МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Н.Э.Баумана

Отчет по лабораторной работе №2

по курсу «Разработка интернет-приложений»

Объектно-ориентированные возможности языка Python

Подготовил

Ионов С.А.

ИУ5-52Б

1. Описание задания
2. Необходимо создать виртуальное окружение и установить в него хотя бы один внешний пакет с использованием pip.
3. Необходимо разработать программу, реализующую работу с классами. Программа должна быть разработана в виде консольного приложения на языке Python 3.
4. Все файлы проекта (кроме основного файла main.py) должны располагаться в пакете lab\_python\_oop.
5. Каждый из нижеперечисленных классов должен располагаться в отдельном файле пакета lab\_python\_oop.
6. Абстрактный класс «Геометрическая фигура» содержит абстрактный метод для вычисления площади фигуры.
7. Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры.
8. 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.
9. Класс «Круг» создается аналогично классу «Прямоугольник», задается параметр «радиус».
10. Класс «Квадрат» наследуется от класса «Прямоугольник». Класс должен содержать конструктор по длине стороны. Для классов «Прямоугольник», «Квадрат», «Круг»:
    * Определить метод "repr", который возвращает в виде строки основные параметры фигуры, ее цвет и площадь.
    * Название фигуры («Прямоугольник», «Квадрат», «Круг») должно задаваться в виде поля данных класса и возвращаться методом класса.
11. 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.
12. Текст программы
13. Файл «Geometric\_figure.py»:

**from** abc **import** ABC, abstractmethod  
  
  
**class** Figure(ABC):  
 @abstractmethod  
 **def** space(self):  
 **pass**

1. Файл «Figure\_color.py»:

**class** Color:  
 **def** \_\_init\_\_(self):  
 self.\_color = **None  
  
 def** get\_color(self):  
 **return** self.\_color  
  
 **def** set\_color(self, value):  
 self.\_color = value  
  
 **def** del\_color(self):  
 **del** self.\_color  
  
 color = property(get\_color, set\_color, del\_color, **"It's the color of the figure"**)

1. Файл «Rectangle.py»:

**import** lab\_python\_oop.Geometric\_figure **as** Geom\_fig  
**import** lab\_python\_oop.Figure\_color **as** Color\_fig  
  
  
**class** Rectangle(Geom\_fig.Figure):  
 \_\_name = **'Прямоугольник'  
  
 def** \_\_init\_\_(self, width, length, color):  
 self.\_width = width  
 self.\_length = length  
 self.\_color = Color\_fig.Color()  
 self.\_color = color  
  
 @staticmethod  
 **def** get\_name():  
 **return** Rectangle.\_\_name  
  
 @property  
 **def** width(self):  
 *"""It's the width of the rectangle"""* **return** self.\_width  
  
 @width.setter  
 **def** width(self, value):  
 self.\_width = value  
  
 @width.deleter  
 **def** width(self):  
 **del** self.\_width  
  
 @property  
 **def** length(self):  
 *"""It's the length of the rectangle"""* **return** self.\_length  
  
 @length.setter  
 **def** length(self, value):  
 self.\_length = value  
  
 @length.deleter  
 **def** length(self):  
 **del** self.\_length  
  
 **def** space(self):  
 *"""It's the space of the rectangle"""* **return** self.\_width \* self.\_length  
  
 **def** \_\_repr\_\_(self):  
 **return '''{} имеет {} цвет, ширину {}   
и длину {}. Его площадь {}'''**.format(Rectangle.get\_name(), self.\_color, self.\_width, self.\_length, self.space())

1. Файл «Circle.py»:

**import** lab\_python\_oop.Geometric\_figure **as** Geom\_fig  
**import** lab\_python\_oop.Figure\_color **as** Color\_fig  
**from** math **import** pi  
  
  
**class** Circle(Geom\_fig.Figure):  
 \_\_name = **'Окружность'  
  
 def** \_\_init\_\_(self, radius, color):  
 self.\_radius = radius  
 self.\_color = Color\_fig.Color()  
 self.\_color = color  
  
 @staticmethod  
 **def** get\_name():  
 **return** Circle.\_\_name  
  
 @property  
 **def** radius(self):  
 *"""It's the radius of the circle"""* **return** self.\_radius  
  
 @radius.setter  
 **def** radius(self, value):  
 self.\_radius = value  
  
 @radius.deleter  
 **def** radius(self):  
 **del** self.\_radius  
  
 **def** space(self):  
 *"""It's the space of the circle"""* **return** float(**'{:.2f}'**.format(pi \* self.\_radius \*\* 2))  
  
 **def** \_\_repr\_\_(self):  
 **return '''{} имеет {} цвет, радиус {}.  
Её площадь {}'''**.format(Circle.get\_name(), self.\_color, self.\_radius, self.space())

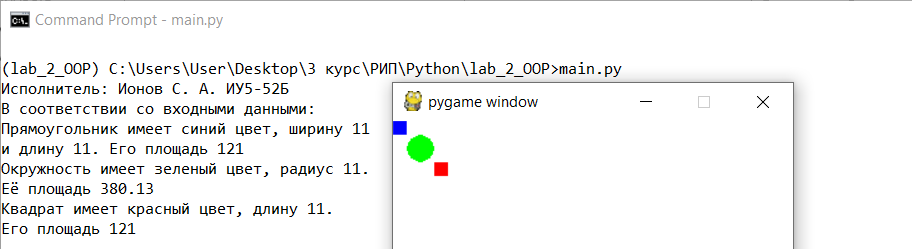
1. Файл «Square.py»:

**import** lab\_python\_oop.Rectangle **as** Rect  
  
  
**class** Square(Rect.Rectangle):  
 \_\_name = **'Квадрат'  
  
 def** \_\_init\_\_(self, length, color):  
 self.\_length = length  
 self.\_width = length  
 self.\_color = color  
  
 @staticmethod  
 **def** get\_name():  
 **return** Square.\_\_name  
  
 **def** \_\_repr\_\_(self):  
 **return '''{} имеет {} цвет, длину {}.  
Его площадь {}'''**.format(Square.get\_name(), self.\_color, self.\_length, self.space())

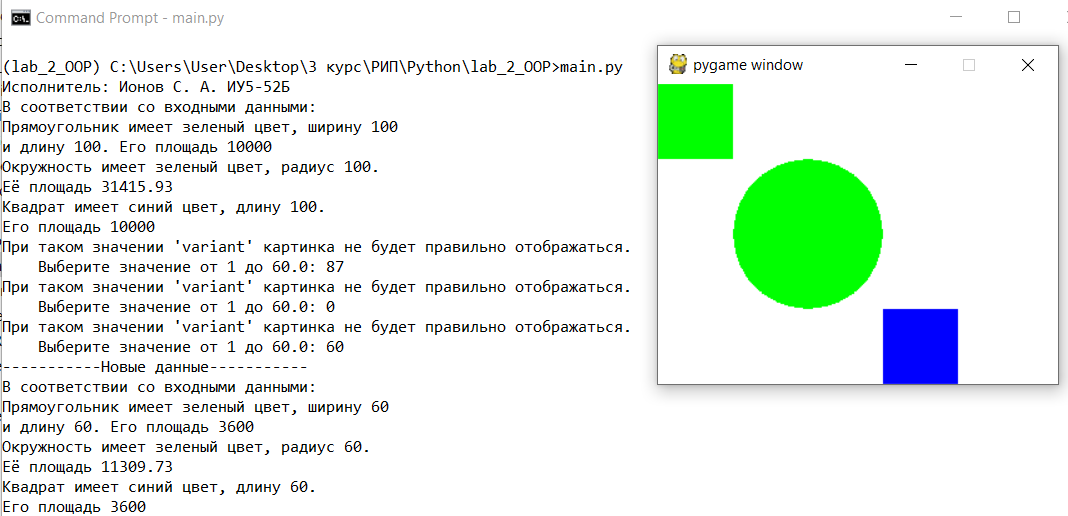
1. Класс «main.py»:

**from** lab\_python\_oop.Circle **import** Circle  
**from** lab\_python\_oop.Rectangle **import** Rectangle  
**from** lab\_python\_oop.Square **import** Square  
**import** pygame *# another module loaded using pip***def** main():  
 *"""It's main function that is called after this module is launched."""* variant = 11 *# variant number in your group* color = [**'синий'**, **'зеленый'**, **'красный'**] *# colors of the figures* call\_my\_packet(variant, color)  
 call\_other\_packet(variant, color)  
  
  
**def** call\_my\_packet(variant, color):  
 *"""Using of my own modules. Variant is number of variant.  
 Color is list of 3rd colors: rectangle, circle, square."""* print(**'В соответствии со входными данными: '**)  
 my\_rect = Rectangle(variant, variant, color[0])  
 print(my\_rect)  
 my\_circle = Circle(variant, color[1])  
 print(my\_circle)  
 my\_square = Square(variant, color[2])  
 print(my\_square)  
  
  
**def** call\_other\_packet(variant, color):  
 *"""Using of another module. Variant is number of variant.  
 Color is list of 3rd colors: rectangle, circle, square."""* flag\_error = **False** *# shows if there is incorrect parameters* fps = 10 *# number of frames per second* start\_pos\_x, start\_pos\_y = 0, 0 *# starting position of print in the window* screen\_size\_x, screen\_size\_y = 320, 240 *# the size of the window* **if** (start\_pos\_x + 4 \* variant) > screen\_size\_x **or** (start\_pos\_y + 4 \* variant) > screen\_size\_y **or** variant < 1:  
 max\_var = min((screen\_size\_x - start\_pos\_x) / 4, (screen\_size\_y - start\_pos\_y) / 4)  
 **while** variant < 1 **or** variant > max\_var:  
 flag\_error = **True** print(**"""При таком значении 'variant' картинка не будет правильно отображаться.  
 Выберите значение от 1 до {}: """**.format(max\_var), end=**""**)  
 **try**:  
 variant = int(input())  
 **except** ValueError:  
 **pass** step = {**'rect\_x'**: start\_pos\_x, **'rect\_y'**: start\_pos\_y, *# coordinates for showing in the window* **'circle\_x'**: start\_pos\_x + 2\*variant,  
 **'circle\_y'**: start\_pos\_y + 2\*variant,  
 **'square\_x'**: start\_pos\_x + 3\*variant, **'square\_y'**: start\_pos\_y + 3\*variant}  
 color\_rgb = {**'rect'**: color[0], **'circle'**: color[1], **'square'**: color[2]} *# for changing text-color to rgb-color* find\_rgb(color\_rgb)  
 **if** flag\_error:  
 print(**"-----------Новые данные-----------"**)  
 call\_my\_packet(variant, color)  
 pygame.init()  
 sc = pygame.display.set\_mode((screen\_size\_x, screen\_size\_y))  
 sc.fill((255, 255, 255))  
 pygame.draw.rect(sc, color\_rgb[**'rect'**], (step[**'rect\_x'**], step[**'rect\_y'**], variant, variant))  
 pygame.draw.circle(sc, color\_rgb[**'circle'**], (step[**'circle\_x'**], step[**'circle\_y'**]), variant)  
 pygame.draw.rect(sc, color\_rgb[**'square'**], (step[**'square\_x'**], step[**'square\_y'**], variant, variant))  
  
 clock = pygame.time.Clock()  
 pygame.display.update()  
 **while True**:  
 clock.tick(fps)  
 **for** i **in** pygame.event.get():  
 **if** i.type == pygame.QUIT:  
 **return  
  
  
def** find\_rgb(color\_rgb):  
 *"""Giving of the conformity text-color and rgb-color.  
 Color\_rgb is dict that keeps the names of colors.  
 This function changes color\_rgb from text to rgb-format"""* name = {**'rect'**: **'прямоугольник'**, **'circle'**: **'окружность'**, **'square'**: **'квадрат'**}  
 **for** i **in** color\_rgb:  
 not\_find = **True  
 while** not\_find:  
 **if** color\_rgb[i] == **'красный'**:  
 color\_rgb[i] = (255, 0, 0)  
 not\_find = **False  
 elif** color\_rgb[i] == **'зеленый' or** color\_rgb[i] == **'зелёный'**:  
 color\_rgb[i] = (0, 255, 0)  
 not\_find = **False  
 elif** color\_rgb[i] == **'синий'**:  
 color\_rgb[i] = (0, 0, 255)  
 not\_find = **False  
 else**:  
 color\_rgb[i] = input(**"""В программе задан неожиданный цвет для фигуры '{}'.   
 Выберите для нее один из цветов: 'красный', 'зеленый', 'синий': """**.format(name[i]))  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 main()

1. Экранные формы с примерами выполнения программы
2. variant = 11; color = [‘синий’, ‘зеленый’, ‘красный’]



1. variant = 100; color = [‘зеленый’, ‘зеленый’, ‘синий’]



1. variant = 11; color = [‘зеленый’, ‘желтый’, ‘синий’]

