Python复合数据类型介绍\*

（供学有余力者参考）

**1. tuple类型（元祖类型）**

某用户预订了商品编号为“ID0010230”、单价为15.68元，数量为36，可将这3个不同类型的简单数据组织成一个复合数据类型——元祖。记作：

BookInfo0=("ID0010230",15.68,36)

另一用户预订了商品编号为“ID2315937”、单价为20元，数量为2，可记作：

BookInfo1=("ID2315937",20,2)

>>> BookInfo0=("ID0010230",15.68,36)

>>> type(BookInfo0)

<class 'tuple'> #返回元祖类型

>>> BookInfo1=("ID2315937",20,2)

>>> BookInfo1[1]

20 #返回元祖BookInfo1中索引为1的项的值

计算今天的订单总金额：

>>> BookInfo0[1]\*BookInfo0[2]+BookInfo1[1]\*BookInfo1[2]

604.48

**2. set类型（集合）**

>>> a=set() #定义变量a为空集合

>>> a

set()

>>> type(a)

<class 'set'> #返回集合类型

>>> a.add(1) #集合a中增加元素1

>>> a

{1}

>>> a.remove(1) #集合a中删除元素1

>>> a

set()

>>> a={1,2,3,4,5,6} #定义集合变量a

>>> len(a) #求集合变量a的长度（即集合中元素个数）

6

>>> b={2,3,7,9}

>>> a&b #求两集合的交集

{2, 3}

>>> a|b #求两集合的并集

{1, 2, 3, 4, 5, 6, 7, 9}

>>> a-b #求两集合的差集：a-b=a-(a&b)

{1, 4, 5, 6}

>>> b-a #求两集合的差集：b-a=b-(b&a)

{9, 7}

>>> 3 in a #测试元素3是否属于集合a

True

>>> {8}<=b #测试集合{8}是否包含于集合b

False

>>> {7}<b #测试集合{7}是否真包含于集合b

True

今天的订单汇总，可以定义为一个集合(集合里的项称为元素，彼此之间没有顺序)：

BookSet={BookInfo0,BookInfo1}

>>> BookSet={BookInfo0,BookInfo1}

>>> type(BookSet)

<class 'set'> #返回集合类型

>>> BookSet

{('ID2315937', 20, 2), ('ID0010230', 15.68, 36)} #返回集合的值

计算今天的订单总金额：

>>> total=0

>>> for i in BookSet:

total+=i[1]\*i[2]

>>> total

604.48

**3. list类型（列表）**

今天的订单汇总，也可以按订单产生的新后顺序组成一个列表(列表里的项是有顺序编号的)：

BookList=[BookInfo0, BookInfo1]

>>> BookList=[BookInfo0, BookInfo1]

>>> type(BookList)

<class 'list'> #返回列表类型

>>> BookList[0]

('ID0010230', 15.68, 36)

>>> type(BookList[0])

<class 'tuple'>

>>> type(BookList[0][0])

<class 'str'>

>>> BookList[1]

('ID0010359', 0.68, 5)

>>> BookList[0][1]\*BookList[0][2]+BookList[1][1]\*BookList[1][2]

604.48 #返回计算结果

**4. dict类型（字典类型）**

>>> a={"盐酸":"HCl","氢氧化钠":"NaOH","氯化钠":"NaCl","水":"H2O"} #定义字典a

>>> a

{'盐酸': 'HCl', '氢氧化钠': 'NaOH', '氯化钠': 'NaCl', '水': 'H2O'}

>>> type(a)

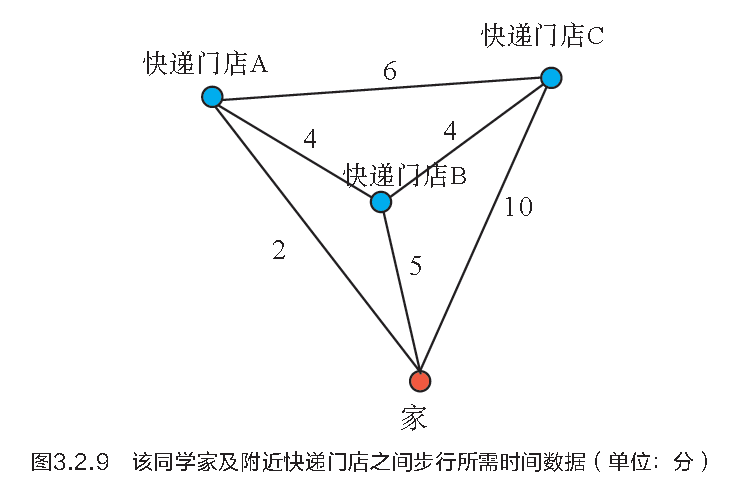
<class 'dict'> #返回字典类型

>>> a["盐酸"] #求字典中键"盐酸"的值

'HCl' #返回字典中键"盐酸"的值

>>> a["盐酸"]+'+'+a["氢氧化钠"]+"="+a["氯化钠"]+'+'+a["水"]

'HCl+NaOH=NaCl+H2O' #返回化学反应方程式



>>>G={'H':{'A':2,'B':5,'C':10},'A':{'H':2,'B':4,'C':6},'B':{'H':5,'A':4,'C':4},'C':{'H':10,'A':6,'B':4}} #上图可用字典变量G完美表示

>>> G['H']['A'] #求顶点“家”（'H'）与“快递门店A”（'A'）之间的用时

2

>>> G['C']['B']

4