Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»



**Отчет**

**Лабораторная работа № 2**

**По курсу «Разработка интернет приложений»**

**ИСПОЛНИТЕЛЬ:**

Группа ИУ5Ц-74Б

Шанаурина Е.Г.

**ПРЕПОДАВАТЕЛЬ:**

Гапанюк Ю.Е.

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

"\_\_"\_\_\_\_\_\_\_\_\_\_\_2021 г.

Москва 2021

1. **Задание**
2. Необходимо создать виртуальное окружение и установить в него хотя бы один внешний пакет с использованием pip.
3. Необходимо разработать программу, реализующую работу с классами. Программа должна быть разработана в виде консольного приложения на языке Python 3.
4. Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab\_python\_oop.
5. Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab\_python\_oop.
6. 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)
7. 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)
8. 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.
9. 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)
10. Класс «Квадрат» наследуется от класса «Прямоугольник». Класс должен содержать конструктор по длине стороны. Для классов «Прямоугольник», «Квадрат», «Круг»:
    1. Определите метод "repr", который возвращает в виде строки основные параметры фигуры, ее цвет и площадь. Используйте метод format - <https://pyformat.info/>
    2. Название фигуры («Прямоугольник», «Квадрат», «Круг») должно задаваться в виде поля данных класса и возвращаться методом класса.
11. 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):
    1. Прямоугольник синего цвета шириной N и высотой N.
    2. Круг зеленого цвета радиусом N.
    3. Квадрат красного цвета со стороной N.
    4. Также вызовите один из методов внешнего пакета, установленного с использованием pip.
12. **Текст программы**

*main.py*

from lab\_python\_oop.rectangle import Rectangle  
from lab\_python\_oop.circle import Circle  
from lab\_python\_oop.square import Square  
import numpy  
  
  
def main():  
 my\_rectangle = Rectangle(17, 17, "синего")  
 my\_circle = Circle(17, "зеленого")  
 my\_square = Square(17, "красного")  
  
 print(my\_rectangle)  
 print(my\_circle)  
 print(my\_square)  
  
 my\_array = numpy.array([[1, 2, 3], [4, 5, 6]], float)  
 print("Метод внешнего пакета NumPy (shape) ", my\_array.shape)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

*figure.py*

from abc import ABC, abstractmethod  
  
  
class Figure(ABC):  
  
 @abstractmethod  
 def square(self):  
 pass

*color.py*

class FigureColor:  
 def \_\_init\_\_(self):  
 self.\_color = None  
  
 @property  
 def colorproperty(self):  
 return self.\_color  
  
 @colorproperty.setter  
 def colorproperty(self, value):  
 self.\_color = value

*rectangle.py*

from lab\_python\_oop.figure import Figure  
from lab\_python\_oop.color import FigureColor  
  
  
class Rectangle(Figure):  
  
 FIGURE\_TYPE = "Прямоугольник"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, width, height, color):  
 self.width = width  
 self.height = height  
 self.fc = FigureColor()  
 self.fc.colorproperty = color  
  
 def square(self):  
 return self.width \* self.height  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета с шириной {} и высотой {} площадью {}.'.format(  
 Rectangle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.width,  
 self.height,  
 self.square()  
 )

*circle.py*

from lab\_python\_oop.figure import Figure  
from lab\_python\_oop.color import FigureColor  
from math import pi  
  
  
class Circle(Figure):  
  
 FIGURE\_TYPE = "Круг"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, radios, color):  
 self.radios = radios  
 self.fc = FigureColor()  
 self.fc.colorproperty = color  
  
 def square(self):  
 return pi \* self.radios \* self.radios  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета с радиусом {} площадью {}.'.format(  
 Circle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.radios,  
 self.square()  
 )

*square.py*

from lab\_python\_oop.color import FigureColor  
from lab\_python\_oop.rectangle import Rectangle  
  
  
class Square(Rectangle):  
  
 FIGURE\_TYPE = "Квадрат"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, side, color):  
 self.side = side  
 self.fc = FigureColor()  
 self.fc.colorproperty = color  
  
 def square(self):  
 return self.side \* self.side  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета со стороной {} площадью {}.'.format(  
 Square.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.side,  
 self.square()  
 )

1. **Экранные формы с примерами выполнения программы**





