3XA3: Test Plan

Team 3
Shuying Chen, 400080161
Ziyang Huang, 400051063
Guiye Wu, 400089784

Oct 25,2018

Revision History

Table 1: Revision History: Test Plan

Developer	**Date**	**Change**	**Revision**
Shuying Chen	Oct 25, 2018	Section 1,2	0
Ziyang Huang	Oct 25, 2018	Section 3, 4	0
Guiye Wu	Oct 25, 2018	Section 5,6,7	0

Contents

List of Tables

List of Figures

1 General Information

1.1 Purpose

The purpose of testing the project is to help improve the understanding of the problems, reduce the ambiguous for the system and build confidence in the software. The testing is the process of validating and verifying if the project satisfies the conditions of each objective and meets the specifies requirements.

1.2 Scope

This plan provides the basis for testing functionality of the software, which is the re-implementation of Minesweeper that is written in python with modules separately. This document works as the outline of testing schedule, method and tools have been used for each module.

1.3 Acronyms, Abbreviations, and Symbols

Table 2: **Table of Abbreviations**

Abbreviation	**Definition**
SRS	Software Requirements Specification
PoC	Proof of Concept

Table 3: **Table of Definitions**

Term	**Definition**
Structural Testing	Testing derived from the software's structure or internal implementation.
Functional Testing	Testing against the functional requirements/specifications
Dynamic Testing	Testing of the dynamic behavior of code(test cases run during execution).
Static Testing	Testing without involving program execution.
Manual Testing	Testing done and supervised by people.
Automated Testing	Testing run automatically by software(pytest, unit test).

1.4 Overview of Document

The minesweeper project will re-implement the open source minesweeper-Java. This game will allow users to play on the newly featured minesweepers with the same logic.

2 Plan

2.1 Software Description

The project allows users to clear the board containing "hidden mine" without triggering any of the mines, with the help from the numbers of surrounded mine in each cell.

2.2 Test Team

The test team members are Shuying Chen, Ziyang Huang and Guiye Wu who are responsible for all the testing process.

2.3 Automated Testing Approach

The program will use the Pytest for the automated testing approach, which go through all the tests automatically, and return the percentage of the codes that are covered in the test. This tool will help to identify the missing test cases.

2.4 Testing Tools

The testing tool that will be used is Pytest which is an external python testing library. It will show the number of passing tests, the total number of test cases and the percentage of test coverage for each module, it will also point to the position where the testing fails.

2.5 Testing Schedule

Table 4: Testing Assignments

Task	**Team Member**	**Date**
Correctness of the image	Ziyang Huang	Nov 1, 2018
Correctness of the frames position	Guiye Wu	Nov 5, 2018
Correctness of the music	Shuying Chen	Nov 7, 2018
Algorithm	Guiye Wu	Nov 15, 2018
Performance	Shuying Chen	Nov 17, 2018
Play-ability	Ziyang Huang	Nov 19, 2018

3 System Test Description

3.1 Tests for Functional Requirements

3.1.1 User Input

Mouse Left Clicks

1. F-MLC-1

   Type: Functional, Dynamic, Manual.

   Initial State: Minesweeper icon is used to be clicks and launches the game.

   Input: Double left clicks on Minesweeper icon.

   Output: The 860 x 640 game screen is popped up.

   How test will be performed: Double clicks on the icon multiple times to check if the proper game screen is popped up successfully, the proper game screen will have the image of the level set up the frame and the background as exactly as the design and implementation.

2. F-MLC-2

   Type: Functional, Dynamic, Manual.

   Initial State: The game screen has four levels to be selected

   Input: Left click on the Beginner level

   Output: The game board which has 16 x 30 cells with 10 bombs. The cell frame is at the width of 20 and height of 20.

   How test will be performed: Provide a test case which calculate the total number of bombs in the board and checks if the total number of bombs is 10 or tests the number of bombs in the module Board or finish the game when 10 bombs are found out.

