Game Name (we don't have one as of now)

## **Elevator Pitch**

As someone who had no experience with Computer Science before college, I was obviously intimidated by computer programming. Any time I would see someone working on a program, I would see the console and immediately realize I had no clue what was going on, let alone simple functions, loops, etc. Going into college, these concepts were difficult for me to grasp right away, so I understand the struggles of new Computer Scientists. Our game’s goal is to teach simple concepts of computer programming, to take away the intimidation factor that comes with novice programming.

## **Influences (Brief)**

* *Influence #1*:
  + **This Is The Only Level** (game)
  + While thinking through game ideas, I was drawn to a similar game, This Is The Only Level, which is a platforming game which you work through puzzles. I thought, an interesting way to present the information would be working through puzzles in a way similar to that game, which I played a lot of in grade school
* *Influence #2*:
  + **IDEs**
  + As of now, we hope to design the game to at least have the background of an IDE, or something similar to that.
* *Influence #3*:
  + **LightBot** (game)
  + This game essentially teaches functional decomposition in reverse. The game limits the amount of commands that you can give to the robot(with the goal of having him reach the light bulb). Some levels are impossible to complete if you do not combine commands into groups and use these groups multiple times(for loops).

## **Core Gameplay Mechanics (Brief)**

* Platforming, to reach areas to complete puzzles
  + Use arrows keys(or WASD) to navigate and spacebar to jump
* Puzzles, which allow you to reach extra areas and the end of levels
  + Tasks emulating the theme of coding ideas such as for loops -- a bin of colored soccer balls, put all the soccer balls in the net(is this too detailed?)
* Power-ups that only last through one level or one world(series of levels), to illustrate variable scope

# **Learning Aspects**

## **Learning Domains**

*(dont really know what this is)*

## **Target Audiences**

Newer audiences, like those in k-12 who would play this during computer labs and gain interest, or those of our age, becoming interested in Computer Science (intimidated by computer science)

## **Target Contexts**

K-12 computer labs during free time, high school labs during free time, possibly college

## **Learning Objectives**

* By the end of the game, players will be able to predict the behavior of a for loop in the context of the game, and match the in-game components to components of a real coding for loop
* By the end of the game, players will be able to predict the behavior of a for while loop in the context of the game, and match the in-game components to components of a real coding for loop
* By the end of the game, players will be able to identify whether a variable(power-up) is global or local

## **Prerequisite Knowledge**

* Participants should have knowledge of highschool level math
* Participants would have a much easier time with basic programming knowledge

## **Assessment Measures**

Pre game, we will assess the user’s knowledge of concepts with a quiz or simple project

Post game, we will assess the user’s learnings with a similar quiz or project

# **What sets this project apart?**

* This game presents a visual representation of programming logic that is underutilized or poorly executed upon when trying to introduce coding concepts to new learners. The player engages in a fun game to later have the game mechanics equated to programming components, which generally results in better retention.
* Not many games growing up involve any sort of programming, or major logic puzzles, so children are introduced to these concepts at an earlier age, and can become interested in the field
* Students in early computer labs class tend to play games on coolmathgames.com, or adventuregames.com etc, and could come into contact with this game instead

# **Player Interaction Patterns and Modes**

## **Player Interaction Pattern**

A singular player (as of now unless we can add multiplayer somehow) will traverse through levels with platforming and puzzles. Until we begin level designing, I couldn’t give any examples

## **Player Modes**

* *Player mode #1*: Playthrough mode, where the player is able to play through the game like normal, completing puzzles and platforming
* *Player mode #2*: Help mode, where the player is able to consult the game for help, and direction, most likely coming through after a certain number of attempts, or a time limit is reached

# **Gameplay Objectives**

* *Primary Objective #1*:
  + Description: Beating the game, by getting past the puzzles and platforming
  + Alignment: *These puzzles will introduce the concepts of coding components such as for loops, and demonstrate how they work*
* *Primary Objective #2*:
  + Description: Finding extra power ups, or point pickups to increase your score
  + Alignment: *Scope of variables*
* *Primary Objective #3:*
  + Description: Beat the game as quickly as possible
  + Alignment: *To show the importance of speed and efficiency in a program and teach deadlines*

# **Procedures/Actions**

Players interact with their character with the arrow keys

* Left → left movement
* Right → right movement
* Up → jump action
* Down → possible crouch action?
* E → use or interact with an object
* R → possible shoot ability if we add one

# **Rules**

Players will have a certain amount of lives (to be determined)

Players will have a certain amount of time (to be determined, probably with difficulty levels)

These rules will make it so there is some challenge implemented into the game

# **Objects/Entities**

* We need to design levels to get through
* We need to design enemies (if we add any)
* We need to design puzzles
* We need to design power ups / pickups
* We need to design a lot of things now that I think about it

## **Core Gameplay Mechanics (Detailed)**

* *Core Gameplay Mechanic #1*: As of now, the main concepts of the puzzles are that they are based on for loops with increasing complexity. The player will have to complete a combination of platforming and a mildly repetitive task, such as retrieving numerous objects from the same location. The platforming may increase in difficulty after the first 1 and 2 retrievals. From here, the player will have the option to somehow record their movement as a power-up, and then have that movement be repeated however often they like. This will serve as a function, demonstrating both the concept of a for loop, as well as the concept of scope, as the power-up(function) will be unusable in the next level(or the next world).
* *Core Gameplay Mechanic #2*: *Describe in 2 paragraphs or less, along with how it generally works*
* *Core Gameplay Mechanic #3*: *Describe in 2 paragraphs or less, along with how it generally works*

## **Feedback**

*Explicitly describe what visual/audio/animation indicators there are that give players feedback on their progress towards their gameplay objectives (and ideally the learning objectives).*

*Describe what longer-term feedback you detect and give that guides the player in their learning and lets them know how they are doing in regards to the learning objectives.*

Little popup

# **Story and Gameplay**

## **Presentation of Rules**

Players will hopefully be given short animated tutorials on game objectives, like collecting keys, opening doors, etc. Small signs with pop up text may be added for extra, but will not interfere with gameplay

## **Presentation of Content**

Our game won’t teach someone how to program, rather it is meant to strengthen one’s knowledge of major concepts such as scope, loops, etc. We will be showing these concepts through our puzzles.

## **Story (Brief)**

*The Summary or TL;DR version of below*

## **Storyboarding**

We aren’t really sure yet how to take the story of our game, however we are deciding and hopefully will come up with something between our options throughout the week

# **Assets Needed**

## **Aesthetics**

Old school Super Mario meets a better color scheme and possibly a more futuristic look

## **Graphical**

* Characters List
  + *Characters 1*
  + *Characters 2*
  + *...*
* Textures:
  + *Texture 1*
  + *Texture 2*
  + *...*
* Environment Art/Textures:
  + *Environment Texture 1*
  + *Environment Texture 2*
  + *...*

## **Audio**

*Game region/phase/time are ways of designating a particularly important place in the game.*

* Music List (Ambient sound)
  + *Game region/phase/time*: *Example 1*, *Example 2*
  + *Game region/phase/time*: *Example 3*, *Example 4*

*Game Interactions are things that trigger SFX, like character movement, hitting a spiky enemy, collecting a coin.*

* Sound List (SFX)
  + *Game Interaction*: *Example 1*, *Example 2*
  + *Game Interaction*: *Example 3*, *Example 4*

# **Metadata**

* Template created by Austin Cory Bart acbart@udel.edu, Mark Sheriff, Alec Markarian, and Benjamin Stanley.
* Version 0.0.3