Assignment 6

Play A Game: Minesweeper or Battleship!!!

Due: Sunday, December 3rd, 2017 11:59pm

This will be our last assignment in the class, boohoo☺

Grading: For this assignment, you will be graded traditionally, i.e. **TAs will compile and execute your program for a grade without you demoing it.** However, you need to **make sure you have Assignment #5 demoed/graded by the end of week 10!!!!**

(90 pts) Problem Statement:
For this assignment, you will implement a game. You can pick between minesweeper or battleship. Each game has an extra credit piece!!!

Minesweeper

This game was made popular by the Windows operating system in the early 1990s, and it has continued to be a pre-installed game until Windows 8. You can watch a video of how to play minesweeper on YouTube or read about it on Wikipedia:

https://www.youtube.com/watch?v=Z0EAysRluJk

https://en.wikipedia.org/wiki/Microsoft_Minesweeper

The game is very simple. The object of the game is to open/reveal every cell on the board without detonating a mine. Every cell contains a number or a mine. The numbers tell you how many mines surround the cell in the horizontal, vertical, and diagonal directions, e.g. at most, 8 possible mines surrounding a cell. You can flag a cell, which helps you remember where you think there is a mine to not detonate. If you select to open/reveal a cell that has a mine, then you automatically lose the game!

You will then randomly distribute the mines on the board and setup the numbers describing how many adjacent mines to each cell. After that, you will display a blank board, and ask the user to flag or open a cell on the board until the user selects a cell that contains a mine (losing the game) or selects all cells free of mines (winning the game). After the user wins or loses, you will ask the user if he/she wants to play again. If so, you must get the number of rows, columns, and mines for the new game and create a new board for a new game. **BTW, you can display row and column numbers on board to make it easier to read/know which row and column to select!!!**

Example run of a game:

```
a.out -r 9 -c 9 -m 10

-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
```

```
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------

Flag(1) or Open(2): 2
row, col: 0 4

You lose!!!
-------------------
|1|1|0|1|*|1|0|0|0|
-------------------
|*|2|1|2|2|2|0|0|0|
-------------------
|2|*|2|2|*|1|0|0|0|
-------------------
|1|1|3|*|3|1|0|0|0|
-------------------
|1|1|3|*|2|0|0|0|0|
-------------------
|1|*|3|2|2|0|0|0|0|
-------------------
|1|1|2|*|1|0|0|0|0|
-------------------
|1|2|3|2|1|0|0|0|0|
-------------------
|1|*|*|1|0|0|0|0|0|
-------------------

Do you want to play again (1-yes, 2-no): 1
How many rows, cols? 9 9
How many mines? 10
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------
| | | | | | | | | |
-------------------

Flag(1) or Open(2): 2
row, col: 0 0
-------------------
|1| | | | | | | | |
```

```
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------

Flag(1) or Open(2): 2
row, col: 8 0
------------------
|1| | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
| | | | | | | | |
------------------
|1| | | | | | | |
------------------

Flag(1) or Open(2): 2
row, col: 5 5
------------------
|1| | | | |1|0|0|0|
------------------
| | | | | |2|0|0|0|
------------------
| | | | | |1|0|0|0|
------------------
| | | | |3|1|0|0|0|
------------------
| | | | |2|0|0|0|0|
------------------
| | | | |2|0|0|0|0|
------------------
| | | | |1|0|0|0|0|
------------------
| | | |2|1|0|0|0|0|
------------------
|1| | |1|0|0|0|0|0|
------------------

Flag(1) or Open(2): 1
```

```
row, col: 6 3
-------------------
|1| | | | |1|0|0|0|
-------------------
| | | | | |2|0|0|0|
-------------------
| | | | | |1|0|0|0|
-------------------
| | | | |3|1|0|0|0|
-------------------
| | | | |2|0|0|0|0|
-------------------
| | | | |2|0|0|0|0|
-------------------
| | | |!|1|0|0|0|0|
-------------------
| | | |2|1|0|0|0|0|
-------------------
|1| | |1|0|0|0|0|0|
-------------------
```

…

```
Flag(1) or Open(2): 2
row, col: 7 1
-------------------
|1|1|0|1|!|1|0|0|0|
-------------------
|!|2|1|2|2|2|0|0|0|
-------------------
|2|!|2|2|!|1|0|0|0|
-------------------
|1|1|3|!|3|1|0|0|0|
-------------------
|1|1|3|!|2|0|0|0|0|
-------------------
|1|!|3|2|2|0|0|0|0|
-------------------
|1|1|2|!|1|0|0|0|0|
-------------------
|1|2| |2|1|0|0|0|0|
-------------------
|1|!|!|1|0|0|0|0|0|
-------------------

Flag(1) or Open(2): 2
row, col: 7 2
Congratulations!!!
-------------------
|1|1|0|1|*|1|0|0|0|
-------------------
|*|2|1|2|2|2|0|0|0|
-------------------
|2|*|2|2|*|1|0|0|0|
-------------------
|1|1|3|*|3|1|0|0|0|
-------------------
|1|1|3|*|2|0|0|0|0|
-------------------
|1|*|3|2|2|0|0|0|0|
-------------------
|1|1|2|*|1|0|0|0|0|
-------------------
```

```
|1|2|3|2|1|0|0|0|0|
-------------------
|1|*|*|1|0|0|0|0|0|
-------------------

Do you want to play again (1-yes, 2-no): 2
```

