实验五 Python数据结构与数据模型

班级： 21计科04

学号： B20210302430

姓名： 徐享

Github地址：https://github.com/zzldy4/python-homework

CodeWars地址：https://www.codewars.com/users/zzldy

实验目的

1. 学习Python数据结构的高级用法
2. 学习Python的数据模型

实验环境

1. Git
2. Python 3.10
3. VSCode
4. VSCode插件

实验内容和步骤

第一部分

在Codewars网站注册账号，完成下列Kata挑战：

第一题：停止逆转我的单词

难度： 6kyu

编写一个函数，接收一个或多个单词的字符串，并返回相同的字符串，但所有5个或更多的字母单词都是相反的（就像这个Kata的名字一样）。传入的字符串将只由字母和空格组成。只有当出现一个以上的单词时，才会包括空格。
例如：

```
spinWords( "Hey fellow warriors" ) => returns "Hey wollef sroirraw"
spinWords( "This is a test") => returns "This is a test"
spinWords( "This is another test" )=> returns "This is rehtona test"
```

代码提交地址：
https://www.codewars.com/kata/5264d2b162488dc400000001

提示：

- 利用str的split方法可以将字符串分为单词列表
  例如：

```
words = "hey fellow warrior".split()
# words should be ['hey', 'fellow', 'warrior']
```

- 利用列表推导将长度大于等于5的单词反转(利用切片word[::-1])
- 最后使用str的join方法连结列表中的单词。

第二题：发现离群的数(Find The Parity Outlier)

难度：6kyu

给你一个包含整数的数组（其长度至少为3，但可能非常大）。该数组要么完全由奇数组成，要么完全由偶数组成，除了一个整数N。请写一个方法，以该数组为参数，返回这个 "离群 "的N。

例如：

```
[2, 4, 0, 100, 4, 11, 2602, 36]
# Should return: 11 (the only odd number)

[160, 3, 1719, 19, 11, 13, -21]
# Should return: 160 (the only even number)
```

代码提交地址：
https://www.codewars.com/kata/5526fc09a1bbd946250002dc

第三题： 检测Pangram

难度：6kyu

pangram是一个至少包含每个字母一次的句子。例如，"The quick brown fox jumps over the lazy dog "这个句子就是一个pangram，因为它至少使用了一次字母A-Z（大小写不相关）。

给定一个字符串，检测它是否是一个pangram。如果是则返回`True`，如果不是则返回`False`。忽略数字和标点符号。
代码提交地址：
https://www.codewars.com/kata/545cedaa9943f7fe7b000048

第四题： 数独解决方案验证

难度：6kyu

数独背景

数独是一种在 9x9 网格上进行的游戏。游戏的目标是用 1 到 9 的数字填充网格的所有单元格，以便每一列、每一行和九个 3x3 子网格（也称为块）中的都包含数字 1 到 9。更多信息请访问：http://en.wikipedia.org/wiki/Sudoku

编写一个函数接受一个代表数独板的二维数组，如果它是一个有效的解决方案则返回 true，否则返回 false。数独板的单元格也可能包含 0，这将代表空单元格。包含一个或多个零的棋盘被

认为是无效的解决方案。棋盘总是 9 x 9 格，每个格只包含 0 到 9 之间的整数。

代码提交地址：
https://www.codewars.com/kata/63d1bac72de941033dbf87ae

第五题：疯狂的彩色三角形

难度： 2kyu

一个彩色的三角形是由一排颜色组成的，每一排都是红色、绿色或蓝色。连续的几行，每一行都比上一行少一种颜色，是通过考虑前一行中的两个相接触的颜色而产生的。如果这些颜色是相同的，那么新的一行就使用相同的颜色。如果它们不同，则在新的一行中使用缺失的颜色。这个过程一直持续到最后一行，只有一种颜色被生成。

例如：

```
Colour here:            G G        B G        R G        B R
Becomes colour here:     G          R          B          G
```

一个更大的三角形例子：

```
R R G B R G B B
 R B R G B R B
  G G B R G G
   G R G B G
    B B R R
     B G R
      R B
       G
```

