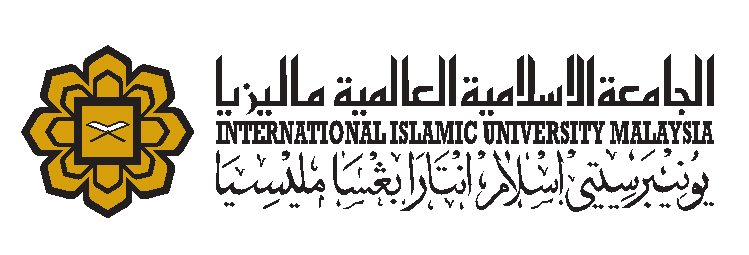
****

**Data Structures and Algorithms**

**CSC 2104**

**Section 4**

**Minesweeper Game**

**Lecturer’s Name:**

DR.[MAZNAH BT AHMAD](http://italeem.iium.edu.my/2016/user/view.php?id=24399&course=3269)

**Group Members:**

1. Ehsanullah Ahmadi 1210731
2. Sayed Faqir Bares 1320587

Table of Contents

1. Introduction 1

2. Algorithm 2

Data Structures: 2

Before Start: 2

Right Click on a Cell: 2

Left Click on a Cell: 2

Multi Click (both Left and Righ Click) on a Cell: 2

3. Screen shots of project 3

4. Conclusion 7

Table of Figures

**Figure 1 Beginner 3**

**Figure 2 intermediate 4**

**Figure 3 Expert 5**

**Figure 4 Custom 6**

**Figure 5 full game 7**

# Introduction

The following project is minesweeper game, which is a popular single player game, distributed with the Microsoft Windows operating system. It consists of a grid of covered cells some of which contain mines. The cells not containing mines are free cells. Each free cell contains information regarding the number of its mined neighbours (where cells adjacent diagonally are also considered as neighbours). At each move, the player may uncover a cell. If the cell is mined, the game is lost; if it is free, the information contained in it is revealed, and the player may use it. The goal of the game is to uncover all free cells, leaving all mined cells covered. Our game has four levels, which are: Beginner, Intermediate, Expert and custom.

Beginner has 10 rows, 10 columns and 10 mines. Intermediate is made from 16 row, 16 columns and 70 mines. Expert level contains 20 rows, 20 columns and 150 mines and lastly, in the Custom the number of rows, columns and mine is up to user.

# Algorithm

## Data Structures:

We may think each cell as a UI structure (e.g. button), which has following attributes:

* colCoord = 0 to colCount
* rowCoord = 0 to rowCount
* isOpened = true / false   (default false)
* hasFlag    = true / false    (default false)
* hasMine  = true / false    (default false)
* neighbourMineCount  0 to 8  (default 0, count of total of mines on neighbour cells)

So, we have a two dimensional "Button[][] cells" data structure to handle game actions.

## Before Start:

1. Assign mines to cells randomly (set hasMine=true) .
2. Calculate neighbourMineCount values for each cell, which have hasMine=false. (this step may be done for each clicked cell while game continues but it may be inefficient)

Note1: Neighbour cells should be accessed with {(colCoord-1, rowCoord-1),(colCoord-1, rowCoord),(colCoord-1, rowCoord+1),(colCoord, rowCoord-1),(colCoord, rowCoord+1),(colCoord+1, rowCoord-1),(colCoord+1, rowCoord),(colCoord+1, rowCoord+1)} coordinates. And don't forget that, neighbour cell counts may be 3 (for corner cells), 5 (for edge cells) or 8 (for middle cells).  
  
Note2: It's recommended to handle mouse clicks with "mouse release" actions instead of "mouse pressed/click" actions, otherwise left or right click may be understood as multi-clicks or vice versa.

## Right Click on a Cell:

* If cell isOpened=true, do nothing.
* If cell isOpened=false, set cell hasFlag=true and show a flag on the cell.

