**Unit 11**

11-1.

|  |  |  |  |
| --- | --- | --- | --- |
| Input | countToFour1 | countToFour2 | countToFour3 |
| 2 | TypeError | 2 3 4 | 2 3 4 |
| 4 | TypeError | 4 | 4 |
| 5 | TypeError | NONE | NONE |
| (nothing) | 0 1 2 3 4 | TypeError | 1 2 3 4 |

11-3.

**def max2**(num1,num2):  
 **return** num1 **if** num1>num2 **else** num2  
  
**def min2**(num1,num2):  
 **return** num1 **if** num1<num2 **else** num2  
  
**def my\_max**(\*args):  
 **return** reduce(max2,args)  
  
**def my\_min**(\*args):  
 **return** reduce(min2,args)  
  
str = "abcdef"  
nums = [1,2,3,4,5]  
**print** my\_min(\*nums)  
**print** my\_max(1,2,3,4,5)

注意my\_min()和my\_max()的调用方式

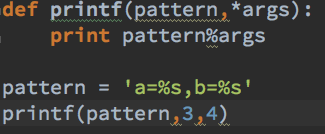
11-4.

**def tranHourtoMinute**(strTime):  
 nums = strTime.split(':')  
 **return** int(nums[0])\*60+int(nums[1])  
  
**def tranMinutetoHour**(strTime):  
 hour,minute = divmod(int(strTime),60)  
 **return** '%d:%d'%(hour,minute)

11-5.

**def tax**(amount,rate=0.12):  
 **return** amount\*rate

11-6.



11-7.

**print** map(**lambda** x,y:(x,y),[1,2,3],['abc','def','ghi'])  
**print** zip([1,2,3],['abc','def','ghi'])

11-8.

**def isLeap**(year):  
 **return** (year%4 == 0 **and** (year%100 != 0 **or** year%400 == 0))  
  
**print** filter(isLeap,range(2000,2020))  
**print** [i **for** i **in** range(2000,2020) **if** isLeap(i)]

11-9.

**def average**(\*numbers):  
 **return** reduce(**lambda** x,y:float(x+y),numbers)/len(numbers)

11-10.

显示folder下所有的非隐藏目录

11-11.

**import** os  
  
**def strip**(str):  
 **return** str.strip()  
  
**def writeNewFile**(lines):  
 **while** True:  
 newfile = raw\_input("Please enter the new filename: ")  
 **if** os.path.exists(newfile):  
 **print** 'The file exists,try again'  
 **else**:  
 **break  
 with** open(newfile,'w') **as** file:  
 **for** line **in** lines:  
 file.write(line+'\n')  
  
**def cleanFile**(filename):  
 **with** open(filename) **as** file:  
 **return** map(strip,file.readlines())  
 #return [strip(line) for line in file]

**def writeOldFile**(lines):  
 **while** True:  
 oldfile = raw\_input("Please enter the old filename: ")  
 **if** os.path.exists(oldfile):  
 **break  
 else**:  
 **print** "the file does not exist,try again"  
 **with** open(oldfile,'w') **as** file:  
 **for** line **in** lines:  
 file.write(line+'\n')  
  
**if** \_\_name\_\_=="\_\_main\_\_":  
 **while** True:  
 filename = raw\_input("Please enter the filename(q to quit): ")  
 **if** filename=='q':  
 **break  
 if** os.path.exists(filename):  
 lines = cleanFile(filename)  
 choice = raw\_input("n to new file, or c to cover file: ")  
 **if** choice.lower()=='n':  
 writeNewFile(lines)  
 **else**:  
 writeOldFile(lines)  
 **else**:  
 **print** 'filename does not exist'

11-12.

**from** time **import** time,sleep  
  
**def timeit**(func):  
 **def \_time**(\*args,\*\*kwargs):  
 start = time()  
 val = func(\*args,\*\*kwargs)  
 delta = time()-start  
 **return** delta,val  
 **return** \_time  
  
@timeit  
**def test**(sec):  
 sleep(sec)

11-13.