你将得到三角形的第一行字符串，你的工作是返回最后的颜色，这将出现在最下面一行的字符串。在上面的例子中，你将得到 "RRGBRGBB"，你应该返回 "G"。
限制条件： 1 <= length(row) <= 10 ** 5
输入的字符串将只包含大写字母'B'、'G'或'R'。

例如：

```
triangle('B') == 'B'
triangle('GB') == 'R'
triangle('RRR') == 'R'
triangle('RGBG') == 'B'
triangle('RBRGBRB') == 'G'
triangle('RBRGBRBGGRRRBGBBBGG') == 'G'
```

代码提交地址：
https://www.codewars.com/kata/5a331ea7ee1aae8f24000175

提示：请参考下面的链接，利用三进制的特点来进行计算。
https://stackoverflow.com/questions/53585022/three-colors-triangles

---

第二部分

使用Mermaid绘制程序流程图

安装VSCode插件：

- Markdown Preview Mermaid Support
- Mermaid Markdown Syntax Highlighting

使用Markdown语法绘制你的程序绘制程序流程图（至少一个），Markdown代码如下：

程序流程图

显示效果如下：

flowchart LR A[Start] --> B{Is it?} B -->|Yes| C[OK] C --> D[Rethink] D --> B B ---->|No| E[End]
查看Mermaid流程图语法-->点击这里

使用Markdown编辑器（例如VScode）编写本次实验的实验报告，包括实验过程与结果、实验考查和实验总结，并将其导出为 **PDF格式** 来提交。

实验过程与结果

请将实验过程与结果放在这里，包括：

- 第一部分 Codewars Kata挑战
- 第二部分 使用Mermaid绘制程序流程图

注意代码需要使用markdown的代码块格式化，例如Git命令行语句应该使用下面的格式：

Git命令

显示效果如下：

```
git init
git add .
git status
git commit -m "first commit"
```

如果是Python代码，应该使用下面代码块格式，例如：

Python代码

显示效果如下：

```
def add_binary(a,b):
    return bin(a+b)[2:]
```

第一题：
点击跳转题目说明

```
def spin_words(sentence):
    lists=sentence.split()
    ret=[]
    for list in lists:
        if len(list)>=5:
            ret.append(''.join(reversed(list)))
        else:
            ret.append(list)
        ret.append(' ')
    ret.pop()
    return ''.join(ret)
```

第二题：

点击跳转题目说明

```
def find_outlier(arr):
    a_numbers = []
    b_numbers = []
    for num in arr:
        if num % 2 == 0:
            b_numbers.append(num)
        else:
            a_numbers.append(num)
    if len(a_numbers) == 1:
        return a_numbers[0]
    elif len(b_numbers) == 1:
        return b_numbers[0]
    else:
        return None
```

第三题：

点击跳转题目说明

```
def is_pangram(s):
    s=s.lower()
    lists=['a','b','c','d','e','f','g','h','i','j','k','l',
        'm','n','o','p','q','r','s','t','u','v','w','x','y','z']
    for list in lists:
        if list not in s:
            return False
    return True
```

第四题：

点击跳转题目说明

```
def validate_sudoku(board):
    for i in range(9):
        if sorted(board[i]) != list(range(1, 10)):
            return False

        column = [row[i] for row in board]
        if sorted(column) != list(range(1, 10)):
            return False

    for block_row in range(0, 9, 3):
        for block_col in range(0, 9, 3):
            block = [board[x][y] for x in range(block_row, block_row + 3) for y in range(block_col
            if sorted(block) != list(range(1, 10)):
                return False

    return True
```

第五题：

点击跳转题目说明

```
def triangle(row):
    color = {'GG': 'G', 'BB': 'B', 'RR': 'R', 'BR': 'G', 'BG': 'R', 'GB': 'R', 'GR': 'B', 'RG': 'B

    if len(row) == 1:
        return row

    new_row = ''
    for i in range(len(row) - 1):
        new_row += color[row[i:i+2]]

    return triangle(new_row)
```

