试卷编号：499
所属语言：Python
试卷方案：模拟考-Python
试卷总分：100 分
共有题型：4 种

一、单选　共 5 题 （共计 20 分）
第 1 题 （4.0 分）　　题号:1129　　难度:中　　第 4 章
以下有关切片描述正确的是哪几项?
1)　　能对 range 进行切片
2)　　能对切片再次进行切片
3)　　能对字符串进行切片
4)　　能对字典进行切片
5)　　能对集合进行切片
A:4,5
B:1,3,5
C:1,2,3
D:2,4,5

答案：C

第 2 题 （4.0 分）　　题号:1130　　难度:中　　第 4 章
vec=[[7,8,9],[4,5,6],[1,2,3]],[a for b in vec for a in b]生成的列表为：
A:[1,2,3,4,5,6,7,8,9]
B:[[7,4,1],[8,5,2],[9,6,3]]
C:[7,8,9,4,5,6,1,2,3]
D:[[7,8,9,4,5,6,1,2,3]]

答案：C

第 3 题 （4.0 分）　　题号:1131　　难度:中　　第 4 章
正则表达式 ‘^[a-z]{2}([a-zA-Z0-9]){4,15}(00)$’匹配的字符串哪一个说法是正确的
A:字符串的总长度为 4～15
B:字符串可以以两个大写英文字母开始
C:字符串中可以出现字符$
D:字符串必须以数字 00 结束

答案：D

第 4 题 （4.0 分）　　题号:1132　　难度:中　　第 4 章

关于以下代码，描述错误的是：

```
with open( ‘abc.txt’ ,’ r+’ ) as f:
        lines = f.readlines()
for item in lines:
        print(item)
```

A:代码功能是打印输出 abc.txt 文件内容
B:变量 item 是字符串类型对象
C:代码中应调用 f.close（）以关闭文件
D:变量 lines 是列表类型对象

答案：C

第 5 题 （4.0 分） 题号:1133 难度:中 第 4 章

对于 Python 来说，下列选项中描述正确的是：（ ）
A:当尝试访问一个没有申明的变量时，一定不会引发 NameError 异常
B:当尝试访问一个没有申明的变量时，可能会引发 NameError 异常
C:当尝试访问一个没有申明的变量时，一定不会引发异常
D:以上都不正确

答案：B

二、程序填空 共 1 题 （共计 15 分）

第 1 题 （15.0 分） 题号:60 难度:中 第 4 章

```
# -*- coding:cp936 -*-
'''-------------------------------------------------------
```

【程序填空】

```
--------------------------------------------------------
```

题目：将股票按照股票的价格从高到低输出。运行样例见样张图片。
程序中有三处代码不完整，请填写完整。

```
--------------------------------------------------------
```

注意：除要求填空的位置之外，请勿改动程序中的其他内容。

```
--------------------------------------------------------'''
def main():
    #股票名称字符串，用","分割，名称前后可能有多余的空格
    stocks=" 中国软件, 金鹰股份 ,财通证券,中电电机, 生益科技,通用股份,  大唐电信,永创智能"

    #股票价格字符串，用","分割，价格与上面股票一一对应
    prices="50.41, 6.61, 10.91, 12.78, 13.33, 7.94, 12.91, 7.96"

    #**********SPACE**********
    【?】
```

```
    #**********SPACE**********
    【?】

    pairs=zip(priceList,stockList)

    #**********SPACE**********
    【?】
        print('{0[1]}({0[0]:.2f}元)'.format(item))


if __name__ == '__main__':
    main()
```

答案：

```
=======(答案1)=======
stockList=[stock.strip() for stock in stocks.split(',') ]

=======(答案2)=======
priceList=[float(price) for price in prices.split(',') ]

=======(答案3)=======
for item in sorted(pairs,reverse=True):
```

三、程序改错　共1题 （共计15分）

第1题 （15.0分）　　题号:59　　难度:中　　第4章

```
# -*- coding:cp936 -*-
'''-------------------------------------------------------
```

【程序改错】

下面的程序实现以下功能：

1. 函数f用于检查所传递的多行字符串text中的每行内容是否为一个魔法数，如果为魔法数，
则输出其对应的行号，提取出的候选魔法数，以及其魔法值。

2. 如果一行的内容中除了空格、制表符、连字符、逗号外，剩下的全部都是数字，则这些数字组合在一起就是该行提取出的"候选魔法数"。
比如456, 556 , 790提取后的候选魔法数为456556790

