Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего **образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Лабораторная работа № 4

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил:
студент группы ИУ5-32Б
Кульктна Д.А.

Проверил:
Канев А.И.

2021 г.

Полученное задание:

1. Необходимо для произвольной предметной области реализовать от одного до трех шаблонов проектирования: один порождающий, один структурный и один поведенческий. В качестве справочника шаблонов можно использовать следующий каталог. Для сдачи лабораторной работы в минимальном варианте достаточно реализовать один паттерн.
2. Вместо реализации паттерна Вы можете написать тесты для своей программы решения биквадратного уравнения. В этом случае, возможно, Вам потребуется доработать программу решения биквадратного уравнения, чтобы она была пригодна для модульного тестирования.
3. В модульных тестах необходимо применить следующие технологии:
    - TDD - фреймворк.
    - BDD - фреймворк.
    - Создание Mock-объектов.

Main.py

```
from builder import Director, BuilderHouse1, BuilderHouse2, BuilderHouse3
from getinto import Furniture, HouseDoor, GetInto
from Information import Wood, Stone, Call

def task1():
    list_of_houses = [BuilderHouse1, BuilderHouse2, BuilderHouse3]
    director = Director()
    for builder_house in list_of_houses:
        builder_house_now = builder_house()
        director.set_builder(builder_house_now)
        director.build_house()
        new_house = builder_house_now.get_house()
        print(new_house, '\n')
    return 1


def task2(object1):
    door = HouseDoor(250, 100)
    text = ''
    print(object1.name)
    temp = GetInto(object1)
    if door.object_inside(temp):
        text = "Занести можно"
    else:
        text = "Занести нельзя"
    return text
if __name__ == '__main__':
    print("Testing builder")
    task_done = task1()

    print('\nTesting getinto')
    list_of_objects = [
        Furniture("диван", 210, 99, 150),
        Furniture("картина", 100, 50),
        Furniture("палка", 300)
    ]
    for object1 in list_of_objects:
        print(task2(object1))

    print('\nTesting Information')
```

```
    question1 = Wood()
    question2 = Stone()
    call = Call()
    print("Вопрос по дереву:")
    call.give_info(question1)
    print("Вопрос по камню:")
    call.give_info(question2)
```

Information.py

```
from abc import ABC, abstractmethod

class Call:
    def give_info(self, question):
        print("Приветствую Вас, Вы обратились в строительную компанию")
        question.give_tip()


class Helper(ABC):
    tip = None

    @abstractmethod
    def give_tip(self):
        pass


class Wood(Helper):
    def __init__(self):
        self.tip = "Информация о дереве"

    def give_tip(self):
        if self.tip:
            print(self.tip)
        else:
            Operator().help_people()


class Stone(Helper):
    def give_tip(self):
        if self.tip:
            print(self.tip)
        else:
            Operator().help_people()


class Operator:
    def help_people(self):
        print("Оператор на связи")
```

test.py

```
mport unittest
from main import task1, task2
from getinto import Furniture
class Test1(unittest.TestCase):
    def test_result(self):
        self.assertEqual(task1(), 1)

    def test_result(self):
        temp = Furniture("диван", 210, 99, 150)
        self.assertEqual(task2(temp), 'Занести можно')
```

getinto.py

```python
class Furniture:
    def __init__(self, name, *args):
        self.name = name
        self.size = []
        for side in args:
            self.size.append(side)


class HouseDoor:
    def __init__(self, high, width):
        self.high = high
        self.width = width

    def object_inside(self, obj):
        if obj.high < self.high and obj.width < self.width:
            return True
        return False


class GetInto:
    def __init__(self, obj: Furniture):
        if len(obj.size) == 1:
            self.high = 0
            self.width = 0
        elif len(obj.size) == 2:
            self.high = min(obj.size)
            self.width = 0
        else:
            self.width = min(obj.size)
            obj.size.remove(min(obj.size))
            self.high = min(obj.size)
```

