Lab09: README.pdf

Ryan Dong
(rfdong@ucsc.edu

Solo Project

I WROTE:

-AGENT.C

-FIELDTEST.C

-FIELD.C

I would like for all of the files above to be graded, although they are incomplete. I wrote all of these files and there are no extra credit opportunities within them. For collaboration, I asked Psi Padhya on how to send CHA, ACC, and REV messages.

LAB REPORT:

The Battleboats System works by two u32 boards communicating with one another. The top level framework was already completed and provided for us. The top level framework consists of a transmission service, a button service, and an agent service. These depend on submodules to work in full. The Message submodule decodes and encodes messages between the two u32 boards. The module translates incoming messages into convenient data structures and also translates outgoing data into messages. The Agent submodule is the main decision maker of the program. The module serves as a large state machine that manages the game playing activity and also updates the Oled display. The Agent submodule relies on the Field and Negotiation modules. The Field submodule manages the BattleBoats location, boat states, hits, and misses on the board. The module holds function definitions that are used within the agent module that does numerous different things such as placing the boats and updating the board after a guess was made. The negotiation submodule provides helper functions to determine the turn order fairly. It uses a commitment scheme to determine a coin flip on who goes first. These systems work in tandem to run the Battleboats system.

My work strategy and approach to this lab did not work at all. I started this lab 3 days prior to the deadline, believing that I would have ample time to finish. I coded much of the agent module and the field module, however my problems arose when I began to test my code. I made

the mistake of not testing my code until relatively last minute which exposed numerous bugs in my code. These bugs were very difficult to fix, and I would not be able to fix my code in time for the deadline. While I thought the code was relatively simple and in my eyes it should be working, I clearly must have made a small mistake which caused my code to not work. I also made the mistake of coding the agent file before testing my field.c file. Because my field module has bugs and is incomplete, my agent module also does not work and is incomplete. While I was testing the FieldAIPlaceBoats() function, I realized that there was a problem with my already tested FieldPlaceBoats function. This was very confusing to me as I already tested placing down the boats. As I began to test more and more, the bugs in my code began to be exposed and I realized the scope of debugging was beyond what I could accomplish in the time limit. Unfortunately, this lab was a clear cut case of "failure to plan, plan to fail". At the beginning of the testing process, the strategy I was using to test my functions was working well. I had a passed tests counter which I would increment if certain if statements were fulfilled.

Through this lab, I learned how to generate random numbers, integrate different submodules to work with each other, further improved in working with finite state machines, and will further improve time management skills in the future. This was a real wake up call in terms of keeping up with material and being on top of work even until the last minute of the class.

In this lab, I enjoyed programming the finite state machine in the agent submodules and coding the function definitions in the field module. I feel that coding function definitions and finite state machines is very satisfying and I enjoy thinking about the logic in the code. I did not enjoy testing my code, as it made me realize the amount of bugs present in the code. I would not change anything about this lab, I feel it is a good final project that encompasses the ideas taught throughout the quarter.