|  |  |
| --- | --- |
| 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 **СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ**

**Отчет по лабораторной работе №2**

**по дисциплине Базовые компоненты интернет технологии**

**Тема работы: "** [**Объектно-ориентированные возможности языка Python**](https://github.com/ugapanyuk/BKIT_2022/wiki/lab_python_oop)**"**

**Выполнил:**

Студент группы ИУ5Ц-54Б

Перевощиков Н.Д. 09.11.22 г.

(дата, подпись)

**Проверил:**

Преподаватель

Канев А.И.

(дата, подпись)

Москва, 2022

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

**СОДЕРЖАНИЕ ОТЧЕТА**

[1. Цель лабораторной работы 3](#_Toc115088672)

[2. Описание задания 4](#_Toc115088673)

[3. Текст программы 6](#_Toc115088674)

[4. Результат 13](#_Toc115088675)

## Цель лабораторной работы

Изучение объектно-ориентированных возможностей языка Python.

## Описание задания

## Необходимо создать виртуальное окружение и установить в него хотя бы один внешний пакет с использованием pip.

## Необходимо разработать программу, реализующую работу с классами. Программа должна быть разработана в виде консольного приложения на языке Python 3.

## Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab\_python\_oop.

## Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab\_python\_oop.

## Абстрактный класс «Геометрическая фигура» содержит абстрактный метод для вычисления площади фигуры. Подробнее про абстрактные классы и методы Вы можете прочитать здесь.

## Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры. Подробнее про описание свойств Вы можете прочитать здесь.

## 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.

## Класс «Круг» создается аналогично классу «Прямоугольник», задается параметр «радиус». Для вычисления площади используется константа math.pi из модуля math.

## 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.

## 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.

## Дополнительное задание. Протестируйте корректность работы Вашей программы с помощью модульного теста.

## Текст программы

**LAB\_02.py**

from lab\_python\_oop.rectangle import Rectangle  
from lab\_python\_oop.circle import Circle  
from lab\_python\_oop.square import Square  
  
import sys  
"""Def – это инструкция, позволяющая создавать функцию."""  
def get\_coef(index, prompt):  
 try:  
 #Пробуем прочитать коэффициент из командной строки  
 coef\_str = sys.argv[index]  
 except:  
 while True:  
 #Вводим с клавиатуры  
 print(prompt)  
 coef\_str = input()  
 # Проверка, есть ли минус и 0?  
 if (int(coef\_str) > 0 and int(coef\_str) < 20):  
 break  
 else:  
 print('Ошибка! Введите номер варианта от 1 до 20')  
 #Переводим строку в действительное число  
 coef = float(coef\_str)  
 return coef  
  
def main():  
 N = get\_coef(1, 'Введите номер вашего варианта по списку журнала:')  
 if (N > 0 and N < 20):  
 r = Rectangle("синего", N, N)  
 c = Circle("зеленого", N)  
 s = Square("красного", N)  
 print(r)  
 print(c)  
 print(s)  
 else:  
 print('Ошибка! Введите номер варианта от 1 до 20')  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Файлы, содержащие отдельных функций, должны располагаться в пакете lab\_python\_oop.

**\_\_init\_\_.py**

class FigureColor:  
 *"""  
 Класс «Цвет фигуры»  
 """* def \_\_init\_\_(self):  
 self.\_color = None  
  
 @property  
 def colorproperty(self):  
 *"""  
 Get-аксессор  
 Get – аксессор, который используется для чтения значения из внутреннего поля класса  
 """* return self.\_color  
  
 @colorproperty.setter  
 def colorproperty(self, value):  
 *"""  
 Set-аксессор  
 Set – аксессор, используемый для записи значения во внутреннее поле класса.  
 """* self.\_color = value

**Circle.py**

from .figure import Figure  
from .color import FigureColor  
import math  
  
class Circle(Figure):  
 *"""  
 Класс «Круг» наследуется от класса «Геометрическая фигура».  
 """* FIGURE\_TYPE = "Круг"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, r\_param):  
 *"""  
 Класс должен содержать конструктор по параметрам «радиус» и «цвет».  
 В конструкторе создается объект класса «Цвет фигуры» для хранения цвета.  
 """* self.r = r\_param  
 self.fc = FigureColor()  
 self.fc.colorproperty = color\_param  
  
 def square(self):  
 *"""  
 Класс должен переопределять метод, вычисляющий площадь фигуры.  
 """* return math.pi\*(self.r\*\*2)  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета радиусом {} площадью {}.'.format(  
 Circle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.r,  
 self.square()  
 )

