Programming Assignment #2

This assignment will give you some practice with object-oriented coding (classes, objects, constructors, destructors, etc.) The task is to simply convert the previous WarBoats assignment (implemented as a procedural program) into an object-oriented version. Because you can re-use almost all of your code from the previous assignment, this shouldn't take too long. The difficult parts have already been written.

Some Details (from the previous handout)
Instead of simply hard-coding size of the board at 10x10, we are going to allow the player to specify the dimensions of the board. (The board is the ocean.) We should be able to handle boards of any size (square and non-square). The program will only be limited by the amount of memory in the computer. We'll also allow the player to specify the number of boats to place in the ocean. The only limit being the size of the ocean, since boats will not be allowed to overlap or leave the ocean. (Contrary to what Ferdinand Magellan's crew claimed, the world is flat, and if you go to far, you will fall off the end of it.)

All implementation will be placed in `Ocean.cpp`. You are not allowed to include any other files in `Ocean.cpp`. There are a couple of files included already, but you will not need any others.

New Stuff
The interface to the game is included in a header file named `Ocean.h`. This file contains all of the information that the client (the player) needs. A partial `Ocean.cpp` file is provided which includes the implementation of a couple of new methods.

New Method Details

Method	Description

`Ocean(int num_boats,`
`int x_quadrants,`
`int y_quadrants);`
	Constructor - The client calls this to create an ocean (game board). Client specifies the dimensions of the ocean and the number of boats that will be placed in it. All private fields are initialized. No return value.

`~Ocean();`	*Destructor* - This method is responsible for deallocating anything that was allocated in the constructor. No return value.

Since 95% of the internal code is going to be the same as the previous assignment, it shouldn't take a lot of time to implement. It is just a matter of going through the existing functions and modifying them to be part of the *Ocean* class, instead of simply being global functions. Each private data member should be named with a trailing underscore to make it clear that the variable is part of the class. You should follow this convention in future assignments when you will be required to create many of your own members. Remember to use **const** wherever appropriate. This includes both function parameters as well as member functions. Also, if you lost points on the previous assignment you should make sure and fix all of the problems so that you won't lose points again for the same thing (e.g. incorrect output, long lines, no comments, memory bugs, etc.).

What to submit

You must submit your header file and implementation file (Ocean.h and Ocean.cpp) in a .zip file to the appropriate submission page. Note that you are not submitting any other files.

Source files	Description
Ocean.cpp	The implementation file. All implementation for the functions goes here. You must document the file (file header comment) and functions (function header comments) using the proper techniques learned in CS120.
Ocean.h	The header file. You will need to modify this. **Absolutely NO implementation is permitted in this file.** You must provide a file header comment and other necessary Doxygen comments.

Usual stuff
Your code must compile (using the compilers specified) to receive credit. The code must be formatted as per the documentation on the website.

Make sure your name and other info is on all documents (paper and electronic).

More on back →

A partial definition of *Ocean* (Ocean.h):

```
namespace CS170
{
  namespace WarBoats
  {
    const int BOAT_LENGTH = 4;  // Length of a boat
    const int HIT_OFFSET = 100; // Add this to the boat ID

    class Ocean
    {
      public:

          /*
           * Other public methods
           */

           // Provided
        const int *GetGrid() const;
        Point GetDimensions() const;

      private:
        int *grid_;        // The 2D ocean
        int x_quadrants_;  // Ocean size along x-axis
        int y_quadrants_;  // Ocean size along y-axis

        /*
         * Other private data
         */

    }; // class Ocean

  } // namespace WarBoats

} // namespace CS170
```

A partial implementation of *Ocean* (Ocean.cpp):

```
#include "WarBoats.h"
#include "Ocean.h"

namespace CS170
{
  namespace WarBoats
  {

    /*
     * ALL STUDENT IMPLEMENTATION GOES HERE
     */

    const int *Ocean::GetGrid() const
    {
      return grid_;
    }

    Point Ocean::GetDimensions() const
    {
      Point pt = {x_quadrants_, y_quadrants_};
      return pt;
    }

  } // namespace WarBoats
```

```
} // namespace CS170
```