

复旦大学计算机科学技术学院

2022-2023 学年第一学期期末试卷

课程名称：_____ 课程代码：_____

卷 别： ☒ A 卷 ☐ B 卷 ☐ C 卷

姓 名：_____ 学 号：_____

提示：请同学们秉持诚实守信宗旨，谨守考试纪律，摒弃考试作弊。学生如有违反学校考试纪律的行为，学校将按《复旦大学学生纪律处分条例》规定予以严肃处理。

题号	一			二		三	四			总分
	1	2	3	1	2	1	1	2	3	
得分										

一、阅读程序写输出（30%）

第 1 题（10%）

```
N = 12
items = list(reversed(range(N)))
print(items[4:8])
items[::-3] = range(N,0,-3)
print(items[-14:-8])
```

第 2 题（10%）

```
list1 = [[1,2,3],[4,5,6],[7,8,9]]
list2 = [[j[i] for j in list1] for i in range(3)]
print(list2)
```

第 3 题（10%）

```
import re
pat = r"([^\s,.\s]+?)大学"
text = "参加的大学生来自复旦大学、清华大学、中国科技大学，还有北京大学等多所大学。"
t = re.findall(pat, text)
print(t[1])
```

二、程序填空（20%）

第1题（10%）

'''

本代码的功能为分析文件中的单词数量（这里的单词为连续的字母串），统计输出出现次数最多的单词，请补全如下代码。

'''

```
import re
def read_file():
    try:
        with open(input('请输入需要分析的文件'), 'r', encoding='utf-8')
as f:
        lines = f.readlines()
        word_list = []
        for line in lines:
            line_words = re.findall(r'\b[a-zA-Z]+\b', line)
            word_list.extend(line_words)
        return word_list
    except FileNotFoundError:
        return list()

def gen_dict(word_list):
    word_dict = {}
    for word in word_list:
        if _____ 1 _____: #填空 1
            word_dict[word] = 1
        else:
            word_dict[word] += 1
    return word_dict

def word_analyse(word_dict):
    max_words = []
    max_count = max(word_dict.values())
    for _____ 2 _____: #填空 2
        if count == max_count:
            max_words.append(word)

print('出现最多的单词是', *max_words)
print('次数为', max_count)
```

```
if __name__ == '__main__':
    word_analyse(gen_dict(read_file()))
```

第 2 题 (10%)

```
'''
```

录入几位学生的语文、数学、英语三门课的成绩，并输出每位学生的平均分等信息。
阅读程序，完成填空。

```
'''
```

```
scores={}          # 使用字典保存学生成绩
```

```
def add_student(name, chinese, math, english):
    scores[name] = (chinese, math, english)
```

```
add_student("王一", 85, 50, 92)
```

```
add_student("黄二", 92, 70, 62)
```

```
add_student("张三", 87, 90, 74)
```

```
add_student("李四", 90, 48, 87)
```

```
add_student("郭五", 70, 75, 57)
```

```
# 输出数学成绩不及格(小于 60 分) 的学生
```

```
print("数学不及格的学生有: ")
```

```
for record in scores.items():
```

```
    (1)
```

```
        print("%s" % record[0])
```

```
# 计算每个学生的三门课程的平均分(保留小数点后一位)，更新学生信息，并按格式输出
```

```
print("\n 学生信息: ")
```

```
for name in scores:
```

```
    (2)
```

```
        scores[name]=scores[name] + (average,)
```

```
        print(name, ", ".join([str(x) for x in scores[name]]))
```

输出结果:

数学不及格的学生有:

王一
李四

学生信息:

王一 85, 50, 92, 75.7

黄二 92, 70, 62, 74.7

张三 87, 90, 74, 83.7

李四 90, 48, 87, 75.0

郭五 70, 75, 57, 67.3

提示：本程序可能调用函数 `round(number, digits)`，该函数返回浮点数 `number` 的四舍五入值，`digits` 表示保留的小数点位数。

三、程序改错（10%）

函数 `f(seq)` 实现了如下功能：

- (1) 要求 `seq` 是列表或者元组对象，如果不是这两种类型的对象，则抛出异常 `ValueError`。
- (2) 判断 `seq` 中的元素是否已经按照从小到大的排序好，如果已经排序好，则返回 `True`，否则返回 `False`。
- (3) 对于空的列表或元组对象，认为其已经排序好。

下面给出了一些调用的示例和期待的结果：

```
f(12345)           # 异常 ValueError 被抛出
f([])              # True
f([1])             # True
f((1, 3, 2))       # False
f([1, 1, 2, 3, 3]) # True
```

提示：`isinstance(object, classinfo)` 函数可以用来判断一个对象是否是一个已知的类型，其中参数 `object` 为实例对象，参数 `classinfo` 可以是直接或间接类名、基本类型或者由它们组成的元组。

下述代码中两处有错误，请在后面的表格中填写错误行的位置（每行最右边注释里的数字。如果代码行后面没有注释，表示该行已经确定不会出错）和改正后的整行代码，改正后的代码只能是一行，不允许跨越多行。

```
def f(seq):
    if not isinstance(seq, list) or not isinstance(seq, tuple):      # 1
        raise ValueError('seq should be a tuple or list')          # 2

    if not seq:                                                        # 3
        return True                                                  # 4

    for idx in range(len(seq)):                                       # 5
        if seq[idx] > seq[idx + 1]:                                  # 6
            return False                                             # 7
    return True                                                        # 8
```

四、编程（40%）

第1题（10%）

编写一个函数 `skip(s)`，将字符串 `s` 中的元音字母及空格去掉，并把所有字母变为小写。

例如原字符串是 'The 1st step is as good as half over. '，则返回的字符串为 'th1ststpssgdshlfvr.'

