1. 顺序排序，每一列都按照从上到下递增的顺序排序。请完成一个函数，输入这样的一

个二维数组和一个整数，判断数组中是否含有该整数。

arr = [[1,4,7,10,15], [2,5,8,12,19], [3,6,9,16,22], [10,13,14,17,24], [18,21,23,26,30]]

def getNum(num ,data=None):

count = 0

for i in range(0,len(data)):

for j in range(len(data[i])-1,-1,-1):

count +=1

if data[i][j] < num:

break;

elif data[i][j] == num:

pos = [i,j]

print count

return pos

else:

pass

return None

if \_\_name\_\_ == '\_\_main\_\_':

position = getNum(18, arr)

if position:

print(position)

else:

print "no match"

备注：算法不是最优 但是还是用到了条件。

1. 【pathon】获取最大公约数、最小公倍数（最大公约数：两个数字连续相互相减，知道相等返回。最大公倍数：两数相乘除以公约数）

a = 36

b = 21

def maxCommon(a,b):

assert a>0 and b>0

while True:

if (a>b):

a = a-b

elif (a==b):

return a

else:

a,b = b,a

def minCommon(a,b):

return (a\*b)/ maxCommon(a,b)

if \_\_name\_\_ == '\_\_main\_\_':

a=14

b=35

print maxCommon(a,b)," ", minCommon(a,b)

1. 【python】求list 中位数？ 确保list为奇数个

l = [1,3,4,53,2]

def median(data):

length = len(data)

assert length%2 != 0

data.sort() #~ 是按位反转 ~0= -1 , ~1=-2 (用在列表倒数的时候 a[1] 对应 a[~1])

return data[length/2]

if \_\_name\_\_ == '\_\_main\_\_':

print median(l)

1. 【python】简单3个数排序

x = input("x=")

y = input("y=")

z = input("z=")

if x>y:

x,y= y,x

if x>z:

x,z= z,x

if y>z:

y,z=z,y

print x,y,z

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

整列输入：

data = raw\_input('x,y,z:')

l = data.split(",")

for i in range(len(l)):

l[i]= int(l[i])

print l

1. 利用递归方法求5!

def func(data):

if data != 0:

return data \* func(data-1)

return 1

print func(5)

1. 求斐波那契数列前10项？（斐波那契数列：第三个开始是前两个的和 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233，377，610，987，15972584，4181，6765，10946，17711。。。）

def func(num):

if num==1 or num ==2:

return 1

else:

return func(num-2)+func(num-1)

if \_\_name\_\_ == "\_\_main\_\_":

count = 10

result = ""

for i in range(1,count+1):

result = result + str(func(i))+","

print result

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

def func(m):

n,a,b = 0,1,1

while n < m:

yield a [塞入一个a 返回list]

a,b = b,a+b

n += 1

for one in func(10):

print one

1. 写个函数使 [1,2,5,[3,[],5,2,[57]],90] => [1, 2, 5, 3, 5, 2, 57, 90]

备注：单纯的递归只适合单深度的，有起伏的要配合while和for

l = [1,2,5,[3,[],5,2,[57]],90]

result = []

def extend(result,data):

while len(data)!= 0:

if type(data[0])!=list:

result.append(data[0])

del data[0]

else:

extend(result,data[0])

del data[0]

return

extend(result,l)

print result

1. 求最大子数组之和问题：给定一个整数数组（数组元素有负有正），求其连续子数组之和的最大值。

s = [12,-4,32,-36,12,6,-6]

def sum(l):

k = 0

for i in l:

k += i

return k

def find\_largest(data):

result = data[0]

for i in range(len(data)): #数组还是建议从0开始

for j in range(0,len(data)-i):

temp = sum(data[j:j+1+i])

if result < temp:

result = temp

return result

print find\_largest(s)

1. 冒泡排序

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

java 好记忆 复杂算法用java

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**void** bubble(**int**[] list){

**for** (**int** i = 0 ;i<list.length-1; i++) {

**for** (**int** j = list.length-2; j>=i; j--) {

**if** (list[j]<list[j+1])

swap(list,j,j+1);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Python比较难记忆 不建议

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import random

def getrandata(num):

a=[]

i=0

while i<num:

a.append(random.randint(0,1000))

i+=1

return a

def poo(data):

for i in range(len(data)-1): #最好在边缘的地方 自己写一下最后一个len-1 ，最后第二个len-2

for j in range(len(data)-2,i-1,-1):#注意倒过来一定要加-1， 倒过来也是左包含右不包含

if (data[j]<data[j+1]):

data[j],data[j+1]=data[j+1],data[j]

return data

if \_\_name\_\_ == "\_\_main\_\_":

num = 30

l= getrandata(num)

print l

print poo(l)

1. 判断101-200之间有多少个素数,并输出所有素数。

def find\_number(left,right,ls=[]):

assert right>left>2

for i in range(left,right+1):

for j in range(2,i):

if (i%j == 0):

break

elif (j==i-1):

ls.append(i)

return ls

if \_\_name\_\_ == "\_\_main\_\_":

left = 101

right = 200

ls = find\_number(left,right)

print len(ls)

print ls

1. 输入一行字符,分别统计出其中英文字母、空格、数字和其它字符的个数。

class judger(object):

def \_\_init\_\_(self,s):

self.str = s

self.intNum = 0

self.letterNum = 0

self.spaceNum = 0

self.otherNum = 0

self.search()

def isLetter(self,ch):

ls =[]

ls.extend([chr(x) for x in range(ord('a'), ord('z') + 1)])

ls.extend([chr(x) for x in range(ord('A'), ord('Z') + 1)])

print ls

return (ch in ls)

def search(self):

for one in self.str:

if one == " ":

self.spaceNum +=1;

elif one.isdigit():

self.intNum+=1;

elif self.isLetter(one): #（.isalpha()不能判断）

self.letterNum+=1;

else:

self.otherNum+=1;

def getInfo(self):

output = "spaceNum:",self.spaceNum,"intNum:",self.intNum,"letterNum:",self.letterNum,"otherNum:",self.otherNum,

return output

if \_\_name\_\_ == "\_\_main\_\_":

str = raw\_input("input:")

cls = judger(str)

print cls.getInfo()

