# Assignment 4: Code Review Report

Tiensa Tran (301577884)

Steven Gong (301562896)

In this code review, we have selected a portion of code in the Game and Player classes and have identified eight code smells. Our focus was improving the code readability and maintainability but also preserving the existing functionality.

To start, the Game class exhibited characteristics of a "God Class" by encompassing over 400 lines of code and managing responsibilities beyond its primary function. These included handling unrelated aspects such as rendering, sound management, and time tracking. To resolve this issue, a dedicated Time class was introduced, separating the time-related functionality and improving modularity and cohesion within the codebase. Commit: 72f8587

Another code smell we found was long methods. The paintComponents(graphics g) contained repetitive code for rendering different enemies. To reduce redundancy, we put the enemies into one arraylist, and then called the draw method. This is recorded in commit: ea86662.

We also had the issue of long methods, so we separated enemy update logic into smaller methods with detailed names. This improved readability and made the code more modular and easier to maintain. We recorded this in commit: 3b73fce.

There were many examples of poorly structured code in our game, as well. Unnecessary wrapper functions for sound effects were created in Game class. Instead, a getter for the sound object was created in Game class. Since the sound object is final, there is no problem with it being unknowingly modified. This is in commit 3cceffb. Furthermore, the player class had multiple methods related to player position, player movement, and some getter functions, but were scattered around the class. To improve clarity, we grouped related methods together. This is in commit: 2b08212

The Game class directly managed objects such as Player, Santa, and Board, leading to high coupling and making these objects difficult to test or extend independently. To address this issue, dependencies were refactored to ensure better encapsulation and reduced interdependencies, though the exact solution should be detailed further. Commits are everywhere.

To handle low cohesion, the key input was moved from the Player class to its own class, allowing the Player class to focus more on player related behaviour, as shown in commit 792dae1.

Besides these, previously, multiple classes frequently accessed the Game class's getTileSize() method, illustrating the code smell, Feature Envy. To reduce unnecessary dependencies and enhance encapsulation, the tileSize value, which is constant, is now passed directly into constructors as needed. This refinement is detailed in commit c798143.

The screenWidth and screenHeight variables, which represent the screen configuration, were previously passed and stored as individual values, creating data clumps. To improve code organization and address the issue of data clumps, these variables were encapsulated into a single screen dimension object. Commit: c87234f

Overall, this code review identified numerous code smells and changes that can be made. By refactoring large, complex methods, reducing unnecessary dependencies, and reorganizing scattered functionality, we enhanced the cohesion and structure of the codebase. With the improvements made, we have created a more testable and extendable project.