Howard University
School of Engineering &Architecture
Department of Electrical Engineering & Computer Science

Large Scale Programming
Spring 2022

Final Exam
Due: Fri., April 22, 2022 by 11:59 AM

100 pts.

Name: Janiya Deshommes

Part I – Essays (40 pts.)

In your summaries say:

- state the name of the pattern
- what kind of problem(s) you can solve with that pattern and when you use it, maybe with a short example
- how the pattern works, what the basic idea of the pattern is
- what the main advantage and what the main disadvantage is of using this pattern

Pattern Classifications:

Creational:

Factory Method Pattern

One of the most popular design patterns, Factory Method Pattern allows a class to defer instantiation to subclasses.The main goal of this design pattern is to solve a fundamental problem in instantiation. You can use a Factory Method Pattern in various instances. For example, You can use a Factory Method Pattern when the current implementation cannot comfortably accommodate new change. One advantage to using Factory Method Pattern is the way you can extend the software. You are able to add new classes without changing the application at all. One disadvantage is that there will be an increase of integrated classes. Which requires a lot more effort.

Abstract Factory

Abstract Factory is another creational design pattern that allows you to produce families of related objects without specifying their concrete classes. It has many similarities to the Factory Method design pattern. Abstract Factory pattern is used when a family of related product objects is designed to be used together, and you need to enforce this constraint. One advantage is that it makes exchanging product families easy. On the other hand, one disadvantage is that supporting new kinds of products is difficult.

Structural:

Adapter Pattern

Adapter pattern is a family of related product objects is designed to be used together, and you need to enforce this constraint.An adapter wraps one of the objects to hide the complexity of conversion happening behind the scenes.You can introduce new types of adapters into the program without breaking the existing client code, as long as they work with the adapters through the client interface.The overall complexity of the code increases because you need to introduce a set of new interfaces and classes. Sometimes it's simpler just to change the service class so that it matches the rest of your code.

Bridge Pattern

Bridge pattern is a structural design pattern that lets you split a large class or a set of closely related classes into two separate hierarchies—abstraction and implementation—which can be developed independently of each other. You can use the Bridge if you need to be able to switch implementations at runtime. One advantage is you can create platform-independent classes and apps. However, one disadvantage similar to the other design patterns is you could potentially make the code more complicated by applying the pattern to a highly cohesive class.

Behavioral:

Observer Pattern

The observer pattern is a behavioral design pattern in which an object, named the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state

changes, usually by calling one of their methods. Observer patterns' main goal is to define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically. For example, a news agency can notify channels when it receives news.

Mediator Pattern

Mediator is a behavioral design pattern that lets you reduce chaotic dependencies between objects. The pattern restricts direct communications between the objects and forces them to collaborate only via a mediator object. Mediator pattern and Observer pattern are similar patterns. The main difference between them is that the Observer pattern distributes communication by introducing "observer" and "subject" objects, while a Mediator design pattern encapsulates the communication between other objects.

1) How is design pattern reuse different from code reuse? Please do not write a dissertation, a few sentences should suffice.

The concept of code reuse is described as using existing code for a new function of software. While design pattern reuse is similar just on a larger scale. Design pattern reuse involves more than one class and the interconnection among objects from different classes.