**Color.py**

class FigureColor:  
 *"""  
 Класс «Цвет фигуры»  
 """* def \_\_init\_\_(self):  
 self.\_color = None  
  
 @property  
 def colorproperty(self):  
 *"""  
 Get-аксессор  
 """* return self.\_color  
  
 @colorproperty.setter  
 def colorproperty(self, value):  
 *"""  
 Set-аксессор  
 """* self.\_color = value

**Figure.py**

from abc import ABC, abstractmethod  
  
class Figure(ABC):  
 *"""  
 Абстрактный класс «Геометрическая фигура»  
 """* @abstractmethod  
 def square(self):  
 *"""  
 содержит виртуальный метод для вычисления площади фигуры.  
 """* pass

**Rectangle.py**

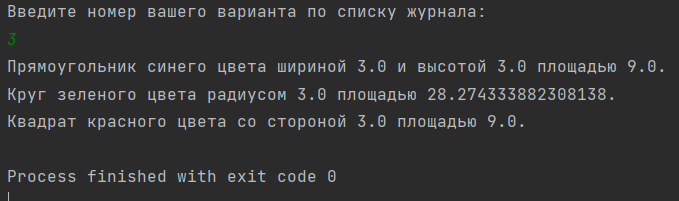
from .figure import Figure  
from .color import FigureColor  
  
class Rectangle(Figure):  
 *"""  
 Класс «Прямоугольник» наследуется от класса «Геометрическая фигура».  
 """* FIGURE\_TYPE = "Прямоугольник"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, width\_param, height\_param):  
 *"""  
 Класс должен содержать конструктор по параметрам «ширина», «высота» и «цвет».  
 В конструкторе создается объект класса «Цвет фигуры» для хранения цвета.  
 """* self.width = width\_param  
 self.height = height\_param  
 self.fc = FigureColor()  
 self.fc.colorproperty = color\_param  
  
 def square(self):  
 *"""  
 Класс должен переопределять метод, вычисляющий площадь фигуры.  
 """* return self.width\*self.height  
  
 def \_\_repr\_\_(self):  
 *"""  
 Метод \_\_repr\_\_ в Python выдает текстовое или строковое представление сущности или объекта.  
 """* return '{} {} цвета шириной {} и высотой {} площадью {}.'.format(  
 Rectangle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.width,  
 self.height,  
 self.square()  
 )

**Square.py**

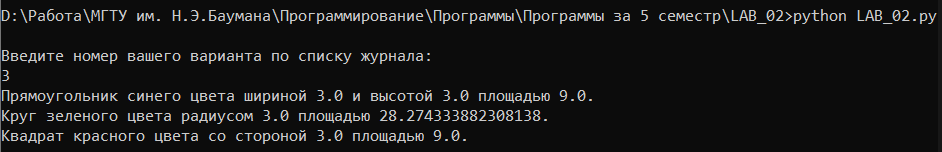
from .rectangle import Rectangle  
  
class Square(Rectangle):  
 *"""  
 Класс «Квадрат» наследуется от класса «Прямоугольник».  
 """* FIGURE\_TYPE = "Квадрат"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, side\_param):  
 *"""  
 Класс должен содержать конструктор по параметрам «сторона» и «цвет».  
 """* 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.fc.colorproperty,  
 self.side,  
 self.square()  
 )

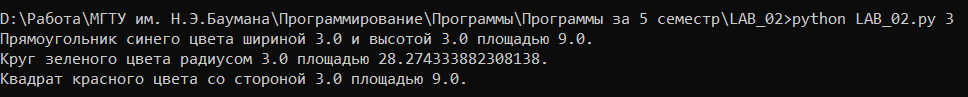
## Результат

**PyCharm**

****

**Командная строка**

****

****