Requirements for game:

- You must provide a usage message, if the user enters incorrect command-line arguments, and they can come in any order!
- You must not have functions over 15-20 lines long (-10 automatically)
- You must not use global variables (-10 automatically)
- You must ask the user if he/she wants to play again
  - If no, then end
  - If yes, then prompt for rows, cols, and mines for new game.
- You must not have any memory leaks (-10 automatically)
- You must detect these errors:
  - Invalid row or column to flag or open
  - Opening a cell that has already been opened

(10 pts) Extra Credit: Recursively open all cells adjacent to 0 cell
If the user selects a cell that has no mines surrounding it, then you will help the user by recursively opening all adjacent cells to empty cells, until you reach cells with a surround mine.

Battleship

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. https://www.hasbro.com/common/instruct/Battleship.PDF

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:

cout << "\033[22;31m X \033[01;37m";

\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:

```
  A   B   C   D   E   F   G   H   I   J
1   |   |   |   |   |   |   |   |   |
-----------------------------------------
2   |   |   |   |   |   |   |   |   |
-----------------------------------------
3   |   |   |   |   |   |   |   |   |
-----------------------------------------
4   |   |   |   |   | 0 |   |   |   |
-----------------------------------------
5   |   |   |   |   |   |   |   |   |
-----------------------------------------
6   |   |   |   |   |   |   |   |   |
-----------------------------------------
7   | 0 |   |   |   |   |   |   |   |
-----------------------------------------
8   | 1 | 0 |   |   |   |   |   |   |
-----------------------------------------
9   | 0 |   |   |   |   |   |   |   |
-----------------------------------------
10  |   |   |   |   |   |   |   |   |
Player 1: What position do you choose? 8 A

Hit!!!
You sunk my ship!!!

  A   B   C   D   E   F   G   H   I   J
1   |   |   |   |   |   |   |   |   |
-----------------------------------------
2   |   |   |   |   |   |   |   |   |
-----------------------------------------
3   |   |   |   | 0 |   |   |   |   |
-----------------------------------------
4   |   |   | 1 | 1 | 0 |   |   |   |
-----------------------------------------
5   |   |   |   | 0 |   |   |   |   |
-----------------------------------------
6   |   |   |   |   |   |   |   |   |
-----------------------------------------
7   |   |   |   |   |   |   |   |   |
-----------------------------------------
8   |   |   |   |   |   |   |   |   |
-----------------------------------------
9   |   |   |   |   |   |   |   |   |
-----------------------------------------
10  |   |   |   |   |   |   |   |   |
Player 2: What position do you choose? 4 C
```

```
Hit!!!
You sunk my ship!!!

  A    B   C   D   E   F   G   H   I   J
1    |   |   |   |   |   |   |   |   |
-----------------------------------------
2    |   |   |   |   |   |   |   |   |
-----------------------------------------
3    |   |   |   |   |   |   |   |   |
-----------------------------------------
4    |   |   |   |   | 0 |   |   |   |
-----------------------------------------
5    |   |   |   |   |   |   |   |   |
-----------------------------------------
6    |   |   |   |   |   |   |   |   |
-----------------------------------------
7    | 0 |   |   |   |   |   |   |   |
-----------------------------------------
8  1 | 1 | 0 |   |   |   |   |   |   |
-----------------------------------------
9    | 0 |   |   |   |   |   |   |   |
-----------------------------------------
10   |   |   |   |   |   |   |   |   |
Player 1: What position do you choose? 2 B

Miss!!!

  A    B   C   D   E   F   G   H   I   J
1    |   |   |   |   |   |   |   |   |
-----------------------------------------
2    |   |   |   |   |   |   |   |   |
-----------------------------------------
3    |   |   |   | 0 |   |   |   |   |
-----------------------------------------
4    |   | 1 | 1 | 1 | 0 |   |   |   |
-----------------------------------------
5    |   |   |   | 0 |   |   |   |   |
-----------------------------------------
6    |   |   |   |   |   |   |   |   |
-----------------------------------------
7    |   |   |   |   |   |   |   |   |
-----------------------------------------
8    |   |   |   |   |   |   |   |   |
-----------------------------------------
9    |   |   |   |   |   |   |   |   |
-----------------------------------------
10   |   |   |   |   |   |   |   |   |
Player 2: What position do you choose? 7 H

Miss!!!
```

(10 pts) Extra Credit:
Play a one player Battleship, where the user can play the computer. The algorithm you use doesn't have to be smart, it just needs to work!!!

(10 pts) Program Style/Comments
In your implementation, make sure that you include a program header in your program, in addition to proper indentation/spacing and other comments! Below is an example header to include. Make sure you review the style guidelines for this class, and begin trying to follow them, i.e. don't align everything on the left or put everything on one line!

http://classes.engr.oregonstate.edu/eecs/fall2017/cs161-001/161_style_guideline.pdf

```
/*************************************************
** Program: game.cpp
** Author: Your Name
** Date: 12/03/2017
** Description:
** Input:
** Output:
*************************************************/
```

Electronically submit your C++ program (.cpp file, not your executable!!!) by the assignment due date, using TEACH.