python练习题

一、操作题：

1.jieba分词练习

1. 使用jieba.cut()函数对“蛋白质或核苷酸三维结构存储在PDB文件中”进行分词，分别采用精确模式和全模式，输出结果。

import jieba

test\_str='蛋白质或核苷酸三维结构存储在PDB文件中'

seg\_list = jieba.cut(test\_str, cut\_all=False)

print("Default Mode: " + "/ ".join(seg\_list))  # 精确模式

seg\_list = jieba.cut(test\_str, cut\_all=True)

print("Default Mode: " + "/ ".join(seg\_list))  # 全模式

Building prefix dict from the default dictionary ...

Loading model from cache C:\Users\pc\AppData\Local\Temp\jieba.cache

Loading model cost 1.722 seconds.

Prefix dict has been built successfully.

Default Mode: 蛋白质/ 或/ 核苷酸/ 三维/ 结构/ 存储/ 在/ PDB/ 文件/ 中

Default Mode: 蛋白/ 蛋白质/ 白质/ 或/ 核苷/ 核苷酸/ 三维/ 结构/ 存储/ 在/ PDB/ 文件/ 中

1. 使用jieba.lcut()函数对“蛋白质或核苷酸三维结构存储在PDB文件中”进行分词，分别采用精确模式和全模式，输出结果。

import jieba

test\_str='蛋白质或核苷酸三维结构存储在PDB文件中'

seg\_list = jieba.lcut(test\_str, cut\_all=False)

print("Default Mode: " + "/ ".join(seg\_list))  # 精确模式

seg\_list = jieba.lcut(test\_str, cut\_all=True)

print("Default Mode: " + "/ ".join(seg\_list))  # 全模式

Building prefix dict from the default dictionary ...

Loading model from cache C:\Users\pc\AppData\Local\Temp\jieba.cache

Loading model cost 1.575 seconds.

Prefix dict has been built successfully.

Default Mode: 蛋白质/ 或/ 核苷酸/ 三维/ 结构/ 存储/ 在/ PDB/ 文件/ 中

Default Mode: 蛋白/ 蛋白质/ 白质/ 或/ 核苷/ 核苷酸/ 三维/ 结构/ 存储/ 在/ PDB/ 文件/ 中

1. 使用jieba.lcut()函数对“蛋白质或核苷酸三维结构存储在PDB文件中”进行分词，将核苷酸三维结构定义为词语，输出结果。

import jieba

test\_str='蛋白质或核苷酸三维结构存储在PDB文件中'

jieba.add\_word('核苷酸三维结构')

seg\_list = jieba.cut(test\_str, cut\_all=False)

print("Default Mode: " + "/ ".join(seg\_list))  # 精确模式

Building prefix dict from the default dictionary ...

Loading model from cache C:\Users\pc\AppData\Local\Temp\jieba.cache

Loading model cost 1.756 seconds.

Prefix dict has been built successfully.

Default Mode: 蛋白质/ 或/ 核苷酸三维结构/ 存储/ 在/ PDB/ 文件/ 中

1. 使用jieba.lcut()函数对“蛋白质或核苷酸三维结构存储在PDB文件中”进行分词，输出词语和词性。

import jieba

import jieba.posseg as ps

test\_str='蛋白质或核苷酸三维结构存储在PDB文件中'

r=ps.lcut(test\_str)

print(r)

Building prefix dict from the default dictionary ...

Loading model from cache C:\Users\pc\AppData\Local\Temp\jieba.cache

Loading model cost 1.917 seconds.

Prefix dict has been built successfully.

[pair('蛋白质', 'n'), pair('或', 'c'), pair('核苷酸', 'n'), pair('三维', 'm'), pair('结构', 'n'), pair('存储', 'j'), pair('在', 'p'), pair('PDB', 'eng'), pair('文件', 'n'), pair('中', 'f')]

2.组合数据类型

1. 请输入一组词语，输出各个词语及其出现的次数（要求使用字典完成次数统计)

