**Technical Design Document**

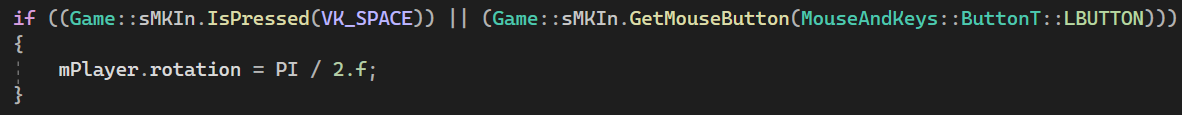
**Milena Wilkinson**

**Hammer is used**

* On mouse click/spacebar, the hammer will rotate its orientation by 90 degrees to be on its side

A picture containing shape

Description automatically generated



* Return the hammer back to it’s original state if the mouse is not being clicked:

Graphical user interface, text

Description automatically generated

**Mole Renders in a random place**

* Every few seconds a mole will appear in a random hole on the screen
* A new mole will only appear after the old mole has disappeared
* Moles will appear in a random screen area
* If the mole and the hammer sprite interact the mole will disappear

Diagram, schematic

Description automatically generated

Figure 1 Deciding coordinates to render moles

* Although seemingly random, the moles will still have to render in one of the holes, so we have to choose from pairs of coordinates
* These pairs of coordinates will be stored in an array.
* We can start by rendering a mole in every single hole, to check if coordinates we guess look accurate.
* For this, we will also keep an array of 9 moles that each coordinate will pair up with

Text, letter

Description automatically generated

Figure 2 Pseudocode example

* An int array containing the number of moles we want to render (num\_moles)
* A for loop that loop through num\_moles and renders each mole

Graphical user interface, text

Description automatically generated

Figure 3 Code for rendering multiple moles: INCORRECT

* The code seemingly works with no errors, but instead of drawing multiple moles, the same mole is redrawn over the previous one. How can we make this code draw multiple moles instead of the same one?
* mMole contains one mole sprite
* instead of one mole sprites, we need at least 9. So we will use the initial idea of a vector to contain an indefinite amount of moles.

Text

Description automatically generated

* This for loop will go through our vector of moles and setup each mole
* Each mole has a corresponding coordinate in hole\_coordinates

A screenshot of a computer

Description automatically generated with medium confidence

* (These coordinates need adjusting)
* We set the mole position to the x and y coordinate for each element in the vector2

Graphical user interface, text, application

Description automatically generated

* I then push back a sprite for each mole I want to render

Shape, circle

Description automatically generated

**Mole Collision**

**Shape

Description automatically generated**

**What we will need**

* A timer to measure the amount of time it has been before the mole can reappear again
* A collision function that detects when the hammer and the mole have interacted

**Collision function/code**

* For collisions we will use AABB (axis aligned bounding boxes) collisions
* We will use the top left and bottom right axes
* If we know the top left position, we can calculate the bottom right by adding the sprites size to the top-left position vector
* We check that the shapes are overlapping
* If both horizontal and vertical edges overlap then there is collision

Pseudocode

Pseudocode of AAB collision function for checking collisions



Figure 4 Pseudocode of AABB collision checking function

Text

Description automatically generated

Figure 5 C++ Code of collision checking

* Now we must decide where check\_collisions will be called in the code
* This ideally should make its way into the update function, as collisions will need to be checked constantly, every frame

**Rendering another mole after one has ‘disappeared’**

* Initially I had the idea of doing this after a certain amount of time, but we can simplify the process by rendering another one once the previous mole’s colour is set to transparent

Graphical user interface, text, application

Description automatically generated

**Vector holding scores**

* Score is updated each time the player whacks a mole
* The final score is output in RenderEnd()
* It is reset to 0 in UpdateEnd()
* We need to take each score before it is updated and store it in the vector.