1. Code Smell: Dead Code:
Where: In the render(Graphics g) method (commit b464fd7a2c705aedf373fdf3717aa4a60186916a) on Line 153.
What: Dead code was found. Previously, we had a rendering method that drew entities based on their position. To accommodate the concept of a Game Camera, which offsets renderings based on the camera's position, a new rendering method was introduced. The previous rendering method was left obsolete and unused in the codebase as the new method subsumed it.
Solution: The redundant rendering method was removed during a comprehensive refactoring on April 6th. This removal did not require further testing, as the rendering method render(Graphics g, int xOffset, int yOffset) had already replaced it.

2. Code Smell: Data Clump:
Where: In commit b464fd7a2c705aedf373fdf3717aa4a60186916a. Refactoring took place in each concrete Entity subclass.
What: A data clump was observed in each of the entity classes. Each entity held its image representation and associated dimensions. This resulted in each entity having a cluster of 4 to 5 variables solely to accommodate its image, leading to redundancy, decreased cohesion, and decreased maintainability.
Solution: Restructured by encapsulating image representation and dimensions into the GameImage class, enhancing cohesion and reducing redundancy. Each Entity now holds a single GameImage object.

3. Code Smell: Duplicated Code/Long Parameter List
Where: In AnimatedGameEntity (commit b464fd7a2c705aedf373fdf3717aa4a60186916a).
What: AnimatedGameEntity pushed responsibility of animating an Entity's image to its subclasses due to minute differences in how they animated them. This resulted in the constructors for its subclasses ballooning with parameters specific to setting up animations. Likewise, the AnimatedGameEntity class contained redundant variables and methods, with subclasses duplicating similar animation logic.
Solution: Animation was moved from AnimatedGameEntity to its own class, enhancing cohesion by providing a unified interface for managing animations. Now, each AnimatedGameEntity instance includes an Animation object, simplifying animation handling. Additionally, entities are created alongside a GameImage to simplify the animation process.

4. Code Smell: Lack of Documentation
Where: The Animation file. Present in b464fd7a2c705aedf373fdf3717aa4a60186916a, fixed in commit a5ed0012bc4ac36801cc724aa24ab5edf676ba32).
What: A lack of documentation was noted in the Animation file. This hindered clarity for developers and obfuscated the class's purpose.
Solution: Added JavaDocs and documentation, improving understanding, code readability, and team collaboration.

5. Code Smell: Data Clump/Low Cohesion
Where: GameEntity class (Lines 135 to 144, Line 33, and Line 161. Commit b464fd7a2c705aedf373fdf3717aa4a60186916a)
What: Collision boxes, utilised for handling unit collision and interaction within the game, were represented by rectangles surrounding each entity. However, the implementation suffered from scattered variables such as IMAGE_X_OFFSET, IMAGE_Y_OFFSET, and various size parameters, which were redundantly present in every entity class. Additionally, the GameEntity class contained a 2dRectangle.float representing the collision box, leading to a lack of cohesion as entities were burdened with managing their collision boxes instead of focusing on their primary functionality.
Solution: Refactored by removing 2dRectangle.float representation from the GameEntity class, and the offset/dimension variables were extracted from the subclasses. Subsequently, entities no longer bear the responsibility of managing their collision boxes, except for updating their positions if they move. All relevant data pushed to the CollisonBox class.

6. Code Smell: Dead Code
Where: Within the Goblin, Dynamite, and Enemies files, spanning specific lines corresponding to commits fdbb6601e1ecc2d7aaba71882a8cf4f69d15de13 (Goblin: lines 60 to 65), a5e2498aadd389b767dc85d4e90c7d6be072700e (Dynamite: lines 36 to 45), and d27d8a73467d3c9f43c0fe636970c05bccaeb4cd (Enemies: lines 27 to 32).
What: Redundant "punish" methods were identified, no longer needed due to the implementation of the "interactable" interface for standardized player interactions.
Solution: Removed redundant "punish" methods from the Enemies, Goblin, and Dynamite files, streamlining the codebase and promoting clarity and maintainability.

7. Code Smell: Low Cohesion
Where: Within the Goblin file, spanning lines 237 to 297 and identified by commit b464fd7a2c705aedf373fdf3717aa4a60186916a
What: Low cohesion was identified in the Goblin class. There was excessive coupling to pathfinding logic, causing Goblin to dedicate considerable portions of its code to path generation utilising methods. This decreased the cohesion of the Goblin class, whose purpose is to represent a Goblin in the game, not generate paths to objects.
Solution: To remedy this, pathfinding logic was moved to a dedicated class, allowing Goblin to focus solely on player pursuit. This restructuring improves code organization and readability.

8. Code Smell: Dead Code
Where: Commit fdbb6601e1ecc2d7aaba71882a8cf4f69d15de13, lines 238 to 246
What: There was dead code in the form of the unused setEnragedRange method. The code was unused, and its purpose was counter to the game’s design.
Solution: The redundant method was removed to declutter the codebase and align with project design principles.