lambda 表达式

作用:

创建一个匿名函数对象

同def 类拟,但不提供函数名

语法:

lambda [函数形参列表]: 表达式

注: [] 内部的形参列表可以省略

说明:

lambda 表达式创建的函数只能包含一条表达式

lambda 比函数简单,且可以随时创建和销毁,有利于减少程序

的偶合度

看懂下面的程序在做什么?

def fx(f, x, y):

print(f(x, y))

fx((lambda a, b: a + b), 100, 200)

fx((lambda c, d: c \*\* d), 3, 4)

eval 函数 exec 函数

eval 函数:

作用:

把一个字符串当成一个表达式来执行,返回表达式执行后的结果

格式:

eval(source, globals=None, locals=None)

参数:

source 是要解释执行的字符串(此字符串必须是符合python

语法的表达式)

globals 是在source运行时的全局作用域的变量的字典

locals 是source 运行时局部作用域的变量的字典

exec 函数

作用:

把一个字符串当成程序来执行

格式:

exec(source, globals=None, locals=None)

参数说明:

与eval函数的参数说明相同

函数式编程

函数式编程是指用一系列函数解决问题

函数式编程的好处:

用每个函数完成小的问题,一系列函数的任意组合可以完成大问题

函数仅接收输入并产生输出,不包含任何能影响输出的内部状态

函数的可重入性

当一个函数没有访问除局部变量以外的变量,则此函数为可重入函数

说明:

可重入函数输入一定,则输出结果必须一定

高阶函数 High Order Function

什么是高阶函数:

满足下列两个条件之一的函数即为高阶函数

1. 函数接受一个或多个函数作为参数传入

2. 函数返回一个函数

python 内建的高阶函数

map filter sorted

map 函数

map(func, iterable1, iterable2, ....)

形参列表可能定义如下:

def map(func, \*args)

作用:

返回一个可迭代对象,此可迭代对象用func函数对可迭代对象

iterable 中的每一个元素作为参数传入func计算,得到计算后

的结果再返回, 当最短的一个可迭代对象不再提供数时时,此可

迭代对象生成结束

filter函数

格式:

filter(func, iterable)

作用:

筛选可迭代对象iteralbe 中的数据,返回一个可迭代对象,此

可迭代对象只返回iterable提供的数据中满足条件的数据

参数:

func 含有一个形参的数据处理函数,此函数传入的值为iterable

中提供的数据,返回True,则保留此数据,返回False则将此数据

丢弃

iterable为可迭代对象,此可迭代对象提供的数据将传入func

判断后决定是否提供给调用者

思考:

1. 能否用filter把1~100之间的所有素数放在一个列表内?

第一步,定义一个函数

def isprime(x):

....

return True/False

第二步:

L = list(filter(isprime, range(1, 100)))

sorted 函数

作用:

将原可迭代对象的数据进行排序,生成排序后的列表

格式:

sorted(iterable, key=None, reverse=False)

参数说明:

iterable 可迭代对象

key 函数,用来提供一个值,这个值将作为排序的依据

reverse 标志用来设置是否降序排序(默认为升序排序)

递归函数 函数直接或间接的调用自身

递归示意:

函数直接调用自身

def f():

...

f()

...

f()

函数间接调用自身

def fa():

fb()

def fb()

fa()

fa()

说明:

递归一定要控制递归的层数,当符合某一条件时要终止递归调用

几乎所有的递归都能用while循环来代替

递归求阶乘示例

5! = 5 \* 4 \* 3 \* 2 \* 1 # 5! = 5 \* 4!

