**The Nightmare**

Story:

Enter Luigi – Plumber Extraordinaire, Vanquisher of Evil and forever living in his brother’s long shadow. The outside world sees him as the honourary side-kick of his brother in red, and hapless derivative of their beloved “Hero”. It’s a very different view from the inside.



Luigi has suffered with a severe inferiority complex because of his brother and his “heroic” deeds. Drugs, mental breakdowns and suicide attempts are repeatedly scarred in his past.

Every night, when others sleep their fatigue and stresses away, Luigi finds no solace. The external enemies may be defeated, but his own inner demons find him every night without fail.

Will he ever escape their hold, his self-loathing and most of all his brother’s shadow?

The Game:

The Nightmare pits players against Luigi’s internal enemies and vices by representing the very familiar environment, of the regular Super Mario Brothers, in a nightmarish visage. Night has fallen in the mind sleeping mind of Luigi and the fiendish critters have been twisted into terrible creatures of dread. The green pipes that used to represent a safe haven or moving forward are now empty husks that give no escape. Each level is more difficult than the last, representing the inner struggle of Luigi and the futility of his efforts.

The task is difficult, but not impossible. There is hope for his salvation. What awaits him at the end of it all…?

Controls:

|  |  |  |  |
| --- | --- | --- | --- |
| Key | Function | | |
|  | Main Menu | Pause Screen | In Game |
| UP ARROW | Move Up | Move Up | Jump |
| DOWN ARROW | Move Down | Move Down | Nothing |
| RIGHT ARROW | Nothing | Nothing | Walk to the Right |
| LEFT ARROW | Nothing | Nothing | Walk to the Left |
| SPACE | Select | Select | Shoot |
| ESC | Nothing | Nothing | Pause |

Software Development Methodology

An agile methodology was chosen as it allowed all members to collaborate in a dynamic

way. A design process was followed, which began with outlining the game and the basic

code that would be needed. This was divided amongst the group, who each worked on

developing their allocated area.

Bug testing was done individually at first, in an isolated environment, for different portions

of code. This allowed the member responsible for a particular piece of code to better detect

the reasons for bugs in that code later on, when different pieces of code were integrated.

Each member followed a process of continuous iteration, with the code being improved for

efficiency and simplicity.

The agile methodology proved to be successful as when working in a large team, it is easier

to follow a dynamic approach of solving problems when they occur and implementing new

ideas when they are found, rather than planning the whole process before beginning

development.

Gameplay Implementation Methods

Brief descriptions of all the methods that are used in implementing the game are given below:

2.3.1. Creating a level

* For every level completed our main character goes into the next level where he will encounter enemies, pipe lines and snakes that is in a different position than in the previous level.

2.3.2. Blocks

* create: The block is drawn with an allegro command.
* block\_limit: Checks to see player has exceeded the bounds of the block.
* box\_right: Ensures player does not pass through block.
* box\_left: Ensures player does not pass through block.
* noblock: Ensures player does not pass through block.

2.3.3. Pipes

* createPI: Pipes are drawn with Allegro command.
* checkpipe: ensures that player does not exceed the bounds of the pipe.
* pipe\_right: Ensures player does not pass through pipe.
* pipe\_left: Ensures player does not pass through pipe.

2.3.4. Spikes

* createSpike: Pipes are drawn with Allegro command.
* spikeCollide: The player dies if walks into/onto spike.

2.3.5. Projectiles

* initproj: Projectiles are initialised to a speed.
* drawproj: Projectiles are drawn with Allegro command.
* shoot: A shoot sound is played, and if enemy is hit, is killed.
* update: Allows projectile to move.
* HittingPipes: Plays a bumping sound if hits pipe.
* HittingBlocks: Plays a bumping sound if hits block.

2.3.6. Enemies

* setEnemies: Bounds of enemy’s movement is set.
* DrawEnemies: Enemies are drawn with Allegro command if alive.
* StartEnemies: Enemies are added.
* Collision: Player score increases and enemy dies if hit by projectile.
* CollideEnemy: Player dies if collides with enemy, dying sound is played.
* ISEEDEADPEOPLE: All the enemies are resurrected.
* Apocalypse: All the enemies are killed.