**CSCE 155N Matlab Programming Project 2 – Summer 2017**

**Assigned: Tuesday 7/30/2017**

**DUE DATE:**

**Electronic: Friday 8/4/2017 at 11:59 PM**

**Hardcopy: Monday 8/7/2017 in class**

**Sea Battle**

**Problem Statement:**

This project has elements of the classic Battleship game, Stratego, and Rickoshay, all of which are developed to some extent on my class web site. There is some setup required prior to play beginning. Placing one’s pieces (ships, reflectors, and mines) is done without the opponent being able to see where they are on the board. During actual play it may be possible to deduce where they are and to take countermeasures.

The primary goal of the game is to reach the home base of the opponent with one of one’s ships. If one’s ships are all sunk, the opponent instantly wins.

Off the edge of the board each player has a long-range bomber that can fire either laser beams or torpedoes rectilinearly into the board at any of the numbered points around the edge. The laser weapon differs from the torpedo in one important detail: The laser beam will reflect off the reflectors and either (eventually) hit and damage a ship or exit the board, where its exit point can be detected. Using the laser, one may be able to locate and determine the orientation of reflectors. The torpedo can explode and destroy a reflector or damage a ship or miss everything and exit on the opposite side of the board. Note that one’s weapons are capable of damaging one’ own pieces!

The sample board shows ships of various lengths labeled with upper case letters, pointed in any of 8 points of the compass. There are reflectors (the line segments) similarly oriented. There are mines, capable of damaging any ship that hits one. There is a diagonal line dividing the board in two sectors. Each player places all his pieces in his/her own sector. Only the ships can be moved, ideally crossing the dividing line and entering the base square of the opponent (at the far upper left and lower right corners). Note some of the mines being strategically placed near the bases.

Play alternates between the two players. Following the initial setup, each play may consist of one of the following:

1. Rotate one of your reflectors to a direction of your choice.
2. Rotate one of your ships to a course of your choice.
3. Move one of your ships one step in its current course. If it hits another ship broadside, it damages that ship. If the hit is not broadside, both ships are damaged. If it hits a mine, the mine explodes and your ship is damaged. A ship will be dead in the water if the back segment is damaged. A ship sinks if all segments are damaged. Ships move safely between reflectors without hitting them.
4. Fire your laser from off the board and monitor it until it hits a ship or exits the board.
5. Launch a torpedo from off the board and similarly follow its path.

Along with any move (before or after), a player may place buoy(s) to mark the suspected location(s) of enemy pieces. Confirmation is returned, and in case of a ship it will move along with that segment of the ship. It is essential to redraw the screen between plays so that a player only sees his/her own pieces along with all the buoys.

**Collaboration:**

Work together as a team on any or all aspects of the research and design. Ideally take advantage of the talents of each member of the team. Use your teams to finalize each of the multiple design options. It is essential to keep track of who did what and where any useful information was found. You need to keep track when you help someone and when you receive help from someone. This includes students from other teams and those outside the class, TAs and me (the instructor). Keeping a log is highly recommended.

**What and How to Submit:**

Read and have your program conform to the “Program Documentation Guidelines” which were provided previously.

As you make progress with the project, periodically handin (electronically) updates numbered as mine are on my site. How many updates really depends on what seems natural. It might be 5 or it might be 15. Anyway, we should see a progression of “working” programs handed in as ***seaBattle1.m***, ***seaBattle2.m***, etc.

By the deadline hand in electronically the two files, seaBattle.m (the Matlab function file for the game), and seaBattle.doc (which contains summaries, documentation, and sample runs).

Each team member should submit his/her own analysis of the relative contributions of all the members toward the project. This should be submitted electronically using each member’s handin account. This is in addition to the acknowledgement section of the main report. Assuming allocation is fairly even, all will receive the same grade.

The Word document should contain the following, all carefully labeled:

* Cover page with name(s) and the account under which it is submitted, title, date submitted, etc.
* A discussion of the features you implemented in the project. Describe how they work and what Matlab options were used to program them. This should be at a fairly high level, not a line-by-line analysis of the code.
* An “instruction manual” that a non-programmer can use to set up and run the game.
* An annotated cut and paste sample dialog sampling of the running of the game. (Hint: Use the ‘diary’ command or cut & paste as appropriate.)
* A discussion of the testing that was performed. This should include testing of each component as it was being built, and testing of the final program ensuring that it works properly under a comprehensive range of conditions.
* An annotated cut and paste of a sample dialog, demonstrating how your program responds to extreme and faulty input. (This could be combined with the previous section.)
* Acknowledge all collaborations (both internal to the team and external), detailing what each person contributed individually, and what was done jointly. Indicate approximate percentages of the work contributed by each person in design, coding, testing, documentation, and report preparation.

**Grading Criteria:**

* Properly running features – 30%
* Program logic is well designed – 20%
* Progress versions – 20%
* Documentation guidelines are followed – 10%
* Handin Documents formatted and arranged as specified – 10%
* Testing is comprehensive – 10%