- 第一题

graph TD A[输入句子] -->|分割为单词| B(遍历单词) B -->|单词长度 >= 5| C{反转单词} C -->|是| D[追加反转后的单词] C -->|否| E[追加原始单词] D --> F(下一个单词) E --> F F -->|结束单词| G[连接单词] G -->|输出| H[生成的句子]

- 第二题

graph TD A[输入数组] -->|遍历元素| B(检查奇偶性) B -->|偶数| C{追加到偶数数组} C -->|是| D[下一个元素] C -->|否| E{追加到奇数数组} E -->|是| D E -->|否| F[判断奇数数组长度] F -->|长度1| G[返回奇数数组元素] F -->|长度不为1| H{判断偶数数组长度} H -->|长度1| I[返回偶数数组元素] H -->|长度不为1| J[返回空值] I --> J D -->|结束遍历| K[返回空值]

- 第三题

graph TD A[输入字符串] -->|转换为小写| B(初始化字母列表) B -->|遍历字母列表| C{检查字母是否在字符串中} C -->|是| D[下一个字母] C -->|否| E[返回False] D -->|结束遍历| F[返回True]

- 第四题

graph TD A[输入数独数组] -->|检查每行| B(检查行是否包含1到9的所有数字) B -->|否| C[返回False] B -->|是| D[检查每列] D -->|否| C D -->|是| E[检查每3x3块] E -->|否| C E -->|是| F[返回True]

- 第五题

graph TD A[输入字符串] -->|长度为1| B[返回字符串] A -->|长度不为1| C(循环遍历字符串) C -->|判断颜色组合| D[追加颜色到新字符串] D --> E[递归调用triangle] E --> A
代码运行结果的文本可以直接粘贴在这里。

注意：不要使用截图，因为Markdown文档转换为Pdf格式后，截图会无法显示。

实验考查

请使用自己的语言并使用尽量简短代码示例回答下面的问题，这些问题将在实验检查时用于提问和答辩以及实际的操作。

1. 集合（set）类型有什么特点？它和列表（list）类型有什么区别？

特点：

- 无序：集合中的元素没有固定的顺序。
- 唯一性：集合中的元素是唯一的，不允许重复。
- 可变性：可以添加或删除元素。

区别：

列表是有序的，可以包含重复元素；而集合是无序的，不包含重复元素。

2. 集合（set）类型主要有那些操作？

- 增加元素：add
- 删除元素：remove, discard
- 集合运算：并集(union), 交集(intersection), 差集(difference), 对称差集(symmetric_difference)等。

3. 使用`*`操作符作用到列表上会产生什么效果？为什么不能使用`*`操作符作用到嵌套

的列表上？使用简单的代码示例说明。

4. *操作符在列表上重复元素，将列表中的元素重复指定的次数。
5. 不能用于嵌套列表，因为*操作符只会复制对列表的引用，而不会创建新的嵌套列表。修改一个嵌套列表的元素会影响所有引用该嵌套列表的变量

```
# 列表重复
original_list = [1, 2, 3]
repeated_list = original_list * 3
print(repeated_list)  # 输出: [1, 2, 3, 1, 2, 3, 1, 2, 3]

# 嵌套列表问题
nested_list = [[0, 1]] * 3
nested_list[0][0] = 99
print(nested_list)  # 输出: [[99, 1], [99, 1], [99, 1]]
```

4. 总结列表,集合，字典的解析（comprehension）的使用方法。使用简单的代码示例说明。

列表解析：

```
Copy code
# 创建平方数列表
squares = [x**2 for x in range(5)]
```

集合解析：

```
Copy code
# 创建平方数集合
square_set = {x**2 for x in range(5)}
```

字典解析：

```
Copy code
# 创建字典，键为数字，值为数字的平方
square_dict = {x: x**2 for x in range(5)}
```

实验总结

总结一下这次实验你学习和使用到的知识，例如：编程工具的使用、数据结构、程序语言的语法、算法、编程技巧、编程思想。
通过这次实验学习和巩固了对列表和字典的使用，同时也更加清楚这两者之间的区别。