## Left Click on a Cell:

* If cell isOpened=true, do nothing.
* If cell isOpened=false:
  + If cell hasMine=true, game over.
  + If cell hasMine=false:
    - If cell has neighbourMineCount > 0, set isOpened=true, show neighbourMineCount on the cell. If all cells which hasMine=false are opened, end game with SUCCESS.
    - If cell has neighbourMineCount == 0, set isOpened=true, call Left Click on a Cell for all neighbour cells, which hasFlag=false and isOpened=false.

Note: Last step may be implemented with recursive call or using a stack data structure.

## Multi Click (both Left and Righ Click) on a Cell:

* If cell isOpened=false, do nothing.
* If cell isOpened=true:
  + If cell neighbourMineCount == 0, no nothing.
  + If cell neighbourMineCount > 0:
    - If cell neighbourMineCount != "neighbour hasFlag=true cell count", . do nothing.
    - If cell neighbourMineCount == "neighbour hasFlag=true cell count":
      * If all neighbour hasFlag=true cells are not hasMine=true, . game over.
      * If all neighbour hasFlag=true cells are hasMine=true (every flag is put on correct cell), call Left Click on a Cell for all neighbour cells, which hasFlag=false and isOpened=false.

Note: Last step may be implemented with recursive call or using a stack data structure.

