Московский государственный технический университет имени Н.Э. Баумана

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Факультет РТ Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

**Отчет по лабораторной работе № 2 по курсу**

**Базовые компоненты**

"Объектно-ориентированные возможности языка Python"

9

 (количество листов)

|  |  |  |
| --- | --- | --- |
| Исполнитель |  |  |
| студент группы РТ5-31Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Билалов А.К. |
|  |  | “15” ноября 2022 г. |
|  |  |  |
| Проверил |  |  |
| Доцент кафедры ИУ5 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гапанюк Ю.Е. |
|  |  | “\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2022 г. |

Москва 2022

# Оглавление

[Оглавление 2](#_Toc113796368)

[Описание задания 3](#_Toc113796369)

[Текст программы 5](#_Toc113796370)

[Результаты выполнения программы 9](#_Toc113796371)

# Описание задания

1. Необходимо создать виртуальное окружение и установить в него хотя бы один внешний пакет с использованием pip.
2. Необходимо разработать программу, реализующую работу с классами. Программа должна быть разработана в виде консольного приложения на языке Python 3.
3. Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab\_python\_oop.
4. Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab\_python\_oop.
5. Абстрактный класс «Геометрическая фигура» содержит абстрактный метод для вычисления площади фигуры. Подробнее про абстрактные классы и методы Вы можете прочитать здесь.
6. Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры. Подробнее про описание свойств Вы можете прочитать здесь.
7. 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.
8. Класс «Круг» создается аналогично классу «Прямоугольник», задается параметр «радиус». Для вычисления площади используется константа math.pi из модуля math.
9. Класс «Квадрат» наследуется от класса «Прямоугольник». Класс должен содержать конструктор по длине стороны. Для классов «Прямоугольник», «Квадрат», «Круг»:
10. Определите метод "repr", который возвращает в виде строки основные параметры фигуры, ее цвет и площадь. Используйте метод format - https://pyformat.info/
11. Название фигуры («Прямоугольник», «Квадрат», «Круг») должно задаваться в виде поля данных класса и возвращаться методом класса.
12. 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):
13. Прямоугольник синего цвета шириной N и высотой N.
14. Круг зеленого цвета радиусом N.
15. Квадрат красного цвета со стороной N.
16. Также вызовите один из методов внешнего пакета, установленного с использованием pip.
17. Дополнительное задание. Протестируйте корректность работы Вашей программы с помощью модульного теста.

# Текст программы

figure.py:

from abc import ABC, abstractmethod

class Figure(ABC):

    @abstractmethod

    def area(self):

        pass

color.py:

class Color:

    def \_\_init\_\_(self):

        self.\_color = None

    @property

    def colorproperty(self):

        # getter

        return self.\_color

    @colorproperty.setter

    def colorproperty(self, value):

        # setter

        self.\_color = value

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, width, height, color):

        self.width = width

        self.height = height

        self.color = Color()

        self.color.colorproperty = color

    def area(self):

        return self.width \* self.height

    def get\_figure\_type(self):

        return self.FIGURE\_TYPE

    def \_\_repr\_\_(self):

        return '{} {} цвета шириной {} и высотой {} площадью {}.'.format(

            self.get\_figure\_type(),

            self.color.colorproperty,

            self.width,

            self.height,

            self.area()

        )

square.py:

from lab\_python\_oop.rectangle import Rectangle

from lab\_python\_oop.color import Color

class Square(Rectangle):

    FIGURE\_TYPE = "Квадрат"

    def \_\_init\_\_(self, side, color):

        super().\_\_init\_\_(side, side, color)

    def get\_figure\_type(self):

        return self.FIGURE\_TYPE

    def \_\_repr\_\_(self):

        return '{} {} цвета со стороной {} площадью {}.'.format(

            self.get\_figure\_type(),

            self.color.colorproperty,

            self.width,

            self.area()

        )

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, radius, color):

        self.radius = radius

        self.color = Color()

        self.color.colorproperty = color

    def get\_figure\_type(self):

        return self.FIGURE\_TYPE

    def area(self):

        return math.pi \* (self.radius \*\* 2)

    def \_\_repr\_\_(self):

        return '{} {} цвета радиусом {} площадью {}.'.format(

            self.get\_figure\_type(),

            self.color.colorproperty,

            self.radius,

            self.area()

        )

main.py:

from lab\_python\_oop.circle import Circle

from lab\_python\_oop.rectangle import Rectangle

from lab\_python\_oop.square import Square

from art import \*

def main():

    # У Вас в списке, где мы оставляем ссылки на репозитории, я 16-ый

    r = Rectangle(16, 16, 'синего')

    print(r)

    c = Circle(16, 'зеленого')

    print(c)

    s = Square(16, 'красного')

    print(s)

    art = text2art("KRAINIKOV RT5-31B", font='small')

    print(art)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

test.py:

import unittest

from lab\_python\_oop.circle import Circle

from lab\_python\_oop.rectangle import Rectangle

from lab\_python\_oop.square import Square

class TestFigures(unittest.TestCase):

    def test\_rectangle(self):

        r = Rectangle(16, 16, 'синего')

        self.assertEqual(r.area(), 256)

        self.assertEqual(r.get\_figure\_type(), 'Прямоугольник')

        self.assertEqual(r.\_\_repr\_\_(), 'Прямоугольник синего цвета шириной 16 и высотой 16 площадью 256.')

    def test\_circle(self):

        c = Circle(16, 'зеленого')

        self.assertEqual(c.area(), 804.247719318987)

        self.assertEqual(c.get\_figure\_type(), 'Круг')

        self.assertEqual(c.\_\_repr\_\_(), 'Круг зеленого цвета радиусом 16 площадью 804.247719318987.')

    def test\_square(self):

        s = Square(16, 'красного')

        self.assertEqual(s.area(), 256)

        self.assertEqual(s.get\_figure\_type(), 'Квадрат')

        self.assertEqual(s.width, s.height)

        self.assertEqual(s.\_\_repr\_\_(), 'Квадрат красного цвета со стороной 16 площадью 256.')

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main(verbosity=2)

# Результаты выполнения программы



