Luke Hubbard
Sam Austin

Simon’s Climb

Synopsis

The proposed game will be a platformer with vertical linearity where the player controls a person who is climbing a large tree to rescue their dog who was birdknapped by a predatory bird. As the person climbs the tree they will encounter various bugs and obstacles such as ants and beetles that will impede movement. If the player is hit by the bugs, they will be knocked back off the platform, losing progress. The player may collect bird eggs for a special ending.

Game Design

In this game, the player has lost their dog to a predatory bird and must climb a tree to save him. To do this, the player must use the WASD keys in addition to the spacebar to move and jump to progress through the levels. On their way up the tree, the player will encounter various enemies that will serve as obstacles for the player. If the player walks into one, they will be knocked back. In order to defeat an enemy, the player must jump on them (some enemies have specific spots to jump on), similar to how Mario plays. Once the player reaches the top of the tree, they will encounter a predatory bird that they will have to fight in order to save the dog. Upon defeat, the bird will explode into a rotisserie chicken and the dog is saved. The player will be greeted with a victory screen and very short credits. There is also an alternative ending where you befriend the bird if you collect all the eggs.

Development Design

The game engine will use a model-view-controller architecture. The model will consist of classes representing the various game objects, including level information, entity information, game progression information, and character information. The level information will be of most importance as it will contain spawn information for entities, platforms, and collectibles.

The controller will be responsible for managing play input and passing information to the model and view as necessary. The player will use the WASD and spacebar keys to maneuver the environment and defeat enemies.

The view will take information from the model and render the objects accordingly. The view will adjust to follow the player throughout the level, only rendering objects within a certain distance of the character.

There are multiple possible challenges that may arise from this development design. Ensuring good communication between the different MVC modules and classes will require thorough planning and documentation. Rendering and performing logic on only entities that are on screen in a way that doesn’t reduce performance will require a decent amount of logic. Another issue may be level design as ensuring that levels are fun and engaging while not being ridiculously difficult or convoluted will require research of other game’s level design. Additionally, the AI for the enemies could pose a challenge to get them just right and to behave properly. Some of the AI properties (such as the head on the beetle) may pose a challenge as well since it is unique in comparison to the other enemies.

Luke Hubbard
Sam Austin

Division of Labor

- Sam: Art Assets (12hrs), Environment Components(12hrs), Level Design (36hrs), Camera/View(3hrs)
- Luke: Data Organization (2hrs), Entity Interactions(48hrs), Physics Systems(15hrs)

  Potential Conflicts:
    - Other responsibilities(classes, projects, work, me time)
        - Set time aside to work on project
    - One of us dies :)
        - Oh well
    - Illness
        - Work in a way that gives us wiggle room for something like that
    - Conflict of vision for the game
        - Whoever is the lead on the part causing conflict gets the final say
    - Lose all progress
        - We are using git for version control

Timeline

Milestone 1: March 30th

- Class structure:
    - Luke: 100%
    - Model, View, Controller for the time being
    - More to come
- Player sprite, tileable Wood sprites:
    - Sam: 100%
- sketch first tutorial level

Milestone 2: April 12th

- Implement ability to store level information
    - Sam: 40%
- implement level 1 ; possibly level 2
- Implement functions(physics, collision)

Milestone 3: April 12th

- Implement level 2 and 3
- Implement ai
- Add endings

Final Game Submission: April 26th
Final Exam & Presentation: May 4th

Luke Hubbard
Sam Austin

Technical Challenges:
Due to time constraints, we have removed player health so that a death screen and respawning mechanic do not have to be implemented. Sap has also been removed since its purpose was to be an obstacle that increases the chance of the player getting hit and dying.
Health fruits are also removed since there is no longer a health mechanic.