**EECS 448**

**Project 1: Minesweeper!**

**Log of all Meetings**

Date: 02/04/2019

Location: Learned 3153

Members: All were present

Briefings/Notes:

Decided how GIT project would be structured

Finalized on Python and Pygames as project plan

Started delegation of the work – created a sample class structure to build game upon

Pseudocode of board started

Date: 02/05/2019

Location: Spahr Library

Members: Marcus and Ryan

Briefings/Notes:

Discussed Tiles and Board class in further detail to finish pseudocode

Date: 02/08/2019

Location: Spahr Library Private Study

Members: Marcus, Ryan, Taylor, Megana

Briefings/Notes:

Established new class structure after confusion over working for a while

Minesweep (exe)-> Game->UI->GameBoard->Tiles

Tried to clarify and confusion we had

Reworking Tile class implementation and added new methods to code

Date: 02/09/2019

Location: Spahr Library Private Study

Members: Marcus, Ryan, Taylor, Megana, Myra

Briefings/Notes:

Reevaluate Tile class – start coding UI first in order to make this easier

Date: 02/10/2019

Location: Spahr Library Private Study

Members: Marcus, Ryan, Taylor, Megana, Myra

Briefings/Notes:

Clean up Tile class and re-delegate the work accordingly

Pygames and UI

Establish what is needed from pygames

Board creation

Flag creation and usage

Any exceptions needed?

Date: 02/15/2019

Location: Classroom

Members: Marcus, Ryan, Taylor, Megana, Myra

Briefings/Notes:

Finish the board game

Fix bugs, last minute changes, anything to add?

**Work Delegation Among Members**

Most of the work was distributed at the first meeting itself. We all picked a class or certain methods to start working on (as seen in the attached Meetings folder). However, due to confusion in overall plan and implementation of logic, we had to redelegate tasks as they arose. Everyone was in charge of fixing bugs as they arose. Specific tasks included implementation of Pygames within the user interface as well as tile class in order to create the Minesweeper board. For instance, Taylor did a lot of the gameboard and pygame implementation, while Myra and Megana worked on tile methods. Ryan and Marcus heavily focused on pygame flagging, clicking, etc. Over the course of the project, division of work became less rigid and multiple team members began collaborating on similar pieces and methods of the project. Overall, the team worked to make sure we not only worked on our own parts, but also were able to help others as issues arose. The structure of team working was centered around achieving and coding small tasks and bits of code and fixing them as needed.

**Challenges**

The biggest challenge we found was being able to understand the logic that each of us had in our head and to implement that exactly. We spent about a week of the project with certain plans in our head and ended up needing to change the class structure in order to implement this most efficiently. Luckily, our team was very receptive to feedback and other’s opinions, that we were able to easily come to a consensus on plans and changed them accordingly. We felt as if the project wasn’t SO big that the delegation would be a lot on each person. Rather, we found it easy to split up the work into tiny tasks and help each other, especially since Python was new to most all of us. Pygames, an imported library Python provides for game development, was a pain for us to learn and try to implement. But Google became our “go-to”, as we found tutorials and other programmer information to help us through the process of learning.

**Things not in Final Product**

Originally, our class structure was set in such a way that Minesweeper (exec) -> Game (exec) -> User Interface-> Gameboard-> Tiles-> Styles would run the game. However, after redefining our class structure, we got rid of the Game class entirely, seeing it unnecessary when an executable was already present and most of the board pieces were handled within Tiles and Gameboard.

We also discussed possibly having a game timer or something that would count our mines on the actual board, but never got around to doing so.

One last thing, which we never coded but only discussed, was actual implementation. There was talk for a bit about having two boards – one that was unchanged and compared to the board being implemented. We thought it better to have one board and manipulate counting variables to win or lose the game instead.

**Retrospective on Things to do Differently**

Overall, we were very happy with the way the project turned out and how we were able to work together as a group. Walking into this, we all had an EECS 268 mindset on how to divide and conquer this ourselves. But working on this as a group helped us divide the needed work as necessary. Next time, we would like to work on a class structure that we all understand and agree upon from the start. That way dividing the work and creating methods accordingly will be easier. If we establish this within our first meetings or so, we will know exactly how to get to work without confusion. Another thing was to make sure we had some way to test our code (in some sort of testing environment) as we went, to fix bugs as they arose. Some methods were coded early on, and we couldn’t check functionality until other things were implemented. This caused us to believe we were done with a task when we may have actually not been done. Overall, we have had a very positive experience and were successfully able to try to learn Python and Pygames. Although we were not HUGE fans of pygames, we still see it as a viable platform for future projects.