**Recommendations for change to the game engine**

(HIN SENG)

Overall, I am fine with the game engine. One thing I would like to change is the menu hotkeys, the actions player can currently perform is up to 26 options despite the numbers. It is fine if the actions are less than 26, but it gets problems when the actions exceed 26 because the function cannot handle the user input. I suggest the showMenu method in Menu class should use numbers instead of characters so there would be no limit for the actions to be performed because whenever the actions increase, the number increases. User just needs to input the corresponding number to perform an action.

I also faced a problem when I was doing the Mambo Marie task, as we were not allowed to modify the classes in the engine, I was struggling how to keep track whether Marie is alive. I decided to extend the gameMap class and override the tick method to keep track of Marie’s status. It was supposed to be easy if we could just add code in the World Class. I would suggest changing World class to be an abstract class so we can create a class inherited from this class and modify the functions to be used. This is also a good design as we might implement more different rules in the future. Designers might want to apply more different rules or want to build different worlds, so changing World class to abstract class would make the intensions clear.

(AHSAN ZAFAR)

1. One positive that I have taken out of the engine class is with reference to the open/close design principle. From the start of Assignment 1, we are clearly given instruction to not make any changes to the engine class. The engine package system is designed in such a way that it is closed for modification, but open for extension to your system. This is extremely beneficial; by closing modification of the engine package, users that are already using the engine package will not be affected. And to create new functionality, we can extend this base classes through inheritance. So many examples of this are present in our scenario – just to name a few; creating new items such as zombie limbs, gun ammunition, shotgun etc.
2. Another positive that I have taken note off is the ‘Don’t Repeat Yourself (DRY) Principle’. The way the engine package has been set up; the multiple uses of abstract classes allows coders like us (students) to make full use of this principle. So many of our classes, both within the game and engine package are connected through abstract classes. This essentially allows classes to share the same code/functionality across classes and to not have repeated code lying anywhere, which can cause a lot of confusion for programmers.
3. A change that I propose to the engine class is to do with the capabilities class. Initially, it was extremely confusing to me as a student who is learning Java for the first time. Additionally, the documentation and use of this class was not that extensive, so it was really hard for me to understand the use of this class within the whole system. I suggest extra documentation to allow students to understand the use of this class better.
4. Another suggestion I may add ( which I don’t think is possible ) is to have a 10 minute video posted to brielfy explain how the interactions between the classes are working.