# Screen shots of project



Figure 1 Beginner



Figure 2 intermediate

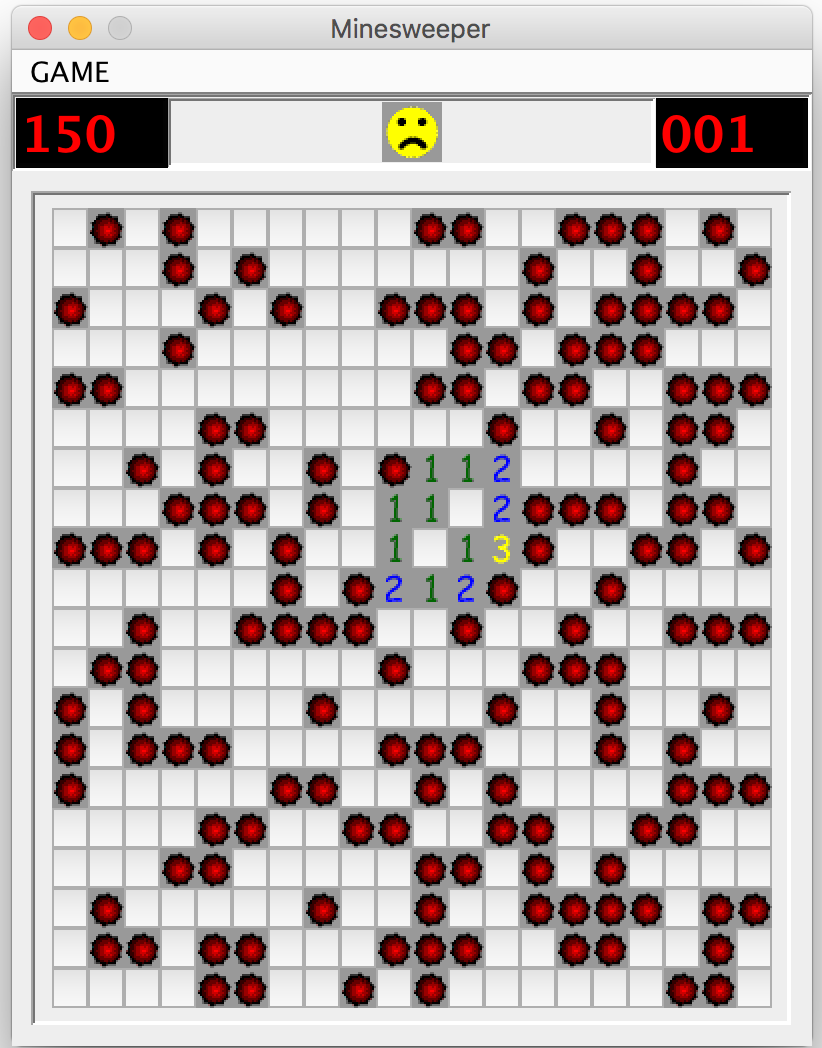


Figure 3 Expert

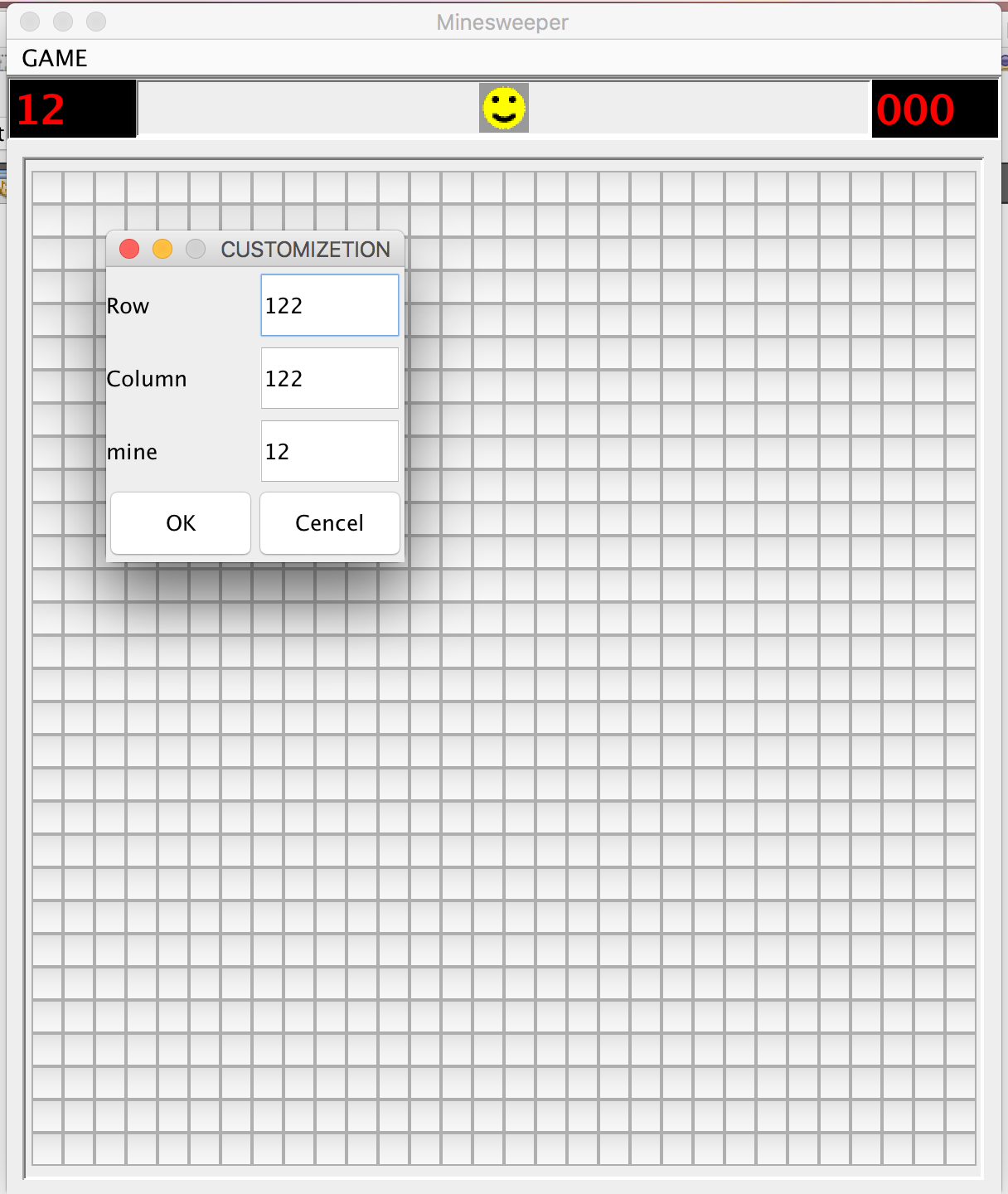


Figure 4 Custom

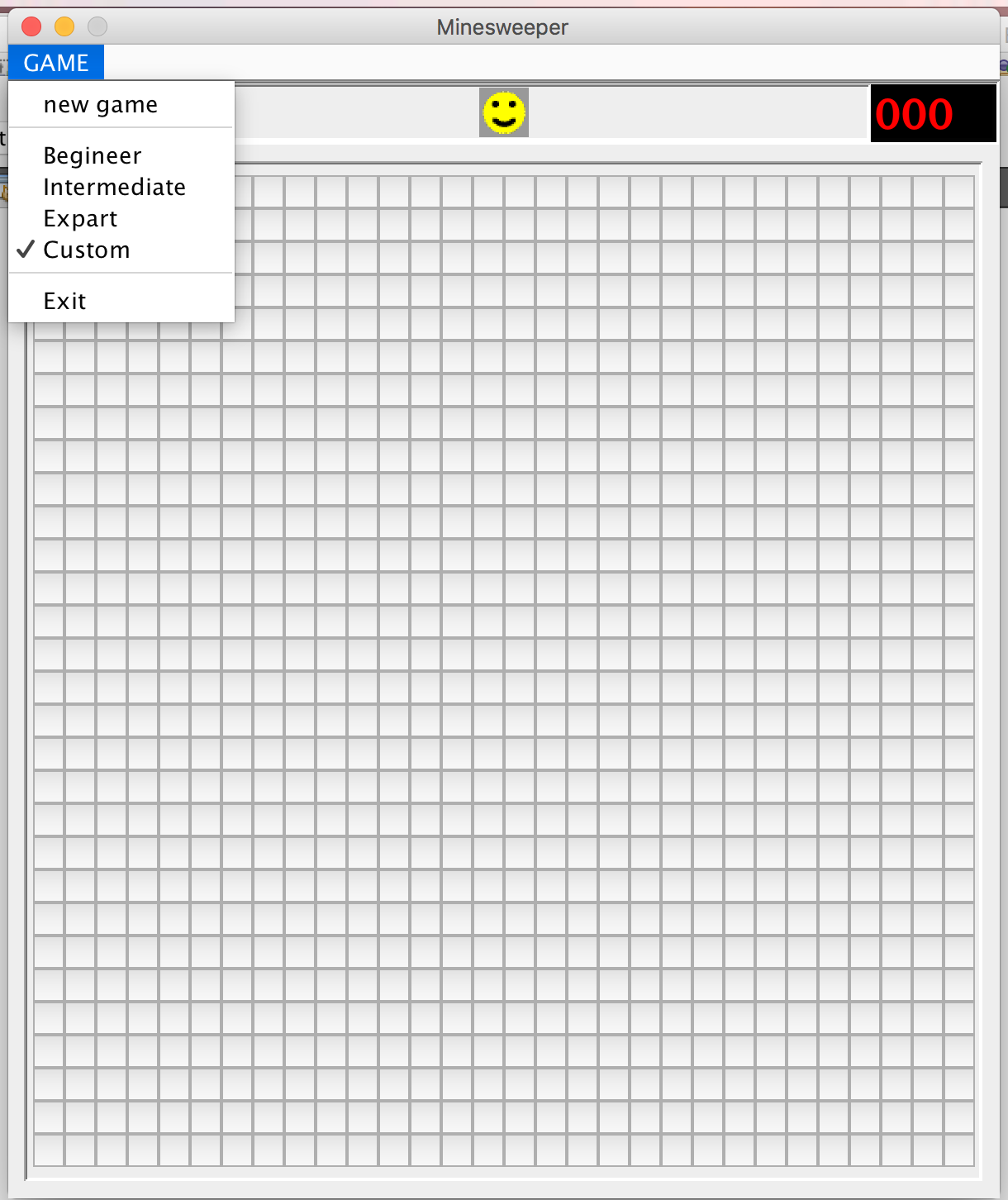


Figure 5 full game

# Conclusion

Minesweeper is a popular single player game, which is very addictive and was developed by windows for the first time. And users must find the mines in order to be winner.