# Games Engine Creation – Semester 2 – Assignment Part 2 – Super Mario Bros.

## Game Description

For my recreation of a Super Mario Bros. game, I took heavy influence from the original Super Mario Bros. My incarnation uses a tile-based level loader and features most of the core mechanics of the original game.

### Features

These are the graphical and ‘quality of life’ improvements the game offers.

* Revamped versions of the original graphics – Mostly brighter and more diverse colors, as well as some entirely reworked textures, like the floor tiles.
* Animated Mario – Mario displays a walking animation in both directions, as well as a jump and a stopped frame.
* Animation effects – When animations occur on objects that are purely for graphical effect, an effect is required. These include, but aren’t limited to: coin bounce from a special block, bricks breaking, score text when points are gained, level transitions, and enemy deaths. As well as these repeated effects, the more complex end level sequence is present at the end of every level. This ‘takes control’ of the player character and completes the level in a smooth fashion.
* Sound design – Most actions and effects have an accompanying noise. There is also music, which loops continuously.
* Debug mode – When debug mode is enabled, the graphics are switched to wire frame renditions of the hitboxes, and the sizes of the various sprites (including tiles, windows and entities, such as enemies) are all color coded based on their characteristics and type.

See Image [11]

### Mechanics

These are the mechanics of the game, the features that effect gameplay, and user input as opposed to graphics.

* Movable player character – Mario can move left, right, and jump. If he lands on an enemy, he kills it. If Mario is in large form, he can punch bricks above him hard enough to break the block.
* Tile class – There are many kinds of tile. The blocks can be passable or impassable, or one of the two unique types: item and flag blocks. All tiles react and look differently.

For a full tile list see bibliography [1].

* Two different enemies – There is the Goomba and the Koopa. The Goomba walks left and right, as does the Koopa, however the Koopa turns into a shell when hit normally, which will rapidly move back and forth along the floor.
* Information panels – There are 3 information panels in the main game. The leftmost is the FPS (frames per second), the right most the player’s score and the middle panel informs the player of the amount of remaining lives.
* Items – There are 4 items in the game, which are not counted as tiles. These are the mushroom, the ‘1-Up’, the star and the fire flower. The fire flower is the only item that has different mechanics than those of the original game, currently only giving a bonus 1500 points, instead of an entire state change for Mario.
* Levels – Each level is made up of tiles and is loaded in from a text file (seen in image [10]). The game has 3 levels currently, the first is modelled on Stage 1-1 from the original game, whilst the second is a lesser rendition of stage 2-1. The final level is a level of my own creation, which contains every type of tile usable block within the game, including all the various pipe orientations and corners.
* Interactive menu(s) – The main menu has two options: Play Game and Quit. These run operations when clicked, and also change color when hovered over.
* Scenes – The scenes are the various states of the game. There are 3 in the game: MAINMENU, MAINGAME and the incomplete LEVELEDITOR. The scene effects which controls are active, the information panels and windows on screen.

### Controls

The game features a variety of controls, many of which would be classed as developer controls.

* Arrow Keys / WA(S)D – Player Control. Left and Right are self-explanatory, whilst up is the control for jumping.
* ESCAPE – Whilst in the main game, ESC will take the player back to the main menu, where as if they are already at the menu, it will close the game.
* Left Click – Dependant on what the cursor is over. The game supports minimizing and force closing (via X). Left click is also used on menu(s) to click buttons.
* Scroll Wheel – Experimental control for zooming in and out. Excessive zooming causes the player character to move illogically when near edges of the level.
* Z – Activates Debug Mode.
* X – Restarts level. Currently also bound to manually switch the current scene.
* C – Skip to the next level.
* Q / E – Navigate the pages of multi-page menus.
* P – Manually toggle Mario’s state (small or large).

