Assignment 4 Report

Alicia Lam, Shiqi Zhong

1. Move the help page to a single class (commit c982e22)
   Code smell: Code duplication and low cohesion.

   The GameScreen and MainMenuScreen both contained separate help screen implementations that both showed the same windows. This would mean that any changes to the layout of the help screen would need to be changed in both implementations. There was also an indication of low cohesion because the GameScreen and MainMenuScreen did more than just show their respective screens.

   To remove this, we extracted the help screens into its own HelpScreen class. Whenever the user pressed the help button, a new help screen would be drawn on the window, instead of drawing it with a help screen function.

2. Move button initialization out of screen constructor (commit 86a33ea)
   Code smell: Method too long.

   The constructor for each screen method was very long and unreadable due to the number of lines instantiating buttons, setting stages, and adding event handlers.

   To make it easier to identify that buttons were being created, a new private method createButtons was made in each screen to group the creation of buttons together.

3. Move the show page out from render in the help page (commit: 86a33ea)
   Code smell: Method too long.

   Originally in the help screen render function, each page of the help screen was drawn based on the condition in the if statements. The body of each if statement would contain the entirety of the help page to be drawn. This made the render function extremely long.

   To make the *render* method more readable, we changed the *render* method to call methods *showPage1, showPage2,* and *showPage3*.

4. Breakdown moveEnemies lines in gamelogic (commit: 5e6437c)
   Code smell: Unreadable code.

   The original lines in the moveEnemies code was difficult to understand, as it was doing multiple functions in one long line.

   We refactored the moveEnemies method in gamelogic to use multiple variables to perform the one line. This would make it more clear and easier for other people looking at the code to know what it is doing.

5. Extracted in-game pause button from the render method(commit 8aff3bc)
   Code smell: Low cohesion.

   The GameScreen render is in charge of the main gameplay loop, including rendering objects. At the end of the function, there were lines of code that rendered the in-game pause button.

   To make it more clear that these lines were rendering the in-game pause button, we created a method *renderPauseButton* in PauseScreen class. This would improve the overall cohesion of the GameScreen class, as we are keeping the objects that are related to the pause screen separate.

6. Move Textures to maze game class (commit 6ea8d4b)
   Code smell: Large class and duplicate code.

   The textures in our game were originally being reinstated and used in each window that needed them. The GameScreen and HelpMenuScreen contained shared textures as well.

   To improve the efficiency of our application as well as shrink the class, we decided to extract these attributes into the MazeGame class. This would allow the textures to be instantiated only once. If a texture was changed, we would also only need to change the path in the MazeGame class, rather than all the other screens.

7. Extract camera into MazeGame class (commit 8f1d979)
   Code smell: Shotgun Surgery.

   A camera object determines what is being shown on the window from the entire map. Because the camera follows the character in GameScreen, we would need to reset it before a new window is opened, otherwise the objects shown would be offset. Additionally, two new cameras were instantiated whenever a new GameScreen was created; one for the full screen view, and one that would move to the character.

   Because the camera object affects all windows in the application despite being instantiated only in GameScreen, we decided to extract it into the MazeGame class, which contains all objects that are shared between screens. This way, we could have only one camera that would control the view on every screen.

8. Add *resetCamera* and *setCamera* methods in MazeGame. (commit 8f1d979)
   Code smell: Code duplication

   In GameScreen, there are multiple lines that intend to reset and set the camera to a new view.

   To refactor this, we made two methods in the MazeGame class, which now contains the camera object. Whenever the camera is changed or reset, these methods can be called, rather than rewriting the lines.