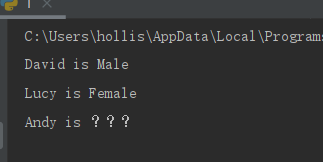
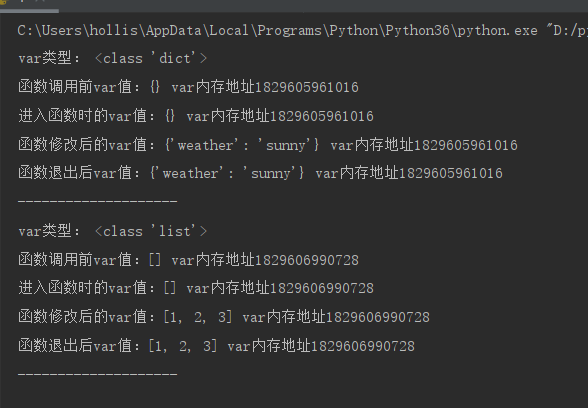
1. 函数的简单入参练习

def my(name,sex='Male'):  
 print("{} is {}".format(name,sex))  
  
my('David') #默认参数  
my('Lucy','Female') #按位置依次传入  
my('Andy',sex='？？？') #指定关键字



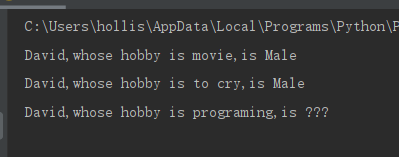
1. 函数可变类型列表，字典入参练习，在函数内改变列表，字典内容后，函数外打印显示发生改变

aDict={}  
aList=[]  
def myfunc(var):  
 print("进入函数时的var值：{} var内存地址{}".format(var,id(var)))  
 if isinstance(var,list):var+=[1,2,3]  
 elif isinstance(var,dict):var['weather']='sunny'  
  
 print("函数修改后的var值：{} var内存地址{}".format(var,id(var)))  
  
test=[aDict,aList]  
for var in test:  
 print("var类型：",type(var))  
 print("函数调用前var值：{} var内存地址{}".format(var,id(var)))  
 myfunc(var)  
 print("函数退出后var值：{} var内存地址{}".format(var, id(var)))  
 print('-'\*20)



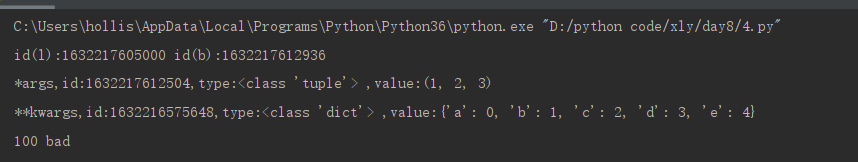
1. 多个缺省参数的传递练习，练习wd\_05\_多个缺省参数.py

def myfunc(name,sex='Male',hobby='movie'):  
 print("{name},whose hobby is {hobby},is {sex}".format(name=name,  
 sex=sex,  
 hobby=hobby))  
  
myfunc('David')  
myfunc('David',hobby='to cry')  
myfunc('David',hobby='programing',sex='???') #关键字参数可以不按位置传递



1. 可变参数练习，元组，字典的传参拆包练习

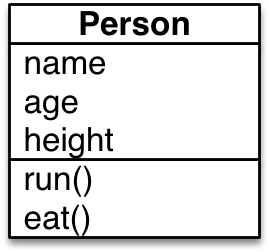
def myfunc(h,\*args,f='good',\*\*kwargs):  
 print("\*args,id:{},type:{} ,value:{}".format(id(args),type(args),args))  
 print("\*\*kwargs,id:{},type:{} ,value:{}".format(id(kwargs),type(kwargs), kwargs))  
 print(h,f)  
  
l=[1,2,3]  
b=dict(zip('a b c d e '.split(),range(5)) )  
  
print("id(l):%d id(b):%d" %(id(l),id(b)))  
myfunc(100,\*l,\*\*b,f='bad')



5、设计一个类，实例化两个对象，然后小明跑步跑完步，会去吃东西

，小美不跑步，小美喜欢吃东西

* **小明** 今年 **18 岁**，**身高 1.75**，每天早上 **跑** 完步，会去 **吃** 东西
* **小美** 今年 **17 岁**，**身高 1.65**，小美不跑步，小美喜欢 **吃** 东西



**class** Person:  
 **def** \_\_init\_\_(self, name, age, height):  
 self.name = name  
 self.age = age  
 self.height = height  
 print(**'{}今年{}岁，身高{}'**.format(name, age, height),end=**''**)  
  
