**Programming Project Report**

Student Blake Williams

Student 010974718

**Problem Statement:**

The goal of this programming assignment was to expand upon the functionality of the previous homework (HW4), by adding the ability for a user to control a player model, and move this player model to collect coins throughout the maze. Once all the coins were collected, then a “YOU WON” is announced.

**Design:**

To achieve this, first the ability to take in user input needs to be added to keyboard functionality. Next once a specific key is read, then update the position of the player model in display(), then re-render the maze with the player position updated. Checks will have to be made to make sure the player is not rendered inside a populated block such as brick, wood, or rock. This will be handled in the keyboard() function. Then a random number generator will handle generating gold on the screen, and another statement inside the display() function will handle when collision with player and gold is made.

**Implementation:**

To implement this, first the player and gold textures are initialized, inside the init() function, same as the other textures. First was making gold generate on the screen. To do this a function was created called treasurePlace() which creates a variable called random. Next a nested for loop acts as a x,y coordinate system running through the maze array. With “i” being equivalent to the y position and “j” being equivalent to the x position. Inside the loop the random variable is set to rand() %2 which generates either a 0 or a 1. Then an if () statement checks to see if the random number is 1, and if the “xy” position in the array is empty, if this is true then a ‘g’ is placed inside the maze.txt at that position, and a global variable called numGold is incremented by 1, representing the number of gold pieces on the board. This function only needs to be run once to place the gold, so it is called in the init() function as to only run once. As if it was called in the display() function every time the render is redrawn there would be a new arrangement of treasure. After this the display() function was modified to recognize the letter “g” added by the treasurePlace() function, and render a treasure block at that location. The treasure render inherits the same properties as the player, so they are the same size. With the Gold being properly counted and added, the next functionality was to add player movement. This was handled inside the keyboard() function, where a new state called MOVE was created, inside that state had 4 directions of movement (up, down, left, and right) corresponding to the “WASD” keys. To physically move the player model, we need to update its xPos and yPos variables to +-1 depending on the direction they want to move. If “w” is pressed then the yPos is incremented by 1, if “s” is pressed yPos is decremented by 1, if “a” is pressed xPos is decremented by 1, and finally is “d” is pressed xPos is incremented by 1. Now inside this is the need to check for collision with other blocks, to do this when a key is pressed the new yPos and xPos is not immediately assigned, instead it is added to 2 temporary variables. Then a if statement makes sure the new positions fall inside the bounds of the maze. If this this true, then another if statement checks the maze at the new position assigned to the 2 temp variables, to see if there is collision with other blocks. If there is no collision then the 2 temp variables values are set the the xPos and yPos values, causing the player position to be updated. If there is collision the if statement never runs, therefore the player remains in place. The final thing to add was erasing gold if the player comes into contact with it, and displaying a “YOU WIN” if all gold is collected. To do this, inside the display() function another if statement was added that checks if maze[yPos][xPos] == ‘g’ which essentially checks if the player is rendered on top of a gold piece. If the are then the value at that position is set to empty, and the numGold variable is decremented. Once the numGold variable reaches 0 a if statement inside the keyboard loop triggers displaying “YOU WIN” to the screen.

**Testing:**

The testing of this program was as follows. First implement a method, such as Gold Placement, run the code, see if the Gold placed on the screen, was there enough Gold, change needed variables, and repeat. For user movement, the initial functionality of moving the player was first added by incrementing and decrementing the xPos and yPos variables. This initial phase, there was no error checking so the player could phase through the wall. Next was adding collision checking. This was tested by attempting to enter each of the 3 texture types, if I failed to enter, then the error checking passed. Finally, I had to add the check to make sure the player does not leave the maze. This was tested by trying to leave the bounds at the entrance and exit of the maze. If I failed to leave, then the error checking passed. Overall, everything worked as expected, the only unknown is that every time the image is re-rendered, it glitches for a second because of the GLUT\_SINGLE being used, when GLUT\_DOUBLE is called instead it breaks the code. Also the provided maze had a single wood render, blocking a large section of the map. This was removed to allow the player to collect all the gold.

A screenshot of a video game

Description automatically generated

(Initial Start)

A screenshot of a video game

Description automatically generated

(Collecting Gold)

A screenshot of a video game

Description automatically generated

(Game Finished)

**Conclusions:**

Overall, the project was a success, as the player model was properly rendered onto the initial maze screen. Then the user can move the player and collect gold pieces. The user is not able to enter textures other than the gold texture. Once all the gold is collected a “YOU WIN” is displayed in the console. If I were to do this differently I would attempt to get the GLUT\_DOUBLE working as to avoid the glitch of rendering the maze. This project took 4 hours with 30 minutes dedicated to the report.