3. F-MLC-3

   Type: Functional, Dynamic, Manual.

   Initial State: The game screen has four levels to be selected

Input: Left click on the Medium level

Output: The game frame will turn out to be a 16 x 30 cells with 60 bombs. The cell frame is at the width of 20 and height of 20.

How test will be performed: Provide a test case which calculates the total number of bombs in the board and checks if the total number of bombs is 60 or tests the number of bombs in the module Board or finish the game when 60 bombs are found out.

4. F-MLC-4

   Type: Functional, Dynamic, Manual.

   Initial State: The game screen has four levels to be selected

   Input: Left click on the hard level

   Output: The game frame will turn out to be a 16 x 30 cells with 99 bombs. The cell frame is at the width of 20 and height of 20.

   How test will be performed: Provide a test case which calculates the total number of bombs in the board and checks if the total number of bombs is 99 or tests the number of bombs in the module Board or finish the game when 99 bombs are found out.

5. F-MLC-5

   Type: Functional, Dynamic, Manual.

   Initial State: The game screen has four levels to be selected

   Input: Left click on the Nightmare level

   Output: The game frame will turn out to be a 16 x 30 cells with 450 bombs. The cell frame is at the width of 20 and height of 20.

   How test will be performed: Provide a test case which calculates the total number of bombs in the board and checks if the total number of bombs is 450 or tests the number of bombs in the module Board or finish the game when 450 bombs are found out.

6. F-MLC-6

Type: Functional, Dynamic, Manual.

Initial State: The game board has all 16 x 30 cells that all cells are clickable.

Input: Left click a hidden value cell on the game board

Output: A value on the cell that represents the number of neighbour bombs.

How test will be performed: Provide a test case which calculates the total number of bombs around the cell and check if the number of bombs is equal to the value or find out all the bombs or clicks out all the cells that around the cell to check if it is the proper value.

7. F-MLC-7

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has all 16 x 30 cells that all cells are clickable.

   Input: Left click a mine cell in the game board

   Output: A hit-mine icon display on the cell.

   How test will be performed: Provide a test case to check if the cell exists a bomb and click on the cell to test if the cell pops up a hit-mine icon.

8. F-MLC-8

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has all 16 x 30 cells that all cells are clickable.

   Input: Left click a blank cell in the game board

   Output: A blank icon display on the cell.

   How test will be performed: Provide a test case to check if the cell is a blank cell and click on the cell to test if the cell pops up a blank cell icon.

9. F-MLC-9

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has a top panel which includes a simile face that is clickable.

   Input: Left click on the simile face

   Output: Reset the game board.

   How test will be performed: Provide a test case to check if all bombs locations are different than the previous locations after the board is reset or click on a random cell to check if the board is reset.

Mouse Right Clicks

1. F-MRC-10

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has all 16 x 30 cells that all cells are clickable.

   Input: Right click on a blank cell.

   Output: The flag icon is popped up on the cell.

   How test will be performed: Right clicks on all the cell on the board to check if any cell on the board is able to be set the flag and check if all the flag is displayed on the proper location.

2. F-MRC-11

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has all 16 x 30 cells that all cells are clickable.

   Input: Right click on a flag cell.

   Output: The blank cell on the right click location.

   How test will be performed: Right clicks on any flag cell in the board to check if the flag is removed and display a blank cell.

3.1.2 Algorithm Area

Algorithm in Board

1. F-AB-12

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has 16 x 30 number of cells and each bombs location is randomly set.

   Input: A 16 x 30 game board with a fixed amount of bombs.

   Output: The number of bombs are set to random locations.

   How test will be performed: Run the board multiple times to check if the bomb locations are different and provide a test case that test if cells in the module Board are set to be bombs or not different for each execution.

2. F-AB-13

   Type: Functional, Dynamic, Manual.

   Initial State: The grey blank cells will always around by value cells.

   Input: A grey blank cell is clicked.

   Output: The neighbor cells are unlocked until all the grey blank cell is around by value cells.

