|  |  |  |
| --- | --- | --- |
| Защищено:  Папин А.В..    "\_\_" \_\_\_\_\_\_\_\_\_\_\_2022 г. |  | Демонстрация:  Папин А.В..    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

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

**базовые компоненты интернет-технологий (БКИТ)**

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

10

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-54Б Папин Алексей | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Гапанюк Ю.Е. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

Москва, МГТУ - 2022

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

**СОДЕРЖАНИЕ ОТЧЕТА**

[1. Цель лабораторной работы 2](#_Toc120618936)

[2. Описание задания. 2](#_Toc120618937)

[3. Листинг программы: 4](#_Toc120618938)

[3.1. Lab\_02.py 4](#_Toc120618939)

[3.2. \_\_init\_\_.py 5](#_Toc120618940)

[3.3. circle.py 5](#_Toc120618941)

[3.4. color.py 6](#_Toc120618942)

[3.5. figure.py 6](#_Toc120618943)

[3.6. rectangle.py 7](#_Toc120618944)

[3.7. square.py 7](#_Toc120618945)

[4. Результаты работы программы: 9](#_Toc120618946)

[4.1. В IDE JetBrains PyCharm 9](#_Toc120618947)

[4.2. Через cmd / powershell 9](#_Toc120618948)

# **Цель лабораторной работы**

Изучение объектно-ориентированных возможностей языка Python.

# **Описание задания.**

Разработать программу для решения биквадратного уравнения.

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»:
10. Определите метод "repr", который возвращает в виде строки основные параметры фигуры, ее цвет и площадь. Используйте метод format - https://pyformat.info/
11. Название фигуры («Прямоугольник», «Квадрат», «Круг») должно задаваться в виде поля данных класса и возвращаться методом класса.
12. 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):
13. Прямоугольник синего цвета шириной N и высотой N.
14. Круг зеленого цвета радиусом N.
15. Квадрат красного цвета со стороной N.
16. Также вызовите один из методов внешнего пакета, установленного с использованием pip.
17. Дополнительное задание. Протестируйте корректность работы Вашей программы с помощью модульного теста.

# **Листинг программы:**

# **Lab\_02.py**

from lab\_python\_oop.rectangle import Rectangle  
from lab\_python\_oop.circle import Circle  
from lab\_python\_oop.square import Square  
  
import sys  
def get\_coef(index, prompt):  
 try:  
 # Пробуем прочитать коэффициент из командной строки  
 coef\_str = sys.argv[index]  
 except:  
 while True:  
 # Вводим с клавиатуры  
 print(prompt)  
 coef\_str = input()  
 # Проверка, есть ли минус числа и нулевой коэффициент?  
 if (int(coef\_str) > 0 and int(coef\_str) < 30):  
 break  
 else:  
 print('Ошибка! Введите номер варианта от 1 до 30')  
 # Переводим строку в действительное число  
 coef = float(coef\_str)  
 return coef  
  
def main():  
 N = get\_coef(1, 'Введите номер вашего варианта по списку журнала')  
 if (N > 0 and N < 30):  
 r = Rectangle("синего", N, N)  
 c = Circle("зеленого", N)  
 s = Square("красного", N)  
 print(r)  
 print(c)  
 print(s)  
 else:  
 print('Ошибка! Введите номер варианта от 1 до 30')  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

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

class FigureColor:  
 *"""  
 Класс «Цвет фигуры»  
 """* def \_\_init\_\_(self):  
 self.\_color = None  
  
 @property  
 def colorproperty(self):  
 *"""  
 Get-аксессор  
 """* return self.\_color  
  
 @colorproperty.setter  
 def colorproperty(self, value):  
 *"""  
 Set-аксессор  
 """* self.\_color = value

# **circle.py**

from .figure import Figure  
from .color import FigureColor  
import math  
  
  
class Circle(Figure):  
 *"""  
 Класс «Круг» наследуется от класса «Геометрическая фигура».  
 """* FIGURE\_TYPE = "Круг"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, r\_param):  
 *"""  
 Класс должен содержать конструктор по параметрам «радиус» и «цвет». В конструкторе создается объект класса «Цвет фигуры» для хранения цвета.  
 """* self.r = r\_param  
 self.fc = FigureColor()  
 self.fc.colorproperty = color\_param  
  
 def square(self):  
 *"""  
 Класс должен переопределять метод, вычисляющий площадь фигуры.  
 """* return math.pi\*(self.r\*\*2)  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета радиусом {} площадью {}.'.format(  
 Circle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.r,  
 self.square()  
 )

# **color.py**

class FigureColor:  
 *"""  
 Класс «Цвет фигуры»  
 """* def \_\_init\_\_(self):  
 self.\_color = None  
  
 @property  
 def colorproperty(self):  
 *"""  
 Get-аксессор  
 """* return self.\_color  
  
 @colorproperty.setter  
 def colorproperty(self, value):  
 *"""  
 Set-аксессор  
 """* self.\_color = value

# **figure.py**

from abc import ABC, abstractmethod  
  
class Figure(ABC):  
 *"""  
 Абстрактный класс «Геометрическая фигура»  
 """* @abstractmethod  
 def square(self):  
 *"""  
 содержит виртуальный метод для вычисления площади фигуры.  
 """* pass

# **rectangle.py**

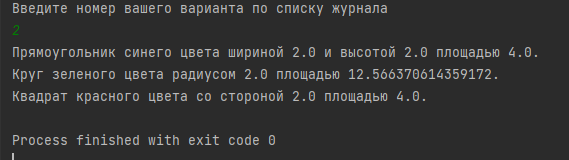
from .figure import Figure  
from .color import FigureColor  
  
  
class Rectangle(Figure):  
 *"""  
 Класс «Прямоугольник» наследуется от класса «Геометрическая фигура».  
 """* FIGURE\_TYPE = "Прямоугольник"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, width\_param, height\_param):  
 *"""  
 Класс должен содержать конструктор по параметрам «ширина», «высота» и «цвет». В конструкторе создается объект класса «Цвет фигуры» для хранения цвета.  
 """* self.width = width\_param  
 self.height = height\_param  
 self.fc = FigureColor()  
 self.fc.colorproperty = color\_param  
  
 def square(self):  
 *"""  
 Класс должен переопределять метод, вычисляющий площадь фигуры.  
 """* return self.width\*self.height  
  
 def \_\_repr\_\_(self):  
 *'''Метод \_\_repr\_\_ в Python выдает текстовое или строковое представление сущности или объекта. '''* return '{} {} цвета шириной {} и высотой {} площадью {}.'.format(  
 Rectangle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.width,  
 self.height,  
 self.square()  
 )

# **square.py**

from .rectangle import Rectangle  
  
  
class Square(Rectangle):  
 *"""  
 Класс «Квадрат» наследуется от класса «Прямоугольник».  
 """* FIGURE\_TYPE = "Квадрат"  
  
 @classmethod  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, side\_param):  
 *"""  
 Класс должен содержать конструктор по параметрам «сторона» и «цвет».  
 """* self.side = side\_param  
 super().\_\_init\_\_(color\_param, self.side, self.side)  
  
 def \_\_repr\_\_(self):  
 return '{} {} цвета со стороной {} площадью {}.'.format(  
 Square.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.side,  
 self.square()  
 )

# **Результаты работы программы:**

# **В IDE JetBrains PyCharm**



# **Через cmd / powershell**

