CSCE 1030 – Homework 6

Due: 11:59 PM on Tuesday, November 25, 2014 CST

Problem Statement:

In this C program, you will start building the software infrastructure that will allow you to build a simplified version of the classic game of Battleship that will display on the screen/console. In homework 7, you'll continue and finish the development of the game.

In this program, the user will attempt to find a randomly placed aircraft carrier and battleship on a 10 by 10 board implemented as a two-dimensional array. You should define this SIZE as a constant in case you want to expand or shrink the board later.

Your program's output should initially display the department and course number, program number, your name, your EUID, and your e-mail address. This is true for all of your CSCE 1030 programs!

Your program will then ask the user to enter the difficulty level of the game as a string, where "easy" means the user has 30 attempts to sink both the aircraft carrier and battleship, "normal" means the user has 25 attempts to do this, and "hard" means the user has only 20 attempts to sink the two ships. If the user does not enter a valid response, you will keep repeating this process until a valid response is entered.

To start the game, you will display an introductory message, giving details about the game, including the number of attempts the user gets based on their chosen level of difficulty and the size of the board. This shall be implemented by a function.

You shall declare and initialize a two-dimensional array using dynamic memory allocation in `main()` to represent a 10 x 10 board. This means that you will have to pass the array by reference when needed to the functions that use it. Each element on this 10 x 10 data structure can be visualized as a single location that may host one unit length of an aircraft carrier or battleship. The board shall visually appear similar to the Battleship game as follows:

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

In the above board, the 5 A's on the board represent a randomly generated aircraft carrier placed horizontally having a total length of 5 units, while the 4 B's on the board represent a randomly generated battleship placed vertically having a total length of 4 units. The aircraft carrier in this case, for example, is located on elements B4, B5, B6, B7, and B8.

You shall write a function to assign a ship to a location on the board, given the data structure for the board passed by reference and another parameter providing some identifying factor, such as the length or type of the ship. This function is called by `main()` and is to only assign (or attempt to assign) one ship that is passed to it. You shall randomly generate the orientation (i.e., either vertical or horizontal) of the ship. You shall also randomly generate the row and column position for the start of the ship. If the ship does not fit on the board at this starting location, you can either return information indicating that the ship could not be assigned (and so it will be up to `main()` to keep calling this function until it is successfully assigned) or keep randomly generating row and column positions for the ship's starting position until one is found that can be used to successfully assign the ship. A ship may be assigned anywhere on the board (as long as it fits), but it may not be assigned on top of another ship that exists at that location. This must be a single function designed to handle all ships.

You will also write a function to display the game board, giving the data structure for the board passed by reference. It is called by `main()` and is invoked anytime the board has been updated (for this homework, it is invoked after the aircraft carrier and battleship have been placed on the board). This function will display the row and column headers for the board as well as the board itself (see sample output). Open space on the board (i.e., where no ships have been assigned) shall be indicated by a blank space on the board. The aircraft carrier shall be represented by an 'A' for each of its 5 unit length, while the battleship shall be represented by a 'B' for each of its 4 unit length.