builder.py

```python
from abc import ABC, abstractmethod


class HouseSquare:
    Square1 = 100
    Square2 = 200
    Square3 = 300


class HouseRooms:
    Rooms1 = 2
    Rooms2 = 4
    Rooms3 = 6


class HouseFloors:
    Floor1 = 1
    Floor2 = 2


class HouseDoors:
    Doors1 = 1
    Doors2 = 2
    Doors3 = 3


class HouseWindows:
    Windows1 = 4
    Windows2 = 7
    Windows3 = 10
```

```
class House:
    def __init__(self, name):
        self.name = name
        self.rooms = None
        self.square = None
        self.doors = None
        self.windows = None
        self.floors = None

    def __repr__(self):
        return
"\tname:\t{}\n\trooms:\t{}\n\tsquare:\t{}\n\tdoors:\t{}\n\twindows:{}\n\tfloo
rs:\t{}".format(self.name,

self.rooms,

self.square,

self.doors,

self.windows,

self.floors)


class Builder(ABC):
    @abstractmethod
    def select_square(self):
        pass

    @abstractmethod
    def select_rooms(self):
        pass

    @abstractmethod
    def select_doors(self):
        pass

    @abstractmethod
    def select_windows(self):
        pass

    @abstractmethod
    def select_floors(self):
        pass


class BuilderHouse1(Builder):
    def __init__(self):
        self.house = House("super house 1")

    def select_square(self):
        self.house.square = HouseSquare.Square1

    def select_doors(self):
        self.house.doors = HouseDoors.Doors1

    def select_rooms(self):
        self.house.rooms = HouseRooms.Rooms1

    def select_windows(self):
        self.house.windows = HouseWindows.Windows1
```

```
    def select_floors(self):
        self.house.floors = HouseFloors.Floor1

    def get_house(self):
        return self.house


class BuilderHouse2(Builder):
    def __init__(self):
        self.house = House("super house 2")

    def select_square(self):
        self.house.square = HouseSquare.Square2

    def select_doors(self):
        self.house.doors = HouseDoors.Doors2

    def select_rooms(self):
        self.house.rooms = HouseRooms.Rooms2

    def select_windows(self):
        self.house.windows = HouseWindows.Windows2

    def select_floors(self):
        self.house.floors = HouseFloors.Floor2

    def get_house(self):
        return self.house


class BuilderHouse3(Builder):
    def __init__(self):
        self.house = House("super house 3")

    def select_square(self):
        self.house.square = HouseSquare.Square3

    def select_doors(self):
        self.house.doors = HouseDoors.Doors3

    def select_rooms(self):
        self.house.rooms = HouseRooms.Rooms3

    def select_windows(self):
        self.house.windows = HouseWindows.Windows3

    def select_floors(self):
        self.house.floors = HouseFloors.Floor2

    def get_house(self):
        return self.house


class Director:
    def __init__(self):
        self.builder = None

    def set_builder(self, builder: Builder):
        self.builder = builder

    def build_house(self):
        self.builder.select_square()
        self.builder.select_floors()
        self.builder.select_rooms()
```

```
        self.builder.select_doors()
        self.builder.select_windows()
```

Результат

```
Testing builder
    name:   super house 1
    rooms:  2
    square: 100
    doors:  1
    windows:4
    floors: 1

    name:   super house 2
    rooms:  4
    square: 200
    doors:  2
    windows:7
    floors: 2

    name:   super house 3
    rooms:  6
    square: 300
    doors:  3
    windows:10
    floors: 2


Testing getinto
диван
Занести можно
картина
Занести можно
палка
Занести можно

Testing Information
Вопрос по дереву:
Приветствую Вас, Вы обратились в строительную компанию
Информация о дереве
Вопрос по камню:
Приветствую Вас, Вы обратились в строительную компанию
Оператор на связи
```

```
Ran 1 test in 0.002s

OK
```