**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Программирование на основе классов и шаблонов»

Отчет по лабораторной работе №5

«Объектно-ориентированные возможности языка python»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Выполнил: | |  | | Проверил: | |
|  | студент группы ИУ5-31Б | |  | | преподаватель каф. ИУ5 | |
|  | Хомутинников Александр | |  | |  | |
|  | Подпись и дата: | |  | | Подпись и дата: | |
|  | |  | |  | |  |

Москва, 2023 г.

**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.
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»:
   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.
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):
    1. Прямоугольник синего цвета шириной N и высотой N.
    2. Круг зеленого цвета радиусом N.
    3. Квадрат красного цвета со стороной N.
    4. Также вызовите один из методов внешнего пакета, установленного с использованием pip.

**Текст программы:**

**main.py**

from lab\_python\_oop import \*  
import matplotlib.pyplot as plt  
  
  
def main():  
 rect1 = Rectangle(10, 12, 'r1', ['black', 50])  
 print(rect1.area())  
 rect1.repr()  
 print('-' \* 20)  
 circ1 = Circle(10, 'circ1', ['red', 50])  
 print(circ1.area())  
 circ1.repr()  
 print('-' \* 20)  
 square1 = Square(10, 'Square1', ['white', 50])  
 print(square1.area())  
 square1.repr()  
 print('-' \* 20)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**\_\_main\_\_.py**

from lab\_python\_oop import \*  
import matplotlib.pyplot as plt  
  
def main():  
 rect1 = Rectangle(10, 12, 'r1', ['black', 50])  
 print(rect1.area())  
 rect1.repr()  
 print('-' \* 20)  
 circ1 = Circle(10, 'circ1', ['red', 50])  
 print(circ1.area())  
 circ1.repr()  
 print('-' \* 20)  
 square1 = Square(10, 'Square1', ['white', 50])  
 print(square1.area())  
 square1.repr()  
 print('-' \* 20)  
 """Тест matplotlib"""  
 x = [1, 2, 3, 4, 5]  
 y = [25, 32, 34, 20, 25]  
 plt.plot(x, y)  
 plt.show()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

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

from .Rectangle import \*  
from .Circle import \*  
from .Square import \*  
  
NAME = "Figures"

**Figure.py**

from abc import ABC, abstractmethod  
  
  
class Figure(ABC):  
 def \_\_init\_\_(self, name=None, args=None):  
 if args is None:  
 args = [None, 0]  
 try:  
 color, transparency = args  
 except ValueError:  
 ValueError("Input [color, transparency] in \_\_init\_\_")  
 else:  
 self.\_color = Color(color, transparency)  
 self.\_name = name  
  
 @property  
 def name(self):  
 return self.\_name  
  
 @name.setter  
 def name(self,value):  
 self.\_name = value  
  
 @property  
 def color(self):  
 return self.\_color  
  
 @color.setter  
 def color(self, args):  
 try:  
 color, transparency = args  
 except ValueError:  
 ValueError("Input [color, transparency] in setter")  
 else:  
 self.\_color.color = color  
 self.\_color.transparency = transparency  
  
 @abstractmethod  
 def area(self):  
 pass  
  
  
class Color:  
 def \_\_init\_\_(self, color, transparency):  
 self.color = color  
 self.transparency = transparency

**Circle.py**

from .Figure import \*  
from math import pi  
  
  
class Circle(Figure):  
 def \_\_init\_\_(self, r, name, color\_transp):  
 super().\_\_init\_\_(name, [color\_transp[0], color\_transp[1]])  
 self.\_r = r  
  
 @property  
 def r(self):  
 return self.\_r  
  
 @r.setter  
 def r(self, value):  
 self.\_r = value  
  
 def area(self):  
 return pi \* self.\_r \*\* 2  
  
 def repr(self):  
 print("Круг, радиус r = {}, имя: {}, цвет: {}, прозрачность: {}%, площадь S = {}"  
 .format(self.\_r, self.\_name, self.\_color.color, self.\_color.transparency, self.area()))

**Rectangle.py**

from .Figure import \*  
  
  
class Rectangle(Figure):  
 def \_\_init\_\_(self, a, b, name, color\_transp):  
 super().\_\_init\_\_(name, [color\_transp[0], color\_transp[1]])  
 self.\_a = a  
 self.\_b = b  
 self.\_name = name  
  
 @property  
 def a(self):  
 return self.\_a  
  
 @a.setter  
 def a(self, value):  
 self.\_a = value  
  
 @property  
 def b(self):  
 return self.\_b  
  
 @b.setter  
 def b(self, value):  
 self.\_b = value  
  
 def area(self):  
 return self.\_a \* self.\_b  
  
 def repr(self):  
 print("Прямоугольник, сторона a = {}, сторона b = {}, имя: {}, цвет: {}, прозрачность: {}%, площадь S = {}"  
 .format(self.\_a, self.\_b, self.\_name, self.\_color.color, self.\_color.transparency, self.area()))

**Square.py**

from .Rectangle import \*  
  
  
class Square(Rectangle):  
 def \_\_init\_\_(self, a, name, color\_transp):  
 super().\_\_init\_\_(a, a, name, color\_transp)  
 del self.\_b  
  
 @property  
 def a(self):  
 return self.\_a  
  
 @a.setter  
 def a(self, value):  
 self.\_a = value  
  
 def area(self):  
 return self.\_a \*\* 2  
  
 def repr(self):  
 print("Квадрат, сторона a = {}, имя: {}, цвет: {}, прозрачность: {}%, площадь S = {}"  
 .format(self.\_a, self.\_name, self.\_color.color, self.\_color.transparency, self.area()))

**Пример выполнения программы:**