## Screenshots

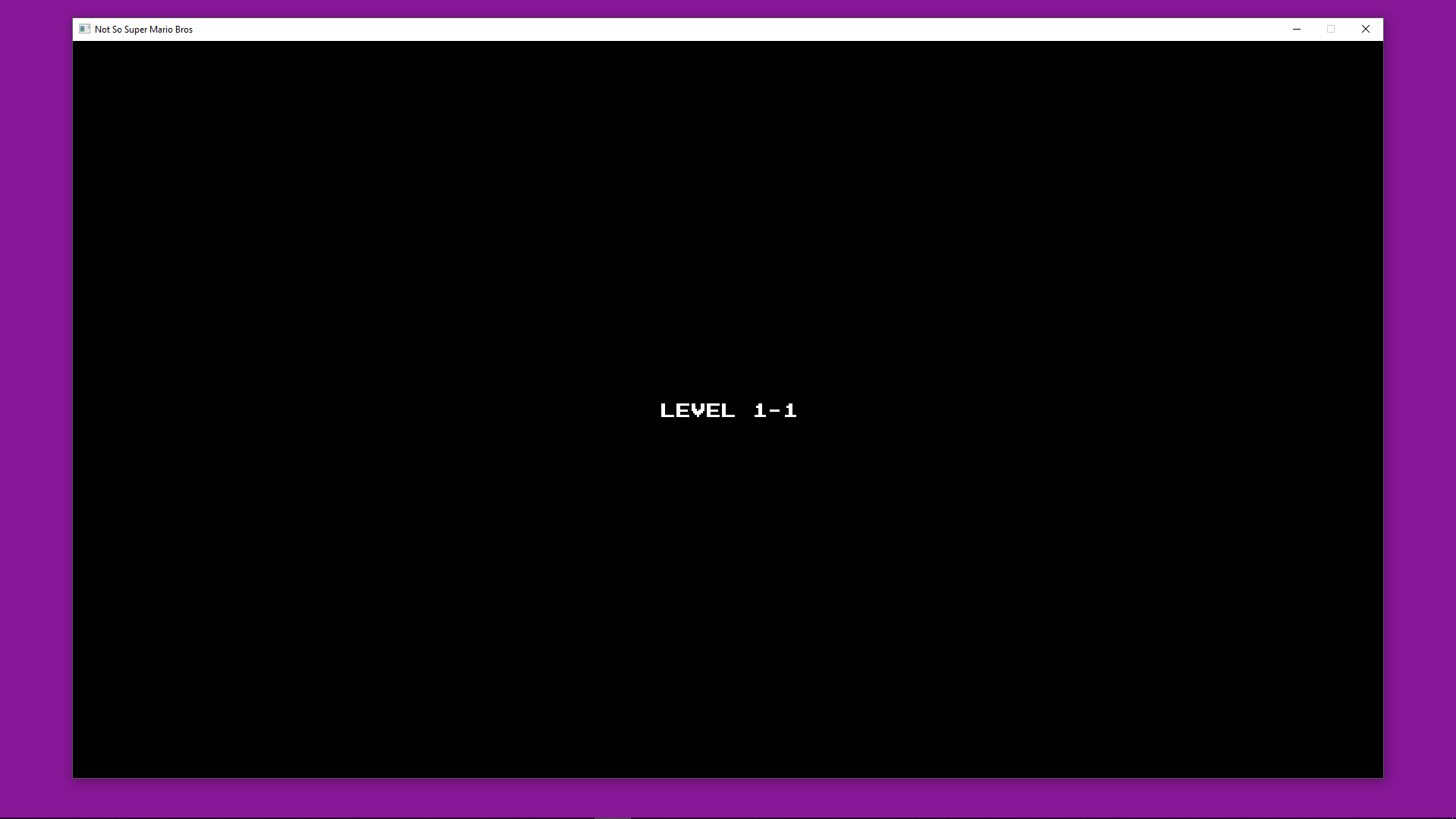
1)

2)

3)

4)







