**MINESWEEPER**

**Isabelle André**

**ICS 3UI**

**Mr. J-D**

**Kitchener Collegiate Institute & Vocational School**

**25 January 2016**

Table of Contents

**Introduction 3**

**Development 4**

**Bibliography 8**

**Introduction**

Abstract

This report outlines the design and development of a computer game software created as a final project for the grade 11 computer science course. The software created is a functional copy of the well-known mini game Minesweeper written in Turing language, then generated to run as an independent (standalone) program. The game contains instruction and game settings as well as an exit button. It is played using mouse buttons and inputs. The results of the numerous tests and bugs will later be investigated. The report includes a full user manual as well as an activity log of the testing and planning.

**Development**

User Manuel

MINESWEEPER GAME INSTRUCTIONS AND CONTROLS

INSTRUCTIONS

Select a difficulty level. If you're new to Minesweeper, start with Beginner.

WARNING: Once a level is chosen, you cannot change it unless program is restarted.

The number in the upper-left corner shows the number of mines on the board.

The number on the right is the timer.

Click any square to begin. Most Minesweeper players click random squares until a GROUP of squares "opens" up. If only one square opens after your click, then it would be better to find another random square to try to open a larger group of tiles.

As you uncover tiles, you will see numbers revealed.

A number means that there is that number of bombs touching that tile (horizontally, vertically, or diagonally). For example, if you see a 1 on the board, it means that square is touching exactly 1 mine.

When you're certain you've found a mine, identify it by right-clicking the square.

This will put a flag on the mine, and prevent you from clicking on it.

If you are unsure, you are free to double click and mark the tile with a question mark.

The game ends when the player has won by opening all cells and flagging all mines,

or has clicked on a mine tile, in which case the player loses.

#MOUSE CONTROLS

LEFT clicking uncovers the contents of a given tile.

RIGHT clicking ONCE marks a tile as a suspected bomb with a flag.

RIGHT clicking TWICE marks a tile as an unknown tile with a question mark.

Activity Log

**14-15 December 2015**

* The game Minesweeper was chosen as a final project for the course. Research was made on other websites and applications about Minesweeper’s settings and interface.

**18-19 December 2015**

* A sketched plan was made on paper of the game’s designs and size. The cells would be 16x16 pixels, and the borders, 10 pixels in width. There would be a panel on top of the board with other gadgets such as a mine count, timer and perhaps a New Game button.

**20 December 2015**

* The map was drawn with draw commands on Turing. Pictures for the tiles and other gadgets were researched and downloaded from the original game from Chez Poor.

**21-23 December 2015**

* Pictures were downloaded and converted from PNG to BMP. Global variables were declared, and map settings procedure was made. The map setting procedures contained the cell size, board size, and other game conditions set to false.

**27-29 December 2015**

* A procedure to hide mines inside board game was made. However, the problem lied where maps would at times overlap themselves as their placement was random. Therefore, a second procedure was made, where the first procedure’s settings accompanied with multiple conditions would ensure that mines would be placed without overlapping. Furthermore, an IF statement was also added in the first procedure in order to start placing numbers around mines. One placed mine would add 1 to the surrounding cells. A few bugs prevented the program from running properly.

**30-31 December 2015**

* A procedure to draw the map itself using the map prep procedure was made. It contained the settings to draw the board background as well as numerous IF and CASE statements in order to place image according to the user state and mouse location. For instance, if the user clicked on a mine tile, a mine picture would be called in that specific emplacement. Again, the procedure held many bugs, however the objective was completed to complete the rough outline of the game.

**3 January 2016**

* A procedure to reveal all map when player clicked on a mine was successfully created. The code was simple enough that no bugs were found this time.

**4-5 January 2016**

* A procedure to find out if player has won by opening all cells and flagging all mines was created. However the procedure was unfortunately not working and was hence changed to a function. When all the tiles are opened and all mine cells are flagged, the result would become true.

**6-11 January 2016**

* The main code was finally started by calling multiple procedures such as the map prep, map drawing and mine hiding procedures. 1 main loop was created containing newly created mouse settings. In order to use the mouse, a “mouse grid” was created in order to fit into the game board, enabling the tiles to be clicked and revealed. Much research was also made in order to learn how to utilize the mouse and its buttons. A CASE statement for the right clicking issue was made, letting the user right click multiple times in order to switch from flags, to question marks, back to blank.

**12 January 2016**

* The game went through its first test to ensure that tiles could be clicked and be revealed. Many bugs were found, and fixed. The mouse grid was also altered for further accuracy in clicking the tiles. In the final attempt, a tile finally opened, though at the wrong emplacement showing a number, proving that the mine hiding and map prep procedures were indeed functional. The problem therefore laid with the map drawing procedure itself.

**13 January 2016**

* Another procedure was added after the map reveal procedure in order to reveal a blank group of tiles when a single blank square was clicked. A bug prevented the corner squares to reveal themselves properly, however, that problem was fixed by creating an algorithm with FOR loops to reach the corners.

**14 January 2016**

* All of the procedures created up to this point were all implemented into the main code. For instance the reveal map, check for win, and such procedures were all placed in several IF statements in case the user won or lost. The program was now fully functional to the exception of a few blinking spots and glitches that were fixed using “offscreenonly” and “View.Update”.

**­**

**15-18 January 2016**

* A procedure to count the mines was created and placed to the top left of the map using the downloaded digit pictures. A timer was also being created using the same pictures, however, as the clock command was not working, it was replaced by a counter, converted into seconds by being multiplied by 10000.

**18-22 January 2016**

* A second loop was created in the main code in order to make a level selection option as well as a play again option for the user. And intro and help windows were created and added at the beginning of the program.

**22-24 January 2016**

* Testing period of the program in order to fix any bugs or glitches remaining.

**Bibliography**

Muranski, Kamil. How to Play Minesweeper Like a Pro, PCWorld. Retrieved from

<http://www.pcworld.com/article/238724/how_to_play_minesweeper_like_a_pro.html>,

(20 january 2016).

Shep Poor, D. (1997). Chez Poor. [Online Image]. Retrieved from

<http://www.chezpoor.com/minesweeper/minecore.html> , (20 january 2016).

Minesweeper Blue, (2013). Google Play [Online Image]. Retrieved from

<https://play.google.com/store/apps/details?id=appistic.products.minesweeper>, (20 january 2016).

Musics8bits, (2012). Tetris Theme (Dubstep Remix), Youtube, Retrieved from

<https://www.youtube.com/watch?v=E8FQBjVlERk>, (20 january 2016).

MSM, Therok. (2010). Super Mario Bros. Level Clear, Youtube, Retrieved from

<https://www.youtube.com/watch?v=FxjT1RvGnhU> , (20 january 2016).

Rogers, Jim. (2011). Shotgun Blast. Sound Bible. Retrieved from

<http://soundbible.com/1919-Shotgun-Blast.html> , (20 january 2016).

Lamb, Joe. (2011). Sad Trombone. Sound Bible. Retrieved from

<http://soundbible.com/1830-Sad-Trombone.html> , (20 january 2016).