# Python学习笔记

高等理工学院 19231188 李健健

shift+r 🡪 在终端运行

1. 字符串处理
2. 删除空白：str.lstrip() 和 str.rstrip() 分别确保字符串前后无空白(left 和right)

str.strip()确保两端无空白

这种删除只是暂时的，只是返回了一个修改的字符串，要真正地更改，还需要将修改后的字符串保存到原来的字符串之中。

1. 字符串强制转换：str(一个数)
2. 对列表元素的相关操作

列表的对应关系是 index🡨🡪value

1. 确定列表的长度：len(list)
2. 返回一个去重的列表：set(list)
3. 在列表的结尾中添加元素，使用list.append(value)
4. 在列表中插入元素，使用list.insert(index,value)
5. 已知索引删除元素，

del list[index]

list.pop(index)，返回被删除的元素的值

1. 已知值来删除元素，list.remove(value)
2. 对列表永久的排序：list.sort()——按照字母顺序排序，且无法逆转
3. 对列表暂时的排序： sorted list
4. 反转列表：list.reverse()
5. 操作列表
6. 使用函数range()生成一系列数字
7. 使用range()生成数字列表

如：list1=list(range(1,5))，

list2=lsit(range(1,20,5))，这是在指定步长

1. 解析列表

squares=[value\*\*2 for value in range(1,5)]

将得到squares=[1,4,9,16,25]

1. 得到最大最小值：

max(range(1,4))

min(squares)

1. 切片与差一现象

range(1,4)的数字们是：1，2，3

list1=list(range(1,4))，list[1:3]是[1,2]

list[:3]就是list[0:3]，list[1:]就是list[1:len(list)]，

list[-3:]总能提取出最后的三个元素

list[:]就是输出整个列表

1. 列表的复制：

应该使用： list2=list1[:]，这样就有两个列表

如果使用：list2=list1[]，这样只是相当于C语言中的指针，对list2的操作就是对list1的操作

1. 元组（tuple）
2. 不能单独对元组中的元素修改（赋值），只能将整个元组重新赋值（覆盖）

如：tuple1=(1,2,3)

tuple[0]=4 🡪 error

tuple1=(2,3,4) 🡪 right

1. 元组也是可迭代的(iterative)
2. 条件判断if语句

if条件将返回一个布尔值：True 或 False

1. 判断字符串 相等/不相等，是区分大小写的
2. 判断数字 相等/不相等/大于（等于）/小于（等于）
3. 简易逻辑：
4. 用 and 来判断条件：

a>20 and b<30

1. 用 or 来判断条件：

a>20 or b<=30

1. 检查某个元素是否在/不在列表中：
2. print(value in list)，在的话就输出 True
3. print(value not in list)
4. if-elif-else 结构：

满足条件就直接跳出，如果有多个条件要测试，则需要使用多个if语句

1. 字典(dictionary)

字典是 key-value结构，dictionary={‘name’:’lihua’,‘color’:’green’}

1. 添加键值对——直接给新的key赋值上value

dictionary[‘xPosition’]=25

1. 修改值：直接dict[key]=newValue
2. 永久删除键值对：del dict[key]
3. 字典中的遍历
4. 键值对的遍历：

for k,v in dictionary.items():

items()方法返回一个键值对表，for循环依次将其存储到指定的两个变量中。这种遍历的顺序和存储时的顺序时不一定想同的。

1. 键的遍历

for key in dictionary.keys():

for key in (sorted(dictionary.keys()))

1. 值的遍历

for value in dictionary.values():

1. 嵌套

users = {

'aeinstein': {

'first': 'albert',

'last': 'einstein',

'location': 'princeton',

},

'mcurie': {

'first': 'marie',

'last': 'curie',

'location': 'paris',

},

}

1. 交互

input()函数：

接受一个参数，将其显示在终端上，并接受一个输入字符串，把它存起来：

name=input(“What’s your name? ”)

1. 循环
2. 有while determine\_statements :

…

1. break：直接跳出循环
2. continue：回到开头
3. 列表中的元素的while：

while list :

接下来的语句，可以直接对list中的value 进行操作

当 list 为空时，相当于 while 0 .

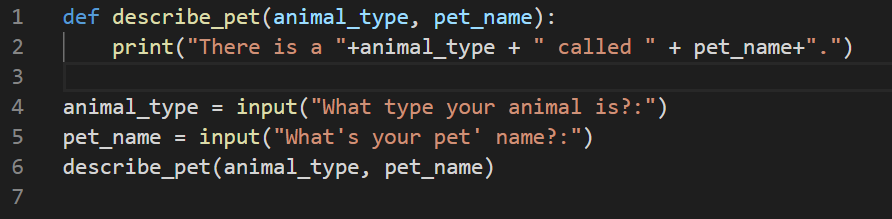
1. 函数

基本语法：

def funcname(parameter\_list):

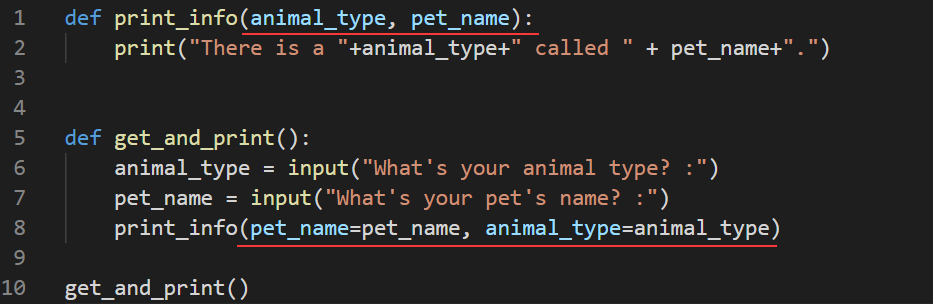
    pass

1. 传递参数
2. 位置实参（positional arguments），实参和形参的位置一一对应



1. 关键字实参

将“参数=变量名”放在函数名后面的那个括号里



1. 默认值

在定义函数时，直接将数值赋给参数，这样当调用函数时，可以两个参数都穿进去，也可以只传那个未被默认的参数。

注意的是，此时Python有一个默认的语法：

default argument follows non-default argument

所以要把未添加默认值的参数放在前面，添加了默认值的函数放在后面，否则会报错。

1. 返回值

函数结尾return sth 就好，可以是字符串、整数或字典

1. 函数中对列表的操作

一般情况下，调用函数操作列表，会修改列表，而function(list[:])，可以避免函数修改列表，而是传了一个列表的副本进去。

1. 传递任意数量的实参
2. C语言中也有类似的语句，def function(\*tuple):

将传进来的数个参数存储到元组里

1. 结合位置实参和任意数量实参def make\_pizza(size, \*toppings):
2. 使用任意数量关键词参数：def build\_profile(first, last, \*\*user\_info):

用两个\*来让python创建一个空字典user\_info，并将接受到的参数封装的这个字典里。

1. 将函数存储在模块中
2. 通过模块导入函数
3. 导入整个模块

新建一个module.py，然后import module

调用模块中的函数要写作：module.fuction(...)

1. 导入模块中的特定函数

from module import function\_1, function\_2 , function\_3

这种调用方法就不用打“ . ”来调用被引入的函数

1. 通过as来给import的函数/模块起别名
2. 给模块起别名import module as m
3. 给模块中的函数起别名form module import function as func
4. 导入模块中的所有函数

form module import \*