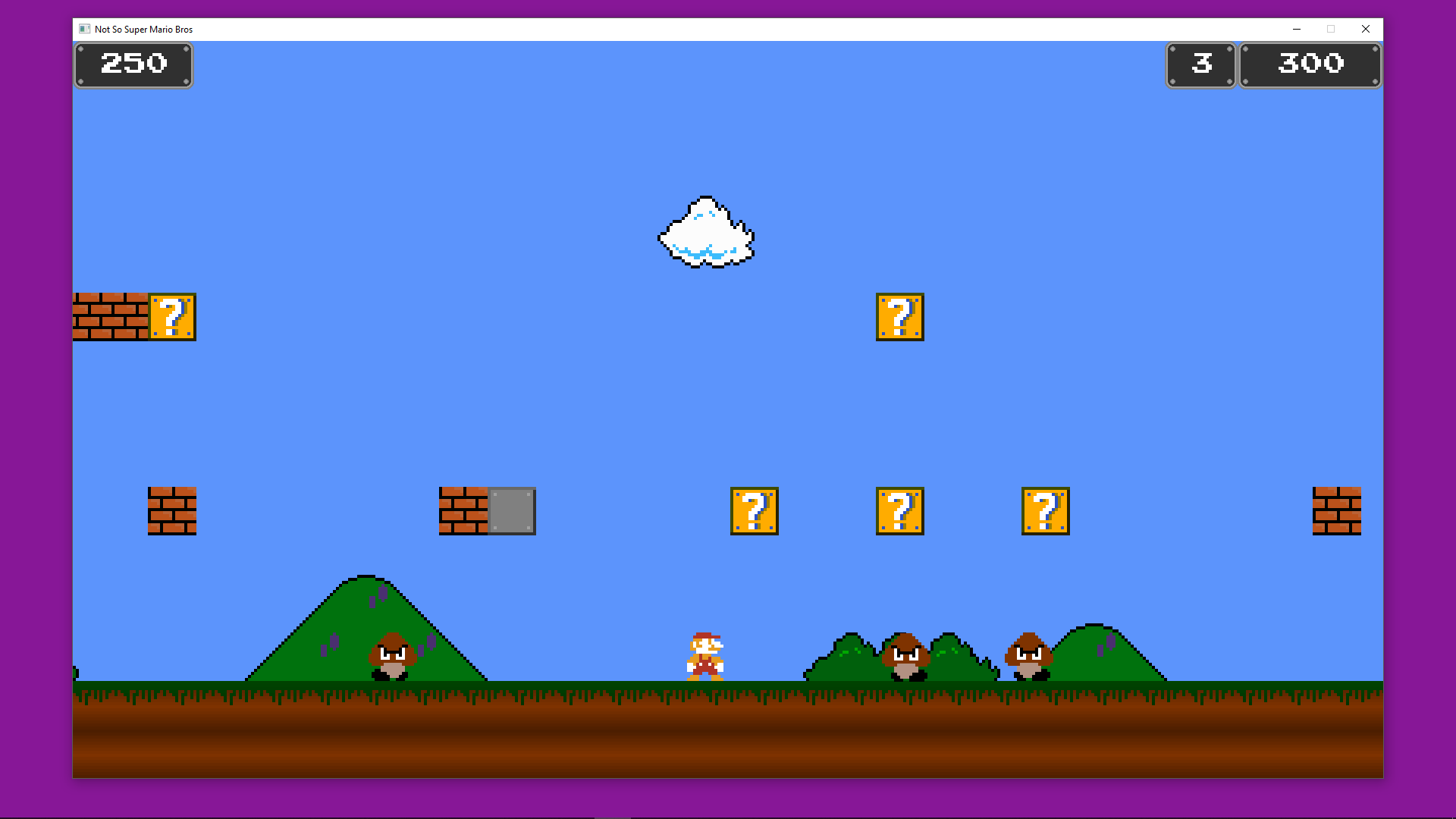
