1. 自己写一个Student类,此类的对象有属性name, age, score, 用来保存学生的姓名,年龄,成绩:  
   1) 写一个函数input\_student读入n个学生的信息,用对象来存储这些信息(不用字典),并返回对象的列表  
   2) 写一个函数output\_student 打印这些学生信息(格式不限)

*#\_author :NineSun  
#data: 2019-09-30***class Student**()**:  
 def \_\_init\_\_**(self,*name*,*age*,*score*)**:** self.\_name**=***name* self.\_age**=***age* self.score**=***score* **@property  
 def name**(self)**:  
 return** self.\_name  
 **@***name*.setter  
 **def name**(self,*name*)**:** self.\_name**=***name* **@property  
 def age**(self)**:  
 return** self.\_age  
  
 **@***age*.setter  
 **def age**(self, *age*)**:** self.\_age **=** *age* **@property  
 def score**(self)**:  
 return** self.\_score  
  
 **@***score*.setter  
 **def score**(self, *score*)**:** self.\_score **=** *score* **def \_\_str\_\_**(self)**:  
 return "姓名："+**self.name**+",年龄："+**self.age**+",成绩："+**self.score  
students**=**[]  
**def input\_student**()**:** n **=** input(**"请输入学生的数量:"**)  
 n**=**int(n)  
 **while** (n**>** 0)**:** name **=** input(**"请输入姓名："**)  
 age **=** input(**"请输入年龄："**)  
 score **=** input(**"请输入成绩："**)  
 student **=** Student(name, age, score);  
 students.append(student)  
 n**-=**1  
**def output\_student**()**:  
 for** student **in** students**:** print(student.\_\_str\_\_())  
input\_student()  
output\_student()

1. 定义一个类Huamn(人类),定义函数input\_human录入信息,main调用显示信息:  
   有三个属性:姓名name,年龄age,家庭住址address (可以省略没有)   
   有方法:show\_info 用来显示人的信息,update\_age用来让这个人的年龄增加一岁

**class Human**()**:  
 def \_\_init\_\_**(self,*name*,*age*,*address*)**:** self.\_name**=***name* self.\_age**=***age* self.\_address**=***address* **@property  
 def name**(self)**:  
 return** self.\_name  
 **@***name*.setter  
 **def name**(self,*name*)**:** self.\_name**=***name* **@property  
 def age**(self)**:  
 return** self.\_age  
  
 **@***age*.setter  
 **def age**(self, *age*)**:** self.\_age **=** *age* **@property  
 def address**(self)**:  
 return** self.\_address  
  
 **@***address*.setter  
 **def address**(self, *address*)**:** self.\_address **=** *address* **def show\_info**(self)**:  
 return "姓名："+**self.name**+",年龄："+**str(self.age)**+"地址："+**self.address  
 **def update\_age**(self)**:** self.age**=**int(self.age)  
 self.age**+=**1  
hus**=**[]  
**def input\_human**()**:** name**=**input(**"姓名："**)  
 age**=**input(**"年龄:"**)  
 address**=**input(**"地址:"**)  
 hu **=** Human(name,age,address)  
 **del** hus[**:**]  
 hus.append(hu)  
input\_human()  
  
**for** hu **in** hus**:** print(hu.show\_info())  
 (hu.update\_age())  
 print(hu.show\_info())

1. 写一个Bicycle(自行车)类,有run(骑行)方法, 调用时显示骑行里程km:  
   再写一个电动自行车类EBicycle继承自Bicycle,添加电池电量valume属性, 同时有两个方法：  
   1. fill\_charge(vol) 用来充电, vol 为电量(度)  
   2. run(km) 方法用于骑行,每骑行10km消耗电量1度,当电量消耗尽时调用Bicycle的run方法骑行并显示骑行结果

**class** Bicycle:  
 **def** run(self,km):  
 print(**"骑行历程%s公里："**%km)  
**class** EBicycle(Bicycle):  
 **def** \_\_init\_\_(self,valume):  
 self.valume=valume  
  
 **def** fill\_charge(self,vol):  
 self.valume += vol  
  
 **def** run(self,km):  
 **while** self.valume>=1:  
 km+=10  
 self.valume-=1  
 print(**"骑行历程%s公里："** %km)

1. 模拟英雄联盟写一个游戏人物的类:  
   (1)创建一个 Game\_role的类.  
   (2) 构造方法中给对象封装name,ad(攻击力),hp(血量).三个属性.  
   (3) 创建一个attack方法,此方法是实例化两个对象,互相攻击的功能:  
   例: 实例化一个对象 盖伦,ad为10, hp为100  
   实例化另个一个对象 剑豪 ad为20, hp为80  
   盖伦通过attack方法攻击剑豪,此方法要完成 '谁攻击谁,谁掉了多少血, 还剩多少血'的提示功能

**class** Game\_role:  
 **def** \_\_init\_\_(self,name,ad,hp):  
 self.ad=ad  
 self.name=name  
 self.hp=hp  
 **def** attack(self,n):  
 n.hp=n.hp-self.ad  
 print(**"%s攻击了%s,%s掉了%s血,还剩%s"**%(self.name,n.name,n.name,self.ad,n.hp))  
  
g1=Game\_role(**'盖伦'**,10,100)  
g2=Game\_role(**'剑豪'**,20,80)  
g1.attack(g2)