[Insert Rad name here]’s Retrospective Write-up

Log of Meetings:

Color code for meetings-

Sandy	**Evan**	**Alex**	**Clay**

Meeting One - [2/8/2019]
Location - Engineering
Overview

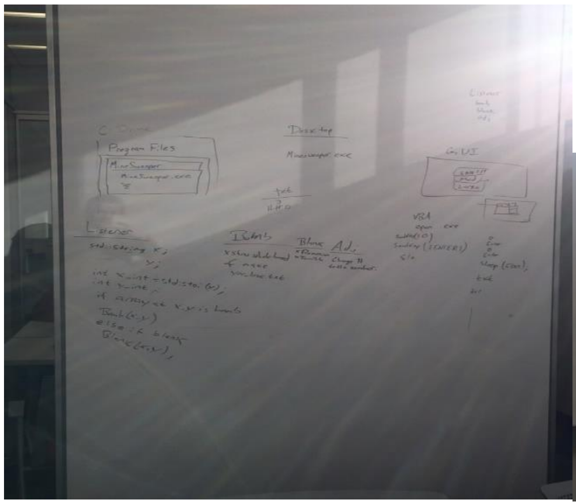
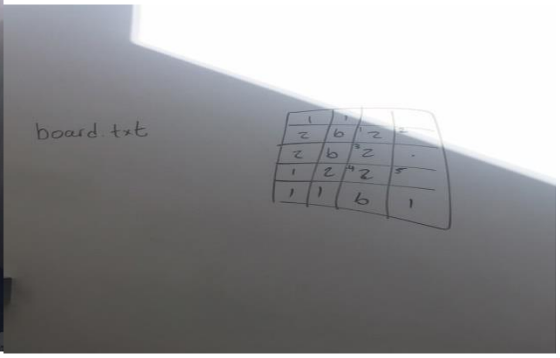
- Focused on the directory structure of the program itself including all text and executable files
- Formally described the keys within the text file
    - Bomb = B, Blanks = - , Hidden = H , Adjacents = 1~8
- C++ Style guides
    - Google Style Guides
- Established goals for the next few days

Goals

- By the end of the weekend (Feb 11th), have the following functions *attempted* with questions brought to team (*ideally* 1 per person).
    - Listener [**Clay**]
        - Takes in the array coordinate as input
        - *if statement* to choose the next function to use
    - Bomb [**Sandy**]
        - End the game
        - *X shows whole board & makes you_lose.txt*
    - Blank [**Alex**]
        - Recursively display all the surrounding blank/adjacent coordinates
        - *Recursion*
        - *Terrible*
            - *Follow up: fix backtracking issue - Evan*

    - Adjacent [**Clay**]
        - Reveal *only* that coordinate and it’s number prescribed to it
        - *Change H to the number*
- Make button clicks happen through visual basic [**Evan**]

Meeting Two - [2/11/19]
Location - Engineering

Overview

- Resolved potential merge conflicts and github techniques
- Discussed progress thus far, most complex functions are still being completed
- Pushed if user loses/ bomb is found funct onto github

Goals

- Complete all *path* functions by wednesday (2/13/19)

Meeting Three - [2/12/19]
Location - Engineering
Progress

- VBA display works with buttons and flags and *board.txt* is edited upon a button click
- VBA *still* needs to update tiles at button clicks
- Walked through VBA code, what is happening as VBA accesses mouse buttons and reads in from text file
- Evan helped everyone get a copy of VBA for each team member to locally test the game

Goals

- *Game over* function, "win condition" **Sandy**
    - Once all bombs have been flagged and all adjacent/none tiles have been revealed, display a *meme* and terminate the game
- Guarantee a single button works **Evan**
    - Follow up goal: Copy and Paste VBA button code **Clay**
- *updateTiles* function **Alex**
    - Take the text file in and "re-skin" all the buttons

Meeting Four - [2/13/19]
Location - Engineering
Goals

Same as before:

- *Game over* function, "win condition" **Sandy**
    - Once all bombs have been flagged and all adjacent/none tiles have been revealed, display a *meme* and terminate the game
- Guarantee a single button works **Evan**
    - Follow up goal: Copy and Paste VBA button code **Clay**
- *updateTiles* function **Alex**
    - Take the text file in and "re-skin" all the buttons

Bugs to Fix

- CheckLoss has a segfault **Evan**

Notes for *UpdateTiles*

- If image on the tile is the flag, then Button.BackColor = Color.FromARGB(0,0,0)
- If the button is unclicked then Button.BackColor = Color.FromARGB(0,0,64)
- If the button has image 1.png then Button.BackColor = Color.FromARGB(0,0,1)
- If the button has image 2.png then Button.BackColor = Color.FromARGB(0,0,2)
- etc.

Meeting Five - [2/16/19]
Location - Starbucks
Last Things to Do:

1. Write up a final, formal, retrospective. - **Sandy**
    a. It should be a single document, 3-5(+) pages.
    b. It should contain a log of ALL meetings.
    c. It should contain a description of how all work was split.
    d. Challenges and how they were overcome.
    e. Any features that did not make demo and why.
    f. A retrospective on what we would have done differently.
2. Plug in all 100 buttons, the first 20 are done. - Clay
3. Finish beautifying VBA forms. - Evan

Additional Comments

Evan - We should all group call tomorrow and make sure everyone has the most recent, final, production version and that it works on all machines.

Description:

Starting out this project, we decided to code the backend of our minesweeper in C++ since it was a language that we have all mastered through Programming 168 and 268. For the frontend, we wanted to challenge ourselves and learn something new and make the minesweeper game easy and intuitive for the user to play. Thus, with Evan's expertice, we decided to code the whole user interface in VBA to accomplish this goal. We decided to tackle the backend of our game first to create a solid foundation for our program to interact with VBA. Through VBA, the user "clicks" or selects a game tile where the coordinates are sent to our C++ program which tells our VBA from what that action entails. This means that VBA is essentially playing the C++ version of the game.

How work was divided:

Backend in C++:

Evan -

- Created and and randomized our “game board” with bombs, adjacent number and blanks.
- Fixed backtracking issue in / finished recursive method.
- Created methods to write to files / reset files to empty on program start.
- Created comments for all methods.

Clay -

- Dealt with user input from the coordinates of the game tile that the user “clicked”.
- Called methods if the tile was a bomb, adjacent number, or a blank.
- Created method if tile was an adjacent number which showed the number of bombs around that tile
- Linked VBA code

Alex-

- Created the initial recursive reveal method for blank tiles.

Sandy -

- wrote the the method if a bomb was found.
- checking the tile and making there was a bomb.
- created a you_lose.txt file that will be looked for in VBA to trigger game end.

Frontend in VBA:

Evan -

- Created initial forms.
- Finished implementation of UpdateTiles
- Created a button array.
- Handled all button click code.
- Handled linking of VBA to C++ with the VBA SendKeys / Shell / AppActivate methods.

Clay -

- Linked code to 100 buttons.

Alex-

- Started implementation of UpdateTiles

Sandy -

- Wrote the winning condition
  - Checked if all bombs were flagged by user.
  - Checked if user only flagged bombs (not adjacent or blank tiles.)
  - Created winning message box
  - Created message box if user flagged non-bombs

Challenges:

VBA button generation- since we chose to do frontend in VBA and we had a large game board of 10x10 (100) buttons, this mean that we needed to different unique properties for each button. This was overcome by the brute force of copy and paste. Clay cranked out this strenuous task with the support of inspirational music.

VBA communication with C++- Since the user only interacted with the VBA form, these user inputs had to be translated to our C++ form so that the program knew what to do once the user selects a tile. This proved more complicated once we realized we had to recognize time as a factor. Sometimes button clicks from VBA came faster than C++ could deal with, so we had to use some thread sleeping methods.

Features That Did Not Make the Demo:

- The “Memesweeper”- have bank of memes that would appear when user won the game
- Other sized board(s)- we only had a small and large board size for user to select

What We Would Have Done Different:

- Not used VBA so we could have easily generated more board sizes as specified by the user
- Learn Python framework instead to generate game boards