变量：变化的量

可以变化存储数据的量：变量

Python中这些变量随意定义？

Python规则：

1. 可以使用a~z，A~Z，52个
2. 0~9，10个数字
3. \_

总共：63个

Python命名规范（道德：虽然违反，但不犯法）

1. 数字不能开头
2. 不能使用python中的关键字：help(“keywords”)查询有多少个关键字 35个关键字

False class from or

None continue global pass

True def if raise

and del import return

as elif in try

assert else is while

async except lambda with

await finally nonlocal yield

break for not

1. python3 虽然提供中文命名法，但是禁止使用。

center，自动居中，空缺部分以自定义进行补全

例如：print(“欢迎来到Jason水果商城”.center(70),”-“))

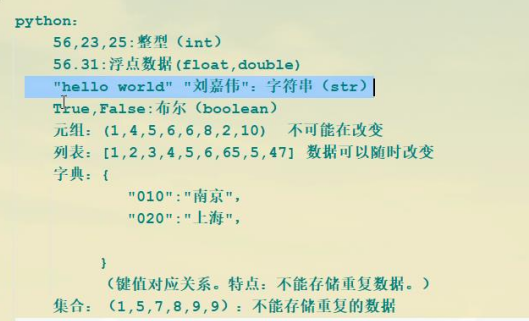
Print(---------------Jason水果商城--------------)

Int能使小数变整数，不会进行四舍五入

例如int(3.6)=3

***name=input("请输入姓名:")  
id=input("请输入身份证号:")  
age=int(input("请输入年龄:"))  
sex=input("请输入性别:")  
height=int(input("请输入身高:"))  
weight=int(input("请输入体重:"))  
  
 info='''  
--------------个人信息--------------  
姓名:{name}  
年龄:{age}  
身份证:{id}  
身高:{height}  
体重:{weight}  
性别:{sex}  
--------------------------------------  
'''  
print(info.format(name=name,age=age,id=id,height=height,weight=weight,sex=sex))***

/t 表示4个空格



二进制：0 1

表现：ob

例子：ob1011001

八进制：0 1 2 3 4 5 6 7

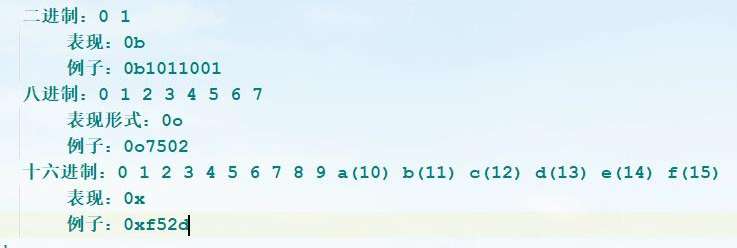
表现形式：0o

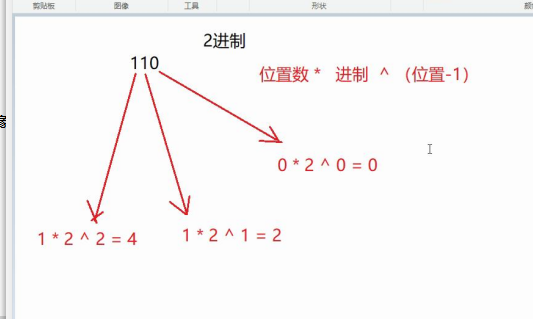
例子：0o7502

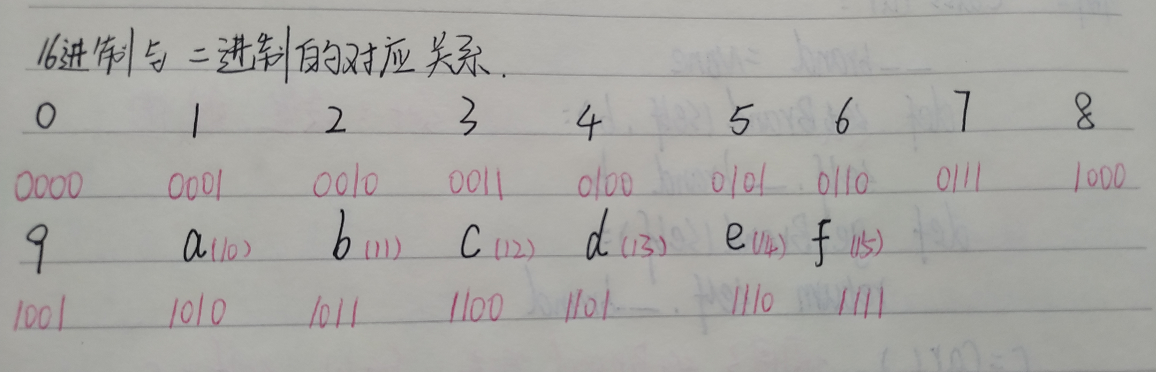
十六进制：0 1 2 3 4 5 6 7 8 9 a(10) b(11) c(12) d(13) e(14) f(15)

