**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №2

«Объектно-ориентированные возможности языка Python»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б |  | преподаватель каф. ИУ5 |
| Турдиев Ихтияр |  | Гапанюк Юрий Евгеньевич |
|  |  |  |
|  |  |  |

Москва, 2024 г.

Постановка задачи

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

1. 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.

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

Текст программы

shape.py

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

shape\_color.py

class Color:  
 def \_\_init\_\_(self):  
 self.\_\_color\_property = None  
  
 @property  
 def color(self) -> str:  
 return self.\_\_color\_property  
  
 @color.setter  
 def color(self, color):  
 self.\_\_color\_property = color  
  
 def \_\_repr\_\_(self):  
 return str(self.\_\_color\_property)

rectangle.py

from lab\_python\_oop.shape import Shape  
from lab\_python\_oop.shape\_color import Color  
  
class Rectangle(Shape):  
 SHAPE\_NAME = "Rectangle"  
  
 @classmethod  
 def get\_name(cls):  
 return cls.SHAPE\_NAME  
  
 def \_\_init\_\_(self, hight, width, color="Blue"):  
 if hight < 0 or width < 0:  
 raise TypeError  
 self.\_\_hight = hight  
 self.\_\_width = width  
 self.color = Color()  
 self.color.color = color  
  
 def area(self):  
 return self.\_\_hight \* self.\_\_width  
  
 def \_\_repr\_\_(self):  
 return "{} {} with params = {}/{} and area = {}".format(  
 self.color.color,  
 self.get\_name(),  
 self.\_\_hight,  
 self.\_\_width,  
 self.area()  
 )

circle.py

from lab\_python\_oop.shape import Shape  
from lab\_python\_oop.shape\_color import Color  
from math import pi  
  
class Circle(Shape):  
 SHAPE\_NAME = "Circle"  
  
 @classmethod  
 def get\_name(cls):  
 return cls.SHAPE\_NAME  
  
 def \_\_init\_\_(self, radius, color="Blue"):  
 if radius < 0:  
 raise TypeError  
 self.\_\_radius = radius  
 self.color = Color()  
 self.color.color = color  
  
 def area(self):  
 return round(pi \* self.\_\_radius \*\* 2, 1)  
  
 def \_\_repr\_\_(self):  
 return "{} {} with radius = {} and area = {}".format(  
 self.color.color,  
 self.get\_name(),  
 self.\_\_radius,  
 self.area()  
 )

square.py

from lab\_python\_oop.rectangle import Rectangle  
  
class Square(Rectangle):  
 SHAPE\_NAME = "Square"  
  
 @classmethod  
 def get\_name(cls):  
 return cls.SHAPE\_NAME  
  
 def \_\_init\_\_(self, side, color):  
 self.\_\_side = side  
 super().\_\_init\_\_(self.\_\_side, self.\_\_side, color)  
  
 def area(self):  
 return self.\_\_side \*\* 2  
  
 def \_\_repr\_\_(self):  
 return "{} {} with side = {} and area = {}".format(  
 self.color.color,  
 self.get\_name(),  
 self.\_\_side,  
 self.area()  
 )

test\_lab4.py

import unittest  
from lab\_python\_oop.rectangle import Rectangle  
from lab\_python\_oop.circle import Circle  
  
  
class TestShapesMethods(unittest.TestCase):  
 # rectangle tests  
 def test\_rectangle\_get\_name(self):  
 rect = Rectangle  
 self.assertEqual(rect.get\_name(), "Rectangle")  
  
 def test\_rectangle\_area(self):  
 self.assertEqual(Rectangle(12, 21).area(), 252)  
 self.assertEqual(Rectangle(2.4, 11.2,).area(), 26.88)  
 with self.assertRaises(TypeError):  
 Rectangle(-9, 4)  
  
 # circle tests  
 def test\_circle\_get\_name(self):  
 circle = Circle  
 self.assertEqual(circle.get\_name(), "Circle")  
  
 def test\_circle\_area(self):  
 self.assertEqual(Circle(12).area(), 452.4)  
 self.assertEqual(Circle(9).area(), 254.5)  
 with self.assertRaises(TypeError):  
 Circle(-9)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 unittest.main()

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(21, 21, "Blue")  
 circle = Circle(21, "Green")  
 square = Square(21, "Red")  
 print(rect, circle, square, sep='\n')

Обработка результатов