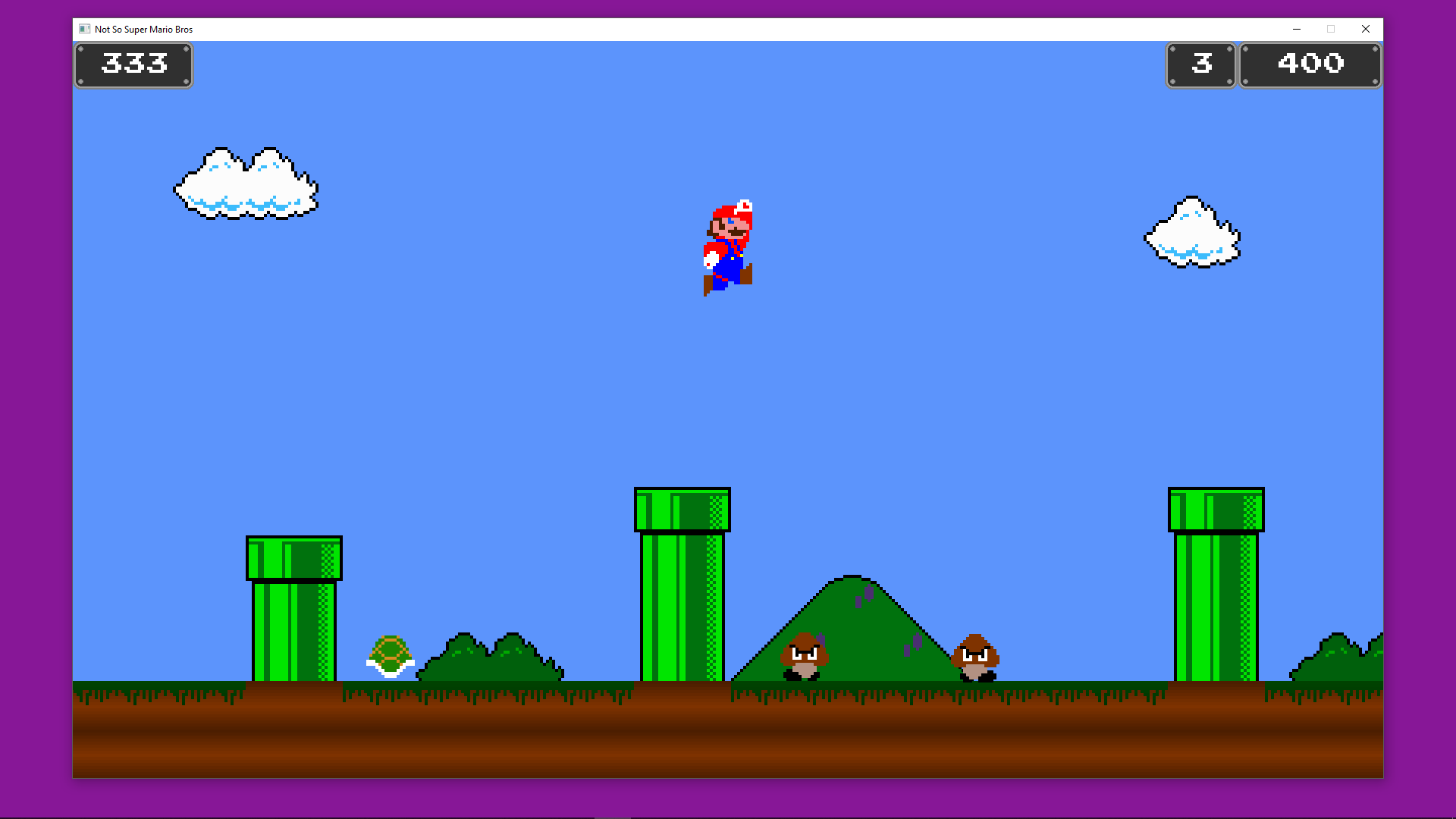