在txt中输入

import sys

reload(sys)

sys.setdefaultencoding('utf8')

import json

import time

path='test.txt'

a= open(path).readlines()

print type(a)

arr01=[]

for x in a:

    y=json.loads(x)

    for z in y:

        if z =='tz':

              print  str(z)+'=>'+str(y[z])

              arr01.append(y[z])

    print '--------------------------------'

print arr01

def get\_counts(sequence):

    counts={}

    for x in sequence:

        if x in counts:

            counts[x] += 1

        else:

            counts[x]=1

    return counts

aa= get\_counts(arr01)

print aa

print type(aa)

for k in aa:

    print k+'=>'+str(aa[k])

1. 有如下运动员成绩

P：9.74

G：9.79

B:9.69

A：9.85

M：9.78

L：9.86

编写一个程序，按名次输出排名、姓名和成绩。

listtmp=[('P',9.74),('G',9.79),('B',9.69),('A',9.85),('M',9.78),('L',9.86)]

listtmp.sort(key=lambda x:(x[1],x[0]))

listtmp.reverse()

for i in range(len(listtmp)):

    print('Rank:'+str(i+1)+' '+str(listtmp[i]))

Rank:1 ('L', 9.86)

Rank:2 ('A', 9.85)

Rank:3 ('G', 9.79)

Rank:4 ('M', 9.78)

Rank:5 ('P', 9.74)

Rank:6 ('B', 9.69)

1. 有如下表1和表2，编写程序完成下列任务
2. 输出表1和表2均包含的商品信息

sheet1={"orange":12.5,"dragonfruit":15.8,"pear":4.5}

sheet2={"apple":5.5,"banana":4.8,"pear":4.5,"watermelon":5.6}

result\_set=(set( sheet2.keys() ) & set( sheet1.keys() ))

result\_sheet={}

for i in result\_set:

    value=''

    if(str(i) in sheet1.keys()):

        value=sheet1[str(i)]

    elif (str(i) in sheet2.keys()):

        value=sheet2[str(i)]

    temp={str(i):value}

    result\_sheet.update(temp)

print(result\_sheet)

{'pear': 4.5}

1. 输出属于表1，但不属于表2的商品信息

sheet1={"orange":12.5,"dragonfruit":15.8,"pear":4.5}

sheet2={"apple":5.5,"banana":4.8,"pear":4.5,"watermelon":5.6}

result\_set=(set( sheet2.keys() ) - set( sheet1.keys() ))

result\_sheet={}

for i in result\_set:

    value=''

    if(str(i) in sheet1.keys()):

        value=sheet1[str(i)]

    elif (str(i) in sheet2.keys()):

        value=sheet2[str(i)]

    temp={str(i):value}

    result\_sheet.update(temp)

print(result\_sheet)

{'apple': 5.5, 'banana': 4.8, 'watermelon': 5.6}

1. 输出属于表1但不属于表2，或者属于表2但不属于表1的商品信息

sheet1={"orange":12.5,"dragonfruit":15.8,"pear":4.5}

sheet2={"apple":5.5,"banana":4.8,"pear":4.5,"watermelon":5.6}

result\_set=(set( sheet2.keys() ) ^ set( sheet1.keys() ))

# print(result\_set)

result\_sheet={}

for i in result\_set:

    # print(str(i))

    value=''

    if(str(i) in sheet1.keys()):

        value=sheet1[str(i)]

    elif (str(i) in sheet2.keys()):

        value=sheet2[str(i)]

    temp={str(i):value}

    result\_sheet.update(temp)

print(result\_sheet)

{'banana': 4.8, 'watermelon': 5.6, 'orange': 12.5, 'dragonfruit': 15.8, 'apple': 5.5}

1. 输入商品名称，查询其价格

sheet1={"orange":12.5,"dragonfruit":15.8,"pear":4.5}

sheet2={"apple":5.5,"banana":4.8,"pear":4.5,"watermelon":5.6}

result\_set=(set( sheet2.keys() ) | set( sheet1.keys() ))

# print(result\_set)

result\_sheet={}

for i in result\_set:

    # print(str(i))

    value=''

    if(str(i) in sheet1.keys()):

        value=sheet1[str(i)]

    elif (str(i) in sheet2.keys()):

        value=sheet2[str(i)]

    temp={str(i):value}

    result\_sheet.update(temp)

print(result\_sheet)

print('input a product name:')

a=input()

print(result\_sheet[a])