1. 计算 sum = a+aa+aaa+..... 共b个的和

def func(a,b):

assert a>0 and b>0

sum = 0

for i in range(1,b+1):

for j in range(1,i+1):

sum = sum+ a\*(10\*\*(j-1))

return sum

if \_\_name\_\_=="\_\_main\_\_":

print func(2,3)

1. 打印菱形，输入中线的层数

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

def printStar(level):

assert level>0

for i in range(1,level+1):

print " "\*(level-i)+ "\*"\*(2\*(i-1)+1)

for i in range(level-1,0,-1):

print " "\*(level-i)+"\*"\*(2\*(i-1)+1)

printStar(4)

1. 求1+2!+3!+...+n!的和

def sum(a):

if a==1:

return 1

else:

return a\*sum(a-1)

if \_\_name\_\_ == "\_\_main\_\_":

n = 4

result = 0

for i in range(1,n+1):

result = result + sum(i)

print result

1. 递归方法字符串倒转

str = "hello"

def reverse(s,index,result=""):

if index == 1:

return result+s[0] #结束条件和结束返回

else:

result = result+s[index-1] #每次递归前do something

return reverse(s,index-1,result) #每次递归index-1 这个是必须要做的

print reverse(str,len(str))

print str[::-1] (内置方法)

1. 有5个人坐在一起.

问第五个人多少岁？他说比第4个人大2岁。

问第4个人岁数，他说比第3个人大2岁。

问第3个人岁数，他说比第2个人大2岁。

问第2个人岁数，他说比第1个人大2岁。

最后问第一个人，他说是10岁。请问第五个人多大？

(输入first\_P\_age,last\_P\_num, 求last\_P\_age?)

def func(current\_P\_num, first\_p\_age):

if current\_P\_num == 1:

return first\_p\_age

else:

return 2+func(current\_P\_num-1,first\_p\_age)

print func(5,10) #第5人 第一个人10岁

1. 给一个不多于5位的正整数，要求：一、求它是几位数，二、逆序打印出各位数字。

def find(n,result):

assert n>0

ls = []

while n > 0:

ls.append(n%10) #除10取余数

n = n/10 #除10向下取整

result["count"]=len(ls)

for i in range(len(ls)):

result["reverse"]=result["reverse"]+ls[~i]\*(10\*\*(i)) # ~倒查数列方便， \*\*指数

#注意1,3,5,7,9 是2\*n+1；1，10，100，1000 是10\*\*n

return result

result={"count":0,"reverse":0}

print find(123456789,result)

1. 输入正整数 ，判断是不是回文数

def judge(data):

assert data>0

ls = [];

while data>0:

ls.insert(0,data%10) #insert（0，）从前插入和append相反

data = data/10

i=0;

j=len(ls)-1;

while i<j:

if ls[i] != ls[j]:

return False

i+=1;

j-=1;

return True;

n=12321

print judge(n)

1. 对字典进行排序。

def sort(d):

ls = d.items() #字典本身没有顺序，生成列表[('a', 1), ('c', 2), ('b', 4), ('e', 3), ('d', 5)],里面是元组

for i in range(0,len(ls)-1):

for j in range(len(ls)-2,i-1,-1):

if ls[j][1]>ls[j+1][1]:

ls[j],ls[j+1]=ls[j+1],ls[j]

return ls

d = {'a':1,'b':4,'c':2,'d':5,'e':3}

print sort(d)

1. 有1、2、3、4个数字,能组成多少个互不相同且无重复数字的三位数? 分析 4的3次方

for i in range(1,5):

for j in range(1,5):

for k in range(1,5):

if i!=j and j!=k and i !=k: #如果没有这条则是全遍历

print i,j,k

1. 给定数组，最大的数与第一个数交换 ， 最小的数最后一个交换

def func(data):

assert len(data)>0

maxIndex = minIndex = 0

max = min = data[0]

for i in range(len(data)): #最好在边缘的地方 自己写一下最后一个len-1 ，最后第二个len-2

if max<data[i]:

max = data[i]

maxIndex = i

if min>data[i]:

min=data[i]

minIndex=i

data[0], data[maxIndex] = data[maxIndex],data[0]

if minIndex==0: #这里是关键,万一最小数被第一次替换掉了 记录替换的位置

minIndex = maxIndex

data[len(data)-1], data[minIndex] = data[minIndex], data[len(data)-1]

return data

1. max 使用 lambda

Max = lambda x,y:(x>y)\*x+(y>x)\*y

Min = lambda x,y:(x>y)\*y+(y>x)\*x

print Max(3,2),Min(4,5)

1. 第k大质数

public boolean isPrime(int n){

for(int i=2;i<=n/2;i++){

if(n%i == 0)

return false;

}

return true;

}

public int judge(int k){

int n=2;

int count = 0;

while(true) {

if(isPrime(n))

count++;

if (count == k)

return n;

n++;

}

}

System.***out***.println(judge(32));