|  |  |
| --- | --- |
|  | **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 «Системы обработки информации и управления»**

Гонов Марат ИУ5-35Б

Парадигмы и конструкции языков программирования

**ОТЧЁТ ПО**

**Лабораторной работе №2**

Москва

2023

**Задание.**

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

Создать пакет "lab\_python\_oop" и внутри него создать следующие файлы и классы:

* geometry.py: Создать абстрактный класс "Геометрическая фигура" с абстрактным методом для вычисления площади.
* color.py: Создать класс "Цвет фигуры" со свойством для описания цвета геометрической фигуры.
* rectangle.py: Создать класс "Прямоугольник", который наследуется от "Геометрической фигуры". Включить конструктор, переопределить метод вычисления площади и определить метод repr для возврата информации о прямоугольнике.
* circle.py: Создать класс "Круг" аналогично "Прямоугольнику", задав параметр "радиус" и использовать math.pi для вычисления площади.
* square.py: Создать класс "Квадрат", который наследуется от "Прямоугольника". Включить конструктор по длине стороны.

В каждом из классов "Прямоугольник", "Круг" и "Квадрат" определить метод repr, который будет возвращать информацию о фигуре, ее цвете и площади в виде строки.

В главном файле "main.py" создать объекты этих классов, установить им цвет и вывести информацию о них с помощью метода repr.

Запустить консольное приложение, чтобы убедиться, что оно работает правильно.

**Текст программы.**

main.py

from bub\_oo.rectangle import Rectangle

from bub\_oo.circle import Circle

from bub\_oo.square import Square

def main():

    r = Rectangle(5, 5, "синего")

    c = Circle(5, "зеленого")

    s = Square( 5, "красного")

    print(r)

    print(c)

    print(s)

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

    main()

figure.py

from abc import ABC, abstractionmethod

class Figure(ABC):

    @abstractionmethod

    def area(self):

        pass

color.py

class Color:

    def \_\_init\_\_(self, color = None):

        self.\_color = color

rectangle.py

from bub\_oo.figure import Figure

from bub\_oo.color import Color

class Rectangle(Figure):

    type = "Прямоугольник"

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

        self.width = width

        self.length = length

        self.r\_color = Color(color)

    def area(self):

        return self.width \* self.length

    def \_\_repr\_\_(self):

       return '{} {} цвета с длиной стороны {}, шириной стороны {}, площадью {}.'.format(

            self.type,

            self.r\_color.\_color,

            self.length,

            self.width,

            self.area()

        )

circle.py

from bub\_oo.figure import Figure

from bub\_oo.color import Color

import math

class Circle(Figure):

    type = "Круг"

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

        self.radius = radius

        self.c\_color = Color(color)

    def area(self):

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

    def \_\_repr\_\_(self):

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

            self.type,

            self.c\_color.\_color,

            self.radius,

            self.area()

        )

square.py

from bub\_oo.rectangle import Rectangle

class Square(Rectangle):

    type = "Квадрат"

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

        self.side = side

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

    def \_\_repr\_\_(self):

        return '{} {} цвета с длиной стороны {}, площадью {}.'.format(

            self.type,

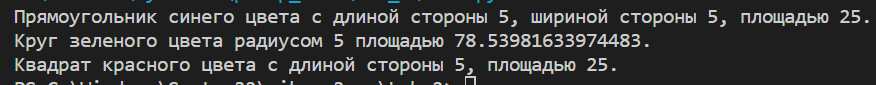
            self.r\_color.\_color,

            self.side,

            self.area()

        )

Экранные формы:

****

Pip:

