Assignment #5 - Game: Battleship
Due: Wednesday, 11/21/12, 11:59pm

(75 pts) You will write a program that plays the game Battleship. In this program, you will be graded on having functions, as well as the ability to play the game correctly. The one requirement for using functions is that your functions, including main(), must not have over 15 lines of code, this doesn’t include comments or blank lines.

Some example functions you might want to include are an initialize_board(), which initializes the boards to spaces, a determine_player_choice() that allows players to pick their spot on their opponents grid, fill_board(), which fills the board with the player’s choice, a print_board() that prints the board to the screen after each user’s turn, a check_for_winner(), which checks to see if there is a winner, and a print_winner_results() that prints the results of the game to the screen.

This is traditionally a 2-person game, where each play picks where to put their ships on a 10 x 10 matrix, and the ships can only be arranged horizontally or vertically on the board, and they can’t hang over the grid!!! Ships can touch each other, but can’t both occupy the same spot. Your computer game will simulate this process by first asking player 1 where he/she wants to put the ships, and then asking player 2. You can clear the screen after each player chooses their positions for their ships by using system(“clear”) from #include <stdlib.h>. Each player is given the following ships, and each ship takes up a specific number of spots on the 10 x 10 grid:

1 Aircraft Carrier, 5 spots
1 Battleship, 4 spots
1 Destroyer, 3 spots
1 Submarine, 3 spots
1 Patrol Boat, 2 spots

After you determine where each player wants to put their ships, then you can prompt each player to choose a position on the opponent’s board. If there is a boat in the corresponding position on the opponent’s board, then it was a hit, and you can use whatever you want to symbolize that it was a hit, i.e. X, 1, etc., in your own board. You can even turn the X’s or 1’s red by using the following code:

\033[22;31m turns your text red, and \033[01;37m turns your text back to white.

In the case of a hit, the other player must mark the hit on his setup board containing the ships. Whenever a ship is sunk, you must announce to your opponent that he/she has sunk your ship. If there isn’t a boat in the opponent’s position, then it is a miss, and you can use whatever you want to symbolize a miss on your board, i.e. 0, N, etc.

The player who sinks all of his/her opponent’s ships first is the winner!!!!
Example Battleship 2 Rounds:

Player 1: What position do you choose? 8 A

Hit!!!
You sunk my ship!!!

Player 2: What position do you choose? 4 C

Hit!!!
You sunk my ship!!!
Player 1: What position do you choose? 2 B

Miss!!

Player 2: What position do you choose? 7 H

Miss!!
(10 pts) In your implementation, make sure that you include a program header and function headers/comments, in addition to proper indentation/spacing and other comments! Read the class style guideline for more information: [http://classes.engr.oregonstate.edu/eecs/fall2012/cs161-001/161_style_guideline.pdf](http://classes.engr.oregonstate.edu/eecs/fall2012/cs161-001/161_style_guideline.pdf)

You are graded on having a header, proper comments, and readable code with indentation and vertical spacing that is CONSISTENT throughout your program. DO NOT align your entire program on the left side. This will cause you to automatically lose the full 10 points. In addition, do not forget your program header!!!

(15 pts) You are required to turn in a written document (as a pdf) addressing Polya’s steps to solving a problem with step 3 being the C code you write to carry out/implement your plan. With this said, your written document must include these three sections:

**Understanding the Problem**
In your own words, explain what YOU think the problem is asking you to do. In this section, document your uncertainties about the problem and anything else that you feel was unclear or vague. This is to ensure that YOUR understanding matches MY understanding of the problem.

**Devising a Plan/Design**
At a minimum, provide an algorithm/pseudo code you designed to help solve the problem. In addition, include pictures/flow charts you used to help you devise your plan, as well as any other design decisions you made such as how to manage your time, how to decompose the problem, where to start first, etc. You can scan any handwritten work and attach it to the document as needed.

**Looking Back/Self-Reflection**
Report any checking/self-reflection you did while solving the problem. For instance, how did you make sense of the output from the implementation? This includes things such as using a calculator to make sure the output is correct, testing to make sure your code executes correctly and behaves the way you expect under specific circumstances, using external sources of information such as the internet to make sense of the results, etc. Also, include a statement about what you learned from the assignment.

**Extra Credit (20 pts):**
Prompt the user for 1 or 2 players, and if the user chooses 1 player, then he/she will play the computer. In other words, player 2 is the computer, and the computer will have to randomly put the ships on the board. This will entail generating a spot on the board and a direction for the ship. However, if the ship cannot fit in one direction, then another direction needs to be chosen.

Electronically submit your pdf document and C++ program file by the assignment due date, using TEACH.