Code Refactoring Report
Project Name: City Escape Game (Group 9)
Author: Vansh, Sanika

A. Scope of Refactoring

- **Affected modules/classes in Game Logic:**
  - Map class
  - CityEscape class
  - Character class
  - Thief class
  - Cop class
- **Affected modules/classes in Game UI:**
  - Tile class
  - Terrain handler class

B. Original Issues

1. ***Code Smell:*** The moveThief method in the Map class was too large, with repetitive logic for handling different directions.

2. Violation of Single Responsibility principle: The main method in the CityEscape class was too large and disobeyed the single responsibility principle as it was creating everything from scratch for all the functionalities of the game.

3. Lack of Encapsulation: The city Escape class was dealing on the lower level to activate and deactivate Nitros(special reward) by using getters of Map class to obtain Nitros and then making the required changes.

4. Limited functionality*:* The character class was not allowed to set a new position for the character and had no method to check if a given point overlaps the character object.

5. Repetitive Methods*:* Repetitive code in the thief and cop class as a superclass had abstract methods of moveLeft, moveRight, moveUp, and moveDown methods.

6. ***Large Method:*** The initializeMap method in the terrain handler class was trying to do too much. While initializing the game items, it was also placing the rewards and hurdles onto the map.

7. Long list of Parameters: The renderTiles and renderCharacter methods in Terrain Handler used a long list of screen settings and offset value parameters to render objects at the correct locations. Thus making it hard to keep track of parameters when calling the functions.

8. Code Duplication: The Tile class that consists of a tile constructor is also used to assign collision property to different kinds of tiles. However, this is already being done by other classes.

9. Poorly Structured Code*:* The terrain handler class had the initializeMap method, which was initializing the map tiles as well as the array of game items, making it difficult to separate both and having multiple loops to iterate through the map placing objects.

C. Refactored Changes

Modularization

1.) The moveThief method was rewritten to invoke smaller methods to accomplish the same task of moving the thief after validating its move. New methods such as simulatePosition, overLappingWithCops, isBound, and onRoad were introduced to simulate and validate the thief's new position before updating the thief's actual position.

2.) The main method was refactored by breaking down its code into multiple smaller methods, each responsible for a single, well-defined task. Instead of performing all tasks directly within the main method, these newly created methods are now invoked within the main method.

6.) The initializeMap method places both the hurdles and rewards onto the map, making it difficult to comprehend using several variables, conditions, and loops. Thus, we break it into two methods, mapHurdlesToTiles, and mapRewardsToTiles, both having a loop each and only their concerned variables.

9.) The initializeMap method, instead of initializing the game items array too, is refactored into calling on just three methods, making it very simple to understand and code. The initialize GameItems method is coded to separate the game item array declaration.

Delegated Nitro Activation Responsibility

3.)The responsibility for nitro activation was moved from the City Escape class to the Map class. Instead of the main class retrieving the necessary data from the Map class via a getter method and then handling nitro activation, the Map class now directly manages this responsibility for the City Escape class.

Enhanced Functionality

4.) Added setLocation and overlap methods in the character class to simplify the moveThief method of the map class.

Removed Duplication

5.) Removed direction-specific methods from the thief and cop class and implemented them in the character class to avoid re-implementing the same code twice in the thief and cop class.

8.) Removed the collision property for each tile from the constructor in the tile class, as it was already being assigned in other classes and there was no need to do the same twice.

Reduced number of method Parameters

7.) Initially, we were passing 4-5 different integer values of offsets and screen measurements in the rendered character and render tile method just to ensure the correct positioning of the map tiles. However, we refactored it into an array of integers named offset that consisted of X and Y offsets and screen measurements at different indexes, which can be used as needed in the methods.