**Engine Code Recommendations:**

Working with Assignment 2 and 3, we came to realise that the engine code is hard to inherit because of the engine code.

**Class with issue: Actions**

Firstly, we the execute() method of the Action class takes only a few parameters like Actor and GameMap this means there is only a few things that can be done in the Action class reducing the how much an Action can do. Ideally, if a few more paraments like Location and another Actor parameter would help us do a lot with the game like making transportation from one map to another and like attacking across a large distance. We dealt with this by extending action class and adding private attributes to pass on data to the specific class and performing the necessary actions using that.

**Class with issue: Actor**

The problem pertaining to the Actor class begins from the use of the Actions. This instance of Actions holds what the player is capable of doing at any given point in the game. This is very problematic as it is obtained randomly in the playturn of that particular player. So everytime we extend this actor class, it's very troublesome to override the playTurn method for the Actor, as the Actions parameter holds several actions that the player can execute, this makes it very hard for us to use this class for any kind of players as some players may require all actions and another type of player will only need a few. Having many actions will cause create other dependencies to concrete implementations of the classes rather than abstractions.

For example; When we give an item to an enemy player in the game, we would not want him to drop it until he is knocked out, but this can’t be fixed as the engine code simply gives every actor all possible types of Actions to every actor subclass that inherits from it. To fix this we had to manually remove every instance of DropItemAction for every instance where an enemy actor was created.

So to fix this you can either reduce the randomness of getting actions or give just the default actions to each actor.

**Class with Issue: World**

When the world class runs it processes the turn of every character in the whole game. This makes the game run in a larger time and space complexity than if the game runs with only the actors on the map the player is currently on. So our suggestion is that if we only process the turns of the Actors on the map the player is on it would make the program more efficient even if there were more than two worlds.

**Class with Issue: In general**

One thing with the engine code is that the Common Closure Principle is not followed which means the Classes are everywhere within the engine package. To help with further developments of the code, if the classes that are likely to change together are in the same package, developers would find it easy to extend the functionality easily without dealing with a problem that leads to another problem.

Another problem we faced was with understanding the engine code because the JavaDoc given are not available for some classes like Range, Skills e.t.c, so the Developer is left without a clue of what’s happening in the code in the first glance. Even the classes which have JavaDoc miss some inline comments which if present would make the life of Developers easier.