Critical Analysis of code

Dilip Venkatesh 2020A7PS1203P
Sarang Sridhar 2020A7PS0297P

1. Encapsulate what varies

We inherit Manager class from user class and inherit some of its methods. We encapsulate what varies to make sure that we can alter the functions that we don't require later.

2. Composition Over inheritance

We use composition over inheritance multiple times in our program for example we use instances of date, property in the other classes. This gives our design higher flexibility.

3. Code to an interface Not an implementation

In our program, we have only programmed to an implementation. Instead, if we program to an interface it hides the things you do not need to know making it simpler to use the object. It provides the blueprint of how the object will function. The classes that twe have used like user and manager could have been programmed to an interface.

4. Strive for loose coupling between objects that interact

We use loosely coupled designs between objects that interact with each other. They help us build flexible programs that reduce dependencies.

Our program doesn't use loosely coupling but can if we make each class more independent.