**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 вычислительная техника»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Отчет по лабораторной работе №2

«Объектно-ориентированные возможности языка Python»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-52Б |  | преподаватель каф. ИУ5 |
| Веревкина Диана В. |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2021 г.

# ****Цель лабораторной работы****

Изучить объектно-ориентированные возможности 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.

# Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры.

# 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.

# 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.

# Текст программы

**main**

from lab\_python\_oop.Rectangle import \*  
from lab\_python\_oop.Quadrate import \*  
from lab\_python\_oop.Circle import \*  
  
def main():  
 rec = Rectangle(7, 8, 'Синий')  
 qud = Quadrate(7, 'красный')  
 cir = Circle(7, 'зеленый')  
 print(rec.repr())  
 print(cir.repr())  
 print(qud.repr())  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**Figure**

from abc import abstractmethod  
from lab\_python\_oop.ColorFigure import \*  
  
class Figure:  
 def \_\_init\_\_(self, color, h, name, par):  
 self.CF = ColorFigure(color)  
 self.\_h = h  
 self.\_name = name  
 self.\_par = par  
 #self.\_s  
  
 @abstractmethod  
 def square(self):  
 self.\_s = self.\_h \* self.\_h  
  
 def repr(self):  
 return ('{}, {}, с площадью {}, {}'.format(self.\_name, self.\_par, self.\_s, self.CF.get\_color()))  
  
 def get\_name(self):  
 return self.\_name

**Circle**

from lab\_python\_oop.Figure import \*  
import math  
  
class Circle(Figure):  
 def \_\_init\_\_(self, r , color):  
 super().\_\_init\_\_(color, r,"круг",'С радиусом {}'.format(r))  
 self.square()  
  
  
 def square(self):  
 self.\_s = math.pi \* self.\_h \* self.\_h

**ColorFigure**

class ColorFigure:  
 def \_\_init\_\_(self, color):  
 self.\_\_color: str = color  
  
 def get\_color(self):  
 return self.\_\_color  
  
 def set\_color(self, color):  
 self.\_\_color = color  
  
 property = (get\_color, set\_color)

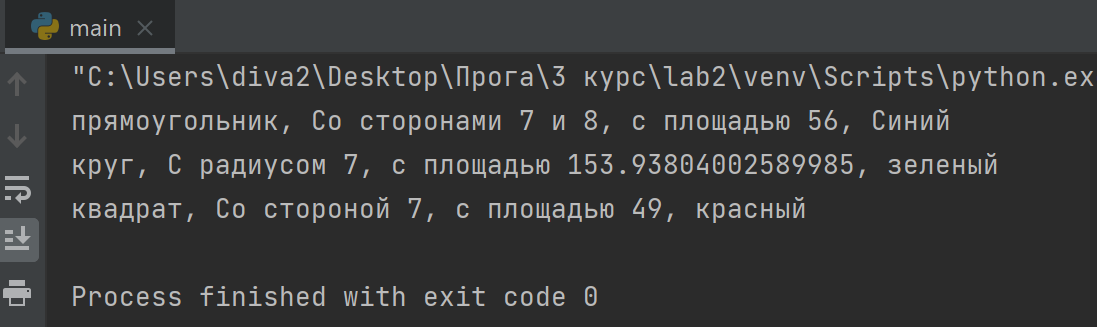
**Quadrate**

from lab\_python\_oop.Rectangle import \*  
  
class Quadrate(Rectangle):  
 def \_\_init\_\_(self, h, color):  
 super(Rectangle, self).\_\_init\_\_(color, h, "квадрат",'Со стороной {}'.format(h))  
 self.square()  
  
  
 def square(self):  
 self.\_s = self.\_h \* self

**Rectangle**

from lab\_python\_oop.Figure import \*  
  
class Rectangle(Figure):  
 def \_\_init\_\_(self, h, w, color):  
 self.\_\_w = w  
 super().\_\_init\_\_(color, h, "прямоугольник", 'Со сторонами {} и {}'.format(h, w))  
 self.square()  
  
  
 def square(self):  
 self.\_s = self.\_h \* self.\_\_w

# Экранные формы с примерами выполнения программы



# Вывод

В ходе выполнения данной лабораторной работы были повторены основные конструкции языка Python. Также были изучены объектно-ориентированные возможности языка Python я.