Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка Интернет-Приложений»

Отчет по лабораторной работе №2
Вариант №6

Выполнил:
студент группы ИУ5-53Б
Ветошкин Артём
Подпись и дата:
25.10.21

Проверил:
Юрий Евгеньевич Гапанюк
Подпись и дата:

Москва, 2021 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.
6. Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры.
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
    - 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:

circle.py

```
from lab_python_oop.color import Color
from lab_python_oop.geometry_figure import IGeometryFigure

import math

class Circle(IGeometryFigure):
    def __init__(self):
        super().__init__()
        self.radius = 0

    def __init__(self, radius):
        super().__init__()
        self.radius = radius

    def __init__(self, radius, color : Color):
        super().__init__(color)
        self.radius = radius

    def __init__(self, radius, red : int, green : int, blue : int):
        super().__init__(red, green, blue)
        self.radius = radius

    def square(self):
        return math.pi * (self.radius ^ 2)

    def __repr__(self) -> str:
        return 'круг цвета {}; радиусом {}; площадью {}.'.format(
            self.color,
            self.radius,
            self.square()
        )
```

color.py

```
class Color:

    def __init__(self):
        self._R = 0
        self._G = 0
        self._B = 0

    def __init__(self, red : int, green : int, blue : int):
        self._R = red
        self._G = green
        self._B = blue

    @property
    def R(self) -> int:
```

```
        return self._R

    @R.setter
    def R(self, value : int):
        self._R = value

    @property
    def G(self) -> int:
        return self._G

    @G.setter
    def G(self, value : int):
        self._G = value

    @property
    def B(self) -> int:
        return self._B

    @B.setter
    def B(self, value : int):
        self._B = value

    def __repr__(self) -> str:
        return 'RGB ({} {} {})'.format(
            self.R,
            self.G,
            self.B
        )
```

geometry_figure.py

```
from abc import ABCMeta, abstractmethod, abstractproperty
from lab_python_oop.color import Color


class IGeometryFigure():
    __metaclass__ = ABCMeta

    def __init__(self):
        self._color = Color()

    def __init__(self, color : Color):
        self._color = color

    def __init__(self, red : int, green : int, blue : int):
        self._color = Color(red, green, blue)

    @abstractmethod
    def square():
        raise NotImplementedError()

    @property
    def color(self) -> Color:
```

```
        return self._color

    @color.setter
    def color(self, color : Color):
        self._color = color

    @color.setter
    def color(self, red : int, green : int, blue : int):
        self._color = Color(red, green, blue)

    @abstractmethod
    def __repr__(self) -> str:
        raise NotImplementedError()
```

restangle.py

```
from lab_python_oop.color import Color
from lab_python_oop.geometry_figure import IGeometryFigure


class Rectangle(IGeometryFigure):

    def __init__(self):
        super().__init__()
        self.height = 0
        self.width = 0

    def __init__(self, height : int, width : int):
        super().__init__()
        self.height = height
        self.width = width

    def __init__(self, height : int, width : int, color : Color):
        super().__init__(color)
        self.height = height
        self.width = width

    def __init__(self, height : int, width : int, red : int, green : int, blue :
int):
        super().__init__(red, green, blue)
        self.height = height
        self.width = width

    def square(self):
        return self.height * self.width

    def __repr__(self) -> str:
        return 'прямоугольник цвета {}; размером {}x{}; площадью {}.'.format(
            self.color,
            self.height,
            self.width,
            self.square()
        )
```

square.py

```python
from lab_python_oop.color import Color
from lab_python_oop.rectangle import Rectangle


class Square(Rectangle):

    def __init__(self):
        super().__init__()

    def __init__(self, line : int):
        super().__init__(line, line)

    def __init__(self, line : int, color : Color):
        super().__init__(line, line, color)

    def __init__(self, line : int, red : int, green : int, blue : int):
        super().__init__(line, line, red, green, blue)

    def __repr__(self) -> str:
        return 'квадрта цвета {}; со стороной {}; площадью {}.'.format(
            self.color,
            self.height,
            self.square()
        )
```

main.py

```python
from lab_python_oop.rectangle import Rectangle
from lab_python_oop.circle import Circle
from lab_python_oop.square import Square


def main():
    r = Rectangle(6, 6, *(0, 0, 255))
    c = Circle(6, *(0, 255, 0))
    s = Square(6, *(255, 0, 0))
    print(r)
    print(c)
    print(s)

if __name__ == "__main__":
    main()
```

Пример работы:

```
PS C:\Users\vet_v\Desktop\RIP> & c:/Users/vet_v/Desktop/RIP/lab_4/.venv/Scripts/python.exe c:/Users/vet_v/Desktop/RIP/lab_2/main.py
прямоугольник цвета RGB (0 0 255); размером 6x6; площадью 36.
круг цвета RGB (0 255 0); радиусом 6; площадью 12.566370614359172.
квадрта цвета RGB (255 0 0); со стороной 6; площадью 36.
PS C:\Users\vet_v\Desktop\RIP>
```