write a test plan, on how you will evaluate this functionality (and what you will do with major errors). Focus more on plain English or pseudocode instead of C++ syntax.

Example:

//Print Fleet Grid

//PARAMETERS: Fleet Grid

//RETURN VALUE: None

/\*Start at top corner.

For each column

For each row

print value of cell

print a tab

print a newline /\*

/\*TEST PLAN:

Write a 2X2 grid on paper. Fill with ‘A’,’B’ in first row, ‘C’, ‘D’ in second row

Test function with those inputs

Validate through print statements (and later debugging) that rows and columns are as expected

**Grading Rubric**

Hand in a single .cpp file. Ensure your name is on every file. You may (and are encouraged to) discuss general strategy with classmates (or other help) but you MUST document all extra help other than the textbook (including web sites) with comments. Failure to document extra help may result in a zero for the assignment. While you should help your classmates, stop short of giving exact lines of code or pseudocode. If we can’t determine who shared and who copied, we may give you both a zero for the assignment.

|  |  |
| --- | --- |
| Outline Framework | 5 |
| Pseudocode | 10 |
| Test Plan | 5 |