National University of San Agustin
System Engineer School

TO - Lab 07 - Solid Principles: Liskov Substitution and Interface Segregation

Christian E. Portugal-Zambrano

November 06, 2018

1 Solid Principles

S.O.L.I.D is an acronym for the first five object-oriented design(OOD) **principles** by Robert C. Martin, popularly known as Uncle Bob.
These principles, when combined together, make it easy for a programmer to develop software that are easy to maintain and extend. They also make it easy for developers to avoid code smells, easily refactor code, and are also a part of the agile or adaptive software development. This principles are:

- Single Responsibility.
- Open-Closed.
- Liskov Substitution.
- Interface Segregation.
- Dependency Inversion.

2 Liskov Substitution

It extends the Open/Closed Principle by focusing on the behavior of a superclass and its subtypes. As I will show you in this article, this is at least as important but harder to validate than the structural requirements of the Open/Closed Principle.

Barbara Liskov posed her principle represents a concept of subtyping, or subtype polymorphism. This type of polymorphism is the most common in Object-oriented programming and is usually referred to as simply **polymorphism** formally it states that:

LET $\varphi(x)$ BE A PROPERTY PROVABLE ABOUT OBJECTS X OF TYPE T. THEN $\varphi(y)$ SHOULD BE TRUE FOR OBJECTS Y OF TYPE S WHERE S IS A SUBTYPE OF T.

Robert Martin also says that :

FUNCTIONS THAT USE POINTERS OF REFERENCES TO BASE CLASSES MUST BE ABLE TO USE OBJECTS OF DERIVED CLASSES WITHOUT KNOWING IT.

So, what it means on code? Here we only present the main class of the code which is:

```
public class LiskovSubstitution {
   public static void main(String [] args){
      CircularList clist = new CircularList();
      clist.insert();
      SimpleList slist = new SimpleList();
      slist.insert();

      ListGraphVizPrinter lgraphPrint = new ListGraphVizPrinter();
      lgraphPrint.print(clist);
      lgraphPrint.print(slist);
      ListXMLPrinter lxmlPrint = new ListXMLPrinter();
      lxmlPrint.print(clist);
      lxmlPrint.print(slist);
   }
```

Please, refer to the code accompanying this practice for further information.

3 INTERFACE SEGREGATION

Simply it states that:

NO CLIENT SHOULD BE FORCED TO DEPEND ON METHODS IT DOES NOT USE.

or this too:

MANY CLIENT-SPECIFIC INTERFACES ARE BETTER THAN ONE GENERAL PURPOSE INTERFACE.

We can resume all the principles as: The Single Responsibility Principle is about actors and high level architecture. The Open/Closed Principle is about class design and feature extensions. The Liskov Substitution Principle is about subtyping and inheritance. The Interface Segregation Principle (ISP) is about business logic to clients communication.

4 Warm up

Until know, you had already implemented this principle through all the practice you did before, but keep in mind the existence of this principle.

5 To do

From this point all the code you do must accomplish one responsibility per interface, it must promote the reutilization through abstraction and keep low coupling and high cohesion. For this practice there is not to do assignment!

6 Deep inside

This section is just for anyone who wants to make a deep inside into the theory and like challenges[1], you will have to prepare a presentation of just 5 minutes to explain and run the code, then you will have to defend yourself another five minutes of questions. I want that all students benefit from your presentation, remember that we are here to learn. If you want to do this please email to cportugalz@unsa.edu.pe

7 Deadline

This practice does not have deadline. All question and doubts must be done to the email.

References

[1] R. C. Martin, *Clean architecture: a craftsman's guide to software structure and design.* Prentice Hall Press, 2017.

[1] This must not be included into the report