{'apple': 5.5, 'orange': 12.5, 'watermelon': 5.6, 'pear': 4.5, 'banana': 4.8, 'dragonfruit': 15.8}

input a product name:

apple

5.5

|  |  |
| --- | --- |
| **商品名称** | **价格** |
| apple | 5.5 |
| banana | 4.8 |
| watermelon | 5.6 |
| pear | 4.5 |

|  |  |
| --- | --- |
| **商品名称** | **价格** |
| orange | 12.5 |
| dragonfruit | 15.8 |
| pear | 4.5 |

3.函数

定义一个函数，计算n个数的平均值。显示结果如下：

请输入n个数值：1,3,6,78,58,98

1+3+6+78+58+98平均值= 40.666666666666664

a=[]

tmp\_str=input()

a=tmp\_str.split(',')

avg=0.0

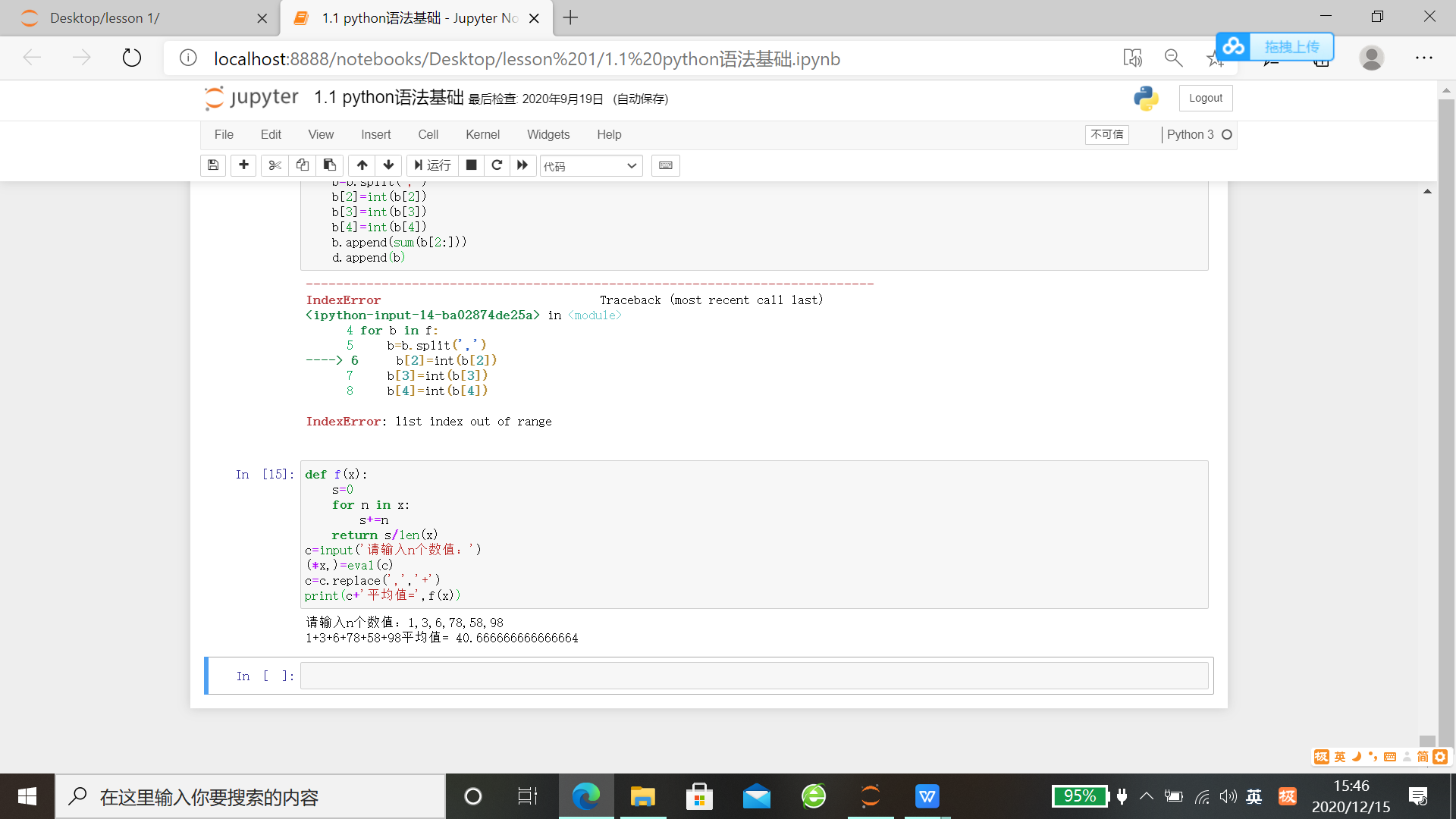
for i in a:

    avg+=float(i)/len(a)

print(avg)

