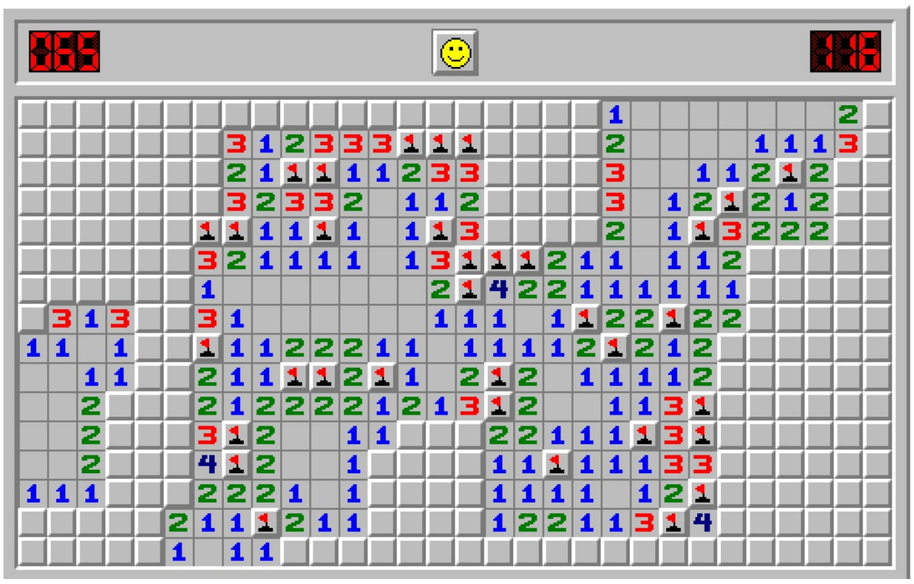
Beijing-Dublin International College

COMP1004J - Introduction to Program Construction 1
Individual Assignment: MineSweeper



Assignment Details

Due date: 2nd of December 2022 (No Exceptions)
Language: Solution must be completed in Python

Task

You are required to implement a text based version of the minesweeper game. This will all be completed printing characters to the screen to show the minefield and the player will have to type the moves that they want to make.

Game Description

MineSweeper is a classic puzzle game where a number of mines are randomly located within a two-dimensional grid. The purpose of the game is to find the location of all of the mines and clear all of the other locations. There are moves the player can make, they can select a square on the grid they believe is clear or they can place a flag on a square the believe is a mine.

Placing a flag on a square, only marks if for you to help you remember. If you are incorrect and there is not a mine, you can remove it by repeating the action.

Selecting a grid square that you believe is clear can have three possible outcomes;

1. There is a mine - In this case, the game is over
2. There are mines in some of the surrounding 8 squares - In this case, the square reveals how many mines are contained in the 8 squares around it by showing a number between 1 and 8.
3. There are no mines in the surrounding 8 squares - In this case, a chain reaction is set off. All spaces that are directly touching this one and also have no mines in the surrounding squares are cleared as well as the mines around them that show a number.

Lets look at an example of each of these outcomes using an example mine field. Here the coordinates of the grid are shown along the top and left sides. Coordinates are shown as a pairing of a letter (shown on the left) and a number (shown on the top).

	0	1	2	3	4	5	6	7	8	9
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										

Example 1: Selecting a Mine

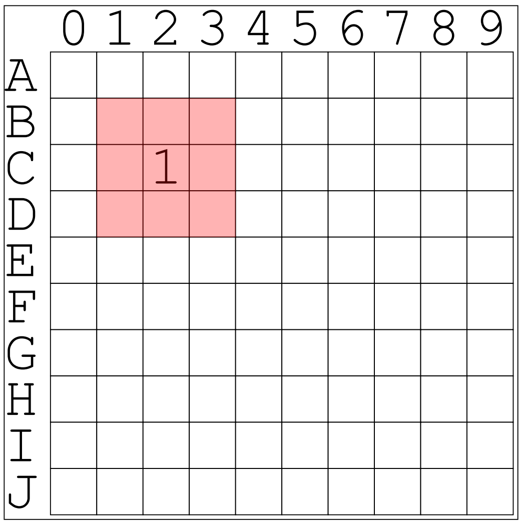
Lets assume that we select the grid square `C3`, but that square contains a mine, then we would get the following output (Here we use an X to show that there was a mine):

	0	1	2	3	4	5	6	7	8	9
A										
B										
C				X						
D										
E										
F										
G										
H										
I										
J										

At this point the game is over and we need to start again.

Example 2: Selecting a clear square with a mine as a neighbour

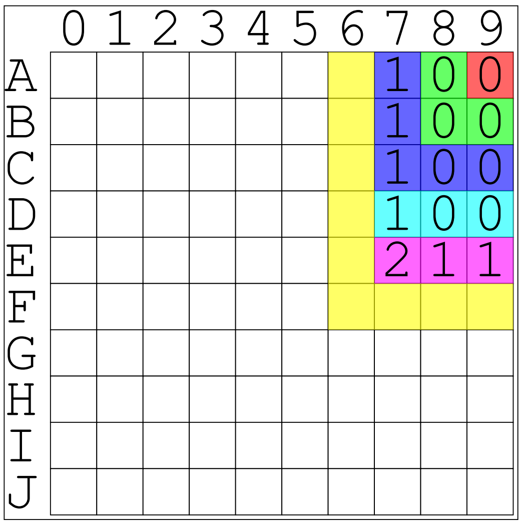
If instead of selecting `C3` we had chosen `C2` then this is the output we would see:



The number in grid square tells us how many of the 8 neighbour squares (shown in red) contain a mine. In this case, there is only one.

Example 3: Selecting a clear square with no mines around it

If we choose a square and there are no mines in any of it's neighbouring squares, then the square is cleared (in our example we will show this as a 0). After this, the neighbouring squares are automatically selected and cleared, if those squares also have no mines as neighbours, then the squares neighbouring those squares are cleared. This action is repeated for all of the neighbouring squares, stopping only when we reach a squares with a mine as a neighbour.



The steps that the algorithm goes through are highlighted in different colours.

1. Red shows the first selected square (A9)
2. Green shows the squares that are selected because they are neighbours (A9, B8, & B9)
3. Blue shows the squares that are selected because they are neighbours of the green squares (A7, B7, C7, C8, & C9)
4. Cyan shows the squares that are selected because they are neighbours of the blue squares that did not contain a number (D7, D8, & D9)
5. Magenta shows the squares that are selected because they are neighbours of the cyan squares that did not contain a number (E7, E8, & E9)

Yellow shows the squares that are not cleared at all.

Making Moves

There are only 2 moves that can be made. Placing a flag on a square when you think there is a mine there and clearing a grid square.

Placing a Flag

Given the grid below, we can work out that there must be a mine in location `A7`. We can calculate this because grid square `B8` shows the number 1 and all other neighbouring squares are already cleared. In cases like this, we want to put a flag on the location to mark it.

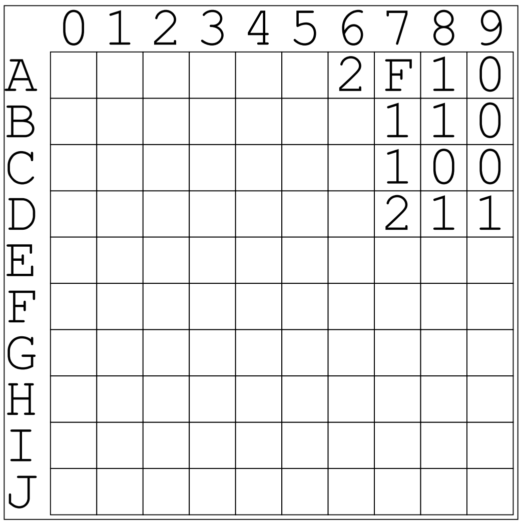
	0	1	2	3	4	5	6	7	8	9
A									1	0
B								1	1	0
C								1	0	0
D								2	1	1
E										
F										
G										
H										
I										
J										

When the game asks for the next move, we use the command `F R C` to instruct the game to places a flag on the grid square `RC` (R for row, C for column). This should only work if the grid square `RC` has not already been cleared, if the grid square has already been cleared no action should be performed. The output of the command F A 7 should look something like this:

	0	1	2	3	4	5	6	7	8	9
A								F	1	0
B								1	1	0
C								1	0	0
D								2	1	1
E										
F										
G										
H										
I										
J										

Clearing a Square

After looking at the grid after our last move, we realise that there are squares that we know are not mines. `A6`, `B6`, and `C6` cannot be mines because `B7` shows the number 1 and we already know there is a mine in square `A7`. So we will want to clear these grid squares. To clear a grid square, when the game asks for our next move we use the command `R C` to instruct the game to clear the grid square `RC` (R for row, C for column). So our command would be `A6`.

