1. Program Design:

The entire program consists of two packages: Tanks package and Tools package.

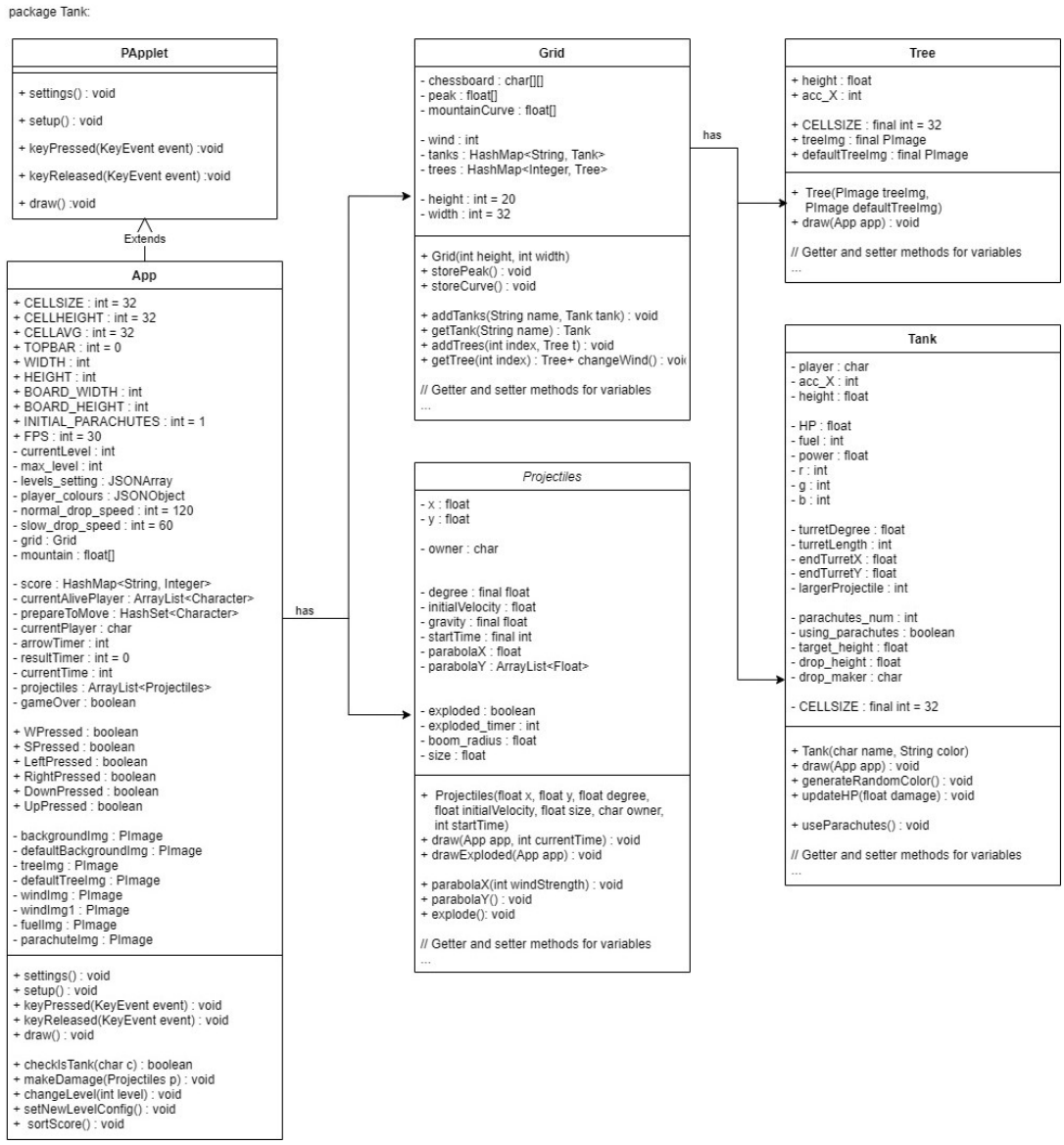
The Tanks package constitutes the core of the game.

The Tools package solely includes the MathTools class, which contains several static methods for mathematical calculations. These methods include Simple Moving Average, linear equations, and distance calculations.

Since these mathematical formulas are static and unlikely to change, I extracted them separately. This approach ensures that modifications to calculation methods do not require changes to the core game.

Within the Tanks package:

The current UML diagram of the program is as follows:



Throughout the game, I've structured all objects in the game into a single class to adhere to object-oriented principles.

a. APP Class: This class serves as the entry point of the game, responsible for initializing the game, drawing the UI, and recording some basic information in the game such as score, player survival count, and level status.