**from** time **import** time  
  
**def timeit**(func):  
 **def \_time**(\*args,\*\*kwargs):  
 start = time()  
 val = func(\*args,\*\*kwargs)  
 delta = time()-start  
 **return** delta,val  
 **return** \_time  
  
  
**def mult**(x,y):  
 **return** x\*y  
  
@timeit  
**def factorial1**(num):  
 **return** reduce(mult,range(1,num+1))  
  
@timeit  
**def factorial2**(num):  
 **return** reduce(**lambda** x,y:x\*y,range(1,num+1))  
  
  
**def factorial3**(num):  
 **if** num==1:  
 **return** 1  
 **else**:  
 **return** num\*factorial3(num-1)  
  
@timeit  
**def factorial**(num):  
 **return** factorial3(num)  
  
**if** \_\_name\_\_=="\_\_main\_\_":  
 **print** factorial1(100)  
 **print** factorial2(100)  
 **print** factorial(100)

11-14.

**def getFibonacci**(num):  
 **if** num==1 **or** num==2:  
 **return** 1  
 **else**:  
 **return** getFibonacci(num-1)+getFibonacci(num-2)

11-15.

**def reverse\_show\_str**(str):  
 **if** str:  
 **print** str[len(str)-1],  
 reverse\_show\_str(str[:len(str)-1])  
  
  
reverse\_show\_str('abc')  
**print  
  
def cross\_show\_str**(str):  
 **if** str:  
 **print** str[0],  
 cross\_show\_str(str[1:])  
  
cross\_show\_str('abc')

11-16.

**from** operator **import** add,sub,mul,div  
**from** random **import** randint,choice  
  
ops = {'+':add,'-':sub,'\*':mul,'/':div}  
MAXTRIES = 2  
  
**def doprob**():  
 nums =[randint(1,10) **for** i **in** range(2)]  
 nums.sort(reverse=True)  
 op = choice('/')  
 **if** op **in** '+-\*':  
 ans = ops[op](\*nums)  
 **else**:  
 nums = [float(i) **for** i **in** nums]  
 ans = round(ops[op](\*nums),6)  
 pr = '%d %s %d='%(nums[0],op,nums[1])  
 oops = 0  
 **while** True:  
 **try**:  
 **if** float(raw\_input(pr))==float(ans):  
 **print** 'correct'  
 **break  
 elif** oops==MAXTRIES:  
 **print** 'answer\n%s%d'%(pr,ans)  
 **break  
 else**:  
 **print** 'incorrect...try again'  
 oops += 1  
 **except** (KeyboardInterrupt,EOFError,ValueError):  
 **print** 'invalid input...try again'  
  
**def main**():  
 **while** True:  
 doprob()  
 **try**:  
 opt = raw\_input('Again?[y]').lower()  
 **if** opt **and** opt[0]=='n':  
 **break  
 except** (EOFError,KeyboardInterrupt):  
 **break  
  
if** \_\_name\_\_=='\_\_main\_\_':  
 main()

11.17.

（a）描述偏函数应用和currying之间的区别

偏函数要解决的问题是：如果一个函数有几个参数，想要固化某几个参数为固定值

currying要解决的问题是：如果有几个单参数函数，如何实现一个多参数函数

（b）偏函数应用和闭包之间有什么区别

闭包是在一个内部函数里对外部作用域（不包括全局作用域）的变量进行引用

（c）最后，迭代器和生成器是怎么区别开的

生成器是用def函数定义的，函数中包含yield语句，可用for in遍历

可通过send()方法向生成器发送值

迭代器是可用for in的对象

迭代器是生成器的真子集

生成器的编写方法和函数定义类似，只是在return的地方改为yield。

生成器中可以有多个yield。当生成器遇到一个yield时，会暂停运行生成器，返回yield后面的值。当再次调用生成器的时候，会从刚才暂停的地方继续运行，直到下一个yield。

生成器自身又构成一个迭代器，每次迭代时使用一个yield返回的值。

迭代器可以看作是一个特殊的对象，每次调用该对象时会返回自身的下一个元素，从实现上来看，一个可迭代的对象必须是定义了\_\_iter\_\_()方法的对象，而一个迭代器必须是定义了\_\_iter\_\_()方法和next()方法的对象。生成器的概念要比迭代器稍显复杂，因为生成器是能够返回一个迭代器的函数，其最大的作用是将输入对象返回为一个迭代器。