МГТУ им. Н. Э. Баумана

Кафедра ИУ5, курс “Разработка интернет-приложений”

**Отчёт**

**Лабораторная работа №3**

Python. Объектно-ориентированные возможности.

Студент: Белкина Е.В.

Группа ИУ5-51

Преподаватель: Гапанюк Ю.В.

Москва, 2019

**Задание:**

**Необходимо разработать программу, реализующую работу с классами.**

1. Программа должна быть разработана в виде консольного приложения на языке Python 3.
2. Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab\_python\_oop.
3. Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab\_python\_oop.
   1. 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>
   2. 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>
   3. 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.
   4. 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>
   5. Класс «Квадрат» наследуется от класса «Прямоугольник». Класс должен содержать конструктор по длине стороны.
   6. 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.
4. 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:
   1. Прямоугольник синего цвета шириной 3 и высотой 2.
   2. Круг зеленого цвета радиусом 5.
   3. Квадрат красного цвета со стороной 5.

**Исходный код:**

Lab1\_main.py

from lab\_python\_oop.Rectangle import Rectangle  
from lab\_python\_oop.Circle import Circle  
from lab\_python\_oop.Square import Square  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 rect = Rectangle(3, 2, 'blue')  
 circle = Circle(5, 'green')  
 square = Square(5, 'red')  
 print(rect)  
 print(circle)  
 print(square)

GeomFigure.py

from abc import ABC, abstractmethod  
  
  
class GeomFigure(ABC): # абстрактный класс  
 \_\_name = ''  
  
 def return\_name(self):  
 return 'This figure is a {}'.format(self.\_\_name)  
  
 # абстрактный метод, который будет необходимо переопределять для каждого подкласса  
 @abstractmethod  
 def square(self):  
 pass

Property.py

class Colour:  
 def \_\_init\_\_(self):  
 self.\_col = None  
  
 @property  
 def col(self):  
 return self.\_col  
  
 @col.setter  
 def col(self, value):  
 self.\_col = value  
  
 @col.deleter  
 def col(self):  
 del self.col

Rectangle.py

from lab\_python\_oop import GeomFigure  
from lab\_python\_oop import Property  
  
  
class Rectangle(GeomFigure):  
 def \_\_init\_\_(self, width, height, color):  
 self.\_\_width = width  
 self.\_\_height = height  
 self.\_\_c = Property.Colour()  
 self.\_\_c.col = color  
 self.\_\_name = 'Rectangle'  
  
 def square(self):  
 return self.\_\_height \* self.\_\_height  
  
 def \_\_repr\_\_(self):  
 return 'Parameters of figure: \n \  
 Width = {} \n \  
 Height = {} \n \  
 Colour = {} \n \  
 Square = {}'.format(self.\_\_width, self.\_\_height, self.\_\_c, self.square())

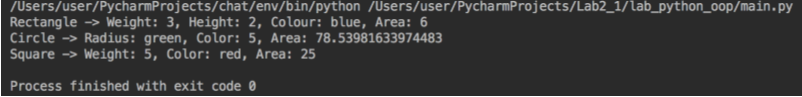
Circle.py

from lab\_python\_oop import GeomFigure  
from lab\_python\_oop import Property  
import math  
  
  
class Circle(GeomFigure):  
 def \_\_init\_\_(self, radius, color):  
 self.\_\_radius = radius  
 self.\_\_c = Property.Colour()  
 self.\_\_c.col = color  
 self.\_\_name = 'Circle'  
  
 def square(self):  
 return math.pi \* self.\_\_radius \*\* 2  
  
 def \_\_repr\_\_(self):  
 return 'Parameters of figure: \n \  
 Radius = {} \n \  
 Colour = {} \n \  
 Square = {}'.format(self.\_\_radius, self.\_\_c, self.square())

Square.py

from lab\_python\_oop import Rectangle  
from lab\_python\_oop import Property  
  
  
class Square(Rectangle):  
 def \_\_init\_\_(self, side, color):  
 self.\_\_side = side  
 self.\_\_c = Property.Colour()  
 self.\_\_c.col = color  
 self.\_\_name = 'Square'  
  
 def square(self):  
 return self.\_\_side \*\* 2  
  
 def \_\_repr\_\_(self):  
 return 'Parameters of figure: \n \  
 Side = {} \n \  
 Colour = {} \n \  
 Square = {}'.format(self.\_\_side, self.\_\_c, self.square())

**Результат:**



**Диаграмма классов:**

