FIT1048 Assignment 3
Briano Goestiawan, 31482228

Object Oriented Project Implementation Reflection

This document discuss the motivation of the project design, reflection on the implementation and things to consider in the future when creating an object oriented project design.

Text-based Graphical user interface

I know from the begining that I wanted to create something graphical. Because I think that graphical UI makes for a more natural and engaging user experience. The implementation was fine for a project of this size and it achieved what I was aiming for. But it's not very flexible, if I wanted to change how the game looks and keep the functionality, there are many places that the code must be changed. in the future, I think it would be better to offload drawing functionality to a separate class that handles all the drawing functionality given a list of 'game objects'. This way, the data and view is not tightly linked together, modifying the graphics is easier and the code is overall more flexible and reusable.

Class dependency - moving items

In the project plan Assignment 2, classes are too closely related to each other. For example, the Room class which represents the room stores a collection of items. Adding and removing an item from the room requires specific methods from the room and player class. A new copy of the item have to be made in the new locationg and deleting the old item. This design was messy and the code is even worse. Almost all classes import almost all other classes. In the new design and implementation, all game objects are created and stored in the game class. The item class just stores a pointer to the current `location` which can easily be changed. This is a much simpler solution and is a design choice that should be considered first in the future. The process of changing from the first design to second design took a lot of effort, we'll disscuss this in the next section.

Implement it last

As said previously, it is not trivial to change an implementation once it had been written especially if the design was poor. The transitioning from one design to another is not trivial, a lot of code ended up being re-written completely which is a waste of time and effort. Spend more time on the design and dont stop once a solution is found, think if there is a better way of approaching the class design. Try to reduce the dependency between the classes and think if a class is even helpful for solving the problem. I find writing all the header files before implementing them is a good way of finding things that could be improved.