я

|  |  |  |
| --- | --- | --- |
| Защищено:  Нардид А.Н.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |  | Демонстрация:  Нардид А.Н.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**Отчет по лабораторной работе № 2 по курсу**

**Парадигмы и конструкции языков программирования**

#### Тема работы: "Объектно-ориентированные возможности языка Python"

4

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-41Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Ахмеров Д.И. | "29" сентября 2024 г. |

Москва, МГТУ – 2024

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

**Описание задания**

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.
   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. 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**

**geometric\_figure.py**

from abc import ABC, abstractmethod

class Geometric\_figure(ABC):

@abstractmethod

def area(self):

pass

**figure\_color.py**

class Figure\_color:

def \_\_init\_\_(self, color):

self.color = color

@property

def color(self):

return self.\_color

@color.setter

def color(self, value):

self.\_color = value

**circle.py**

from math import pi

from .geometric\_figure import Geometric\_figure

from .figure\_color import Figure\_color

class Circle(Geometric\_figure):

figure\_type = "Круг"

def \_\_init\_\_(self, radius, color):

self.radius = radius

self.color = Figure\_color(color)

def area(self):

return pi\*(self.radius \*\* 2)

def \_\_repr\_\_(self):

return "{} {} цвета, радиусом {}, площадью {:.2f}".format(

self.figure\_type, self.color.color, self.radius, self.area()

)

**rectangle.py**

from .geometric\_figure import Geometric\_figure

from .figure\_color import Figure\_color

class Rectangle(Geometric\_figure):

figure\_type = "Прямоугольник"

def \_\_init\_\_(self, width, height, color):

self.width = width

self.height = height

self.color = Figure\_color(color)

def area(self):

return self.width \* self.height

def \_\_repr\_\_(self):

return "{} {} цвета, шириной {}, высотой {}, площадью {:.2f}".format(

self.figure\_type, self.color.color, self.width, self.height, self.area()

)

**square.py**

from .rectangle import Rectangle

class Square(Rectangle):

figure\_type="Квадрат"

def \_\_init\_\_(self, side\_length, color):

super().\_\_init\_\_(side\_length, side\_length, color)

def \_\_repr\_\_(self):

return "{} {} цвета, стороной {}, площадью {:.2f}".format(

self.figure\_type, self.color.color, self.width, self.area()

)

**main.py**

from lab\_python\_oop.rectangle import Rectangle

from lab\_python\_oop.circle import Circle

from lab\_python\_oop.square import Square

from colorama import Fore, Style

def main():

n = 1

rectangle = Rectangle(n, n, "синего")

print(Fore.BLUE + str(rectangle))

circle = Circle(n, "зеленого")

print(Fore.GREEN + str(circle))

square = Square(n, "красного")

print(Fore.RED + str(square) + Style.RESET\_ALL)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Экранные формы с примерами выполнения программы**

