函数

一、 函数的创建和调用

1、 创建函数：

```
def function([parameterlist]):
    [functionbady]
```

2、 调用函数:

```
function([parametersvalue])
```

二、 参数传递

1、 形式参数与实际参数：

根据实际参数的类型不同，可以分为值传递和引用传递两种方式。值传递改变形参的值，但不改变实参的值；而引用传递在改变形参的同时实参发生改变。当实际参数为不可变对象（str）时，进行值传递；当实参为可变对象（list）时，进行引用传递。

```
>>> def demo(obj):
...     print('原值',obj)
...     obj+=obj
...
...
>>> mot='测试样例'
>>> demo(mot)
原值 测试样例
>>> print(mot)
测试样例
>>> mot=['测','试','样','例']
>>> demo(mot)
原值 ['测', '试', '样', '例']
>>> print(mot)
['测', '试', '样', '例', '测', '试', '样', '例']
```

2、 位置参数：

调用函数时，实参和形参的数量和位置必须一一对应。如果觉得牢记参数位置麻烦，可以使用关键字参数来调用函数。例如 def function(a,b,c) 调用时可以 function(b=1,a=2,c=3)

3、可变参数：

定义可变参数时主要有两种方式：*parameter 和**parameter

(1)*parameter：

这种形式表示接受任意多个实际参数并将其放入一个元组中（不可改变其值）

```
def func(*args):
    j=0
    for i in args:
        j+=i
    print(j)
func(1,3,4,5,3)
```

(2)**parameter：

这种形式表示接受任意多个类似关键字参数一样的显式赋值的实际参数，并将其放入一个字典中。

```
def printf(**args):
    for i,j in args.items():
        print(i,j)

printf(ind='射手座',t='天蝎座')
```

三、 函数的高级话题

1、 lambda 函数

```
lambda argument1,argument2…argument n:expression using argument
```

lambda 返回了一个函数，可以选择性的赋值给另一个变量名。Lambda 仅限于表达式，其功能比通常

的 def 功能要小，lambda 是为了编写简单的函数而设计的。

Eg:

f=lambda x,y,z:x+y+z

f(1,2,3)

2、 map 函数

map 函数会对一个序列对象的每一个元素应用被传入的函数，并且返回了包含所有函数调用结果的列表。例如：

```
>>> def inc(x):
...     return x+10
...
>>> list(map(inc,[1,2,3,4]))
[11, 12, 13, 14]
```

事实上，由于 map 期待传入一个函数，它恰好就是 lambda 通常出现的位置之一。

```
>>> list(map(lambda x,y:x**y,[1,2,3],[5,6,7]))
[1, 64, 2187]
```

3、 filter 和 reduce：函数式编程工具

函数式编程工具就是对序列应用一些函数的工具。map 就是简单的函数式编程工具的代表。

filter 函数用于通过某一函数测试过滤出一些元素，这种内置函数可以通过 for 循环和条件判断来等效，但是通过 filter 这样内置函数可以效率较快地运行。

```
>>> list(filter(lambda x:x>0,range(-5,5)))
[1, 2, 3, 4]
```

reduce 函数接受一个迭代器处理，返回一个单一的结果。

```
>>> from functools import reduce
>>> reduce(lambda x,y:x+y,[1,2,3,4])
10
```

reduce 在内部一般性运作是：

```
>>> def Reduce(function,sequence):
...     tally=sequence[0]
...     for next in sequence[1:]:
...         tally=function(tally,next)
...     return tally
...
>>> Reduce(lambda x,y:x*y,[1,2,3,4])
24
```