**Project 1**

<Battleship>

CSC 17A - 42824

Name: David Macias

Date: 5/03/2016

**Introduction**

**In Battleship, two players compete to sink each others ships placed on their respective 'Grids' or X and Y tables by calling out a single row and column at a time. A two-sided vertical grid acts a call record and blocks a player's ship grid from the oppenent's view.**

**I have decided to continue a game program from my previous programming class adapting new concepts learned in this class.**

**Development Summary**

* **Program size: 341 lines**
* **Structures Utilized: 4**
* 3 dynamic structures
* 1 internal structure
* Functions Used: 8
* 2 functions return pointers to structures
* 4 functions with pointers in arguments
* 6 arguments in functions passed by reference
* 3 character arrays
* 1 binary file utilized as output and input

The most challenging and part of the project was adapting my existing program with structures. As I was constructing the structures, I discovered that instead of declaring and initializing 4 different variables for each ship, I could just make an array of structures that contained all of the ship variables. From there, I took that idea and adapted it to a structure for the number of players and a structure for number of character arrays in the game which as a whole, dramatically reduced the size of my program. The program roughly took 8 hours to build.

**Description**

The program first generates two 15x15 character arrays. The one the player sees is filled with one character that represents water or ocean and the other that is filled with 5 different sized ships. Each ship is placed at a point randomly as well as position vertically or horizontally as the character 'O'. After the ship grid has been generated, the grid without the ships is displayed to the player along with ship length details, remaining bombs, total ship hits, and total misses.



The player is then instructed on how the input should be formatted and an option to quit the game if desired. The input is read in as a string but then extracted into individual variables of different data types.

Proper formatting is as follows:

* The first character of the input is the letter of the column of the desired strike coordinates.
* The next character is the number of the row of the desired strike coordinates.
* Then if the player wants, a third character 'B' is entered to tell the game that the strike is a bomb strike.

After entering the desired strike coordinates, the game does a comparison between the coordinates entered and the ship grid array. Depending if the index of the ship array contains the character 'O', the game changes the character on the array the player sees to 'H' or 'X'. 'H' meaning a ship was hit or 'X' the strike was a miss.

**Flow Chart**

**Pseudo Code**

**Major Variables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Variable Name** | **Description** | **Location** |
| character | usrGrid[] | Array that the user sees | main() |
|  | shpGrid[] | Array that contains the ships | Main() |
|  | lttrIn | User character input for the array location in the column | Main() |
| short | rows | Number of rows in both grids | Main() |
|  | clmns | Number of columns in both grids | Main() |
|  | shipX(a) | Column location of ship(a) a=[1-5] | Main() |
|  | shipY(a) | Row location of ship(a) a=[1-5] | Main() |
|  | shipL(a) | The number of characters the ship(a) is. a=[1-5] | Main() |
|  | shipCnt | Counter for the number of ship characters in the ship array | Main() |
|  | hitCnt | Counter for the ships that are hit | Main() |
|  | rowIn | User input for the row location | Main() |
|  | dfflty | Number input to change the array sizes | Main() |
|  | menuNum | Number option for start game, change difficulty, or see ranks | Main() |
| Boolean | hit | Becomes true when user column, row input matches ship column, row | Main() |
|  | quit | Becomes true when user inputs the command to quit the game | Main() |
|  | jstStrt | Becomes false after the user grid is printed | Main() |
|  | modeSet | Becomes true after the option to play the game is inputted by user | Main() |
| string | intl | User’s initials | Main() |
|  | fLine | Lines read from the external in file | Main() |

**Reference**

* Textbook
* <http://www.cplusplus.com/forum/lounge/52458/> -Reading in From file

**Program**