The APP class mainly consists of two primary classes, Grid class and Projectiles class, responsible for recording map information and ammunition information within

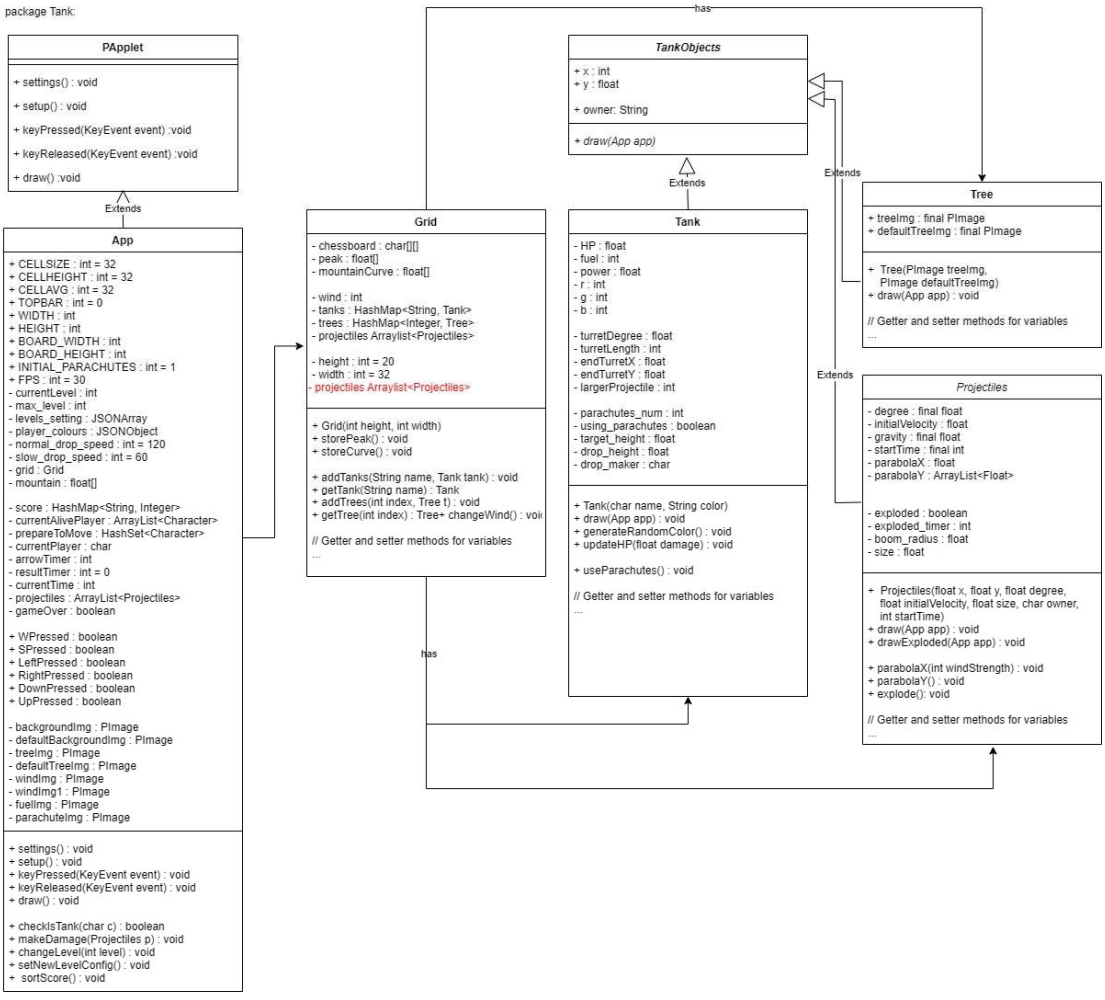
the scene, respectively.

b. Grid Class contains the actual objects in the map, the curves of mountains, wind direction, Tank class, and Tree class.

c. Tank class and Tree class are responsible for recording specific information about tanks and trees, respectively.

d. Projectiles class is responsible for recording information related to current ammunition.

Regarding improvements: Based on the current UML, I plan to restructure my program as follows:



a. Create an abstract class TankObjects: This new class contains the coordinates (x, y) of the object, and the owner of the object. It also has an interface method, draw(APP app). All objects in the game, including Tank, Tree, and Projectiles, should inherit from this class. This makes it easier to expand the program in the future, such

as adding new objects (e.g., monsters) to the game interface.

b. The map should possess the Projectiles class, making the distinction between APP and Grid classes clearer.

2. Extensions:
I'm enrolled in the COMP9003 course. Therefore, in this aspect, my extensions implement the functionality of INFO1113; namely, Repair Kit and Additional Fuel. The implementation of these two functionalities is straightforward. Upon detecting user input, if the user inputs the keys for these two functionalities, they are executed.

For the Repair Kit, if the user's current health points (HP) are greater than or equal to 100, no points are deducted, and no increase in health points occurs. If the HP is greater than 81 and less than 100, the user's HP is restored to 100.

3. Javadoc:
I have already created Javadoc comments in each class.
However, I didn't see any requirement to submit a Javadoc folder in the assignment details. Therefore, I'm not sure whether I should generate HTML files for the Javadoc. If required, I have already uploaded them to GitHub: link.
https://github.sydney.edu.au/ysun9174/tanks_scaffold-Javadoc