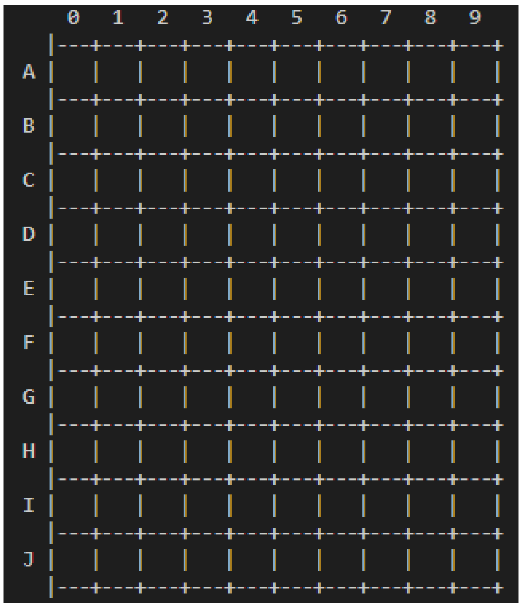


	0	1	2	3	4	5	6	7	8	9
A							2	F	1	0
B								1	1	0
C								1	0	0
D								2	1	1
E										
F										
G										
H										
I										
J										

Text Version

The expected version of the game is a text version, you may if you want to try and make a version of the game that plays using the mouse. However, you will have to study the required materials on your own time. Additionally, you must submit a text version of the game, this is what your grade will be based on. If you also submit a version with a GUI I may allow some extra marks, but this will only be a small amount and the biggest part of your grade will be based on the text version.

Here is an example of how the game might look when you make a text version:



```
     0   1   2   3   4   5   6   7   8   9
   |---+---+---+---+---+---+---+---+---+---+
 A |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 B |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 C |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 D |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 E |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 F |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 G |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 H |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 I |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
 J |   |   |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+---+---+
```

Example

To get an example of the how the minesweeper game plays, the game can be played on most windows computers and can be found for phones and other systems on the internet. For information about the game see the following website http://www.minesweeper.info/wiki/Windows_Minesweeper

Difficulty Levels

The game should have a menu that is presented to the user giving them an option of what difficulty level to play. The game should have three difficulty settings, beginner: 8 x 8 grid with 10 mines, intermediate: 16 x 16 grid containing 40 mines and expert: 24 x 24 grid containing 99 mines. An example of how the menu should look and function is given here:

```
Welcome to Minesweeper.
Please choose your difficulty:
  1: Beginner 8 x 8 grid with 10 mines
  2: Intermediate 16 x 16 grid with 40 mines
  3: Expert 24 x 24 grid with 99 mines
1
Mines: 10    Flags: 0   No visible squares: 0
     0   1   2   3   4   5   6   7
   +---+---+---+---+---+---+---+---+
A  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
B  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
V  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
D  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
E  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
F  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
G  |   |   |   |   |   |   |   |   |
   |---+---+---+---+---+---+---+---+
H  |   |   |   |   |   |   |   |   |
   +-------------------------------+
   Please enter your move:
```

Assessment

This section gives the submission requirements and a breakdown of the approximate marking criteria for the assignment. The final marking scheme may vary slightly but will be relatively similar.

Submission

The completed project should be submitted as a zip file and must contain the following:

- All of the source files (.py) associated with the project
- A PDF document listing your name and UCD student number and an appraisal of your work on the assignment (complete the assignment report template)

Marking Scheme

The marking scheme shown in table 1 is indicative only. This means that it may be changed at any time without notice. However, it will have approximately the same criteria. Criterion may be removed and additional criterion may be added.

It can however be used as a guide to the level required for the different parts of the assignment.

Item	Fail	Pass	Excellent
Design	Not using any functions at all, code is written as one big list of instructions	Some use of functions, but many are too large, do too much, are repeated, or have other problems	Excellent use of functions to break the problem down into manageable chunks
Global Variables	Almost all information is remember using global variables	Several global variables are used to remember data in the program	There are no global variables used in the program
Display	Minefield is not drawn correctly on the screen	Minefield is drawn, but does not align very well or does not have labels for the rows and columns	Minefield is drawn, is well aligned and contains labels of the row and column numbers
Input	User input is not completed	User input is completed but some commands are not read correctly	User input is completed all commands work correctly for uppercase and lowerccase commands
Menu	No menu or other options are shown, game moves straight into gameplay	Main menu is shown before gameplay starts, but options are not implemented properly	Game contains a main menu, and all difficulty levels can be attempted in the game
Clearing Algorithm	Clearing algorithm does not work correctly	Clearing algorithm works but does not clear all of the squares that it should or clears too many.	Clearing algorithm works and correctly clears all of the correct grid squares.
Gameplay	Game plays only for a set number of turns or does not stop when the game is won	Game stops when the game is won or lost, but not both	Game stops correctly when the game is won or lost showing a message and the final game board to the player

Table 1: Indicative Marking Scheme for Assignment

Additional Considerations

Along with the criteria mentioned above, I will also consider the following when I calculate your grade:

- Names - Variables and functions should be named sensibly (if I can't understand what a variable or function is used for based on its name, then it has not been named well)
- Code Formatting - Code should be well formatted and indented correctly
- Comments - You code should be lightly documented (explaining intent of the algorithms)
- Report - The contents and **accuracy** of the report you submit