3. WRITTEN RESPONSES

3 a.

3.a.i.

The purpose is to provide a pathway for entertainment for kids 3-7 by providing an interactive maze game that requires the use of timing and decision-making while also being fun. This allows kids to burn some time while learning from it.

3.a.ii.

The user utilizes the WASD keys to move around a character while shooting water at balls. The balls are inside a maze that the character must go through without touching anything. Each touch will send the user-controlled character back and lose one health point. Once at the goal, the game starts up a boss which changes based on whether the user picked easy or hard at the beginning. The game ends when either the boss is defeated or when the health point is 0.

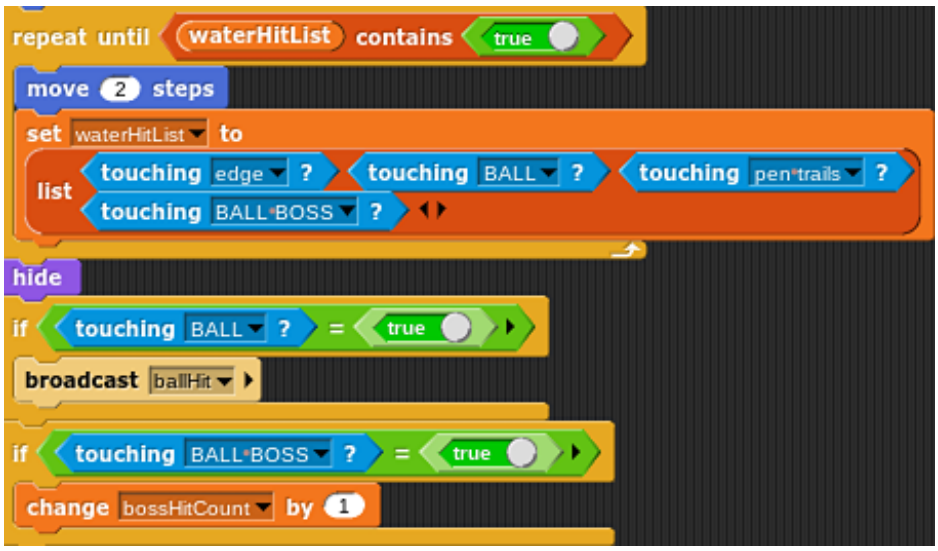
3.a.iii.

The inputs shown in the video are: when the green flag is clicked, when you type in either “easy” or “hard”, when you press “W”,” A”,” S”,” D” when you press “space”, and lastly when the user-controlled character touches anything within the "touchList".
The outputs shown in the video are: the “difficultyAnswer” block which makes “diff” = 1 or “diff” = 2 which activates the “bossHits” block and the “easy/hardSpawn” block. Additional outputs are the “UP” block, “LEFT” block, “DOWN” block, “RIGHT” block, “waterMovement” block, and the “checkTouch” block.

3 b.

3.b.i.

3.b.ii.



3.b.iii.

waterHitList

3.b.iv.

The list contains objects that when hit should hide the water which is constantly moving. In addition, the list is reset after use to allow for repeated uses(allows the user to shoot water on each press of the "space").

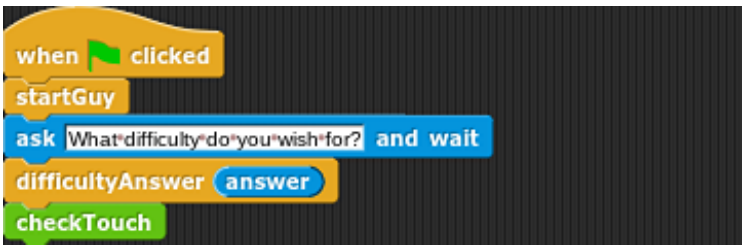
3.b.v.

This code manages complexity by holding multiple values that, if this list were not to be there, would have to be done individually. The checking of these values would have to be in multiple “ifs” that would be constantly repeated thus slowing down the code and making it difficult to figure out where a problem is if one were to arise in the process of coding. To reset the code for repeated use, there would be a need to constantly set each value to false leading to a further blurring of the code. With this list, the code is less jumbled, easier to understand, easier to fix, and easier to reset for repeated use.

3 c.

3.c.i.

3.c.ii.



3.c.iii.

The procedure asks the user for the difficulty they want and broadcasts the answer which determines how hard the game is and if the boss fight occurs. If the user doesn’t input either easy or hard, it tells the user to put either of those and repeats the question.

3.c.iv.

First, the algorithm takes a variable called “reset” and sets it to 1 which prevents the ball spawning from previous resets to continue. Afterward, it has a “repeat until” which locks the code into an infinite loop with only 2 escapes. It does this by checking if "repeatQuestion" is 1 and then checking the difficulty parameter to see if it has either “easy” or “hard”. The difficulty parameter is set to be the "answer" to the question. This then sets the “diff” variable to 1 if the difficulty parameter is "easy" and 2 if it’s "hard". Then it broadcasts “easy/hardRegister” which sends a message to determine whether easy or hard has been picked and changes the speed of respawn, whether there is a boss fight, how much health the boss has, and sets repeatQuestion to 1 which stops the repeat. Each time “easy” or “hard” are not inputted, the code says, “Please say easy or hard” and asks the question again.

3 d.

3.d.i.

First call:
difficultyAnswer(easy)

Second call:
difficultyAnswer(eezy)

3 d.ii.

Condition(s) tested by first call:
This is testing the "if" part of the "if-else" statement which goes into another "if" statement. The first "if" takes only lets the user through if they had inputted "easy" or "hard" and the second "if" takes this and activates the code within either if "easy" or if "hard".

Condition(s) tested by second call:
This is testing the else part of the if-else statement. This part should repeat if the user doesn't input "easy" or "hard".

3.d.iii.

Results of the first call:
The diff is set to 1 and the broadcast is sent resulting in the spawn speed of the balls being 5 seconds, the boss health being 5, and the "repeatQuestion" variable being changed to 1. When the user wins, they will be sent to the “win” screen rather than the “YOUWIN?” screen which is for hard mode/ when “diff” is 2.

Results of the second call:
“The guy”(user-controlled character) says, “Please say easy or hard” and asks the question again. Which will run through the if-else statement again using the answer as the difficulty to which the if-else statement is reading for.