|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 ИУ5 «Системы обработки информации и управления»**

Курс «Разработка интернет-приложений»

Отчет по лабораторной работе №2

Выполнила:

студент группы ИУ5-53Б

Латыпова К.Н.

Москва, 2020 г.

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»:
    * 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.
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):
    * 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.
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 requests  
  
  
def main():  
 r = Rectangle("синего", 14, 24)  
 c = Circle("зеленого", 14)  
 s = Square("красного", 14)  
 print(r.\_\_repr\_\_())  
 print(c.\_\_repr\_\_())  
 print(s.\_\_repr\_\_())  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**color.py:**

class GeomFigureColor:  
  
 def \_\_init\_\_(self,value):  
 self.\_color\_ = value  
  
 @property  
 def color(self):  
  
 return self.\_color\_  
  
 @color.setter  
 def color(self, value):  
  
 self.\_color\_ = value

**circle.py:**

from lab\_python\_oop.figure import GeomFigure  
from lab\_python\_oop.color import GeomFigureColor  
import math  
  
  
class Circle(GeomFigure):  
 \_radius\_ = 0  
 \_square\_ = 0  
 FIGURE\_TYPE = "Круг"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color, radius,):  
 self.\_radius\_ = radius  
 self.colour = GeomFigureColor(color)  
  
 def square(self):  
 return float(math.pi) \* self.\_radius\_ \* self.\_radius\_  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета радиусом {} и площадью {}.'.format(  
 Circle.get\_figure\_type(),  
 self.colour.color,  
 self.\_radius\_,  
 self.square()  
 )

**figure.py:**

from abc import ABC, abstractmethod  
  
  
class GeomFigure():  
 \_\_metaclass\_\_ = ABC  
 \_square\_ = 0  
  
 @abstractmethod  
 def square(self):  
  
 pass

**square.py:**

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, color\_param, side\_param):  
  
 self.side = side\_param  
 super().\_\_init\_\_(color\_param, self.side, self.side)  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета со стороной {} площадью {}.'.format(  
 Square.get\_figure\_type(),  
 self.colour.color,  
 self.side,  
 self.square()  
 )

**rectangle.py:**

from lab\_python\_oop.figure import GeomFigure  
from lab\_python\_oop.color import GeomFigureColor  
  
  
class Rectangle(GeomFigure):  
 \_square\_ = 0  
 \_width\_ = 0  
 \_height\_ = 0  
 FIGURE\_TYPE = "Прямоугольник"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, width\_param, height\_param):  
  
 self.\_width\_ = width\_param  
 self.\_height\_ = height\_param  
 self.colour= GeomFigureColor(color\_param)  
 ##self.fc = GeomFigureColor()  
 ## self.fc.colorproperty = color\_param  
  
 def square(self):  
  
 return self.\_width\_\*self.\_height\_  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета шириной {} , высотой {} и площадью {}.'.format(  
 Rectangle.get\_figure\_type(),  
 self.colour.color,  
 self.\_width\_,  
 self.\_height\_,  
 self.square()  
 )

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

