Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №3
«Объектно-ориентированные возможности языка Python.»

Выполнил:
студент группы ИУ5-31Б
Орлова Анна
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Подпись и дата:

Москва, 2024 г.

Задание:

Необходимо создать виртуальное окружение и установить в него хотя бы один внешний пакет с использованием pip.

Необходимо разработать программу, реализующую работу с классами. Программа должна быть разработана в виде консольного приложения на языке Python 3.

Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab_python_oop.

Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab_python_oop.

Абстрактный класс «Геометрическая фигура» содержит абстрактный метод для вычисления площади фигуры. Подробнее про абстрактные классы и методы Вы можете прочитать здесь.

Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры. Подробнее про описание свойств Вы можете прочитать здесь.

Класс «Прямоугольник» наследуется от класса «Геометрическая фигура». Класс должен содержать конструктор по параметрам «ширина», «высота» и «цвет». В конструкторе создается объект класса «Цвет фигуры» для хранения цвета. Класс должен переопределять метод, вычисляющий площадь фигуры.

Класс «Круг» создается аналогично классу «Прямоугольник», задается параметр «радиус». Для вычисления площади используется константа math.pi из модуля math.

Класс «Квадрат» наследуется от класса «Прямоугольник». Класс должен содержать конструктор по длине стороны. Для классов «Прямоугольник», «Квадрат», «Круг»:

Определите метод "repr", который возвращает в виде строки основные параметры фигуры, ее цвет и площадь. Используйте метод format - https://pyformat.info/

Название фигуры («Прямоугольник», «Квадрат», «Круг») должно задаваться в виде поля данных класса и возвращаться методом класса.

В корневом каталоге проекта создайте файл main.py для тестирования Ваших классов (используйте следующую конструкцию - https://docs.python.org/3/library/__main__.html).

Создайте следующие объекты и выведите о них информацию в консоль (N - номер Вашего варианта по списку группы):

Прямоугольник синего цвета шириной N и высотой N.

Круг зеленого цвета радиусом N.

Квадрат красного цвета со стороной N.

Также вызовите один из методов внешнего пакета, установленного с использованием pip.

Дополнительное задание. Протестируйте корректность работы Вашей программы с помощью модульного теста.

Текст программы

square.py

```
from lab_python_oop.rectangle import Rectangle


class Square(Rectangle):
    FIGURE_TYPE = "Квадрат"

    def __init__(self, col, a):
        self.side = a
        super().__init__(col, self.side, self.side)

    def __repr__(self):
        return '{} {} цвета со стороной {} площадью {}.'.format(
            self.get_figure_type(),
            self.rect_col.col,
            self.side,
            self.square_fig()
        )
```

rectangle.py

```
from lab_python_oop.figure import Figure
from lab_python_oop.color import Color


class Rectangle(Figure):

    FIGURE_TYPE = "Прямоугольник"

    def __init__(self, color, w, h):

        self.width = w
        self.height = h
        self.rect_col = Color()
        self.rect_col.col = color
    def square_fig(self):
        return self.width*self.height

    def __repr__(self):
        return '{} {} цвета шириной {} и высотой {} площадью {}.'.format(
            self.get_figure_type(),
            self.rect_col.col,
            self.width,
            self.height,
```

```
            self.square_fig()
        )
```

circle.py

```
from lab_python_oop.figure import Figure
from lab_python_oop.color import Color
import math

class Circle(Figure):
    FIGURE_TYPE = "Круг"

    def __init__(self, color, rad):
        self.radius = rad
        self.circ_col = Color()
        self.circ_col._color = color
    def square_fig(self):
        return math.pi*(self.radius**2)

    def __repr__(self):
        return '{} {} цвета радиусом {} площадью {:.4f}.'.format(
            self.get_figure_type(),
            self.circ_col.col,
            self.radius,
            self.square_fig()
        )
```

color.py

```
class Color:
    def __init__(self):
        self._color=None
    @property
    def col(self):
        return self._color

    @col.setter
    def col(self,val):
        self._color=val
```

figure.py

```
from abc import ABCMeta , abstractmethod
class Figure (metaclass=ABCMeta):
    """Абстрактный класс фигура"""
    FIGURE_TYPE = ""

    def __init__(self):
        pass
    """функция для вычисления площади фигуры"""
    @abstractmethod
    def square_fig(self):
        pass
    def get_figure_type(self):
        return self.FIGURE_TYPE
```

main.py

```
from lab_python_oop.rectangle import Rectangle
from lab_python_oop.circle import Circle
from lab_python_oop.square import Square
import requests
```

```
def main():

    rect = Rectangle("синего" ,18,18)
    circ = Circle("зеленого", 18)
    sqr = Square("красного", 18)
    print(rect)
    print(circ)
    print(sqr)
    #response = requests.get('https://sky.pro/media/')
    #print(response.ok)  # проверяем успешен ли запрос?


if __name__ == "__main__":
    main()
```

test.py

```
import math
import unittest
from lab_python_oop.rectangle import Rectangle
from lab_python_oop.circle import Circle
from lab_python_oop.square import Square

class Test_Rect(unittest.TestCase):
    '''тесты для класса прямоугольник'''
    def test_rec(self):
        R1=Rectangle("красный",1,2)
        self.assertEqual(R1.square_fig(),2)
        self.assertEqual(R1.rect_col.col, "красный")
        self.assertEqual(R1.width, 1)
        self.assertEqual(R1.height, 2)
class Test_Square(unittest.TestCase):
    def test_sqr(self):
        S1 = Square("синий", 5)
        self.assertEqual(S1.square_fig(), 25)
        self.assertEqual(S1.rect_col.col, "синий")
        self.assertEqual(S1.side, 5)

class Test_Circle(unittest.TestCase):
    def test_cir(self):
        S1 = Circle("зелёный", 25)
        self.assertEqual(S1.square_fig(), 625*math.pi)
        self.assertEqual(S1.circ_col.col, "зелёный")
        self.assertEqual(S1.radius, 25)



if __name__ == '__main__':
    unittest.main()
```

экранные формы с примерами выполнения программы.


Run
main
C:\Users\annfr\PycharmProjects\lab2\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab2\main.py
Прямоугольник синего цвета шириной 18 и высотой 18 площадью 324.
Круг зеленого цвета радиусом 18 площадью 1017.8760.
Квадрат красного цвета со стороной 18 площадью 324.
Process finished with exit code 0
main.py
11:33
CRLF
UTF-8
4 spaces
Python 3.11


Run
Python tests in test.py
Test Results 10 ms
test 10 ms
Test_Circle 10 ms
Test_Rect 0 ms
Test_Square 0 ms
Tests passed: 3 of 3 tests – 10 ms
C:\Users\annfr\PycharmProjects\lab2\.venv\Scripts\python.exe "C:/Program Files/JetBrains/PyCharm 2023.3.5/plugins/python/helpers/pycharm/_j
Testing started at 0:01 ...
Launching unittests with arguments python -m unittest C:\Users\annfr\PycharmProjects\lab2\test.py in C:\Users\annfr\PycharmProjects\lab2
Ran 3 tests in 0.011s
test.py
19:49
CRLF
UTF-8
4 spaces
Python 3.11 (lab2) (2)