



SE 450: OO Software Development

Assignment 3: Design Principles and Design Patterns

② Problems

Instructor: Vahid Alizadeh

Email: v.alizadeh@depaul.edu

Quarter: Fall 2022



Last update: October 25, 2022

Assignment 3: Design Principles and Design Patterns

Submission


- **Due date:** "Please check the slides and announcements"
- **How to submit:**
 - Prepare your report in a PDF file for all exercises which includes your answers, explanations, any diagrams, or code screenshots where the question asks for.
 - Please provide your name and exercise info as a comment in the header of each source code.
 - Place source codes inside sub-folders related to each question (ex. Q1, Q2,...).
 - Compress the PDF document and all sub-folders into a single file.
 - As a rule for all assignments and exams of this course, NO screenshot, hand-written, hand-drawn, or diagram created in a photo editor are accepted and result in losing the credit. You must create your diagrams with a UML design tool.
 - This assignment has 4 exercises and will be graded out of 100. The value of each exercise is written in front of it.
 - Submit on D2L (Submissions > Assignment 3).
- **If you have any problem:**
 - Review the lecture OR ask for help on the course forum.

Goals

- Understanding and practicing SOLID Design Principles.
- Problem Solving and Architecture Design using Software Design Patterns.




Assignments

 **Exercise 1. (10 pts)** Consider the following code. What is the issue with this design? If there is any, which SOLID design principle is violated? You must **explain** your argument briefly.

```
public abstract class Feline {
    private int age;
    private double weight;
    private double top_speed;
    public void run(double newSpeed) { ... }
    private void eat(double portionSize) { ... }
    // Getters and Setters...
}

public class Tiger extends Feline {
    private String breed;
    public Tiger(int age, double wt, double sp, String br) {
        this.breed = br;
        setAge(age);
        setWeight(wt);
        setTopSpeed(sp);
    }
    private void run(double newSpeed) { ... }
    public void camouflage() { ... }
    // Getters and Setters...
}
```



 **Exercise 2. (20 pts)** Consider the following code. We have a Manager class which represent a person who manages the workers. Also, we have two types of workers, regular and super efficient workers. Both types of workers works and eats. The company brings some robots which work but don't eat. In order to add this new feature, a developer implements the WorkerInterface interface for robots and only uses working behavior.

What is the issue with this design? If there is any, which SOLID design principle is violated? You must **explain** your argument briefly. Also, **write the code** for your solution (abstract-don't write the method bodies-similar to the problem codes).

Please put the codes in the document for this exercise, DO NOT submit as Java file.

```
interface WorkerInterface {
    public void work();
    public void eat();
}

class Worker implements WorkerInterface{
    public void work() { /* Work */}
    public void eat() { /* Eat */}
}

class SuperWorker implements WorkerInterface{
    public void work() { /* More work */}
    public void eat() { /* Eat */}
}

class Manager {
    WorkerInterface worker;
    public void setWorker(WorkerInterface w) {
        worker=w;
    }
    public void manage() {
        worker.work();
    }
}
```

 **Exercise 3. (35 pts)** Consider the following class:

```
public class Assignment3 {  
    public Assignment3() {  
        System.out.print("Instance created at:"  
            + System.currentTimeMillis());  
    }  
}
```

We would like to implement the Assignment3 class as a modified version of **Singleton** design pattern. Let us call the new class SingletonAssignment3. However, instead of allowing one single instance to be created, we would like to enable a maximum of three instances to be created.

Here is a description of the client code, called SingletonTest, that would create instances of SingletonAssignment3 class:

The user is asked whether he/she would like to create an instance of the SingletonAssignment3.


1. If the answer is "Yes": create a new instance only if the number of instances that have been created so far is less than three. Otherwise, display the message "You cannot create more than three instances" and the instance created last is returned.
2. If the answer is "No", display the message "No instance is requested."

You need to provide:

1. The UML class diagram for the SingletonAssignment3 class.
2. Implement the SingletonTest client and SingletonAssignment3 class based on your UML class diagram. Implement a scenario in your SingletonTest in which the user's inputs are "No, Yes, Yes, Yes, Yes" (You don't need get an input from a user. You should hard-code the scenario in your demo which executes the requested scenario.).

Put your source codes in Q3 folder and your UML diagram in the PDF document.



 **Exercise 4. (35 pts)** Suppose you have the task of building a user interface framework that works on top of MS-Windows, Mac OS and Linux. It must work on each platform with the platform's native look and feel. You organize it by creating an abstract class for each type of widget. We consider the following three types: text field, push button, and list box. You need to write a concrete subclass of each of those types for each supported platform. To make this robust, you need to ensure that the widget objects created are all for the desired platform.

We would like to use the **Abstract Factory** design pattern to implement our system. Give the UML class diagram and implement the system. Note that you don't need to implement the GUI components. You can simply print the name of component and OS on the terminal.

Put your source codes in Q4 folder and your UML diagram in the PDF document.

-
- Don't forget to ask for help in the course forum in case you have any problem.
 - Enjoy OO programming!