## Introduction

I have done my design pattern look at the Cmd module included with python. In my analysis of the module itself, I found that the Cmd module acts as a template method. It has hooks which it runs as part of an algorithm which can be hooked on by the subclass. The subclass in this instance is the laid out application that uses the module.

## Where the design pattern is used

Hook methods can be found in the module. They are:

precmd

postcmd

preloop

Postloop

The method for handling the algorithm is cmdloop()

The invariant aspects of the library are found in these functions:

Parseline

Onecmd

Emptyline

Default

Completedefault

Completenames

Complete

Getnames

Complete\_help

Do\_help

Print\_topics

Columnize

## How does the pattern work

When a class is instanced, it is required that the method inherit from the super method (cmd.Cmd) and that the instance of the command is called.

There are attributes which are inherited from the cmd.Cmd class, they are prompt and intro. The prompt is what is displayed before the command line

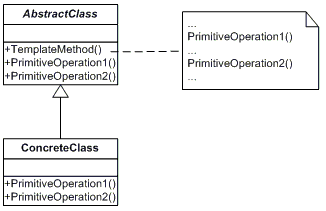
The intro is displayed when the program loads.

The class operates on the methods that are given to it. They must be factored by the name do\_ in this way, all the do\_ commands are hook methods within the subclass.

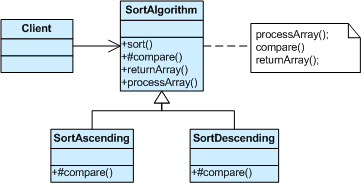
If it doesn’t have do\_ preceding it, it won’t find it. It is mentioned in two methods in the superclass,

Onecmd and do\_help

The template method pattern looks like this:



An example of the Template Method is used in Algorithms:



I’m drawing a conclusion from the way it should be structured in the cmd module. In this example compare is a hook method, as it doesn’t do anything, but the classes that inherit from it have the functionality.

The superclass has the functionality for the way the module works, this means when you start the subclass, you must call the cmdloop method. This enables you to access the functionality of the cmd module.