   How test will be performed: Provide a test case to test the correctness of the unblock method in the module Board or clicks on any grey blank cell to check if it unlocks all the grey blank cells until all the cells are blocked by value cells.

3. F-AB-14

   Type: Functional, Dynamic, Manual.

   Initial State: The game board has some value cells that each value cell represents the number of bombs around the cell.

   Input: A game board with mines.

Output: Any cell that around with mines should have a value cell that the value is equal to the number of mines around the cell.

How test will be performed: Provide a test case to test the correctness of each value cell in the module Board or clicks on any value cell to check if it is the proper value.

4. F-AB-15

   Type: Functional, Dynamic, Manual.

   Initial State: The top panel of the game board has bombs counter that includes the total number of bombs

   Input: A game board with the top panel.

   Output: The accurate number of bombs display on the bombs counter block.

   How test will be performed: Provide a test case to set a different number of bombs for each game board and check if the number of bombs is equal to the bombs counter which displays on the top panel.

5. F-AB-16

   Type: Functional, Dynamic, Manual.

   Initial State: The bombs counter should decrease each time a bomb is flagged.

   Input: A game board with the top panel that has the bombs counter block.

   Output: The accurate number of bombs display on the bombs counter block and decrease each time when a bomb is flagged.

   How test will be performed: Set a flag to a mine cell to check if the bombs counter decrease 1 value, and do it for all the mines to check if the final value is 0.

6. F-AB-17

   Type: Functional, Dynamic, Manual.

Initial State: The goal of the game is achieved.

Input: All the value cells are found out.

Output: Winning animation

How test will be performed: Provide a test case with only one mine which is able to win the game immediately to check if the winning animation pops up successfully or play and win the game multiple time to check the winning animation.

7. F-AB-18

   Type: Functional, Dynamic, Manual.

   Initial State: The goal of the game is not achieved.

   Input: A mine is hit on the game board.

   Output: Ending animation

   How test will be performed: Provide a test case with only full mines which are able to end the game immediately to check if the ending animation pops up successfully or play and lose the game multiple time to check the ending animation.

3.1.3 Animation Area

1. F-AA-19

   Type: Functional, Dynamic, Manual.

   Initial State: The top panel of the game board has a timer

   Input: The difficult level is selected

   Output: Timer block is display on the top panel of the game board and automatically counting the time.

   How test will be performed: Select all the difficult level and check if all the game boards display the timer and check if the timer is counting time.

2. F-AA-20

Type: Functional, Dynamic, Manual.

Initial State: The game's goal is achieved.

Input: Win the game when all values are found out.

Output: A screen with the animation of banana rain.

How test will be performed: Win the game then check if the animation is displayed successfully and check if it is banana dropping rain.

3. F-AA-21

   Type: Functional, Dynamic, Manual.

   Initial State: Any hit mine icon is display on the game board.

   Input: A mine is hit by user on the game board.

   Output: A screen with the animation of funny 'poopoo' rain.

   How test will be performed: Loose the game then check if the animation is displayed successfully and check if it is funny 'poopoo' dropping rain.

3.2 Tests for Nonfunctional Requirements

3.2.1 Usability

1. NF-U-1

   Type: Structural, Manual, Static

   Initial State: program installed on the system via the download from git, but not launched.

   Input/Condition: launch the program and choose the desired level on computers running operations system Windows and Mac OS.

   Output/Result: A blank board with cells should be displayed. When all the "good cells" which do not contain bombs are clicked, the winning animation is displayed.

   How test will be performed: Program will be downloaded on computers with different operating systems. The program will be checked to see if it runs and main functions perform successfully.

2. NF-U-2

   Type: Structural, Manual, Static

   Initial State: Program is installed on the computer and launched.

   Input/Condition: Users are asked to start and win the game in the beginner level with no assistance.

   Output/Result: users with experience can achieve the goal within 5 minutes, and other users can achieve the goal within 30 minutes.