 **def** \_\_str\_\_(self):  
 **return '姓名：%s 年龄：%d 身高：%d'** % (self.name, self.age, self.height)  
  
 **def** \_\_del\_\_(self):  
 **pass  
  
 def** run(self, run\_flage):  
 **if** run\_flage:  
 print(**'每天早上跑完步'**, end=**'，'**)  
 **else**:  
 print(**'%s不跑步'** % self.name, end=**','**)  
  
 **def** eat(self, eat\_flage):  
 **if** eat\_flage == **'like'**:  
 print(**'%s喜欢吃东西'** % self.name)  
 **else**:  
 print(**'会去吃东西'**)  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 xiaoming = person(**'小明'**, 18, 1.75)  
 xiaoming.run(1)  
 xiaoming.eat(**'will'**)  
  
 xiaomei = Person(**'小美'**, 17, 1.65)  
 xiaomei.run(0)  
 xiaomei.eat(**'like'**)

**Result:**

小明今年18岁，身高1.75每天早上跑完步，会去吃东西

小美今年17岁，身高1.65小美不跑步,小美喜欢吃东西

6、设计一个类，实例化1个对象，会实现下面两种行为

**需求**

* 一只 **黄颜色** 的 **狗狗** 叫 **大黄**
* 具有 **汪汪叫 行为**
* 具有 **摇尾巴 行为**

**class** Dog:  
 **def** \_\_init\_\_(self, name, color):  
 self.name = name  
 self.color = color  
 print(**'一只{}颜色的狗狗叫{}'**.format(name, color))  
 **def** \_\_str\_\_(self):  
 **return '这是一条叫做%s的%s色的狗'** % (self.name, self.color)  
  
 **def** \_\_del\_\_(self):  
 **pass  
  
 def** call(self):  
 print(**'汪汪汪'**)  
  
 **def** shake(self):  
 print(**'摇摇尾巴'**)  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 dahuang = Dog(**'大黄'**, **'黄'**)  
 dahuang.call()  
 dahuang.shake()

**Result:**

一只黄颜色的狗狗叫大黄

汪汪汪

摇摇尾巴

7、使用dir查看类里的内容，是否有类属性，是否有对象方法，练习\_\_init\_\_，\_\_str\_\_还有\_\_del\_\_

**class** Dog:  
 name = **''** color = **''  
  
 def** \_\_init\_\_(self, name, color):  
 Dog.name = name  
 Dog.color = color  
 print(**'一只{}颜色的狗狗叫{}'**.format(color, name))  
  
 **def** \_\_str\_\_(self):  
 **return '这是一条叫做%s的%s色的狗'** % (self.name, self.color)  
  
 **def** \_\_del\_\_(self):  
 **pass  
  
 def** call(self):  
 print(**'汪汪汪'**)  
  
 **def** shake(self):  
 print(**'摇摇尾巴'**)  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 print(dir(Dog))

**Result:**

['\_\_class\_\_', '\_\_del\_\_', '\_\_delattr\_\_', '\_\_dict\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_le\_\_', '\_\_lt\_\_', '\_\_module\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', '\_\_weakref\_\_', 'call', 'color', 'name', 'shake']

8、上课已写

9、练习私有属性和私有方法

**class** Person:  
 **def** \_\_init\_\_(self, name, age, height):  
 self.name = name  
 self.\_\_age = age  
 self.height = height  
 print(**'{}今年{}岁，身高{}'**.format(name, self.\_\_age, height))  
  
 **def** \_\_str\_\_(self):  
 **return '姓名：%s 年龄：%d 身高：%d'** % (self.name, self.\_\_age, self.height)  
  
 **def** \_\_del\_\_(self):  
 **pass  
  
 def** \_\_show\_age(self):  
 print(**'我今年%s'** %self.\_\_age)  
  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 xiaomei = Person(**'小美'**, 17, 1.65)  
 xiaomei.\_\_age = 18  
 *# xiaomei.\_\_show(self)* print(xiaomei)

**Result:**

小美今年17岁，身高1.65

姓名：小美 年龄：17 身高：1

10、练习继承、使用super扩展父类，多重继承

**class** A:  
 **def** test(self):  
 print(**'A'**)  
  
**class** B:  
 **def** test(self):  
 print(**'B'**)  
  
**class** C(B,A):  
 **def** test(self):  
 print(**'C'**)  
 super().test()  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 c = C()  
 c.test()

**Result:**

C

B