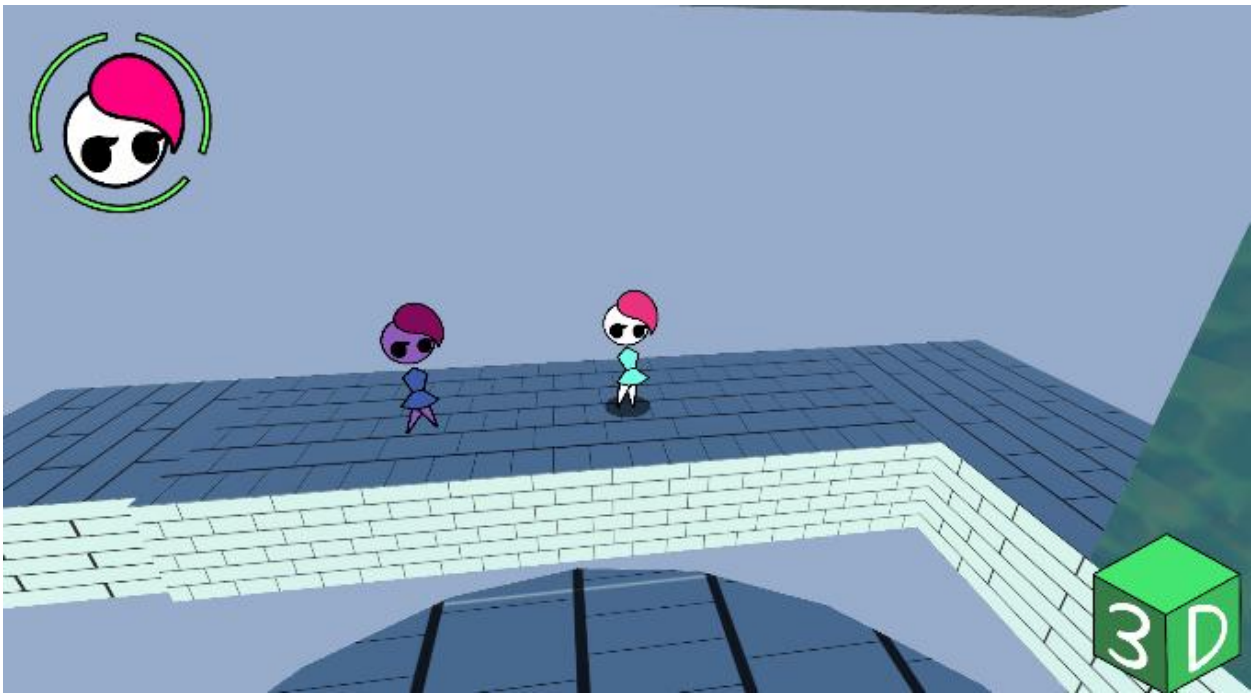
Report

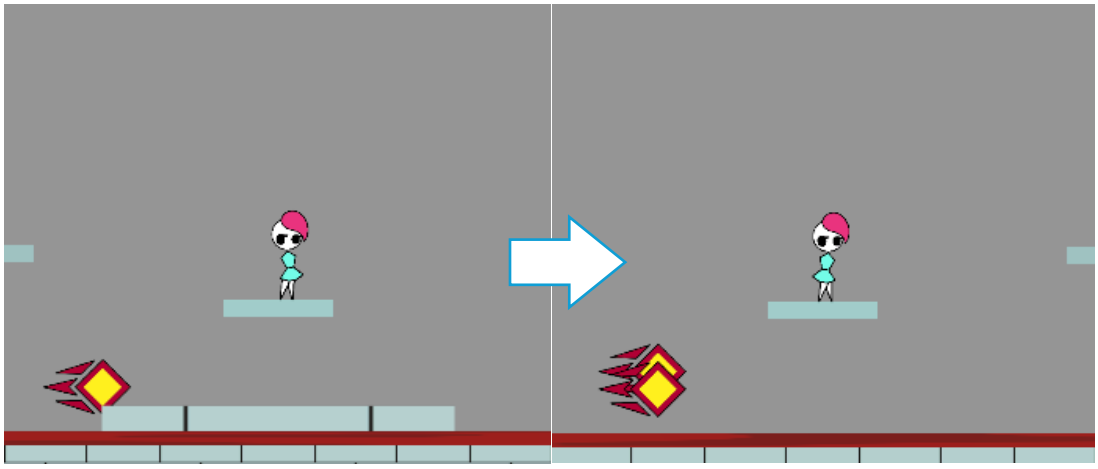
Tier 1



Given the choice between an array and a list I chose to use a list as it is easier to modify in size and add to it after initializing. I created a struct containing sprite and transform information, getting this information from the player at every frame and adding it into a list of said struct. This was used to make an enemy in the game that gets the information from the first element in the list every frame and deletes said element after using it, so the list does not continue growing overtime until it overflows, creating the illusion of a shadow player chasing the actual player. This did not prove very challenging as lists are very intuitive and easy to use and understand.

