# Лабораторная работа №4

Шаблоны проектирования и модульное тестирование в Python

## Цель:

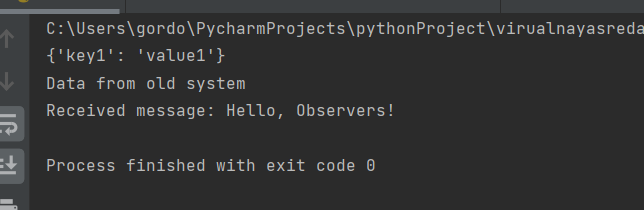
Изучение реализации шаблонов проектирования и возможностей модульного тестирования в языке Python.

## Условие:

Необходимо для произвольной предметной области реализовать от одного до трех шаблонов проектирования: один порождающий, один структурный и один поведенческий. Для сдачи лабораторной работы в минимальном варианте достаточно реализовать один паттерн.  
  
В модульных тестах необходимо применить следующие технологии:  
- TDD - фреймворк.  
- BDD - фреймворк.  
- Создание Mock-объектов.

## Код:

# Singleton Pattern  
class SingletonMeta(type):  
 \_instances = {}  
  
 def \_\_call\_\_(cls, \*args, \*\*kwargs):  
 if cls not in cls.\_instances:  
 instance = super().\_\_call\_\_(\*args, \*\*kwargs)  
 cls.\_instances[cls] = instance  
 return cls.\_instances[cls]  
  
class Database(metaclass=SingletonMeta):  
 def \_\_init\_\_(self):  
 self.data = {}  
  
 def insert(self, key, value):  
 self.data[key] = value  
  
# Adapter Pattern  
class OldSystem:  
 def request(self):  
 return "Data from old system"  
  
class NewSystemAdapter:  
 def \_\_init\_\_(self, old\_system):  
 self.old\_system = old\_system  
  
 def get\_data(self):  
 return self.old\_system.request()  
  
# Observer Pattern  
class Subject:  
 def \_\_init\_\_(self):  
 self.\_observers = []  
  
 def add\_observer(self, observer):  
 self.\_observers.append(observer)  
  
 def notify(self, message):  
 for observer in self.\_observers:  
 observer.update(message)  
  
class Observer:  
 def update(self, message):  
 print(f"Received message: {message}")  
  
# Пример использования  
if \_\_name\_\_ == "\_\_main\_\_":  
 # Singleton Example  
 db1 = Database()  
 db2 = Database()  
 db1.insert("key1", "value1")  
 assert db1 is db2 # Проверка Singleton  
 print(db1.data) # Вывод: {'key1': 'value1'}  
  
 # Adapter Example  
 old\_system = OldSystem()  
 adapter = NewSystemAdapter(old\_system)  
 print(adapter.get\_data()) # Вывод: "Data from old system"  
  
 # Observer Example  
 subject = Subject()  
 observer1 = Observer()  
 subject.add\_observer(observer1)  
 subject.notify("Hello, Observers!")



## Код тестов:

import unittest  
from unittest.mock import MagicMock  
from program import Database, OldSystem, NewSystemAdapter, Subject, Observer  
  
# TDD: тесты для шаблона Singleton  
class TestSingleton(unittest.TestCase):  
 def test\_singleton\_instance(self):  
 db1 = Database()  
 db2 = Database()  
 self.assertIs(db1, db2)  
 db1.insert("test\_key", "test\_value")  
 self.assertEqual(db2.data["test\_key"], "test\_value")  
  
# TDD: тесты для адаптера  
class TestAdapter(unittest.TestCase):  
 def test\_adapter(self):  
 old\_system = OldSystem()  
 adapter = NewSystemAdapter(old\_system)  
 self.assertEqual(adapter.get\_data(), "Data from old system")  
  
# BDD: тесты для наблюдателя  
class TestObserver(unittest.TestCase):  
 def test\_observer(self):  
 subject = Subject()  
 observer = Observer()  
  
 # Mock-объект для наблюдателя  
 observer.update = MagicMock()  
 subject.add\_observer(observer)  
 subject.notify("Test message")  
  
 # Проверяем вызов метода update с правильным аргументом  
 observer.update.assert\_called\_with("Test message")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 unittest.main()

