**Министерство науки и высшего образования Российской Федерации** **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования** **«Московский государственный технический университет** **имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Информатика и системы управления»**

**Кафедра ИУ5 «Системы обработки информации и управления»**

Лабораторная работа №2

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил:

студент группы ИУ5-33Б

Некрасов С. А.

Проверил:

Канев А.И.

2021 г.

**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.
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.](https://docs.python.org/3/library/math.html)
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:**

**Файл main.py**

from lab\_python\_oop.rectangle import Rectangle # прямоугольник  
from lab\_python\_oop.circle import Circle # круг  
from lab\_python\_oop.square import Square # квадрат  
  
from PIL import Image, ImageDraw  
  
  
def get\_picture(r, c, s):  
 imagine = Image.new('RGB', (800, 400), (255, 255, 255)) # размер рамки и цвет фона  
 drawing = ImageDraw.Draw(imagine)  
 scale = 20 # коэффициент  
 drawing.rectangle((10, 10, scale \* r.width, scale \* r.height), fill='blue', outline=(0, 0, 0))  
 drawing.ellipse((200, 10, 200 + 2 \* scale \* c.r, 10 + 2 \* scale \* c.r), fill='green', outline=(0, 0, 0))  
 drawing.rectangle((570, 90, 570 + scale \* s.width, 90 + scale \* s.width), fill='red', outline=(0, 0, 0))  
 # координаты верхнего левого угла, координаты нижнего правого угла, цвет фигуры и рамки  
 drawing.text((50, 350), 'Rectangle', fill='blue')  
 drawing.text((340, 350), 'Ellipse', fill='green')  
 drawing.text((630, 350), 'Square', fill='red')  
 # координаты, название, цвет надписи  
 imagine.show() # рисует фигуры  
  
  
def main():  
 r = Rectangle("синего", 8, 16)  
 c = Circle("зеленого", 8)  
 s = Square("красного", 8)  
 print(r)  
 print(c)  
 print(s)  
  
 get\_picture(r, c, s)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

**Файл circle.py**

from lab\_python\_oop.figure import Figure  
from lab\_python\_oop.color import FigureColor  
import math  
  
  
class Circle(Figure):  
 *"""  
 Класс «Круг» наследуется от класса «Геометрическая фигура».  
 """* FIGURE\_TYPE = "Круг"  
  
 @classmethod # Метод класса — это метод, который привязан к классу, а не к его объекту.  
 # Он не требует создания экземпляра класса  
  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, r\_param): # init-метод, который играет роль конструктора  
 *"""  
 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): #Функция \_\_repr \_\_() возвращает представление объекта.  
 # Это может быть любое допустимое выражение в Python, такое как кортеж, словарь, строка и т.д.  
  
 return '{} {} цвета радиусом {} площадью {}.'.format(  
 Circle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.r,  
 self.square()  
 )

**Файл color.py**

class FigureColor:  
 *"""  
 Класс «Цвет фигуры»  
 """* def \_\_init\_\_(self): # init-метод, который играет роль конструктора  
 self.\_color = None  
  
 @property # property позволяет превратить метод класса в атрибут класса  
 def colorproperty(self):  
 *"""  
 Get-аксессор  
 """* return self.\_color  
  
 @colorproperty.setter # применяется сеттер к методу colorproperty, то есть делаем метод доступным для записи  
 # а не только для чтения  
 def colorproperty(self, value):  
 *"""  
 Set-аксессор  
 """* self.\_color = value

**Файл figure.py**

from abc import ABC, abstractmethod  
  
  
class Figure(ABC):  
 *"""  
 Абстрактный класс «Геометрическая фигура»  
 """* @abstractmethod  
 # Абстрактным называется объявленный, но не реализованный метод  
  
 def square(self): #Self используется для представления экземпляра класса  
 # Благодаря ему можно получить доступ к атрибутам и методам своего класса.  
 *"""  
 содержит виртуальный метод для вычисления площади фигуры.  
 """* pass #возвращает NULL

**Файл rectangle.py**

from lab\_python\_oop.figure import Figure  
from lab\_python\_oop.color import FigureColor  
  
  
class Rectangle(Figure):  
 *"""  
 Класс «Прямоугольник» наследуется от класса «Геометрическая фигура».  
 """* FIGURE\_TYPE = "Прямоугольник"  
  
 @classmethod # Метод класса — это метод, который привязан к классу, а не к его объекту.  
 # Он не требует создания экземпляра класса  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, width\_param, height\_param): # init-метод, который играет роль конструктора  
 *"""  
 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): # Функция \_\_repr \_\_() возвращает представление объекта.  
 # Это может быть любое допустимое выражение в Python, такое как кортеж, словарь, строка и т.д.  
 return '{} {} цвета шириной {} и высотой {} площадью {}.'.format(  
 Rectangle.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.width,  
 self.height,  
 self.square()  
 )

**Файл square.py**

from lab\_python\_oop.rectangle import Rectangle  
  
  
class Square(Rectangle):  
 *"""  
 Класс «Квадрат» наследуется от класса «Прямоугольник».  
 """* FIGURE\_TYPE = "Квадрат"  
  
 @classmethod # Метод класса — это метод, который привязан к классу, а не к его объекту.  
 # Он не требует создания экземпляра класса  
 def get\_figure\_type(cls):  
 return cls.FIGURE\_TYPE  
  
 def \_\_init\_\_(self, color\_param, side\_param): # init-метод, который играет роль конструктора  
 *"""  
 Класс должен содержать конструктор по параметрам «сторона» и «цвет».  
 """* self.side = side\_param  
 super().\_\_init\_\_(color\_param, self.side, self.side)  
  
 def \_\_repr\_\_(self): # Функция \_\_repr \_\_() возвращает представление объекта.  
 # Это может быть любое допустимое выражение в Python, такое как кортеж, словарь, строка и т.д.  
 return '{} {} цвета со стороной {} площадью {}.'.format(  
 Square.get\_figure\_type(),  
 self.fc.colorproperty,  
 self.side,  
 self.square()  
 )

**Экранные формы с примерами выполнения программы**