1,2,3,4

2.5



4.文件和数据组织

文件中test.txt保存了5个学生的课程成绩。

学号，姓名，语文，数学，外语

1. 陈一，99，88，76
2. 刘二，108，84，84
3. 杨三，111，56，120
4. 王五，98，78，89
5. 雷六，90，88，76

编写一个程序，从文件读取成绩，计算总分，按总分从高到低的顺序输出（各字段对齐)

答案：

f=open('C:/users/libei/Desktop/test.txt')

d=[]

t=next(f)

for b in f:

b=b.split(',')

b[2]=int(b[2])

b[3]=int(b[3])

b[4]=int(b[4])

b.append(sum(b[2:]))

d.append(b)

def fk(a):

return a[5]

d.sort(key=fk,reverse=True)

n=1

#输入时注意对齐

xm='{1:{0}<4}'.format(chr(12288),'姓名')

print('%-4s%-5s%s-4s%-4s%s')

%('‘名次','学号',xm,'语文','数学','外语','总分',)

for a in d:

print('{0:^6d}'.format(n),'%-6s'%a[0],

'{1:{0}<4}'.format(chr(12288),a[1]),

'%-7s%-7s%-7s%s'%tuple(a[2:]))

n+=1

f.close()

import os

with open(os.getcwd()+'\\练习1\\stinfo.txt', 'r',encoding='utf-8') as f:

    # print(f.read())

    temp\_list=f.readlines()

    print(temp\_list)

    outputlist=[]

    for i in range(len(temp\_list)):

        outputlist.append(temp\_list[i][2:-1].split(','))

        outputlist[i].append(int(outputlist[i][1])+int(outputlist[i][2])+int(outputlist[i][3]))

    outputlist.sort(key=lambda x:(x[4],x[0]))

    outputlist.reverse()

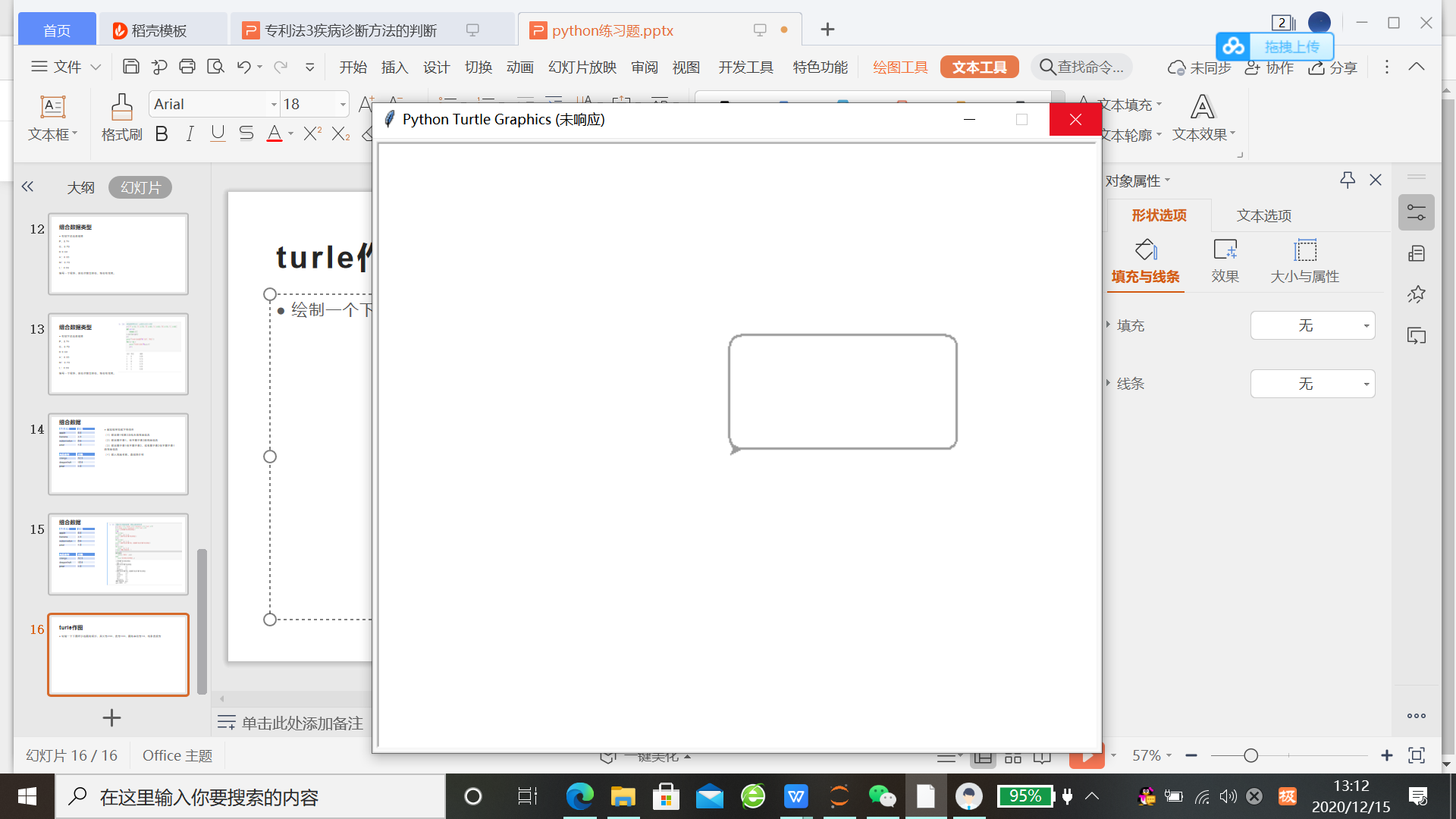
    print(outputlist)

f.close()

[['杨三', '111', '56', '120', 287], ['刘二', '108', '84', '84', 276], ['王五', '98', '78', '89', 265], ['陈一', '99', '88', '76', 263], ['雷六', '90', '88', '76', 254]]

1. turtle作图

绘制一个下图所示的圆角矩形，其长为200，宽为100，圆角半径为10，线条宽度为2。



import turtle as t

t.pensize(3)

for \_ in range(2):

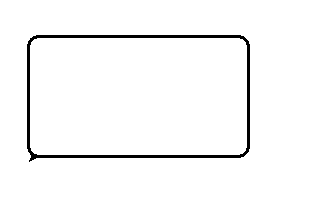
    t.forward(200)

    t.circle(10,90)

    t.forward(100)

    t.circle(10,90)

t.done()



二、选择题

1. 下列关于Python文件处理的说法中，错误的是（D）
2. 能处理JPG图像文件
3. 能处理CSV文件
4. 能处理EXCEL文件
5. 文本文件不能作为二进制文件来处理
6. 下列关于文件操作的说法中，错误的是（B）
7. open()函数用于打开文件
8. 以文本文件方式打开文件时，文件按照字节流方式进行读取
9. 文件使用完毕要用close()方法关闭
10. 能够以文本和二进制两种方式处理文件
11. 关于Python文件打开模式的说法错误的是（b）
12. w模式会覆盖原有的文件
13. a模式有文件末尾添加数据
14. b模式为创建新文件
15. r模式只能从文件读取数据
16. 设文件d:/data.txt的内容如下

Python,java,C++

Javascript,HTML

下面代码的执行结果是（ 这段代码会报错，如果稍加修改语法，应当选A ）

f=open(“d:/data.txt”)

Data=f.read().split(“,”)

f.close()

Print(data)

