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:
студент группы ИУ5-33Б
Буров Р. Ю.

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.

Москва, 2025 г.

main.py

```
from lab_python_oop.Rectangle import Rectangle
from lab_python_oop.Circle import Circle
from lab_python_oop.Square import Square

def main():
    rectangle = Rectangle(4, 4, "синий")
    print(rectangle)
    square = Square(4, "зеленый")
    print(square)
    circle = Circle(4, "красный")
    print(circle)

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

Circle.py

```
from lab_python_oop.FigureColor import FigureColor
from colorama import Fore, Style
from lab_python_oop.GeometricFigure import GeometricFigure
import math

class Circle(GeometricFigure):
    name = "круг"

    def __init__(self, radius, color):
        self.radius = radius
        self.color_parametr = FigureColor(color)

    def AreaCalculation(self):
```

```
        return round(math.pi * pow(self.radius, 2), 2)

    def get_name(self):
        return self.name

    def __repr__(self):
        return "{} {} радиусом {}, площадью - {}".format(
            Fore.RED + self.color_parametr.color + Style.RESET_ALL,
            self.get_name(),
            self.radius,
            self.AreaCalculation())
```

FigureColor.py

```
from colorama import Fore, Style
class FigureColor():
    def __init__(self, color):
        self.color = color
```

GeometricFigure.py

```
from abc import ABC, abstractmethod

class GeometricFigure(ABC):

    @abstractmethod
    def AreaCalculation(self):
        pass
```

Rectangle.py

```
from lab_python_oop.FigureColor import FigureColor
from colorama import Fore, Style
from lab_python_oop.GeometricFigure import GeometricFigure


class Rectangle(GeometricFigure):
    name = "прямоугольник"


    def __init__(self, height, width, color):
        self.height = height
        self.width = width
        self.color_parametr = FigureColor(color)

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


    def get_name(self):
        return self.name


    def __repr__(self):
        return '{} {} шириной {} и высотой,  {} площадью {}.'.format(
            Fore.BLUE + self.color_parametr.color + Style.RESET_ALL,
            self.get_name(),
            self.width,
            self.height,
            self.AreaCalculation()
        )
```

Square.py

```
from lab_python_oop.Rectangle import Rectangle
from colorama import Fore, Style


class Square(Rectangle):
    name = "квадрат"

    def __init__(self, length, color):
        self.length = length
        super().__init__(self.length, self.length, color)

    def AreaCalculation(self):
        return pow(self.length, 2)

    def get_name(self):
        return self.name

    def __repr__(self):
        return "{} {} длиной {}, площадью {}".format(
            Fore.GREEN + self.color_parametr.color + Style.RESET_ALL,
            self.name,
            self.length,
            self.AreaCalculation())
```

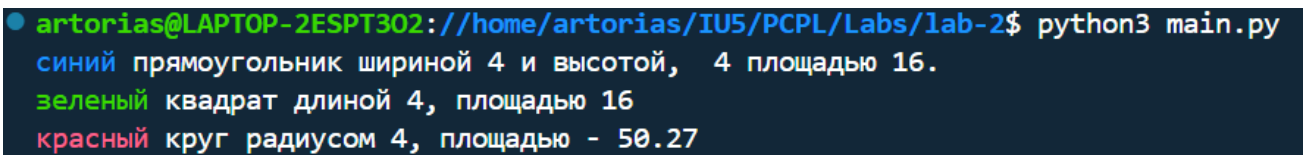
test.py

```
from lab_python_oop.GeometricFigure import GeometricFigure
from lab_python_oop.FigureColor import FigureColor
from lab_python_oop.Square import Square
from lab_python_oop.Circle import Circle
from lab_python_oop.Rectangle import Rectangle
```

```
import unittest
import math
import sys
import os

sys.path.append(os.path.join(os.path.dirname(__file__), 'lab_python_oop'))

class TestRectangle(unittest.TestCase):

    def test_rectangle_creation(self):
        rect = Rectangle(5, 3, "синий")
        self.assertEqual(rect.height, 5)
        self.assertEqual(rect.width, 3)
        self.assertEqual(rect.color_parametr.color, "синий")
        self.assertEqual(rect.name, "прямоугольник")

    def test_rectangle_area(self):
        rect = Rectangle(4, 6, "красный")
        expected_area = 4 * 6
        self.assertEqual(rect.AreaCalculation(), expected_area)

class TestCurcle(unittest.TestCase):

    def test_curcle_creation(self):
        circle = Circle(5, "красный")
        self.assertEqual(circle.radius, 5)
        self.assertEqual(circle.color_parametr.color, "красный")
        self.assertEqual(circle.name, "круг")

    def test_curcle_area(self):
        circle = Circle(3, "зеленый")
```

```
        expected_area = math.pi * 9
        self.assertAlmostEqual(circle.AreaCalculation(),
                               expected_area, places=2)

class TestSquare(unittest.TestCase):

    def test_square_creation(self):
        square = Square(4, "зеленый")
        self.assertEqual(square.length, 4)
        self.assertEqual(square.color_parametr.color, "зеленый")
        self.assertEqual(square.name, "квадрат")
        self.assertEqual(square.height, 4)
        self.assertEqual(square.width, 4)

    def test_square_area(self):
        square = Square(5, "красный")
        expected_area = 25
        self.assertEqual(square.AreaCalculation(), expected_area)

if __name__ == '__main__':
    unittest.main(verbosity=2)
```

Скриншоты работы приложения



```
artorias@LAPTOP-2ESPT3O2://home/artorias/IU5/PCPL/Labs/lab-2$ python3 -m unittest  test.py -v
test_curcle_area (test.TestCurcle.test_curcle_area) ... ok
test_curcle_creation (test.TestCurcle.test_curcle_creation) ... ok
test_rectangle_area (test.TestRectangle.test_rectangle_area) ... ok
test_rectangle_creation (test.TestRectangle.test_rectangle_creation) ... ok
test_square_area (test.TestSquare.test_square_area) ... ok
test_square_creation (test.TestSquare.test_square_creation) ... ok

----------------------------------------------------------------------
Ran 6 tests in 0.001s

OK
```