3. 一个候选魔法数是否是真正的魔法数，是要判断其魔法值是否为奇数。魔法值按照下述方法

计算：考虑候选魔法数的奇数位数字以及其相邻的偶数位数字，其中奇数位作为底，偶数位作为
幂，如果已经是最后一个数字，则等同于幂为 1。这些幂运算的结果相加，各位数字之和就是其"魔法值"。
比如 456556790 的魔法值的计算：
4 ** 5 + 6 ** 5 + 5 ** 6 + 7 ** 9 + 0 ** 1 = 40378032
4 + 0 + 3 + 7 + 8 + 0 + 3 + 2 = 27
--'''

```
import re

def f(text):
    sols = []
    lines = text.splitlines()

    #**********FOUND**********
    for lineno, line in range(len(lines)),lines:
        normalized_line = re.sub('[\s,-]', '', line)
        if not normalized_line.isdigit():
            continue

        total = 0
        for i in range(0, len(normalized_line), 2):
              #**********FOUND**********
              if i < len(normalized_line) :
                     total += int(normalized_line[i]) ** int(normalized_line[i+1])
              else:
                     total += int(normalized_line[i])

        total = sum(int(i) for i in str(total))

        if total % 2:
            #**********FOUND**********
            sols = sols.append(lineno, total, normalized_line)
        for item in sols:
            print('[{:^3d}] {:3d} {}'.format(*item))

if __name__ == '__main__':
        text =  '''456-556-778
        456, 556 , 790
        ab 456
        45655679
        1234
        0
```

```
        ,,,
"""调用如下函数后的输出结果应该为:
[ 1 ]  41 456556778
[ 2 ]  27 456556790
[ 4 ]  27 45655679
"""
f(text)
```

答案：

```
=======(答案 1)=======
for lineno, line in enumerate(lines, 1):

=======(答案 2)=======
if i < len(normalized_line) - 1:
=========或=========
if i + 1 < len(normalized_line):

=======(答案 3)=======
sols.append((lineno, total, normalized_line))
=========或=========
sols.append([lineno, total, normalized_line])
```

四、程序设计　共 2 题 （共计 50 分）

第 1 题 （25.0 分）　　题号:88　　难度:中　　第 4 章

```
# -*- coding:cp936 -*-
'''------------------------------------------------------
【程序设计】
------------------------------------------------------
题目：模拟抢红包。输入数据是一行，含有一个浮点数和一个整数，
分别表示红包总金额 amount 和数量 n。输出 n 个浮点数，分别表示
每个人抢到的金额（至少 0.01），且总和必须等于 amount。
假定用户的键盘输入没有错误，程序无需对其进行错误处理。
提示：使用 random.randint(a,b)可以得到整数 a 到 b 之间的一个
随机整数（包含 a 和 b）。
------------------------------------------------------
注意：仅在注释标志之间填入所编写的若干语句，请勿改动其余部分。
------------------------------------------------------'''
import random
```

```
def dispatch(amount, n):
    """函数参数为红包总金额 amount 和数量 n。
    返回一个列表，包含 n 个浮点数，分别表示每个人抢到的金额
    （至少 0.01），且总和必须等于 amount。"""
    #**********Program**********









    #**********  End  **********
amount=float(input('请输入红包总金额：'))
n=int(input('请输入红包数量：'))
answer=dispatch(amount, n)
for x in answer:
    print('%.2f'%x, end=' ')
```

答案：

```
    a=[]
    for i in range(n-1):
        t=random.randint(1,int(100*amount)-(n-1-i))/100 #预留 1 分钱给最后每个红包
        a.append(t)
        amount-=t
    a.append(amount)
    return a
```

第 2 题 （25.0 分） 题号:87 难度:中 第 4 章

```
# -*- coding:cp936 -*-
"""
```

本题统计电影信息。统计过程分解为多个步骤，要求实现如下两个函数：

1) 函数 construct_movies_by_director(data_list)
参数 data_list 是一个每个元素形如"(年份，电影名，票房，导演)"元组的列表，见代码中的 data_list 对象。函数处理
传入的列表对象 data_list，得到并返回一个字典。返回的字典，key 是导演，value 是每项形如"(年份，电影名，票房)"
元组的电影信息列表。在处理过程中，如果同样的电影信息出现多次，则抛出异常"重复数据"。代码中的 data_list 列表
对象在处理后得到的字典如下
{'Ron Howard': [(2013, 'Rush', 26.9), (2001, 'A Beautiful Mind', 171.0)], 'Steve

McQueen': [(2008, 'Hunger', 154.0)]}

2) 函数 top_directors(movie_dict)
参数 movie_dict 为调用函数 construct_movies_by_director 返回的结果。经处理后返回一个列表，其每个元素是形如
"(导演，总票房)"的元组，其中总票房是该导演所有执导电影的票房之和。且返回的列表按总票房从大到小的顺序排序。
上面的字典，经本函数处理后返回的结果如下
[('Ron Howard', 197.9), ('Steve McQueen', 154.0)]

注意：仅在注释标志之间填入所编写的若干语句，请勿改动其余部分。

"""

data_list = [(2013, 'Rush', 26.9, 'Ron Howard'), (2001, 'A Beautiful Mind', 171, 'Ron Howard'), (2008, 'Hunger', 154, 'Steve McQueen')]

#**********Program**********

#********** End **********

```
if __name__ == '__main__':
    try:
        movie_dict = construct_movies_by_director(data_list)
        sorted_list = top_directors(movie_dict)
        print(sorted_list)
    except Exception as e:
        print(e)
```

答案：
#答案
```
def construct_movies_by_director(data_list):
    movies = {}
    for line in data_list:
        title, director = line[1], line[-1]
```

```
            year, box_office = int(line[0]), float(line[2])

            if director not in movies:
                movies[director] = []
            if (year, title, box_office) in movies[director]:
                raise Exception("重复数据:{}".format(title))
            movies[director] += [(year, title, box_office)]
        return movies

def top_directors(movie_dict):
    directors = []
    for director, lst in movie_dict.items():
        total = 0;
        for _, _, box_office in lst:
            total += box_office
        directors.append( (director, total) )
    return directors
```