Software Implementation and Testing Document

For

Group <6>

Version 1.0

Authors:
Reece B
Benjamin R
Aiden L
Kevin O
Nate L

1. Programming Languages (5 points)

We used C# for our scripts in our Unity project because C# is better for novice game developers like us since it handles garbage collection automatically and avoids memory leakage. We use C# for our Unity scripts to create our game components, trigger game events, modify our component properties over time, and determine how our game responds to user inputs. We currently use C# for our player movement script, user inputs to move left and right, jumping and wall jumping, camera functionality, following the player through the level, and a projectile attack script that allows the user to shoot fireballs and plan to use C# for more of our scripts as we get further into the project.

List the programming languages use in your project, where you use them (what components of your project) and your reason for choosing them (whatever that may be).

2. Platforms, APIs, Databases, and other technologies used (5 points)

We are using Unity as our main platform to develop our 2D puzzle solving platformer. Using Krita for some sprite animation.

List all the platforms, APIs, Databases, and any other technologies you use in your project and where you use them (in what components of your project).

3. Execution-based Functional Testing (10 points)

We have created a basic level in Unity where we have begun to test and implement a player, their movement controls and physics, and their sprite animation. We did so by downloading free assets from the Unity Asset Store and began to learn the animation process for a sprite. We currently have idle, left, and right movement and jumping animations for our placeholder character sprite, gaining the know-how needed to animate the sprites we create. We also have created a character movement script that allows the player to move left and right using the arrow keys or the A and D keys, respectively, and the spacebar to jump and wall jump up vertical platforms. We hope to change the wall jump into a jetpack-like feature that will let the user fly or hover, but only for limited periods.

*Describe how/if you performed functional testing for your project (i.e., tested for the **functional requirements** listed in your RD).*

4. Execution-based Non-Functional Testing (10 points)

We performed non-functional testing on the player's movement as, at times, depending on the angle of the player's wall jumping or platform landing, the sprite will get flipped and land horizontally and is stuck in that position, but the player is still able to move and jump which may cause the sprite to flip again. Also, we are still messing around with the colliders on our platforms, as walking on some of them causes the player's sprite animation to get stuck on one frame and will only allow the user to jump once they get off the platform.

*Describe how/if you performed non-functional testing for your project (i.e., tested for the **non-functional requirements** listed in your RD).*

5. Non-Execution-based Testing (10 points)

We performed non-execution-based testing on our player movement script to try and determine the cause of the sprite being flipped instead of always staying vertical. We are still working on this fix but have reviewed the code several times. We have also worked on a portal script as a filler for our time travel mechanic, where if a player touches a

block with the properties of the portal script, it will take them to another block that has the same script on another part of the level. We are still attempting to implement this correctly and have reviewed the script several times.

Describe how/if you performed non-execution-based testing (such as code reviews/inspections/walkthroughs).