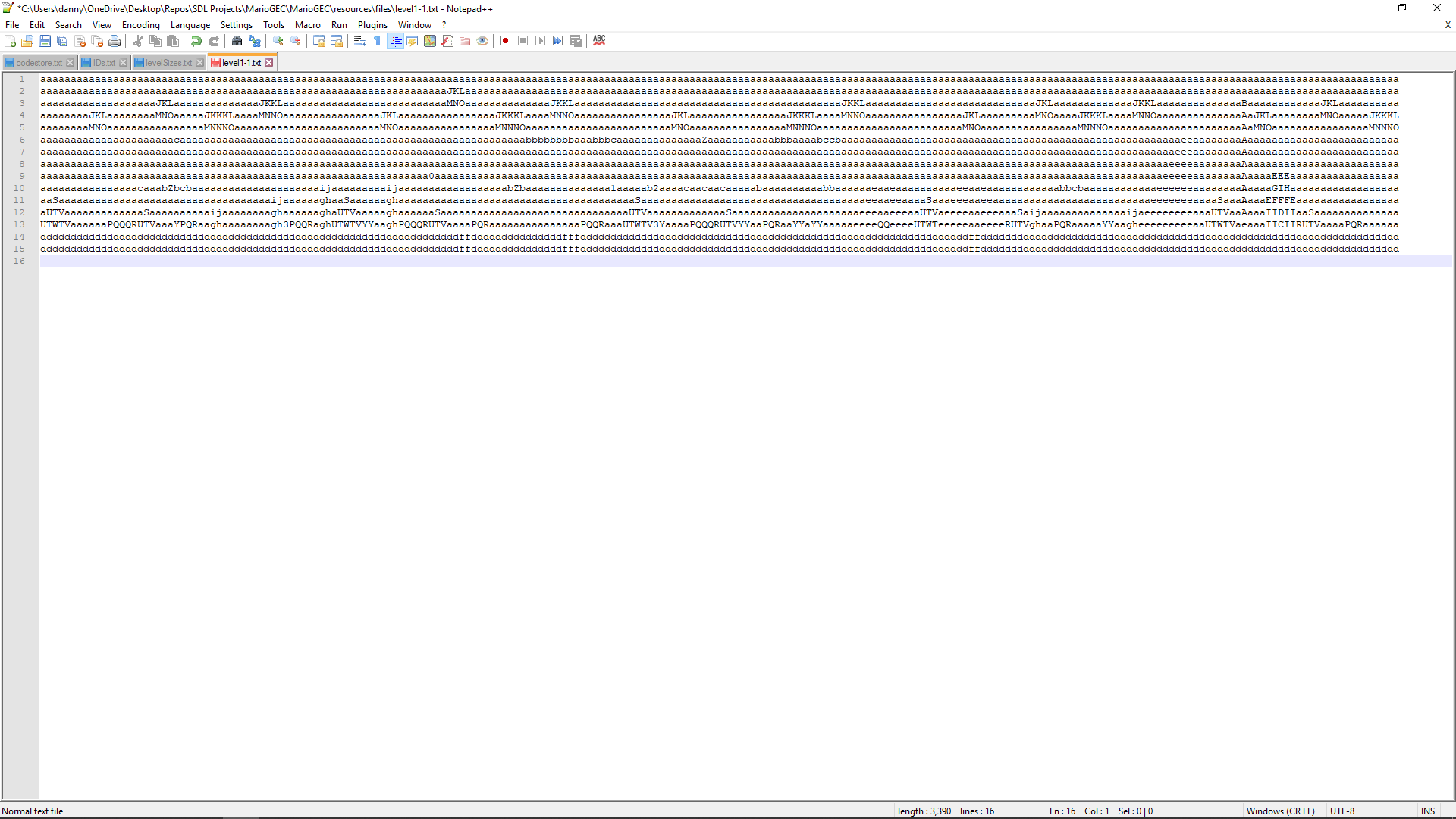
