**Московский государственный технический**

**Университет им. Н.Э. Баумана**

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №2

## «Объектно-ориентированные возможности языка Python»

Выполнил:

студент группы ИУ5-31Б

Корецкий К.В.

Проверил:

Гапанюк Е.Ю.

2022 г.

**Задание**

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»:
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.

**Текст программы**

Директории:

main.py

Pipfile

Pipfile.lock

lab\_python\_oop/

Circle.py

Rectangle.py

Shape.py

ShapeColor.py

Square.py

\_\_init\_\_.py

main.py:

**from lab\_python\_oop import Circle**

**from lab\_python\_oop import Square**

**from lab\_python\_oop import Rectangle**

**import cowsay**

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

**a = Rectangle.Rectangle(6, 6, 'blue', 'pryamougolnik')**

**b = Circle.Circle(6, 'green', 'krug')**

**c = Square.Square(6, 'red', 'kvadrat')**

**cowsay.trex(str(a)+'\n'+str(b)+'\n'+str(c))**

Pipfile:

**[[source]]**

**url = "https://pypi.org/simple"**

**verify\_ssl = true**

**name = "pypi"**

**[packages]**

**cowsay = "\*"**

**[dev-packages]**

**[requires]**

**python\_version = "3.8"**

ShapeColor:

**class ShapeColor():**

**def \_\_init\_\_(self):**

**self.value = None**

**@property**

**def color(self):**

**return self.value**

**@color.setter**

**def color(self, value):**

**self.value = value**

**@color.deleter**

**def color(self):**

**del self.value**

**def color(self):**

**return self.value**

Shape:

**from abc import ABC**

**from abc import abstractmethod**

**from .ShapeColor import ShapeColor**

**class Shape(ABC):**

**def \_\_init\_\_(self, name):**

**self.color = ShapeColor()**

**self.name = name**

**@abstractmethod**

**def area(self):**

**pass**

**@abstractmethod**

**def get\_name(self):**

**return self.name**

**def \_\_repr\_\_(self):**

**return "{}'s area is {:.3f} and color is {}".format(self.get\_name(), self.area(), self.color.value)**

Rectangle:

**from .Shape import Shape**

**class Rectangle(Shape):**

**def \_\_init\_\_(self, width=0, height=0, color=None, name=''):**

**super().\_\_init\_\_(name=name)**

**self.width = width**

**self.height = height**

**self.color.value = color**

**def area(self):**

**return self.width\*self.height**

**def get\_name(self):**

**return self.name**

Circle:

**import math**

**from .Shape import Shape**

**class Circle(Shape):**

**def \_\_init\_\_(self, radius=0, color=None, name=''):**

**super().\_\_init\_\_(name=name)**

**self.color.value = color**

**self.radius = radius**

**def area(self):**

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

**def get\_name(self):**

**return self.name**

Square:

**from .Rectangle import Rectangle**

**class Square(Rectangle):**

**def \_\_init\_\_(self, width=0, color=None, name=''):**

**super().\_\_init\_\_(name=name)**

**self.width = width**

**self.color.value = color**

**def area(self):**

**return self.width\*self.width**

**def get\_name(self):**

**return self.name**

**Экранные формы с примерами выполнения программы**

