|  |  |
| --- | --- |
| Защищено: Гапанюк Ю.Е.  " " **2025** г. | Демонстрация:  Константинов А.А.  " " **2025** г. |

Отчет по лабораторной работе № 2 по курсу Парадигмы и конструкции языков программирования

**Тема работы: " Объектно-ориентированные возможности языка Python"**

5

(количество листов)

|  |  |
| --- | --- |
|  |  |
| студент группы ИУ5Ц-53Б |  |
|  | (подпись) |
| Константинов А.А. | " " **2025** г. |

Москва, МГТУ - **2025**

# **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»:

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.

10.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.

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

# **Листинг программы**

Main.py

from lab\_python\_oop.rectangle import Rectangle

from lab\_python\_oop.circle import Circle

from lab\_python\_oop.square import Square

# внешний пакет - prettytable (pip install prettytable уже установлен)

from prettytable import PrettyTable

def main():

    N = 5  # замените на ваш номер варианта

    rect = Rectangle(N, N, "синий")

    circle = Circle(N, "зелёный")

    square = Square(N, "красный")

    print(rect)

    print(circle)

    print(square)

    table = PrettyTable(["Фигура", "Цвет", "Площадь"])

    table.add\_row([rect.figure\_type, rect.color.color, f"{rect.area():.2f}"])

    table.add\_row([circle.figure\_type, circle.color.color, f"{circle.area():.2f}"])

    table.add\_row([square.figure\_type, square.color.color, f"{square.area():.2f}"])

    print("\nТаблица фигур:")

    print(table)

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

    main()

circle.py

import math

from lab\_python\_oop.figure import GeometricFigure

from lab\_python\_oop.color import FigureColor

class Circle(GeometricFigure):

    figure\_type = "Круг"

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

        self.radius = radius

        self.color = FigureColor(color)

    def area(self):

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

    def \_\_repr\_\_(self):

        return "{} цвета {} радиусом {}. Площадь: {:.2f}".format(

            self.figure\_type, self.color.color, self.radius, self.area()

        )

Color.py

class FigureColor:

    """Класс для описания цвета геометрической фигуры"""

    def \_\_init\_\_(self, color):

        self.\_color = color

    @property

    def color(self):

        return self.\_color

    @color.setter

    def color(self, new\_color):

        self.\_color = new\_color

scuare.py

from lab\_python\_oop.rectangle import Rectangle

class Square(Rectangle):

    figure\_type = "Квадрат"

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

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

    def \_\_repr\_\_(self):

        return "{} цвета {} со стороной {}. Площадь: {:.2f}".format(

            self.figure\_type, self.color.color, self.width, self.area()

        )

Rectangle .py

from lab\_python\_oop.figure import GeometricFigure

from lab\_python\_oop.color import FigureColor

class Rectangle(GeometricFigure):

    figure\_type = "Прямоугольник"

    def \_\_init\_\_(self, width, height, color):

        self.width = width

        self.height = height

        self.color = FigureColor(color)

    def area(self):

        return self.width \* self.height

    def \_\_repr\_\_(self):

        return "{} цвета {} шириной {} и высотой {}. Площадь: {:.2f}".format(

            self.figure\_type, self.color.color, self.width, self.height, self.area()

        )

Figure.py

from abc import ABC, abstractmethod

class GeometricFigure(ABC):

    """Абстрактный класс Геометрическая фигура"""

    @abstractmethod

    def area(self):

        """Вычисление площади фигуры"""

        Pass

# **Результаты работы программы**