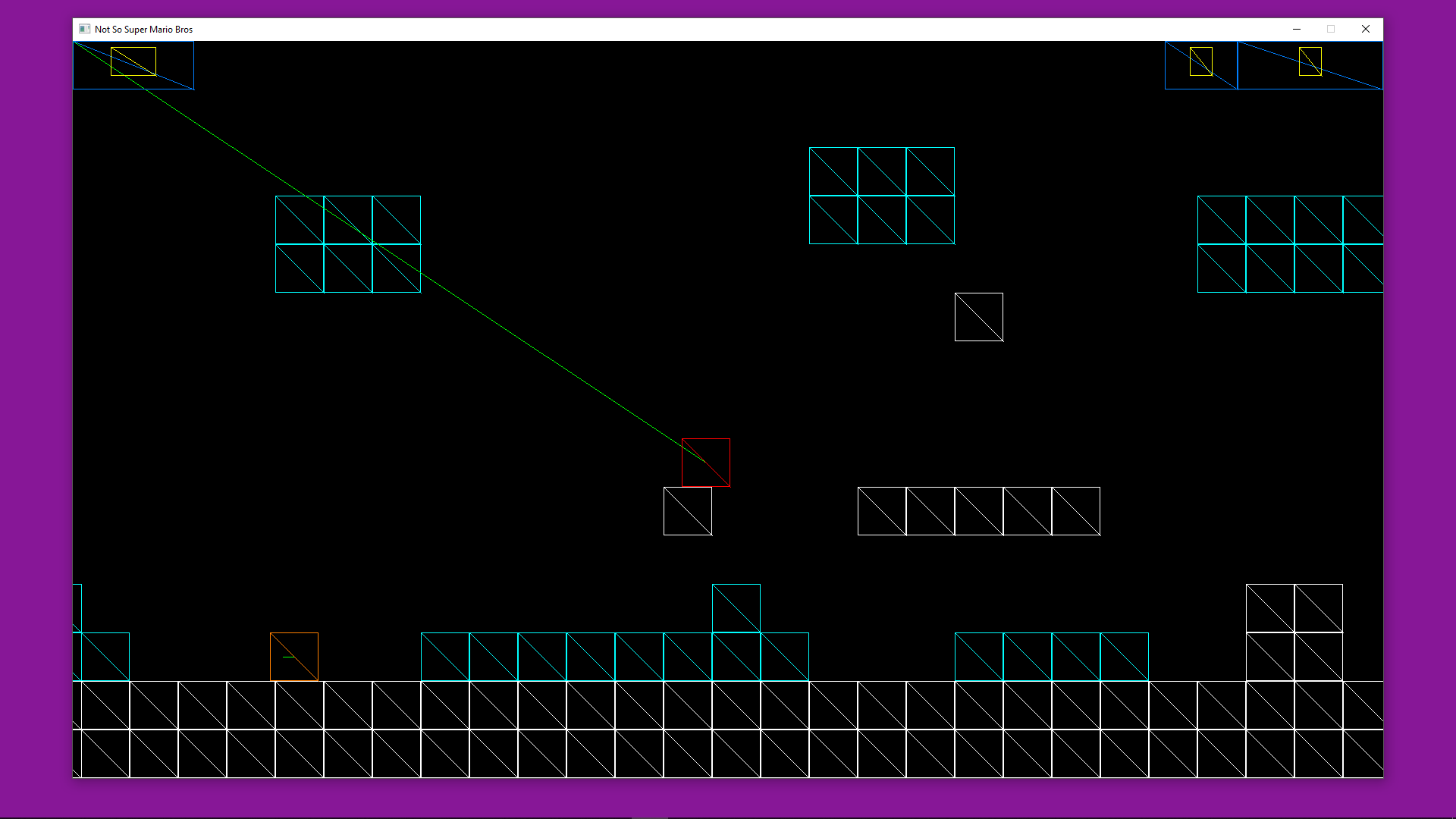
5)

