Game Test Plan: Sky Seeker

CPSC 427 – Video Game Programming

# Team: Falene (Team 7)

Emma Park #33281130;

Elton Tiet #22039192;

Stephane Gallant #72846678;

Benson Wu #76200195;

Cecilia Liu #75597831;

### Testing Scope:

The test plan covers the following aspects:

* Rendering
* Gameplay mechanics
* Assets
* Creative elements

### Test Cases:

#### Rendering:

1. Textured Geometry:
   * Verify that sprites are loaded correctly with proper blending and asset drawing order
   * Pass Criteria:
     1. Sprites are loaded with correct blending settings
     2. The map is correctly parsed and generated from a JSON file and matches with the expected design
     3. Character and enemy sprites are drawn correctly and are visible above the map
     4. No visual glitches or missing textures
2. Basic 2D transformations:
   * Verify translation, rotation, and scaling functionality
   * Pass Criteria:
     1. Position component allows entities to be moved, rotated, and scaled
3. Key-frame/State interpolation
   * Verify smooth interpolation between two values of an entity position
   * Pass Criteria:
     1. Smooth sword swing interpolation (sword moving in a circular arc)
4. Simple rendering effects (basic):
   * Ensure that visual effects display correctly
   * Pass Criteria:
     1. Red flashing effects is rendered correctly on attacked entity (when player collides with the enemy and when enemy is attacked by the player)
     2. Red flashing effect does not otherwise appear and does not remain
5. Text Rendering
   * Verify that text is rendered correctly
   * Pass Criteria:
     1. Start screen has the name of the game and the “Start”, “Tutorial”, “Credits”, and “Settings” text.
     2. Tutorial has all relevant text rendered
6. Particle Rendering
   * Verify that particles render correctly
   * Pass Criteria:
     1. Killing enemies render particles that “explode” from the enemy
     2. Currently, particles should be blue
     3. Particle rendering should succeed with different textures

#### Gameplay:

1. Keyboard/mouse control:
   * Verify player control responsiveness
   * Pass Criteria:
     1. WASD movement inputs are correctly processed and responded by the character (player movements)
     2. LMB mouse input is correctly processed and responded by the character (360 degrees player attacks)
     3. RMB mouse input is correctly processed and a projectile is spawned in the direction of the cursor
2. Random/coded action:
   * Verify that enemies walk towards the player character
   * Verify that enemies dash to players in range
   * Pass Criteria:
     1. Enemies attempt to follow the player
     2. Enemies “flinch” and pause for a moment when they take damage
     3. Enemies stop near player before dashing towards player
3. Well-defined game-space boundaries:
   * Verify that the player remains within the defined boundaries
   * Pass Criteria:
     1. Player and enemies cannot move outside the intended game area
     2. Player and enemies cannot move through the walls (obstacles)
4. Simple collision detection & resolution (e.g. between square sprites):
   * Verify correct collision processing using AABB
   * Pass Criteria:
     1. Collisions between entities are detected accurately, stopping the entities from overlapping
        + Player and wall/obstacles
        + Enemy and wall/obstacles
        + Projectiles and wall/obstacles
     2. Collisions between entities that deal damage
        + Player and enemy
        + Attacks and living entities
5. Mesh-based collision detection (e.g. sword):
   * Verify correct collision processing using triangle mesh
   * Pass Criteria:
     1. Collisions between sword and enemies that deal damage
        + Sword and enemy
        + Attacks and living entities
6. Enemy pathfinding AI (basic):
   * Validate enemy movement and navigation
   * Pass Criteria:
     1. Enemies attempt to maneuver around map obstacles in order to reach the player using a BFS to search a graph of pathfinding nodes
7. Player Parry
   * Validate player parry action
   * Pass Criteria:
     1. Parry can be carried out with the “Q” or “E” key
     2. Parry stops enemy movement and attacks
8. Player Dash
   * Validate player dash action
   * Pass Criteria:
     1. Dash can be carried out with the space bar
     2. Player flashes white to indicate invulnerability
     3. Player takes no damage during dash
     4. Player dashes in direction of current movement
9. Camera Movement
   * Verify correct camera movement
   * Pass Criteria:
     1. Player should be at the center of the screen at all times
10. Player Inventory
    * Verify inventory
    * Pass Criteria:
      1. Inventory shows the correct items stored
      2. Inventory opens and closes with the “I” key
11. Serialization
    * Verify that the game has been saved
    * Pass Criteria:
      1. Game exits without crashing
      2. Upon entering the game again, all upgrades and souls from the previous session appear in upgrades.
12. Upgrade System
    * Verify permanent player upgrade functionality
    * Pass Criteria:
      1. Souls are collected during gameplay and persist between runs
      2. Upgrade system displays available upgrades and their soul cost correctly
      3. Players can click “upgrade” to spend souls to purchase permanent upgrades when they have enough souls
      4. Feedback is given when an upgrade is purchased
13. Item stat block
    * Verify item information display
    * Pass Criteria:
      1. Getting close to an item shows a stat block
      2. Stat block displays correct item stats
      3. Stat block disappears when the player moves away from the items

#### Assets:

1. Basic Integrated Assets
   * Verify correct asset loading, proper rendering and positioning
   * Pass Criteria:
     1. Custom tiles are successfully loaded and appear correctly on-screen without distortions, or missing elements
2. Audio Integration
   * Verify audio integration in game
   * Pass Criteria:
     1. Background music plays in main menu and during the game
     2. Player attacks and item pickups should have sound effects
     3. Menu navigation and selection should have sound effects

#### Creative Elements:

1. Shadows
   * Verify dynamic shadows and lighting
   * Pass Criteria:
     1. Shadows move appropriately with dynamic light sources
     2. Areas with multiple light sources appear brighter
2. External Integration (overlaps with with Player Inventory, Item stat block, and upgrade system)
   * Verify Dear ImGui integration for UI systems
   * Pass Criteria:
     1. Inventory UI opens and closes using ImGui with the “I” key
     2. Upgrade system displays correct upgrade options
     3. Item stat block appears using ImGui when player is close to the items
3. Game Balancing
   * Verify game balance through level design and mechanics
   * Pass Criteria:
     1. Player starts with increased base health compared to previous submissions
     2. Player’s ranged attacks are less powerful
     3. Health pickup drops from enemies
     4. Health pickups restore HP
     5. Global variables are used for tuning key stats