   How test will be performed: a group of people who are students of McMaster University are asked to play the game at the beginner level and achieve the final success. The majority of the group can win the game within 5 minutes.

3. NF-U-3

   Type: Structural, Manual, Static

   Initial State: The URL for download is provided, but not downloaded on the computer.

   Input/Condition: User is asked to download the game and launch it on the computer

   Output/Result: Majority of the user are able to install the game and open the main menu of the game.

   How test will be performed: a group of people who are students of McMaster University is asked to download the game with the given URL and open the main menu. Majority of the group can download and start the game within 5 minutes.

4. NF-U-4

   Type: Structural, Manual, Static

   Initial State: The group of people has finished downloading the game and win the beginning level.

   Input/Condition: User is asked to rate the program on the standards of ease of use, ease of installation, ease of understanding, user satisfaction and level of entertainment of scale from 1 - 10.

Output/Result: The result of the survey received an average of 7.3 on all criteria.

How test will be performed: The group of people will be given a survey that includes a list of criteria and a scale from 1 to 10, which 1 is Very Poor ascending to 10 which is Above Satisfaction. They give their opinions on the project. The average will have remain at least 7 for each criterion.

5. NF-U-5

   Type: Structural, Manual, Static

   Initial State: The group of people has finished downloading the game and win the beginning level.

   Input/Condition:User is asked to rate the source program on the standards of ease of use, ease of installation, ease of understanding, user satisfaction and the level of entertainment of scale from 1 - 10.

   Output/Result: The average rating on the program is higher or equal to the source program. The average rating on the level of entertainment of program has significantly increased compared to the sourcing program.

   How test will be performed: The group of people will be given a survey that includes a list of criteria and a scale from 1 to 10, which 1 is Very Poor ascending to 10 which is Above Satisfaction. They give their opinions on the existing product

3.2.2 Performance

1. NF-P-6

   Type: Structural, Manual, Dynamic

   Initial State: The program is launched and on default settings, with the initial playing level setting frame and background showing.

   Input/Condition: Any clicks on the level setting block.

   Output/Result: The request action is performed under the processing time.

How test will be performed: Users are required to choose the hardness level of the game. The time lapsed after clicking the hardness level is being initiated by the proper option in the program and the program is actually acting on setting up the gaming frame and cell matrix, then switch the setting frame to the playing frame. The time elapsed is different regarding the options.

2. NF-P-7

   Type: Structural, Manual, Dynamic

   Initial State: The program is launched and on default settings, with the initial playing level setting frame and background showing.

   Input/Condition: Once the user clicks on the beginner level, the program starts to set up the gaming frame, cell matrix,and winning method.

   Output/Result: The request action is performed under the processing time.

   How test will be performed: Users are required to choose the hardness level of the game. The time lapsed after clicking the hardness level is being initiated by the proper option in the program and the program is actually acting on setting up the gaming frame and cell matrix, then switch the setting frame to the playing frame. The time elapsed is different regarding the options.

3. NF-P-8

   Type: Structural, Manual, Static

   Initial State: The program is launched and on default settings, with the initial playing level setting frame and background showing, the play frame has been switched and available to play. Players can now click on a blank cell.

   Input/Condition: The player has clicked a blank cell.

   Output/Result: There is a bomb showing after the blank cell is being clicked, in a very short of process time. The game shows that the game is over.

How test will be performed: Users are required to click on any of the blank cells. The time lapsed after clicking the blank cell is being initiated by the proper option in the program and the program is actually acting on covering the image of the blank cell to a bomb, then showing up that the game is ending. The time elapsed is different regarding the options.

