**Daily Report**

30th January, 2018

1. **Map（）函数**

map()是 Python 内置的高阶函数，它接收一个函数 f 和一个 list，并通过把函数 f 依次作用在 list 的每个元素上，得到一个新的 list 并返回

map()函数不改变原有的 list，而是返回一个新的 list

由于list包含的元素可以是任何类型，因此，map() 不仅仅可以处理只包含数值的 list，事实上它可以处理包含任意类型的 list，只要传入的函数f可以处理这种数据类型

1. **Reduce()函数**

reduce()函数接收的参数和 map()类似，一个函数 f，一个list，但行为和 map()不同，reduce()传入的函数 f 必须接收两个参数，reduce()对list的每个元素反复调用函数f，并返回最终结果值

例如，编写一个f函数，接收x和y，返回x和y的和：

def f(x, y):

return x + y

调用 reduce(f, [1, 3, 5, 7, 9])时，reduce函数将做如下计算：

先计算头两个元素：f(1, 3)，结果为4；

再把结果和第3个元素计算：f(4, 5)，结果为9；

再把结果和第4个元素计算：f(9, 7)，结果为16；

再把结果和第5个元素计算：f(16, 9)，结果为25；

由于没有更多的元素了，计算结束，返回结果25。

reduce()还可以接收第3个可选参数，作为计算的初始值。如果把初始值设为100，计算：

reduce(f, [1, 3, 5, 7, 9], 100)

结果将变为125，因为第一轮计算是：

计算初始值和第一个元素：f(100, 1)，结果为101

1. **Filter（）函数**

filter()函数接收一个函数 f 和一个list，这个函数 f 的作用是对每个元素进行判断，返回 True或 False，filter()根据判断结果自动过滤掉不符合条件的元素，返回由符合条件元素组成的新list

s.strip(rm) 删除 s 字符串中开头、结尾处的 rm 序列的字符。

当rm为空时，默认删除空白符（包括'\n', '\r', '\t', ' ')，如下：

a = ' 123'

a.strip()

结果： '123'

a='\t\t123\r\n'

a.strip()

结果：'123'

1. **Sort（）函数**

sorted()函数可对list进行排序

sorted()也是一个高阶函数，它可以接收一个比较函数来实现自定义排序，比较函数的定义是，传入两个待比较的元素 x, y，如果 x 应该排在 y 的前面，返回 -1，如果 x 应该排在 y 的后面，返回 1。如果 x 和 y 相等，返回 0

>>> sorted([36, 5, 12, 9, 21], reversed\_cmp)

[36, 21, 12, 9, 5]

1. **闭包**

内层函数引用了外层函数的变量（参数也算变量），然后返回内层函数的情况，称为闭包（Closure）

闭包的特点是返回的函数还引用了外层函数的局部变量，所以，要正确使用闭包，就要确保引用的局部变量在函数返回后不能变

1. **匿名函数**

例如：

>>> map(lambda x: x \* x, [1, 2, 3, 4, 5, 6, 7, 8, 9])

[1, 4, 9, 16, 25, 36, 49, 64, 81]

通过对比可以看出，匿名函数 lambda x: x \* x 实际上就是：

def f(x):

return x \* x

关键字lambda 表示匿名函数，冒号前面的 x 表示函数参数

1. **Decorate**

decorator 本质上就是一个高阶函数，它接收一个函数作为参数，然后，返回一个新函数

使用 decorator 用Python提供的 @ 语法，这样可以避免手动编写 f = decorate(f) 这样的代码

@log 写死了只含一个参数的返回函数

要让 @log 自适应任何参数定义的函数，可以利用Python的 \*args 和 \*\*kw，保证任意个数的参数总是能正常调用

1. **中偏函数**

functools.partial就是帮助我们创建一个偏函数的，不需要我们自己定义int2()，可以直接使用下面的代码创建一个新的函数int2

>>> import functools

>>> int2 = functools.partial(int, base=2)

>>> int2('1000000')

64

>>> int2('1010101')

85