6)

7)

8)





9)

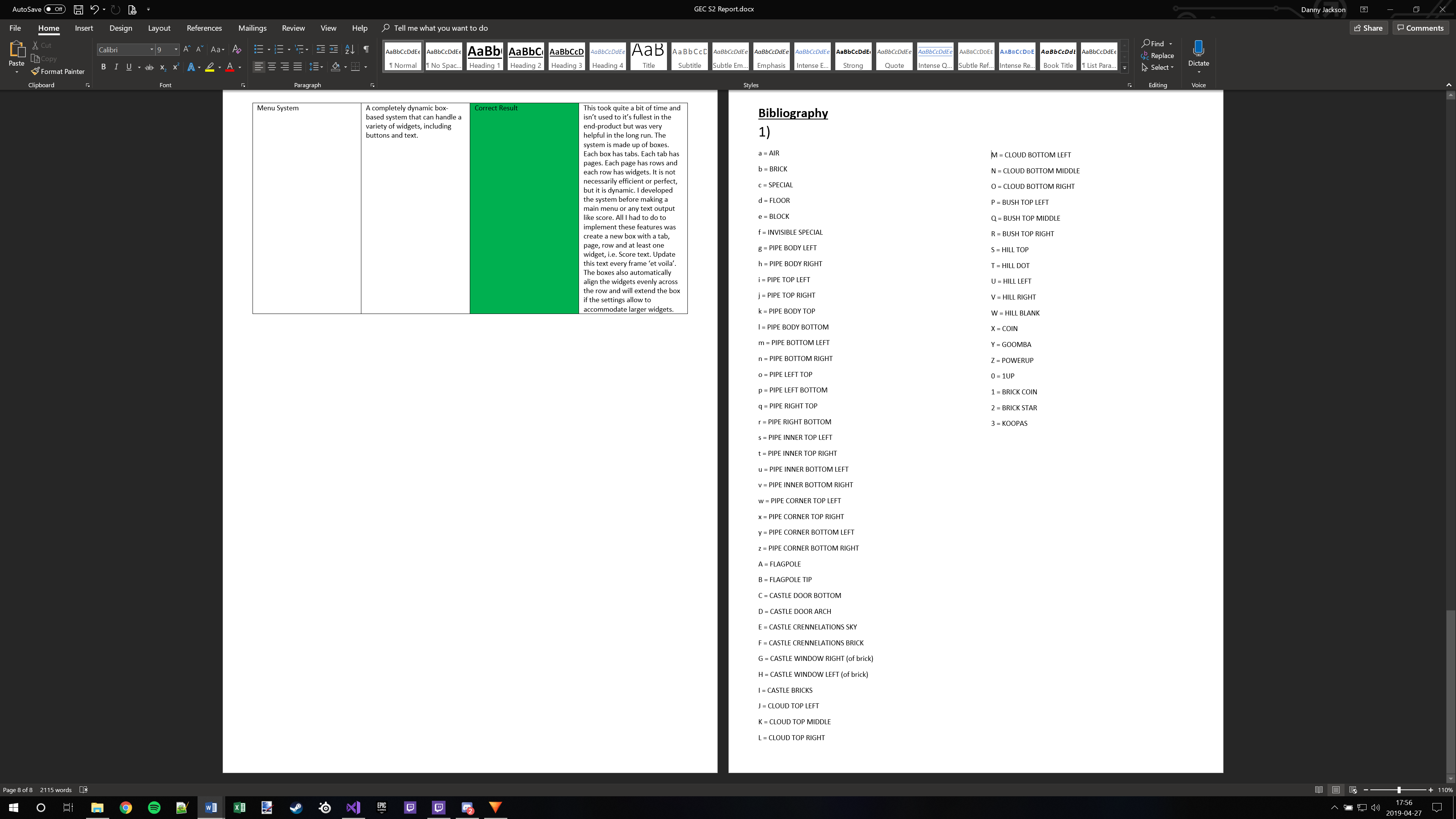
10)

11)

## Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Name** | **Expected Result** | **Actual Result** | **Notes/Review** |
| Window Opens | The SDL window opens on the main menu | Correct Result |  |
| Button Hover | Buttons change text color from white to yellow when cursor is above it. | Correct Result |  |
| Quit Button | The game exits on left click | Correct Result |  |
| Window ‘X’ Button | The game exits on left click | Correct Result |  |
| Play Game Button | The game switches to the MAINGAME scene | Correct Result |  |
| Mario Jump | Upon pressing ‘W’ or ‘Up’, Mario jumps, and a tone is played. The tone changes based on Mario’s size. | Correct Result |  |
| Goomba Stomp | If Mario is falling and lands on a Goomba’s head, the Goomba is killed. An effect replaces the enemy and 100 points are added to the score. | Correct Result |  |
| Special Block (Coin) | If the bottom of a Special Block (Coin) is hit by Mario, 200 points are added to the score and a coin effect is played on top of the special block. A sound effect is also played. The block turns grey. | Correct Result |  |
| Special Block (Powerup) | When the bottom of the block is hit a powerup will appear. If small a mushroom is spawned or, if large, a fire flower. |  |  |
| Brick Block (Coin) | Same as the Special Block (Coin), but will only go grey after all the coins in the block are expended | Correct Result |  |
| Brick Block (Normal) | When this block is hit from underneath when Mario is in large form, the brick will be removed, and a short-animated effect will play. | Correct Result |  |
| Brick Block (Star) | Same as Special Block (Animated) except it spawns a | Correct Result |  |
| Special Block (1-Up) | Same as the Special Block (Powerup) but It begins as invisible and spawns a 1up when hit from beneath. There are also no collisions in any direction other than moving up. | Correct Result |  |
| Level Loading | The level loads from a text file, by translating characters into tile objects. | Correct Result | Originally I used numbers spaced by gaps, but to reduce file size I switched to characters as they did not require the gaps to be read separately. |
| Escape | Pressing ESC will go back to the main menu if you are in the main game. If you aren’t in the main game and are already on the menu, it will quit the game. | Correct Result |  |
| Koopa Stomp | Same as the Goomba stomp but it turns into a shell. | Correct Result |  |
| Shell Moves | If the shell is at stop and a side is collided with it will rapidly move in that direction. It will move indefinitely until it is landed on by Mario. | Correct Result |  |
| Shell Kills Entities | If an entity is hit, it will take damage. | If Mario is hit, he will take damage regularly however it doesn’t kill enemies and just phases through them | The enemies don’t collide correctly with the shell. |
| Mario Death | If Mario is in large form, he flashes with invulnerability, before becoming small. If he is in small form he will die, playing a short animation and deducting one life from the total. | Correct Result |  |
| Game Over | If Mario dies whilst on one life the game plays the death animation and then plays a Game Over transition. It will return the player to the main menu | Correct Result |  |
| Bounding Box Collisions | If two hitboxes intersect on any two game objects, they collide and a reaction occurs, varying from preventing movement to causing damage. | Correct Result | Early on I had problems with Mario colliding with the rectangle in more than one direction. This meant he tried to evaluate in multiple directions and ended up flying up the brick or falling through it. I solved this by limiting the resolution of collision to a single direction (the most prominent one) and the closest colliding block as well. |
| Circle Collisions | Same as above but using circles instead of quads | Not Implemented | The system is a lot more efficient using grouped types (i.e. Tile and enemy). These types should only use one type of collisions for efficiencies sake, in conjunction with necessity for specific edges. |
| Level End | When Mario passes the flag pole, the player object is hidden, and the end level effect is started from where the player was. | Correct Result |  |
| Level Editor | Menu for choosing blocks, grid for placing blocks etc. | Not implemented except for a multi-page menu that has no actions linked to buttons | More of a in the future idea than an actual feature. However, it was useful in developing the menu systems. |
| Left Wall | As Mario moves right the left edge of the screen becomes the new ‘left wall’. Mario cannot pass through this wall thus can only move forward. | Correct Result |  |
| Camera | The camera is linked to the player object. If the player is near an edge wall. The camera stays locked to the centre of the screen. If not, the player is in the middle and the camera is locked to it. The same is true for the y axis. | Correct Result | The camera took a bit of work to get it to follow the player character but lock to the sides of the screen, but I am happy with the result |
| Zooming | The scale changes when scrolling. | This does make the camera locking incorrect when zooming out revels past the left wall. |  |
| Mario Moves | A/D or LEFT/ RIGHT will make Mario move in that direction with a slight acceleration. | Correct Result |  |
| Information Panels | All panels update their information when required (e.g. score). They also change size to accommodate all widgets | Correct Result |  |
| Menu System | A completely dynamic box-based system that can handle a variety of widgets, including buttons and text. | Correct Result | This took quite a bit of time and isn’t used to it’s fullest in the end-product but was very helpful in the long run. The system is made up of boxes. Each box has tabs. Each tab has pages. Each page has rows and each row has widgets. It is not necessarily efficient or perfect, but it is dynamic. I developed the system before making a main menu or any text output like score. All I had to do to implement these features was create a new box with a tab, page, row and at least one widget, i.e. Score text. Update this text every frame ‘et voila’. The boxes also automatically align the widgets evenly across the row and will extend the box if the settings allow to accommodate larger widgets. |

## Bibliography

1)