3.3 Traceability Between Test Cases and Requirements

Table 5: Traceability Matrix

Req't	**PW**	**F-1**	**F-2**	**F-3**	**F-4**	**F-5**	**F-6**	**F-7**	**F-8**	**F-9**
3.2.1.1	5	X	X	X	X	X	X	X	X	X
3.2.1.2	2						X	X	X	
3.2.1.3	4									
3.2.1.4	1						X	X	X	
3.2.2.1	5								X	
3.2.2.2	3							X		
3.2.2.3	2									
3.2.2.4	4									X
3.2.3.1	5									
3.2.3.2	1					X	X			
3.2.3.3	2	X	X	X	X	X				
3.2.3.4	5									X
3.2.3.5	2							X		
3.2.3.6	3						X		X	
3.2.3.7	4						X			
3.2.3.8	2									X
3.2.3.9	5									

Table 6: Traceability Matrix

Req't	**PW**	**F-10**	**F-11**	**F-12**	**F-13**	**F-14**	**F-15**	**F-16**	**F-17**	**F-18**
3.2.1.1	5	X	X		X		X		X	X
3.2.1.2	2				X		X		X	
3.2.1.3	4	X	X					X		
3.2.1.4	1	X		X				X		X
3.2.2.1	5						X		X	
3.2.2.2	3						X			X
3.2.2.3	2	X	X				X			
3.2.2.4	4									
3.2.3.1	5									
3.2.3.2	1									
3.2.3.3	2									
3.2.3.4	5									
3.2.3.5	2						X			
3.2.3.6	3						X		X	
3.2.3.7	4						X		X	
3.2.3.8	2					X	X	X		
3.2.3.9	5	X	X						X	

Table 7: Traceability Matrix

Req't	**PW**	**F-19**	**F-20**	**F-21**
3.2.1.1	5			
3.2.1.2	2	X		
3.2.1.3	4			
3.2.1.4	1		X	
3.2.2.1	5			
3.2.2.2	3		X	
3.2.2.3	2	X		
3.2.2.4	4			
3.2.3.1	5		X	X
3.2.3.2	1			
3.2.3.3	2	X		
3.2.3.4	5		X	
3.2.3.5	2	X		
3.2.3.6	3			
3.2.3.7	4	X	X	X
3.2.3.8	2			
3.2.3.9	5			

Table 8: Traceability Matrix

Req't	**PW**	**NF-1**	**NF-2**	**NF-3**	**NF-4**	**NF-5**	**NF-6**	**NF-7**	**NF-8**	**NF-9**	**NF-10**	**NF-11**	**NF-12**
4.1	4			X				X			X		
4.2	3	X				X		X			X		
4.3.1	4							X					
4.3.2	2		X		X			X			X		
4.3.3	4	X											
4.3.4	3								X		X		
4.3.5	5		X				X			X			X
4.4	1				X								
4.5	2	X					X				X		X
4.6	1		X						X				
4.7	3				X							X	
4.8	2		X										
4.9	1	X								X		X	
5.0	1				X		X						X

4 Tests for Proof of Concept

Proof of Concept testing will be focused on verifying and validating the user interface as well as checking if the interface is performing as the desired design. Therefore, testing on Proof of Concept will involve testing the performance, functionality and the user-friendly of the interface.

4.1 Correctness of the images loading and showing

1. PF-1

   Type: non-functional, Dynamic, Automated.

   Initial State: The user is intended to click on the execution program on the desktop to start the game.

   Input: Left click twice on the execution icon.

   Output: The background of the game will show up in the pop-up game frame, with the right resolution as 860x640 and the background image is in the right position, the set-level block is at the center of the background image.

   How test will be performed: The test output should be clear and straightforward shows up to users.

2. PF-2

   Type: non-functional, Manual, Static.

   Initial State: The level-setting block is available to be clicked.

   Input: Left click on the beginner level block, at any positions inside.

   Output: The beginner playing mode should be correct showed up with the correct blank cell images loaded and showed.

   How test will be performed: The test output should be clear and straightforward shows up to users.

3. PF-3

   Type: non-functional, Manual, Static.

Initial State: The level-setting block is available to be clicked.

Input: Left click on the Medium level block, at any positions inside.

