Jade Braun

Mark Lehr

CSC 17A

20 October 2019

Introduction

For my project I decided to create a text based version of minesweeper. I used netbeans to write text based code to play the game. I wanted to create minesweeper because I have many fond childhood memories of asking my mom if I can play minesweeper with her. This project taught me a lot about how functions interact with each other and how deeply I can nest loops.

Summary

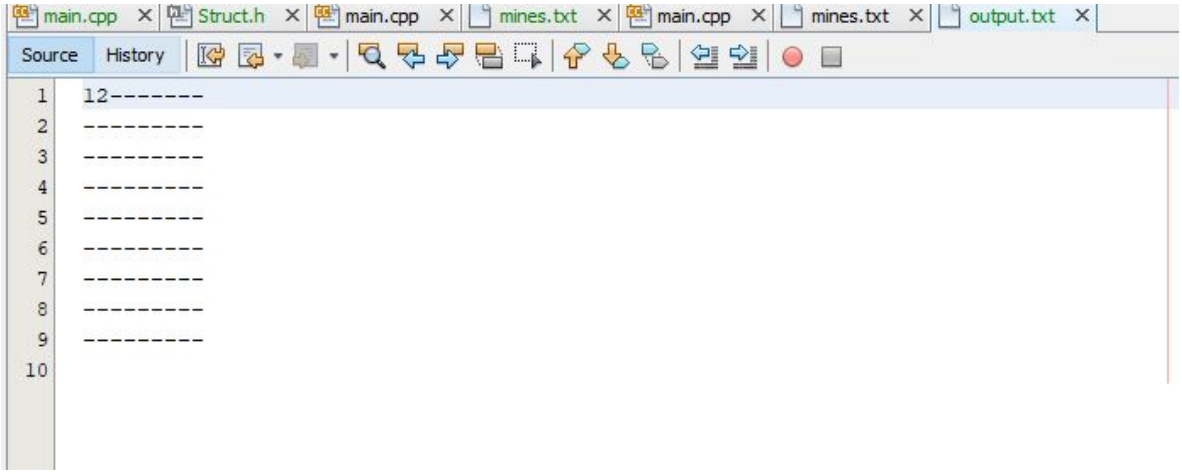
My program is about 19 kilobytes without the build and dist folders. It is 326 lines long but there are also certain spaces in between some functions and lines to make the code look cleaner. My project uses structures to refer to difficulty level and I attempted to create a cheat sheet using a read and write file, but it just outputs into the write file the last attempted row and column input before you hit a mine. I utilized character arrays for the board. I used while loops to check input validation. I used stringstreams because after hacker rank I wanted to try to understand them more. Creating minesweeper took me a full week of four hours daily trying to figure out the setup of the game. At first I wanted to use a 2D pointer array to create the board, but I could not get it to display in the output screen, so I scrapped that and made a regular static 2D character array. One of the challenges I faced was that when I would look up reference code all of the

codes used classes which I am not efficient with yet. I read through Gaddis to understand classes more and how I could potentially take pieces from the references and form the game.

Description

The solution to my problem of setup was to create a 2D character array, create another character array which would hold the randomly placed mines in it. I could use this mine array to count how many mines the tile you just revealed had around it. I could also use the mine array to create those chunks of walls that appear when there are multiple zeros around. That means that whenever there were multiple zeros in the area of the board, it would reveal those spots until it hit a non-zero number.

Sample input/output:



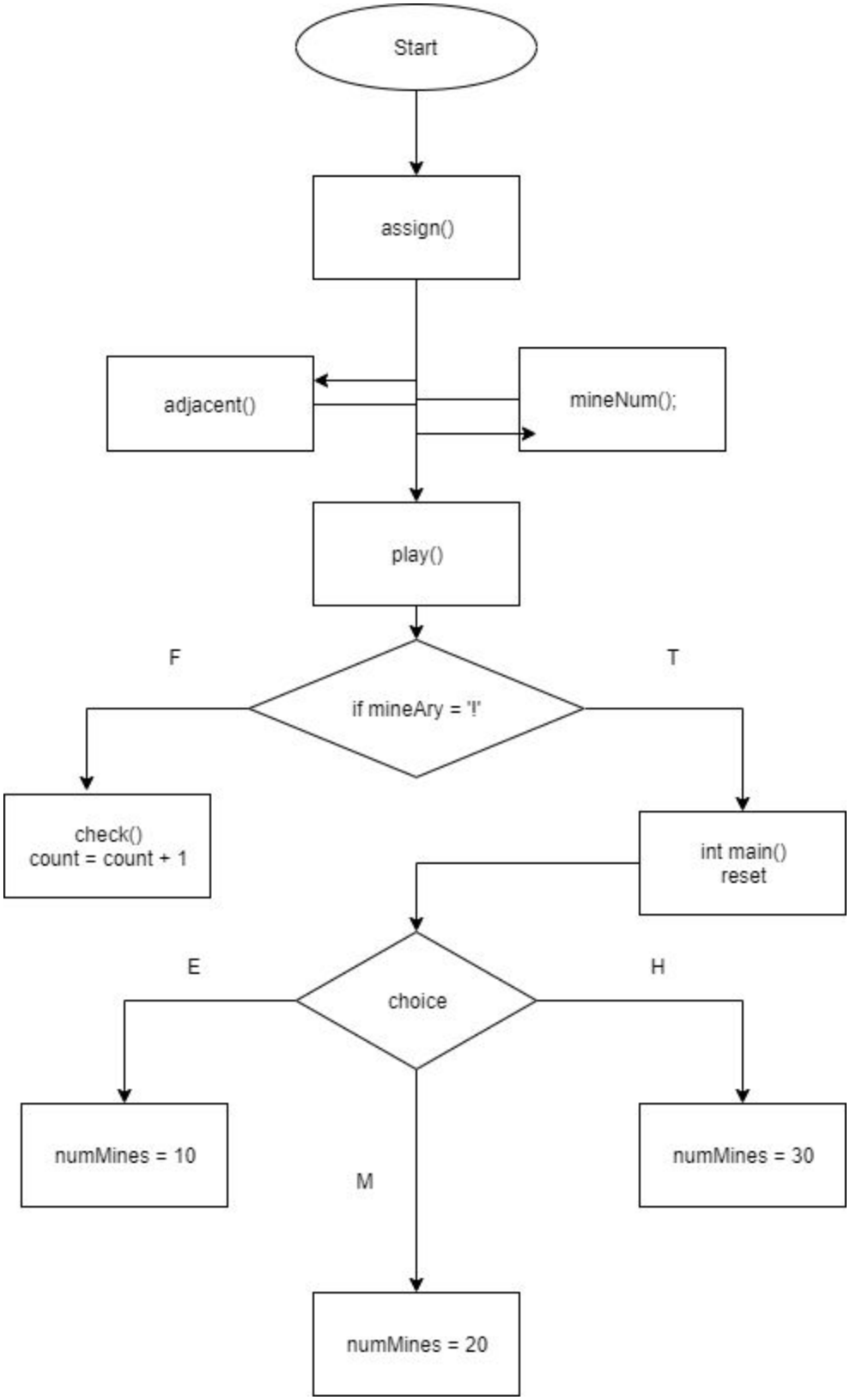
```
Would you like to play easy or hard mode.
Easy mode only has 10 mines intermediate has 20 and hard mode has 30.
Enter E for easy, M for medium and H for hard.
t
Invalid entry, please pick a level difficulty (e/m/h): 32
Invalid entry, please pick a level difficulty (e/m/h): 45
Invalid entry, please pick a level difficulty (e/m/h): -9
Invalid entry, please pick a level difficulty (e/m/h): -1
Invalid entry, please pick a level difficulty (e/m/h): ;
Invalid entry, please pick a level difficulty (e/m/h): -p
Invalid entry, please pick a level difficulty (e/m/h): h
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
Enter the number of the row you'd like to choose a spot (1 - 9)
1
Enter the number of the column you'd like to choose a spot (1 - 9)
2
- 2 - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
Enter the number of the row you'd like to choose a spot (1 - 9)
1
Enter the number of the column you'd like to choose a spot (1 - 9)
1
1 2 - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
- - - - - - - - -
Enter the number of the row you'd like to choose a spot (1 - 9)
```

```
Enter the number of the row you'd like to choose a spot (1 - 9)

Enter the number of the row you'd like to choose a spot (1 - 9)

Enter the number of the row you'd like to choose a spot (1 - 9)
3
Enter the number of the column you'd like to choose a spot (1 - 9)
4
BOOM!
Game Over
1 2 ! ! 2 ! 1 1 1
! 4 5 4 4 2 3 2 !
3 ! ! ! 4 ! 3 ! 3
3 3 5 5 ! ! 3 2 !
1 2 ! ! 4 3 1 1 1
2 3 ! 5 ! 3 1 1 1
2 ! 3 4 ! ! 3 3 !
5 3 3 ! 5 ! 5 ! !
! ! 2 1 3 ! ! 4 !
Play again? (y/n):
```

Flowchart:


Start
assign()
adjacent()
mineNum();
play()
F
T
if mineAry = 'l'
check()
count = count + 1
int main()
reset
E
H
choice
numMines = 10
M
numMines = 30
numMines = 20

Variables:

I used three structure variables to call the mine count. There were two arrays which copied into each other to fill the board with randomly placed mines. I used x and y as position integer variables (rows/ columns) to locate accessible spots on the game board. If the position chosen has already been revealed then a while loop ensures the user can enter a valid position point on the board that hasn't been selected yet.

Concepts:

I learned how to use the rand function and also how to create a random seed. I learned how to use stringstreams which I had never done before this project. I attempted to create a read and write file that created a cheat sheet, but it still writes some of the game board into the ouput.txt file.

References:

http://www.cplusplus.com/forum/general/69782/
https://github.com/mukund26/minesweeper/blob/master/minesweeper.cpp
http://www.cplusplus.com/forum/general/160996/
http://www.cplusplus.com/forum/general/96150/
https://www.youtube.com/watch?v=c8wswUEfnrQ
https://www.daniweb.com/programming/software-development/threads/179212/c-minesweeper-help

Program:

This is the link to my github so you can view the cpp file directly from there and copy and paste into netbeans to play the game yourself.

https://github.com/Jadiepie-658/CSC17A-47827/blob/master/Projects/Project%201%20Mineseweeper/main.cpp