**OOP HW18 Software Engineering Principles**

**DATE: 01th february 2021**

15 CRUZ PANTOJA DARLING MICAELA

TOPIC: INTRODUCTION TO THE SOLID PRINCIPLES

OVERVIEW OF THE SOLID PRINCIPLES SOLID

Its main feature is a mnemonic acronym for five principles

S -ingle Responsibility Principle

O -pen/Closed Principle

L -iskov Substitution Principle

I -nterface Segregation Principle

D -ependency Inversion Principle

Some argue that these are the ”first five” principles

-That claim has not been justified or widely accepted

-However, whether they are the “first five” principles is not very important

- Following these principles can help ensure quality software, primarily from a developers’ perspective

DESIGN PROBLEM

This program is generate a program where a maze must be produced, which must be able to be printed using ASCII characters or drawn on an image.

SINGLE RESPONSIBILITY PRINCIPLE

The Single Responsibility Principle is the first of the five that make up SOLID.

The Single Responsibility principle tells us that an object must do only one thing. It is very common, if we do not pay attention to this, that we end up having classes that have several logical responsibilities at the same time.

OPEN/CLOSED PRINCIPLE

This principle tells us that a software entity should be open to extension but closed to modification.

-A class is open if it is still available for extension

-A class is closed if it is available for use by other class, and therefore should not be modified

INTERFACES,ABSTRACT CLASSES,PURE VIRTUAL CLASSES

- Review: Inheritance allows a specialization (a derived class) to reuse the generalization :

-Data members

-Method declarations

-Method definitions (i.e., their implementations)

OPEN/CLOSED PRINCIPLE

Ways to achieve the open/closed principle

Inheritance

Move public methods into their own abstractions, namely interfaces, abstract classes, or pure virtual classes

Aggregation

Encapsulate behaviors in sub-part objects and allow those sub-part object to change dynamically

Parameterization

Use a generic to capture a template solution and instantiate it with the specific data types and Following the Open/Closed Principle can help developers

FOLLOWING THE LISKOV SUBSTITUTION PRINCIPLE

Let Product be a base class, with one virtual method, called save, whose intent is to save an object to a file.

LISKOV SUBSTITUTION PRINCIPLE

Each class that inherits from another can be used as its parent without the need to know the differences between them.

INTERFACE SEGREGATION PRINCIPLE

Each of the three three responsibilities place in their own interfaces (or virtual classes in Java) and the product class implements all three classes

DEPENDENCY INVERSION PRINCIPLE

Core Ideas:

-Organize the system into layers: some layers, like reusable libraries or frameworks will be more abstract or policy-setting layer, others will be detail oriented

-Components from the abstract layers should not depend on components from the detail layers; instead, they should depend on abstractions that the detailed components implement