Output: The beginner playing mode should be correct showed up with the correct blank cell images loaded and showed.

How test will be performed: The test output should be clear and straightforward shows up to users.

4. PF-4

   Type: non-functional, Manual, Static.

   Initial State: The level-setting block is available to be clicked.

   Input: Left click on the Hard level block, at any positions inside.

   Output: The beginner playing mode should be correct showed up with the correct blank cell images loaded and showed.

   How test will be performed: The test output should be clear and straightforward shows up to users.

5. PF-5

   Type: non-functional, Manual, Static.

   Initial State: The level-setting block is available to be clicked.

   Input: Left click on the Nightmare level block, at any positions inside.

   Output: The beginner playing mode should be correct showed up with the correct blank cell images loaded and showed.

   How test will be performed: The test output should be clear and straightforward shows up to users.

4.2 Correctness of the music loading and playing

1. PFM-1

   Type: non-functional, Manual, Static.

   Initial State: The level-setting block is available to be clicked.

Input: Left click on the Nightmare level block, at any positions inside.

Output: The right music of the setting the game frame should be correct playing.

How test will be performed: The test output should be clear and straightforward shows up to users.

2. PFM-2

   Type: non-functional, Manual, Static.

   Initial State: The level-setting block is available to be clicked.

   Input: Left click on any level blocks, at any positions inside.

   Output: The right music of the playing mode should be clearly played.

   How test will be performed: The test output should be clear and straightforward shows up to users.

3. PFM-3

   Type: non-functional, Manual, Static.

   Initial State: The playing board is now showing up to the players.

   Input: Left click on any blank cells, at any positions inside.

   Output: The right music of the null cell, bomb cell and number cell should be clearly played.

   How test will be performed: The test output should be clear and straightforward shows up to users.

4. PFM-4

   Type: non-functional, Manual, Static.

   Initial State: The playing board is now showing up to the players.

   Input: The right click on any blank cells, at any positions inside.

   Output: The right music of the flagged cell should be clearly played.

   How test will be performed: The test output should be clearly and straightforward shows up to users.

5 Comparison to Existing Implementation

There is one test that compares the program to the Existing Implementation of the program. Refer to:

- test NF-U-5 in Tests for Nonfunctional Requirements - Usability

6 Unit Testing Plan

The Pytest framework will be used to do unit testing for this project.

6.1 Unit testing of internal functions

In order to create unit tests for the internal functions of the program. The unit test will include the input and the boundary constraints that generate the exceptions. This project does not need stubs, drivers and other methods rather than just click the cells. We will be using a coverage matrix to determine how much of the code we have been tested. Our goal is the cover as much code as possible, to make sure that we have tested all functions adequately. We will use test cases to test the correctness on the x and y coordinates, the correctness of the image/cell that is being clicked and the correctness of the flag/bomb/number in the matrix.

6.2 Unit testing of output files

The output file for the program will be the graphic representation of the internal functions. The result of the unit testing of output file should consistent with the result of the unit testing of output files.

7 Appendix

This is where you can place additional information.

7.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

7.2 Usability Survey Questions

There is two tests that require survey Question Refer to:

- test NF-U-4 in Tests for Nonfunctional Requirements - Usability
- test NF-U-5 in Tests for Nonfunctional Requirements - Usability

Question:

- What do you think about the ease of installation? Please rate from 1 - 10, which 1 indicates very poor, and 10 indicates above satisfaction.
- What do you think about the ease of using for this game(easy to manipulate)? Please rate from 1 - 10, which 1 indicates very poor, and 10 indicates above satisfaction.
- What do you think about the ease of understanding? Please rate from 1 - 10, which 1 indicates very poor, and 10 indicates above satisfaction.
- What do you think about the level of entertainment? Is the main menu creative? Please rate from 1 - 10, which 1 indicates very poor, and 10 indicates above satisfaction.

8 Reference

- Shuying Chen, Ziyang Huang, Guiye Wu, 2018, Requirement Documentation