**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «ПиКЯП»

Отчет по лабораторной работе №2

## «Объектно-ориентированные возможности языка Python.»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34Б |  | преподаватель каф. ИУ5 |
| Сафронов Андрей |  | Нардид А. Н. |
|  |  |  |

Москва, 2024 г.

**Цель лабораторной работы:** изучение объектно-ориентированных возможностей языка Python.

**Задание:**

1. Необходимо создать виртуальное окружение и установить в него хотя бы один внешний пакет с использованием pip.
2. Необходимо разработать программу, реализующую работу с классами. Программа должна быть разработана в виде консольного приложения на языке Python 3.
3. Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab\_python\_oop.
4. Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab\_python\_oop.
5. 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)
6. 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)
7. 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.
8. 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)
9. 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.
10. 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.
11. **Дополнительное задание.** Протестируйте корректность работы Вашей программы с помощью модульного теста.

Код 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 as np  
  
def start():  
 rect = Rectangle(22, 22, (0, 0, 255))  
 circle = Circle(22, (0, 255, 0))  
 square = Square(22, (255, 0, 0))  
 arr = np.array([1, 2, 3])  
 print(arr)  
 print(rect.repr())  
 print(circle.repr())  
 print(square.repr())  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 start()

circle.py

from lab\_python\_oop.figure import Figure  
from lab\_python\_oop.color import Color  
import math  
  
  
class Circle(Figure):  
 def \_\_init\_\_(self, r: float, c: tuple[int, int, int]):  
 self.radius = r  
 self.name = "Circle"  
 self.color = Color()  
 self.color.color = c  
  
 def area(self):  
 return math.pi \* self.radius \*\* 2  
  
 def repr(self):  
 return "pi x {0} ^ 2 = {1}, {2}".format(self.radius, self.area(), self.color.color)  
  
 def get\_name(self):  
 return self.name

color.py

class Color:  
 def \_\_init\_\_(self):  
 self.\_\_color = None  
  
 def get\_color(self):  
 return self.\_\_color  
  
 def set\_color(self, value):  
 self.\_\_color = value  
  
 def del\_color(self):  
 del self.\_\_color  
  
 color = property(get\_color, set\_color, del\_color)

figure.py

import abc  
  
  
class Figure(abc.ABC):  
 @abc.abstractmethod  
 def area(self):  
 pass

rectangle.py

from lab\_python\_oop.figure import Figure  
from lab\_python\_oop.color import Color  
  
  
class Rectangle(Figure):  
 def \_\_init\_\_(self, w: float, h: float, c: tuple[int, int, int]):  
 self.width = w  
 self.height = h  
 self.name = "Rectangle"  
 self.color = Color()  
 self.color.color = c  
  
 def area(self):  
 return self.width \* self.height  
  
 def repr(self):  
 return "{0} x {1} = {2}, {3}".format(self.width, self.height, self.area(), self.color.color)  
  
 def get\_name(self):  
 return self.name

square.py

from lab\_python\_oop.rectangle import Rectangle  
  
  
class Square(Rectangle):  
 def \_\_init\_\_(self, s: float, c: tuple[int, int, int]):  
 super().\_\_init\_\_(s, s, c)  
 self.name = "Square"

tests.py

import unittest  
import math  
from lab\_python\_oop.rectangle import Rectangle  
from lab\_python\_oop.circle import Circle  
from lab\_python\_oop.square import Square  
  
  
class MyTestCase(unittest.TestCase):  
 def test\_rect(self):  
 rect = Rectangle(5, 10, (0, 0, 255))  
 self.assertEqual(rect.area(), 50)  
  
 def test\_square(self):  
 square = Square(10, (255, 0, 0))  
 self.assertEqual(square.area(), 100)  
  
 def test\_circle(self):  
 circle = Circle(22, (0, 255, 0))  
 self.assertEqual(circle.area(), math.pi \* 22 \*\* 2)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

Результаты:

