**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» Лаб3

|  |  |
| --- | --- |
| Выполнил: | Проверил: |
| студент группы ИУ5-24Б: | преподаватель каф. ИУ5 |
| Здобняков Фёдор | Гапанюк Ю. |
| Подпись и дата: | Подпись и дата: |

Москва, 2023 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. Подробнее про абстрактные классы и методы Вы можете прочитать [здесь.](https://docs.python.org/3/library/abc.html)
6. Класс «Цвет фигуры» содержит свойство для описания цвета геометрической фигуры. Подробнее про описание свойств Вы можете прочитать [здесь.](https://docs.python.org/3/library/functions.html#property)
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»:

Выполнено не по заданию, а как оговорено с преподавателем лабораторных работ

Код программы:

class Unit:  
 def \_\_init\_\_(self, name, damage=150):  
 self.name = name  
 self.damage = damage  
  
  
class Rassa:  
 def \_\_init\_\_(self, rassa='человек'):  
 self.rassa = rassa  
  
  
class Archer(Unit, Rassa):  
 def \_\_init\_\_(self, damage=10, name="безымянный", weapon="arch", rassa="elf"):  
  
 Unit.\_\_init\_\_(self, name, damage)  
 Rassa.\_\_init\_\_(self, rassa)  
 self.damage\_bonus = 5  
 self.damage = damage + self.damage\_bonus  
  
 def attack(self):  
 print("атака с уроном ", self.damage)  
  
 def rass(self):  
 print("Обладает рассой", rassa)  
  
  
class Swordman(Unit):  
 def \_\_init\_\_(self, damage=10, name="безымянный", weapon="sword"):  
 Unit.\_\_init\_\_(self, name, damage)  
 self.weapon = weapon  
 self.damage\_bonus = 10  
 self.damage = damage + self.damage\_bonus  
  
 def attack(self):  
 print("атака с уроном ", self.damage)  
  
  
class Healer(Unit):  
 def \_\_init\_\_(self, \*args, \*\*kwargs):  
 super(Healer,self).\_\_init\_\_(\*args)  
 self.weapon = "staff"  
 self.damage\_bonus = -5  
 # self.damage = damage + (-self.damage\_bonus)  
  
  
class OldHealer(Healer):  
 def \_\_init\_\_(self, damage=10, name="безымянный", weapon="staff"):  
 Healer.\_\_init\_\_(self, damage, name)  
 self.damage = -damage  
 self.damage\_bonus = -5  
 self.name = name  
 self.damage = damage + (-self.damage\_bonus)  
  
 def MagicSuper(self):  
 print("восстановление здоровья ", self.damage \* 3)  
  
  
  
  
ob1 = Swordman()  
print(ob1.damage)  
ob1.attack()  
print(ob1.weapon, '\n')  
  
ob2 = Healer("aaa")  
print("====")  
print(ob2.damage, '\n')  
  
  
ob3 = OldHealer()  
ob3.MagicSuper()

Результаты:

