**CS 2050**

**Homework Assignment 1**

**Summer - 2015**

**Due**: Friday 19th June, 2015 5:00 PM (No Extension or Exceptions)

**Purpose:**

 Continue to use good programming practices discussed in lab.

 Utilize structs and continue to practice algorithm design.

 Complete a decently large project incorporating things you’ve learned last semester and so far this semester.

**Submission information:**

Submit the homework assignment on BlackBoard. Submit your whole netbeans project.

**Description**

For the homework assignment you will be recreating a mockup of the game “Battleship”. This will involve you playing against the “computer”. The rules for the purpose of this assignment can be found here: <http://www.hasbro.com/common/instruct/battleship.pdf>

You will need three “Game Boards” which should be implemented as two-dimensional character arrays. One game board will be used as a display to the user and will print out where the users ships are, where the computer has shot, and whether the computers shot was a hit or miss. More information can be found in the function descriptions in the template and below. The computer’s game board will keep track of where the computers pieces are and whether or not the user has hit them. Realize, the computers game board is essentially the users game board, with the difference being whose ships are displayed and whose hits and misses are displayed. The way this will be written, it’s as if the computer is actually a second player. The third and final game board will be printed out and shown to the user in addition to the users game board and will be titled “Opponent’s game board”. This will show where the user has shot, whether it was a hit or miss, and the rest of the positions will be marked with a ‘?’ character. This is defined as a const character pointer at the global level, so whenever you need to use it just use the word MYSTERY. There are examples of using the constant character pointer in the template file provided last week.

You will need to implement TWO different ways to place the ship pieces on the board. One will be a “manual” method that the user will use to place the ships where they want them on the user’s game board. The other will be a “random” method to place the computer’s pieces on the computer’s gameboard in “random” positions.

When placing pieces **in both functions** you will need to make sure of a few things:

1. That the position that the piece will be played in does not exceed any of the four boundaries. That is to say, if this were a physical game board, would the position and direction of the piece lead to it hanging off of the edge? If so, you need to reposition the piece. For the manual placement function, ask the user to select a new starting position and direction. For the random position function, simply generate new values for the starting coordinates and direction and try this new location. If you can not place the piece in this new position keep generating new coordinates and directions until the piece is placed successfully.
2. That the new ship will not intersect a ship that is already in place. Handle this conflict the same way you would handle the conflict in number 1.

You will need to call check\_ship\_placement from both the place\_ships\_manually function and the place\_ships\_randomly function.

**SUGGESTED ORDER OF IMPLEMENTATION**

Complete these functions in this order if you are struggling.

1. init\_game\_board
2. check\_ship\_placement
3. place\_ship
4. place\_ships\_randomly
5. place\_ships\_manually
6. check\_strike\_coords
7. acquire\_strike\_coordinates
8. fire\_missile
9. update\_game\_board

If you complete the functions in this order, you will have plenty of opportunity to TEST, TEST, TEST your code as you write it.

**Functions that call other functions**

The following functions call their respective functions. This gives you more of an idea of the layout of the code. The order here is NOT necessarily the order in which they are called. Also, the print\_game\_board function may be called from other functions as needed.

1. main
   1. init\_game\_board
   2. place\_ships\_randomly
      1. check\_ship\_placement
      2. place\_ship
   3. place\_ships\_manually
      1. check\_ship\_placement
      2. place\_ship
   4. print\_game\_board
   5. acquire\_strike\_coordinates
      1. check\_strike\_coords
   6. fire\_missile
   7. update\_game\_board

**Rough Guidelines for Grading HW-1**

**40 Points Possible**

**General**

***If your program does not compile, or produce any input/output (I/O) because most of the source code is commented out then your assignment will receive a grade of NO POINTS.*** For partial credit for homework assignment and future labs your C program must not only compile but also produce some valid I/O that meets the lab specifications.

**-5 points:** Header (with name, paw print, no lab code is required for HW), comments and general coding style.

**10 points:** place\_ships\_manually and place\_ships\_randomly

**10 points:** acquire\_strike\_coordinates and check\_strike\_coordinates

**10 points:** check\_ship\_placement and place\_ship

**10 points:** fire\_missile and update\_game\_board

The ENTIRE program needs to work as specified in the many documents and resources offered for this homework. If it does not, you will lose points and the above guidelines are not a full enumeration of all possible point deductions.