**Московский государственный технический**

**Университет им Н.Э.Баумана**

Факультет «Информатика и системы управление»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №2

Проверил:

Преподаватель каф. ИУ5

Нардид А. Н.

Подпись и дата:

Выполнил:

Студент группы ИУ5-34Б

Малышко А. В.

Подпись и дата:

Москва 2024 г.

**Постановка задачи**

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.](https://docs.python.org/3/library/abc.html)
6. Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры. Подробнее про описание свойств Вы можете прочитать [здесь.](https://docs.python.org/3/library/functions.html#property)
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.](https://docs.python.org/3/library/math.html)
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.

**Текст программы**

**main.py**

from lab\_python\_oop.Rectangle import Rectangle  
from lab\_python\_oop.square import Square  
from lab\_python\_oop.circle import Circle  
import requests  
import urllib3  
  
urllib3.disable\_warnings(urllib3.exceptions.InsecureRequestWarning)  
  
  
def main():  
 N = 15  
 rectangle = Rectangle(N, N, 'blue')  
 circle = Circle(N, 'green')  
 square = Square(N, 'red')  
  
 print(rectangle)  
 print(circle)  
 print(square)  
  
 response = requests.get("https://api.github.com", verify=False)  
 print(f"GitHub API status: {response.status\_code}")  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**square.py**

from lab\_python\_oop.Rectangle import Rectangle  
from lab\_python\_oop.color import Color  
  
class Square(Rectangle):  
 def \_\_init\_\_(self, side: float, color: str):  
 super().\_\_init\_\_(side, side, Color(color))  
 self.side = side  
 self.color=Color(color)  
  
 def square(self):  
 return self.width \* self.height  
  
 def get\_color(self):  
 return self.color.color  
  
 def set\_color(self, value: str):  
 self.color.color = value  
  
 def \_\_repr\_\_(self):  
 return f"Square(side={self.side}, color={self.get\_color()}, square={self.square()})"

**color.py**

class Color:  
 def \_\_init\_\_(self, color=None):  
 self.\_color = color  
  
 @property  
 def color(self):  
 return self.\_color  
  
 @color.setter  
 def color(self, value: str):  
 self.\_color = value

**Rectangle.py**

from lab\_python\_oop.color import Color  
from lab\_python\_oop.geometric import Geometric\_Figure  
  
class Rectangle(Geometric\_Figure):  
 def \_\_init\_\_(self, width: float, height: float, color: str):  
 self.width = width  
 self.height = height  
 self.color = Color(color)  
  
 def square(self):  
 return self.width \* self.height  
  
 def get\_color(self):  
 return self.color.color  
  
 def set\_color(self, value: str):  
 self.color.color = value  
  
 def \_\_repr\_\_(self):  
 return f"Rectangle(width={self.width}, height={self.height}, color={self.get\_color()}, square={self.square()})"

**geometric.py**

from abc import ABC, abstractmethod  
  
class Geometric\_Figure(ABC):  
 @abstractmethod  
 def square(self):  
 pass

**circle.py**

from lab\_python\_oop.color import Color  
from lab\_python\_oop.geometric import Geometric\_Figure  
import math  
  
class Circle(Geometric\_Figure):  
 def \_\_init\_\_(self, radius: float, color: str):  
 self.radius = radius  
 self.color = Color(color)  
  
 def square(self):  
 return math.pi \* self.radius \*\* 2  
  
 def get\_color(self):  
 return self.color.color  
  
 def set\_color(self, value: str):  
 self.color.color = value  
  
 def \_\_repr\_\_(self):  
 return f"Circle(radius={self.radius}, color={self.get\_color()}, square={self.square()})"

**Экранные формы с примерами выполнения программы**



 