人工智能程序设计实验报告（二）

类的设计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 桑浩翔 | 学号 | E4211427 | 成绩 |  |

|  |
| --- |
| **1.实验目标：**  1）针对问题描述，合理设计并实现类；  2）形成良好的函数与变量的命名风格；  3）设计合理的测试用例对类进行测试；  **2.实验任务与要求：**  1）编写一个表示宠物的类Pet，它有两个实例属性name和age，分别表示宠物的名字和年龄；  编写一个成员函数，show，输出宠物的名字与年龄；  编写一个成员函数makeSound，输出宠物的叫声。  2）从Pet类派生一个表示狗的类Dog。它有一个新的实例属性breed，是字符串类型，表示狗的品种。  在构造函数中初始化name,age,breed。  改写show函数，输出狗的名称、年龄以及品种。  改写makeSound函数，输出狗的叫声“wang…wang…wang…”  3）编写脚本，测试上述两个类。  4）提交内容包括：  A.本实验报告，请把实验报告文件名中的########替换为你的学号，XXX替换为你的姓名。  B.源代码。源文件命名规则为AIP-2-########.py，其中########替换为你的学号。  例如学号为E02018007的同学，提交的源代码文件命名为：AIP-2-E02018007.PY。  **3.附加题（选做）**  定义一个三维向量 Vector3D 类  1）定义相应的特殊方法实现两个该类对象之间的加减运算  2）实现该类对象与标量的乘、除运算  3）实现该类向量模长的计算。  实现好类以后，需定义 2 个三维向量和 1 个标量进行测试。  样例1：a=[[1,2,3],[2,3,4],[3,4,5]] b=[[1,3,5],[2,4,6],[7,8,9]  a.add(b) ==>[[2,5,8],[4,7,10],[10,12,14]] |
| **3.把程序运行结果截图粘贴在下方。（因语法错误不能生成可执行文件的，无运行结果，不粘贴截图）** |
| **4.把你完成的源文件插入到下方。（插入方法：先把光标置于本段文字的下一段落的开头，然后在菜单栏中选中插入->对象->文件中的文字，在弹出的对话框中选择你编写的源文件）**  bark = 'wang'  class Pet:  def \_\_init\_\_(self, name, age):  self.\_\_name = name  self.\_\_age = age  def show(self):  print("我的名字是%s，我今年 %d 岁！" % (self.\_\_name, self.\_\_age))  def makeSound(self):  print(bark)  class Dog(Pet):  def \_\_init\_\_(self, name, age, breed):  Pet.\_\_init\_\_(self, name, age)  self.\_\_breed = breed  def show(self):  Pet.show(self)  print("我的品种是%s！" % self.\_\_breed)  def makeSound(self):  print((bark + '...') \* 3)  y = Pet('旺财', 3)  y.show()  y.makeSound()  x = Dog('二哈', 2, '哈士奇')  x.show()  x.makeSound()  import math  class Vector3D:  def \_\_init\_\_(self, x, y, z):  self.x = x  self.y = y  self.z = z  def \_\_add\_\_(self, other):  return Vector3D(self.x + other.x, self.y + other.y, self.z + other.z)  def \_\_sub\_\_(self, other):  return Vector3D(self.x - other.x, self.y - other.y, self.z - other.z)  def len(self):  return math.sqrt(self.x \*\* 2 + self.y \*\* 2 + self.z \*\* 2)  def \_\_mul\_\_(self, k):  return Vector3D(self.x \* k, self.y \* k, self.z \* k)  def \_\_truediv\_\_(self, k):  return Vector3D(self.x / k, self.y / k, self.z / k)  def output(self):  print("(%f, %f, %f)" % (self.x, self.y, self.z))  a = Vector3D(1, 2, 3)  b = Vector3D(4, 5, 6)  c = 3  (a + b).output()  (a - b).output()  (a \* c).output()  (a / c).output()  print("a 的长度为 %f。" % a.len()) |

注：亲爱的老师您好，程序或者实验报告遇到任何问题，请联系我，我将在此基础上修改！

Email:[sanghaoxiang@gmail.com](mailto:sanghaoxiang@gmail.com)

QQ:2631510912

参考来源：

python程序设计 董付国

<https://stackabuse.com/overloading-functions-and-operators-in-python/>

<https://www.w3schools.com/python/python_inheritance.asp>

<https://blog.csdn.net/qq_41573234/article/details/82053752?utm_source=app&app_version=5.3.1&code=app_1562916241&uLinkId=usr1mkqgl919blen> by yjm