Retrospective – Project 2

Official Fortran Fanclub

Meeting Log

1.     Date: Friday, September 21, 2018

Location: In-class

Attended: All members

Outcomes: Discussed ideas for planned feature – multiple lives, GUI, multiplayer functionality, leaderboard

2.     Date: Monday, September 24, 2018

Location: In-class

Attended: All members

Outcomes: Finalized idea for planned feature – chose multiplayer mode, planned meeting schedule

3.     Date: Wednesday, September 26, 2018

Location: In-class

Attended: All members

Outcomes: Discussed multiplayer functionality – cooperative multiplayer vs. competitive multiplayer; how each mode of play looks and functions

4.     Date: Wednesday, September 26, 2018

Location: LEEP2

Attended: Sydney Combs, Daniel Gonzalez, Nathan Pelletier

Outcomes: Decided workload division - divided up necessary classes & functionalities.

5.     Date: Friday, September 28, 2018

Location: In-class

Attended: All members

Outcomes: Discussed progress, discussed multiplayer functionality and other proposed features

6.     Date: Friday, September 28, 2018

Location: Spahr

Attended: All members

Outcomes: Bug fixes (input and bounds checking) implemented, timer implemented, basic cheat mode finished, worked on multiplayer

7.     Date: Monday, October 1, 2018

Location: In-class

Attended: All members

Outcomes: Discussed progress, planned next meeting

8.     Date: Wednesday, October 3, 2018

Location: Eaton

Attended: All members

Outcomes: Cheat mode finished, leaderboard started, continued work on multiplayer modes

9.     Date: Wednesday, October 3, 2018

Location: In-class

Attended: All members

Outcomes: Discussed progress, discussed multiplayer feature

10.  Date: Friday, October 5, 2018

Location: In-class

Attended: All members

Outcomes: Discussed progress & remaining steps, set meeting time

11.  Date: Friday, October 5, 2018

Location: Eaton

Attended: Daniel Gonzalez, Daniel Hidalgo, Nathan Pelletier

Outcomes: Worked on multiplayer modes, figured out final implementation of networking aspect, settled on methods to use in implementation (send\_comm(), etc. as opposed to send\_json, send\_pickle…)

Workload Division

Sydney Combs

* Bug fixes
* Cheat mode

Daniel Gonzalez

* Multiplayer implementation

Daniel Hidalgo

* Networking API
* Network data manipulation

Nathan Pelletier

* Timer
* Leaderboard

Challenges

Sydney Combs - The biggest challenge in my personal part of the work was working through and understanding our inherited code, but, to be honest, it wasn’t a terribly difficult challenge. The most difficult part was creating the cheat\_show function. This was done in order to properly format cheat mode. However, the use of colorama made understanding the existing code a little more confusing. It didn’t take too long to figure out, however, simply because it wasn’t that dramatic.

Daniel Gonzalez - The networking aspect was the biggest hurdle for this project, Daniel H and I had an initial basic idea of how to implement it but ended up changing ideas a lot. We overcame this challenge by talking with other team members, Nathan came up with final design. We also had the problem of the previous team’s implementation not being nicely portable to windows operating systems, we overcame this problem by deciding it would look better keeping it the way it was.

Daniel Hidalgo - I… made a mess. Networking was a fancy new toy and I wanted to play with a ton of different things and so a lot of artifacts and bad decisions were left in the code for too long (and some ultimately stayed). Not deciding on a consistent data format until very late forced me to write various (de)serialization and communication functions and eventually led to a huge mess because it was unclear whether data being passed in string or byte format, whether it had to be (de)encoded and when, etc.

Nathan Pelletier- For me, the biggest problem was actually trying to figure out how to score the minesweeper game.  I had to do quite a bit of research to figure out the correct method of scoring, 3BV. After that, it was fairly straight forward with my coding, although I had to change a bit of code.

Unused Features

·      The team originally discussed implementing a GUI. For whatever reason, we decided that implementing a GUI was not necessary (or might make the multiplayer feature more difficult to implement).

Retrospective

·      Sydney Combs - While I know I did everything that we agreed on - as far as division of work - I feel that I could have done a lot more to contribute to the team. My part of the project was simple and fairly isolated from the networking portion, which was most of the work for this project, since it only dealt with basic game functionality and cheat mode, which we decided only applied to single-player games. In the future, I would like to contribute more fully to the final product, and communicate more effectively with my teammates.

·      Daniel Gonzalez -  I wasn’t able to spend much time working on the project until the final week, so I spent a lot of time feeling useless. For next projects I want to work on getting started right away. I also should have spent more time working out the networking side, as I mostly just waited for Daniel H. to figure it out, could’ve solved any problems we came across earlier. We should also have discussed this more with the other team members, Nathan came up with the final implementation which was much simpler than our ideas.

·      Daniel Hidalgo - I had never done any sort of socket programming before (save for the socket exercise in EECS 563 earlier in the semester) so early on I was playing with a lot of different networking ideas. Unfortunately, I gave all those examples to the rest of the team to take a look at or mess with and it resulted in some confusion (at least on my end) and messy code eventually making its way into the final product.

I had originally told my team that they could use three different data structures to send and receive data but that resulted in confusion because we weren’t sure how to properly implement a client / server set up. We did eventually decide to stick with one data structure (python dictionaries) that helped simplify most communication.

After writing my network API code I barely played with it and **knew** that it was going to be annoying to work with since so much had to be done manually, but since I originally told the team that we could use three different data structures and be okay, the thought kind of slipped my mind. Only after helping finish some multiplayer code did it dawn on me how truly awful it must have been for Daniel G.

Lesson learned: Don’t just write code that works and leave it at that. Like with comments, it’s fairly obvious to the writer how things should work… but the people actually using it might not have any idea and so you have to put yourself in their position for a little while.

\*\*\* Feature creep. I need to stick to what the team decides and stop trying to add ‘interesting’ new ideas

·      Nathan Pelletier- I personally thought that the idea to do a multiplayer was pretty interesting, but honestly had zero knowledge or understanding of the networking aspects so I ended up doing the leaderboard and timer instead.  Overall, I think that my portion of the code was much less work intensive in terms of trying to figure things out. One thing I learned was to make sure to try and state my opinions to the group earlier so that I can try to limit any unnecessary coding.  This project I ended up not commenting on the networking, and so made both Daniels have to do a lot of extra work that they just ended up scrapping. Next project, I’ll try and communicate my ideas sooner and more effectively.