1. [‘python’,’java’,’C++\nJavascript’,’HTML’]
2. [‘python’,’java’,’C++’,’Javascript’,’HTML’]
3. [‘python’,’java’,’C++’,’\n’,’Javascript’,’HTML’]
4. ‘python’,’java’,’C++’,’Javascript’,’HTML’
5. 关于CSV文件的说法错误的是（~~B~~D）。
6. CSV文件的第一行是一组一维数据
7. 文件中的数据可使用自定义符号分隔
8. 整个CSV文件是一组二维数据
9. CSV是一种通用的文件格式，可用于在程序之间转移多种格式的表格数据
10. 下面的程序运行后，文件data.csv中的内容是（这段代码会报错，如果稍加修改语法，应当选C ）。

f=open(“data.csv”,’w’)

a=[1,2,3],[4,5,6],[7,8,9]

for x in a:

for y in [0,1]:

f.write(str(x[y])

f.write(‘,’)

f.write(str(x[2]))

f.write(‘\n’)

f.close()

A.1,2,3,4,5,6,7,8,9

B.9,8,7,6,5,4,3,2,1

C.1,2,3

4,5,6

7,8,9

D.9,8,7

6,5,4

3,2,1

1. 关于数据组织维度的说法中，错误的是（ D ）。
2. 一维数据采用线性方式组织，可表示数学中的数组和集合
3. 二维数据采用表格方式组织，可表示数学中的矩阵
4. 高维数据可用JSON字符串表示
5. Python的字典类型可用于表示一维和二维数据
6. 关于数据组织的维度的说法正确的是（ ~~B~~ D）。
7. 表格中的数据都是二维数据
8. 高维数据由关联的关系数据构成
9. CSV格式的数据是一维数据
10. 一维数据通常采用线性方式存储
11. 下列关于Python函数的说法错误的是：（ C ）
12. 函数是一段可重用的语句组
13. 函数通过函数名进行调用//不一定
14. 每次调用函数提供参数的数据类型必须相同
15. 调用函数时，可以不为带默认值的参数提供实参
16. Python语言中用来定义函数的关键字是（ D ）。
17. 函数使用def语句定义
18. 函数可以没有参数
19. 函数可以有多个参数
20. 函数可以有多个返回值
21. 下列关于函数调用的说法错误的是（ ~~D~~ A ）。
22. 函数调用可以出现在任意位置
23. 函数是一种对象
24. 可将函数名赋值给变量
25. 函数名是一个变量
26. 函数定义如下。

def f(a,b):

return a+b

下列选项中函数调用错误的是（ D ）。

1. f(1,2)
2. f(a=1,b=2)
3. f(b=2,a=1)
4. f((1,2))
5. 执行下面代码的输出结果是（ D ）。

def fun(a,b=1,c=2):

print(a+b+c)

fun(3, ,4)

A.7 B.8 C.6 D.出错

1. 下面程序的输出结果是（ D ）。

def f(y):

y\*=2

f(x)

print(x)

A.3 B.6 C.无输出 D.出错

1. 函数定义如下：

def f(a,\*b)

s=a

for c in b:

s+=c

return s

下列选项中函数调用错误的的（ C ）。

A.f(1) B.f(a=1) C.f(a=1,2) D.f(b=2,a=1)

1. 下列说法中正确的是（ B ）。
2. 局部变量指在函数内部使用的变量
3. 使用global保留字声明的变量是全局变量
4. 简单数据类型的变量只能在函数内部创建和使用
5. 全局变量指在函数外创建的变量，在程序执行的全过程有效
6. 下列选项中错误的是（ D ）。
7. 使用global保留字声明变量后，可以函数内部为其赋值
8. 局部变量与全局变量的名称可以相同
9. 函数运行结束后，局部变量会被释放
10. nonlocal关键字用于在函数内部声明全局变量
11. 执行下列的语句后，输出结果是（ C ）。

def func():

global x

x=200

x=100

func()

print(x)

1. 0 B.100 C.200 D.300
2. 下列关于组合数据类型的说法正确的是：（ ~~B~~ D ）。
3. 集合对象中的元素是有序的
4. 序列对象和集合对象中的元素允许重复
5. 映射类型对象中的关键字只能是同一种类型的数据
6. 组合数据类型可以对多种类型的数据执行相同的处理
7. 下面代码的输出结果是（ B ）。

color={“red”:“红色”,“green”:“绿色”,“blue”:“兰色”}

print(color[‘red’],color.get(‘blue’,’黄色’))

A.红色 黄色 B.红色 兰色 C. 绿色 黄色 D.红色 绿色