# READ ME! v2

CMSC21 LB1A Final Project: Machine Problem: Bills and Bombs

Licup, John Paolo

Santos, Ethan Mark

# Features

- Game-style program

- Room generator with random number of rows, columns, bombs, and moves

- Step-by-step solution

- High scores system stored in a .txt file

- Input handling for every option and detailed error messages

- Unlimited stages and “play again” feature without restarting program

# Notes

- ASCII Art generated from patorjk.com/taag

- Webpages, mostly stackoverflow QnAs, that helped debug / solve problems are linked inside the code for future reference

# Code Map/Outline

Libraries

Structs

Cell

High Score Player

Less and Greater Keys for Sorting Structs

Global Variables

Functions

Clear Screen

Display High Scores

How to Play

Display: Instructions

Main Menu

Player Name Input

Random Integer – returns random integer from range

Update High Scores

Transfer hs.txt Contents to a Vector

Adds New Record

Deleting 6th and Above Entries

Game Over

Play Again Prompt

Do the Thing

Set Default Variable Values

Randomize Room Elements

Place Bombs Randomly

Display

Answer Prompt

THE THING:

- Each cell is represented by a struct cell that contains the value, row, and column that cell belongs to. It

is also arranged in a 2d-vector way for easier accessing later.

- While placing the bombs randomly, add its corresponding row and column details to unsafeRows and

unsafeColumns, respectively.

- Iterate through room and classify whether each cell is supposed to live or die by checking if the row

and column it belongs to is unsafe(see previous). Cells are inserted to either willLive or deathRow.

- Sort deathRow in descending order and willLive in ascending order. This way, when you swap elements

from the two from left to right, you will always get the smartest move that saves the most amount of

bills. Do this up to the number of moves allowed, or up to the number of cells that will live, whichever is

less (see comment inside code for explanation).

- Do swapping by using temp cells. Swap values from willLive and deathRow. Swap struct cells from

vector room. Using temp variables.

- Detonate bombs by changing unsafe cell values in room to \*

- Determine total savable by summing up values in willLive (after swapping).

Answer Checking

Score Accumulator

Next Round/Game Over Preparation

Main

Play Again Loop

Highscores File Integrity Check

Call Main Menu

Call How to Play

Call Do the Thing