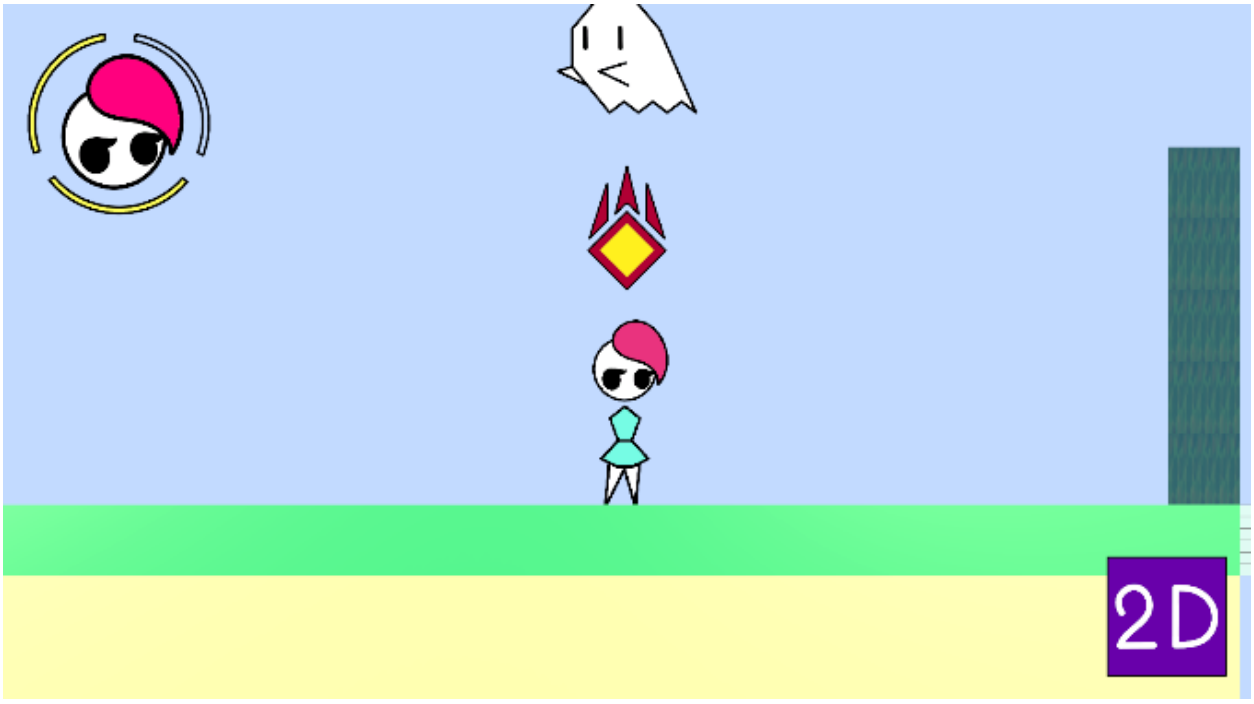
During feedback it was recommended to switch the list for a queue in order to improve the time complexity of the algorithm, which was easily done immediately after.

Tier 2

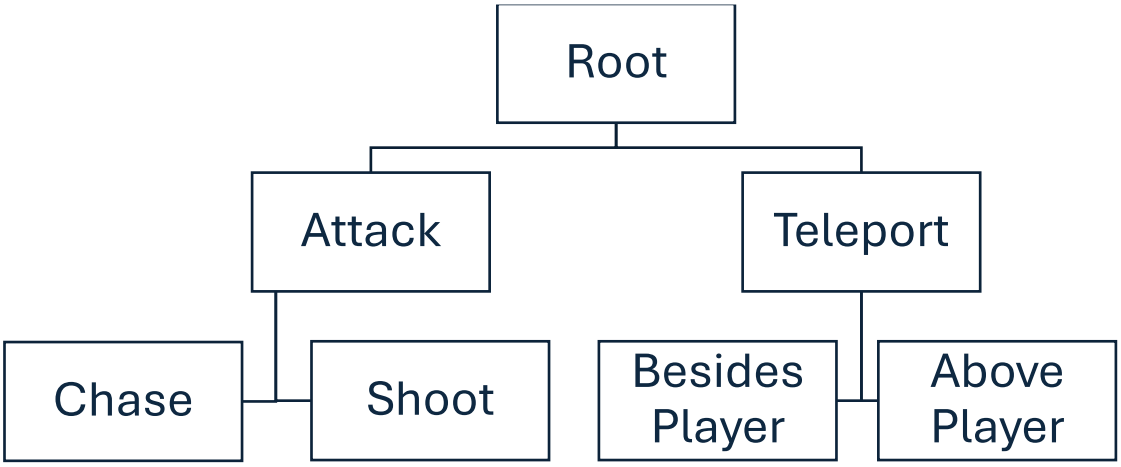


With its dynamic memory allocation and therefore the ease at redefining size, a linked list results useful in creating an attack that gets stronger overtime during a boss battle. Adding a new node at the tail of the list. Adding the linked list felt tricky at first specially modifying a previously existing script that was already likely the most complex in the project, leading to some problems like forgetting the line that advances to the next node in the list at one point, causing Unity to get stuck in a permanent loop. But with enough perseverance I managed to give the boss a new attack that spawns horizontal fireballs at a random height, adding more fireballs by adding nodes to the list as the boss loses health.

Tier 3



Trees allow for more complex systems, like getting the opportunity to create an enemy with multiple random behaviors, like teleporting around, or using one of multiple attacks.



Setting up the tree itself was the hard part, as well as trying to implement the mechanic decently and still looking rough due to lack of time to create proper animations or doing in depth testing.