**Python Design Patterns**

**Section 1: Design Pattern Warm Up**

**1.1 The Course Overview**

This video provides an overview of the entire course.

**1.2 What Are Design Patterns?**

Understand what design patterns are and why they are useful.

* + Explore the concept of a design pattern
  + Look at where design patterns originate from
  + Understand the uses and pitfalls of design patterns

**1.3 Design Pattern Classification**

Introduce the three main classifications of design patterns, and categories of each that will be covered in the course.

* + Introduce the Creational design pattern
  + Introduce the Structural design pattern
  + Introduce the Behavioral design pattern

**1.4 Advanced Python Topics**

Explore some advanced inbuilt Python implementations of some design patterns.

* + Demonstrate how a Python iterator works
  + Understand and explore examples of list comprehension
  + Understand and explore examples of Python decorators

**1.5 Inheritance in Python**

Look at how single and multiple inheritance works in Python, with an understanding of the differences between Python 2 and 3.

* + Demonstrate single inheritance with overriding and overloading
  + Understand multiple inheritance and method resolution order
  + Be aware of different inheritance syntax between Python versions

**Section 2: Producing with Factories**

**2.1 Factory**

Present the Factory pattern.

* + Understand the intention of the Factory pattern
  + Study diagram of the general implementation of the pattern
  + Walk through a concrete implementation of the Factory pattern

**2.2 Abstract Factory**

Present the Abstract Factory pattern.

* + Understand the intention of the Abstract Factory pattern
  + Understand how it’s an extension of the Factory pattern
  + Extend the previous example to demonstrate an Abstract Factory implementation

**2.3 Builder**

Present the Builder pattern.

* + Understand the intention of the Builder pattern
  + Study diagram of the general implementation of the pattern
  + Walk through a concrete implementation of the Builder pattern

**2.4 Prototype**

Present the Prototype pattern.

* + Understand the intention of the Prototype pattern
  + Demonstrate how it can be used to avoid initialization costs
  + Demonstrate how it can be used to copy and modify an instance

**2.5 Singleton Versus Borg**

Present the Singleton and Borg patterns.

* + Understand and demonstrate the Singleton and Borg patterns
  + Compare both patterns and discuss advantages and disadvantages of each
  + Show that Python modules are Singletons

**Section 3: Structuring Around**

**3.1 Model View Controller**

Present the Model View Controller (MVC) pattern.

* + Understand the intention of the MVC pattern
  + Introduce Python web frameworks, Flask and Django
  + Consider an example of MVC in Flask

**3.2 Façade**

Present the Façade pattern.

* + Understand the intention of the Façade pattern
  + Understand the benefits of the Façade pattern
  + Apply the Façade pattern to the mechanics of starting a car

**3.3 Proxy**

Present the Proxy pattern.

* + Understand the intention of the Proxy pattern
  + Example of Proxy pattern using common interface with real subject
  + Example of generic proxy class which can be subclassed

**3.4 Decorator**

Present the Decorator pattern.

* + Understand the intention of the Decorator pattern
  + Show how it can be better than subclassing
  + Example of using decorators to compose different UI windows

**3.5 Adapter**

Present the Adapter pattern.

* + Understand the intention of the Adapter patter
  + Explain when one might want to use the Adapter pattern
  + An example of adapting a European socket to an American one

**Section 4: Behaving Ourselves**

**4.1 Command**

Present the Command Pattern.

* + Understand the intention of the Command pattern
  + Demonstrate the general Command pattern implementation
  + Look at an example of cut and paste from a screen of text

**4.2 Interpreter**

Present the Interpreter pattern.

* + Understand the intention of the Interpreter pattern
  + Demonstrate the general Interpreter pattern implementation
  + Implement a simple rule validator using the Interpreter pattern

**4.3 State**

Present the State pattern.

* + Understand the intention of the State pattern
  + Demonstrate the general State pattern implementation
  + Apply the pattern to changing states of a computer

**4.4 Chain of Responsibility**

Present the Chain of Responsibility pattern.

* + Understand the intention of the Chain of Responsibility pattern
  + Explain when it should and shouldn’t be used
  + Example of a garage handling cars with different servicing requirements

**Section 5: Behaving Ourselves Again**

**5.1 Observer**

Present the Observer pattern.

* + Understand the intention of the Observer pattern
  + Explain its relation with MVC
  + Show an example of stock markets observing a company

**5.2 Strategy**

Present the Strategy pattern.

* + Understand the intention of the Strategy pattern
  + Program to an interface, not an implementation
  + Use the strategy pattern to find prime numbers

**5.3 Memento**

Present the Memento pattern.

* + Understand the intention of the Memento pattern
  + Create an Undoable class using a Memento class
  + Compare with the Command pattern to implement undo

**5.4 Template**

Present the Template pattern.

* + Understand the intention of the Template pattern
  + Understand how it’s used in frameworks
  + Demonstrate an example of using a template to make different meals

**5.5 Reactive Programming**

Present the concept of reactive programming and RxPY.

* + Understand the intention of reactive programming
  + Introduce RxPY - a set of Python libraries for reactive programming
  + Show how to install and start using RxPY

**Section 6: No Pattern (a.k.a AntiPattern)**

**6.1 Spaghetti Code**

Present the spaghetti code AntiPattern.

* + Understand why spaghetti code is an AntiPattern
  + Understand common symptoms, consequences, and causes
  + Demonstrate a spaghetti code example, explain how to improve it

**6.2 Blob**

Present the blob AntiPattern.

* + Understand why blob is an AntiPattern
  + Understand symptoms and consequences of blob
  + Demonstrate how to refactor away from a blob class

**6.3 Functional Decomposition**

Present the functional decomposition AntiPattern.

* + Understand why functional decomposition is an OO AntiPattern
  + Understand the symptoms and consequences of functional decomposition
  + Compare a functional decomposition versus an OO solution for a problem

**6.4 Copy and Paste**

Present the Copy and Paste AntiPattern.

* + Understand why Copy and Paste is an AntiPattern
  + Understand symptoms and consequences of Copy and Paste
  + Explore potential causes of Copy and Paste programming