CS 246 Spring 2019 - Tutorial 10

Josh Rampersad

July 15, 2019

This tutorial focuses on three design patterns. It is expected that readers of this tutorial thoroughly go through the code examples mentioned below.

1 Summary

- Iterator Pattern
- Factory Method Pattern
- Template Method Pattern

2 Iterator Pattern

- **Problem:** We want to iterate through a collection[1] of objects in a way that doesn't break encapsulation. We also want to have an unified interface for such mechanism. The traversing order could be specified or unspecified.
- **Solution:** Create an abstract iterator class that defines the interface of iterators. All concrete iterator class must inherit from this class.
- **Example:** Abstract Iterator (`tut10/iterator`)

3 Factory Method Pattern

- **Problem:** At run-time, we want to create instances of a subclass based on some criteria.
  - Some examples of these criteria are input from the user or a specified probability distribution.
  - We also want to be able to easily change these criteria.
- **Solution:** Create a class which has a method that creates instances of the subclass.
  - The class can be abstract in order to be able to switch the criteria.
- **Example:** Pizza factory (`tut10/factory_method`)

[1]For example: lists, sets, binary trees, and maps

4 Template Method Pattern

- **Problem:** We want to allow subclasses to have different implementations for some sections of a method, but enforcing a structure of the method and not allowing subclasses to have different implementations for other sections.
- **Solution:** Implement an abstract superclass which has public non-virtual method(s) and private virtual method(s).
  - Have the superclass non-virtual method(s) perform the operations which will be the same for all subclasses.
  - The subclasses can implement the virtual portion(s) in a way that suits their needs.
- **Example:** Faces (`tut10/template_method/face`) and turtles (`tut10/template_method/turtle`).