表现：0x

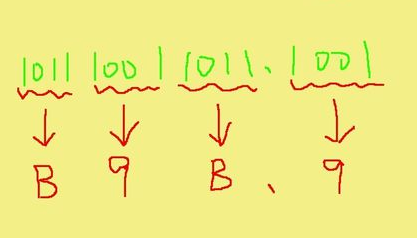
例子：0xf52d



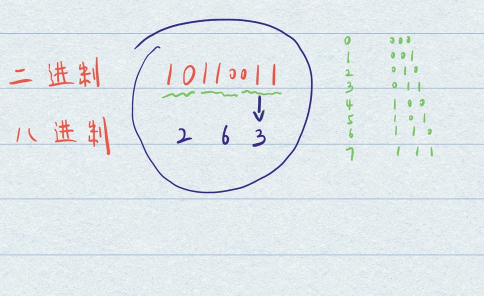


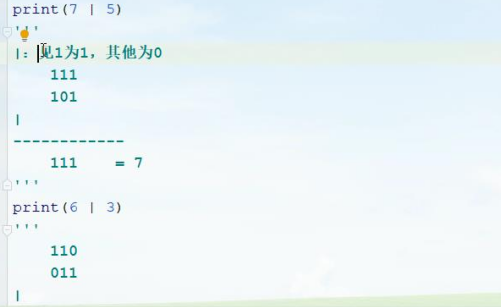


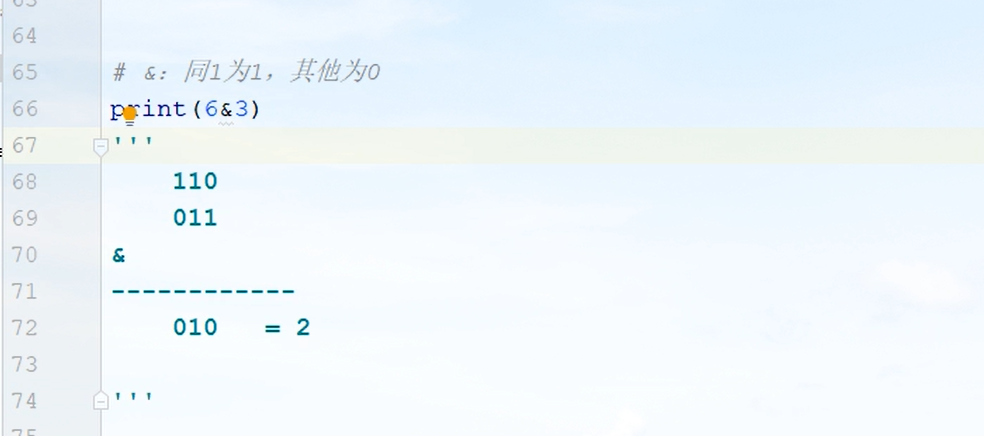
每四位表示一个十六位



二进制与八进制之间的关系：每三位二进制表示一个八进制



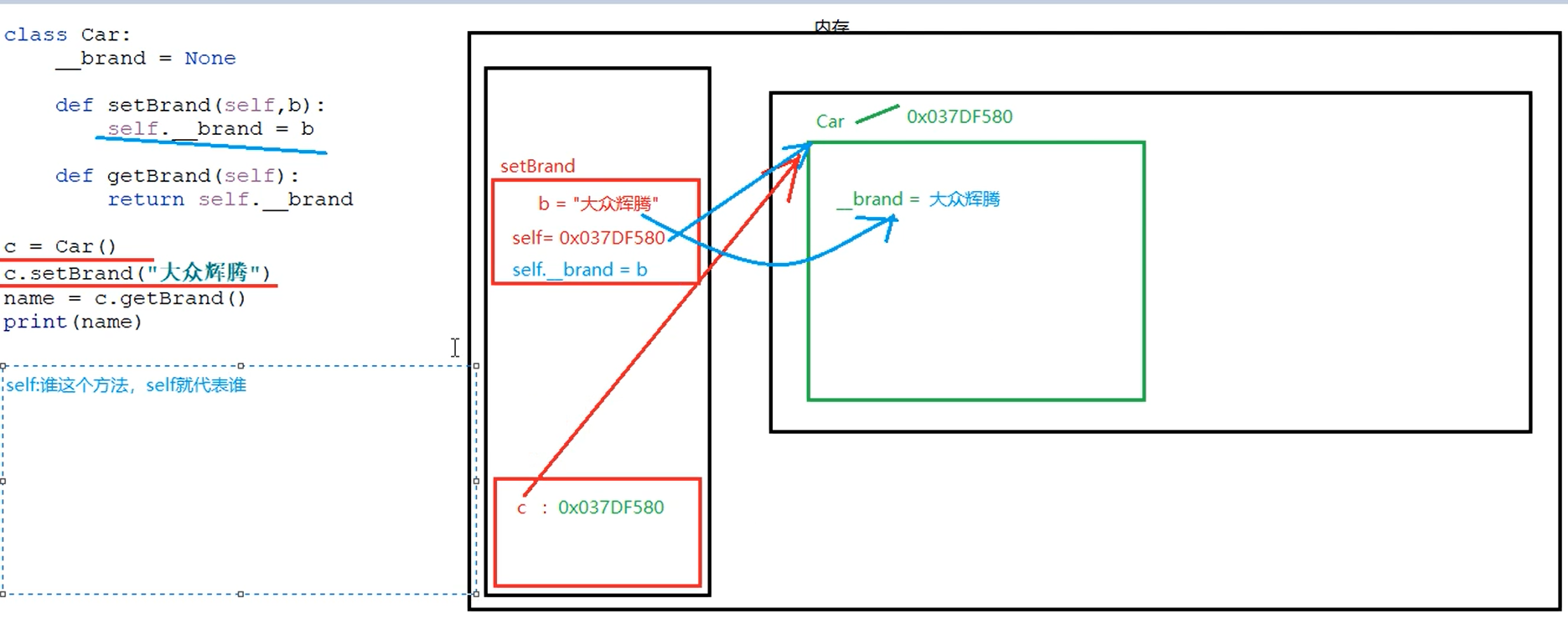






Self：谁调用我，我就代表谁（谁调用这个方法，self就代表谁）

class Car:  
 \_\_brand=None  
  
 def setBrand(self,b):  
 self.\_\_brand=b  
  
 def getBrand(self):  
 return self.\_\_brand  
  
c=Car() ——>调用了setBrand 方法，所以 self=c  
c.setBrand(**"大众辉腾"**) 在pycharm里打印出来的  
name=c.getBrand() C=0X037DF580(十六进制)  
print(name)  
print(c)



import time  
class Animal(object):  
 \_\_color=**""** \_\_weight=**""** \_\_age=**""** def \_\_init\_\_(self,color,weight,age):  
 self.\_\_color=color  
 self.\_\_weight=weight  
 self.\_\_age=age  
  
 def setColor(self,color):  
 self.\_\_color=color  
 def getColor(self):  
 return self.\_\_color  
  
 def setWeight(self,weight):  
 self.\_\_weight=weight  
 def getWeight(self):  
 return self.\_\_weight  
  
 def setAge(self,age):  
 self.\_\_age=age  
 def getAge(self):  
 return self.\_\_age  
  
class Dog(Animal):  
 \_\_brand=**""** def \_\_init\_\_(self,color,weight,age,brand):  
 super().\_\_init\_\_(color,weight,age) *#三个属性交给父类进行初始化 对象.方法（） super() 指向Animal（父类）所以super()=Animal \_\_init\_\_和上面的\_\_init\_\_方法一样，因为上面*(color,weight,age)*前使用了\_\_init\_\_方法，所以这里也使用，这就是子类继承父类*

self.\_\_brand=brand  
  
 def setBrand(self,brand):  
 self.\_\_brand=brand  
 def getBrand(self):  
 return self.\_\_brand  
  
 def Show(self):  
 for i in range(3): *#叫的次数，3次* print(**"汪"**,end=**""**)  
 time.sleep(1) *#叫的间隔时间 1秒* print(**"我是"**,self.getColor(),**"的狗"**,  
 **"我的体重是"**,self.getWeight(),**"kg"**,  
 **"我的最长寿命是"**,self.getAge(),**"年"**,  
 **"我是："**,self.getBrand())  
  
d=Dog(**"黄色"**,10,10,**"犬科"**)  
d.Show()