阶乘的定义

n! 等于:

值为1 if n == 0

值为 n \* (n-1)! if n > 0

递归的优缺点:

优点:

递归可以把问题简单化,让思路更加清晰,代码更简洁

缺点:

递归因系统环境影响大,当递归深度太大时,可能会出现不可

预知的结果

闭包 closure

什么是闭包

闭包是指引用了此函数外部嵌套函数的变量的函数

闭包必须满足以下三个条件:

1. 必须有一个内嵌函数

2. 内嵌函数必须引用外部函数中的变量

3. 外部函数返回值必须是内嵌函数

说明:

由于闭包会使得函数中变量都被保存在内存中,内存消耗很多,所

以不能滥用闭包

装饰器 decorators(专业提高篇)

装饰器是一个函数,主要作用是用来包装另一个函数或类(后面才讲)

作用:

在不修改被装饰函数的源代码和不改变被装饰函数的调用方式的

基上为函数添加新的功能或改变原有功能

装饰器的语法:

def 装饰器函数名(fn):

语句块

return 函数对象

@装饰器函数名 <换行>

def 被装饰函数名(形参列表):

语句块

函数的文档字符串

函数内第一次未赋值给任何变量的字符串是此函数的文档字符串

语法:

def 函数名(形参列表):

'函数的文档字符串'

说明:

文档字符串可以在交互模式下用help(函数名) 查看

函数的文档字符串绑定在函数的 \_\_doc\_\_属性中

函数的\_\_doc\_\_属性

作用:

用于绑定的函数的文档字符串

如:

print(myfun.\_\_doc\_\_)

函数的 \_\_name\_\_ 属性

\_\_name\_\_ 属性用来记录函数名

说明:

以双下划线开头,以双下划线结尾的标识符通常代表python的

特殊变量

函数定义语句(def语句)的完整语法

[@装饰器名1]

[@装饰器名2]

[...]

def 函数名([位置形参], [\*元组形参], [命名关键字形参],

[\*\*字典形参 ]):

'''文档字符串'''

语句块

面试题:

L = [1, 2, 3]

def f(n=0, lst=[]):

lst.append(n)

print(lst)

f(4, L) # [1, 2, 3, 4]

f(5, L) # [1, 2, 3, 4, 5]

f(100) # [100]

f(200) # [100, 200] # 缺省参数里的[] 在def语句执行

# 就已经创建,并一直被f函数绑定

f(300) # [100, 200, 300]

模块 Module

模块是一个包含有一系列数据,函数,类等组成的程序组

模块是一个文件,模块文件名通常以.py结尾

作用:

1. 让一些相关的函数,数据,类有逻辑的组织在一起,使逻辑结构

更加清晰

2. 模块中的数据,函数和类等可以提供给其它模块使用

模块的分类:

内置模块(builtins) 在解析器的内部可以直接使用

标准库模块,安装python时已安装且可以直接使用

第三方模块(通常为开源), 需要自己安装

安装命令:

pip3 install 模块名

或

pip install 模块名(通常用来安装python3的模块)

用户自己编写的模块

模块的导入语句 import

import 语句

import 模块名1 [as 模块新名1], 模块名2 [as 模块新名1],...

示例:

import math

import sys, time

作用:

当某模块整体导入到当前模块中

用法:

模块名.属性名

math.factorial(5) # 返回120

dir(obj) 函数,返回obj对象所有的属性的字符串列表

help(obj) 函数,可以查看模块的文档字符串

from import 语句

语法:

from 模块名 import 模块属性名1 [as 属性新名1],

模块属性名2 [as 属性新名2], ....

作用:

将某模块内的一个或多个属性导入到当前模块的作用域

示例:

from math import factorial as fac

from math import sqrt, pi, sin, cos

print(sin(pi/2)) # 1.0

from import \* 语句

语法:

from 模块名 import \*

作用:

将某模块的所有属性导入到当前模块作用域

示例:

from math import \*

print(sin(pi / 2)) # 1.0

print(factorial(5)) # 120

dir函数

dir([对象])

作用:

如果没有参数调用,则返回当前作用域内所有变量的列表

如果给定一个对象作为参数,则返回这个对象的所有变量的列表

对于一个模块,返回这个模块的全部属性(变量)

对于一个类对象,返回这个对象的所有变量,并递归基类对象

的所有变量

对于其它对象返回所有变量,类变量和基类变量

内建模块

math

time

sys

math 模块

文档参见:

python\_base\_docs\_html/数学模块math.html

time 模块

文档参见:

python\_base\_docs\_html/时间模块time.html

sys 模块

运行时系统相关的信息 (runtime system)

文档参见:

python\_base\_docs\_html/系统模块sys.html