Sample input and output appears below (with input shown in bold) to provide you with direction as to what is expected from the program:

```
+------------------------------------------------------------+
|              Computer Science and Engineering              |
|               CSCE 1030 - Computer Science I               |
|          Student Name     EUID    euid@my.unt.edu          |
+------------------------------------------------------------+

                Welcome to Battleship!

Enter difficulty level of game (easy, normal, hard): easy

+------------------------------------------------------------+
| The computer program will randomly assign an aircraft car- |
| rier and a battleship, that are oriented either vertically |
| or horizontally, to the board. You will have 30 chances to |
| sink both the computer's aircraft carrier and battleship!! |
| You'll play on a 10 x 10 board, where the aircraft carrier |
| will have a length of 5 units and the battleship will have |
| a length of 4 units.                                        |
+-------------------------------------------------------------+

Initializing board... now let's begin!

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

Your functions should receive as arguments only the minimally needed ones of the appropriate type, and should a value of the appropriate type whenever a return value is needed.

Additional requirements for this homework:

1. You will write a function to display your personal information (department and course number, program number, your name, your EUID, and your e-mail address).
2. You will write a function to display the introductory message with details about the game.
3. You will write a SINGLE function to assign a ship (whether an aircraft carrier or battleship) to the board.
4. You will write a function to display the current status of the game board.

Design:

On a piece of paper, write down in English the sequence of steps you will perform to solve the problem. Pretend this is a “recipe” for someone else to follow. Refine your “recipe” until it is clear. Be sure to include the steps for prompting for input, etc. This document should contain the sequence of steps used and some verification of what you did to ensure that your design worked.

Type these steps into a document (Word, txt, PDF, etc.). Also be sure to include your algorithm steps as comments in your code file. Do this before you start coding as completing it afterwards does not help you in learning the design process!

Implementation:

Now that you have a working design, your next step is to translate these steps into C code. Use the algorithm development techniques discussed in class to implement your solution to the problem above. Add your C code a little at a time, and compile and test as you go.

Remember to add your comments to your code to explain your program. Do this before/during programming instead of waiting until the end. At a minimum, you should comment the header (e.g., name, class, date, brief description of the program, etc.), all variables (i.e., what they are used for), and specific "blocks" of code. For example, use comments to describe the inputs, the formulas used, and any other important steps in your code.

Your program will be graded based largely upon whether it works correctly on a CSE Department machine, so you should make sure your program compiles and runs on a CSE machine.

Your program will also be graded based upon your programming style. At the very least, your program should include:

- A consistent indentation style as recommended in the textbook and in class;
- Meaningful variable names;
- A block header comment section that includes: your name, e-mail address, and a brief description of the program.
- Good function headers as described below.

Documentation:

When you have completed your C program, write a short report (2 – 3 paragraphs) describing what the objectives were, what you did to solve the problem, and the status of the program. Does it work properly for all test cases? Are there any known problems?

You will also include an example of your program working. To do so, you will use the "script" command with no parameters so that the output is saved to a file called "typescript".

Save this report in a separate file to be submitted electronically. You should also include any specific instructions required to compile or execute your code, such as linking a specific library (e.g., "`-lm`" for the math library, or "`-std=c99`", etc.).

Testing:

Test your program to check that it operates as desired with a variety of inputs, including any boundary conditions. Compare the answers your code gives with the ones you get from hand-calculations.

Homework Submission:

In this class, we will be using electronic homework submission to make sure that all students hand their programming projects (and labs) on time. You will submit your program source file to the class website through the "**Homework 5**" drop box by the due date and time.

CSCE 1030 – Homework 6

Due: 11:59 PM on Tuesday, November 25, 2014 CST

Note that this project must be done individually. The program will be checked using a code plagiarism tool against other solutions, so please ensure that all work submitted is your own.

Note that the dates on your electronic submission will be used to verify that you met the due date above. All homework up to 24 hours late will receive a 50% grade penalty. Later submissions will receive zero credit, so hand in your best effort on the due date.

Summary:

- You will design a solution to the problem.
- You will implement it on the CSE machines using C. You will make sure to use good style, good variable names, indentation, etc. You will compile, run, and test your code.
- You will write a brief report describing what your code does and how well it works.
- You will submit electronically your C code, your design, and your brief report.

General Guidelines (for ALL of your programming assignments):

- Your program's output should initially display the department and course number, program number, your name, your EUID, and your e-mail address.
- Use meaningful variable names.
- Use appropriate indentation.
- Use comments, including a header. Example header:

```
/*
Author: Jane Doe (Jane.Doe@my.unt.edu)
Date: 9/18/2014
Purpose: This program reads in three numbers and computes their
average
*/
```

- Add a header to each function. Example function header:

```
/*
Function: deposit
Parameters: a double representing the account balance,
and a double representing the deposit amount
Return: a double, the account balance after the deposit
This function computes the account balance after a deposit
*/
```