提示：（1）元音字母包括 `a, e, i, o, u`；（2）可以调用 `lower()` 方法。

注意：部分源程序给出如下。请勿修改程序其他部分的内容，仅需编写在 `Program`，`End` 注释标志之间的若干语句。

```
*****Program*****
```

```
***** End *****
```

```
s='The 1st step is as good as half over. '
print(skip(s))
```

第2题（10%）

对于一个正整数 `n`，定义其逆序数为正整数 `n` 的各个数字逆序组成的整数。比如正整数 `123`，其逆序数为 `321`。若一个正整数（首位不为零）从左向右读与从右向左读都一样，我们就将其称为回文数。比如 `121` 为回文数。

回文数猜想：

对于一个正整数 `n`，记其逆序数为 `n'`，令 `n=n+n'`，得到新的 `n` 和 `n'`，如此重复若干步后，`n` 一定会成为回文数。例如：`n` 为 `37`，则经过以下两步，即可得到回文数 `121`。

`37+73=110`

`110+11=121`

编写程序，输入一个正整数，输出得到回文数的具体步骤。若输入不是正整数，程序应予以提示并要求重新输入。运行示例如下：

```
Enter an integer: ten
Input error.
Enter an integer: 3.14
Input error.
Enter an integer: 87
87+78=165
165+561=726
726+627=1353
1353+3531=4884
```

第 3 题 (20%)

billsList.html 文本文件 (如下面的左图所示) 给出了最近多个月份的账单信息, 关于月份的信息对应其中的一行内容, 该行内容以标签<p>开始, 以标签</p>结尾, 给出了月份、该月份的账单文件的路径名以及消费总金额。

某个月份的账单文件的内容 (如下面的右图所示), 其中那些以<td>标签开始 (前面允许有多个空格), 以</td>标签结束的行给出了该月的消费明细, 包括消费项目及消费金额。

文本文件 billsList.html 内容	文本文件 Januray.html 内容
<pre><html> <head> <title>Bills </title> </head> <body bgcolor="#ffffff" text="#000000"> <table width="100%" bgcolor="#CCF6F6"> <tbody><tr><td class="play" align="left">Bills List </td></tr><tr><td class="nav" align="center"> </td></tr></tbody></table> <p>Januray:Total:\$1001.50
</p> <p>February:Total:\$5078.80
</p> <p>March:<Total:1></Total:1>\$1850.80
</p> <p>April:Total:\$1110.40
</p> <p>May:Total:\$2364.50
</p> <p>June:Total:\$6780.40
</p> </body> </html></pre>	<pre><html> <body> <table id="customers" > <caption>Januray</caption> <tr> <th width=30%>Expenses</th><th width=70%>Amount</th> </tr> <tr> <td>Books</td><td>\$241.10</td> </tr> <tr> <td>Pens</td><td>\$30.00</td> </tr> <tr> <td>Food</td><td>\$730.40</td> </tr> <tr> <th>Total</th><th>\$1001.50</th> </tr> </table> </body> </html></pre>

请完成 **problem3.py** (和 **billsList.html** 同一目录), 该程序首先分析 **billsList.html** 得到最近的月份以及对应的账单文件。然后进一步分析那些账单文件, 获得该月份消费最多的项目及金额。最后将前面两个步骤获得的信息写入到文本文件 **bills.csv** 中。bills.csv 的第一行为标题行, 后续的行对应着下述信息: 月份名、账单文件路径名、该月份消费最多的项目以及消费金额, 这 4 个字段之间以逗号隔开。对于上面的示例, 运行 **problem3.py** 后生成的文本文件 **bills.csv** 示例如下图所示:

```
Month,File,Max,Amount
Januray,./data/Januray.html,Food,730.40
February,./data/February.html,Phone,3500.00
March,./data/March.html,Rent,830.40
April,./data/April.html,Telecoms,450.00
May,./data/May.html,Clothing,1030.00
June,./data/June.html,Computer,4800.00
```

注意: 部分源程序给出如下。请勿修改程序其他部分的内容, 仅需编写在 **Program**, **End** 注释标志之间的若干语句。

```
import re
##提示: 本程序可能用的主要函数如下:
##re.findall(pattern, string, flags=0)
##sorted(iterable, /, *, key=None, reverse=False)
##open(file, mode='r', buffering=-1, encoding=None, errors=None)

def getBillsList(fileName):
    """
    该函数可解析文件 billsList.html。返回下列列表
```

```

[('Januray', './data/Januray.html'), ('February', './data/February.html'),
('March', './data/March.html'), ('April', './data/April.html'),
('May', './data/May.html'), ('June', './data/June.html')]
"""

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

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

def getMaxItem(fileName):
    """
    该函数可解析每月的账单文件，以元组的方式返回该月消费最多的(消费项目，金额)
    如解析 Januray.html 文件，返回("Food","730.40")
    """
    #*****Program*****

    #***** End *****
if __name__ == '__main__':
    fileName = "billsList.html"
    csvName = 'bills.csv'

    billsList = getBillsList(fileName)
    #根据 billsList 提供的账单文件的文件名，分析每个月账单文件，
    #得到每月消费最多的消费项目与金额，并将月份，文件名，消费最多的消费项目与金额等
    #信息写入 bills.csv，bills.csv 文件具体内容如上述所示。
    #*****Program*****

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

```