实验五 Python数据结构与数据模型

班级： 21计科4

学号： 20210301105

姓名： 张湘睿

Github地址：https://github.com/ttZhang0512/PythonClassTasks.git

CodeWars地址：https://www.codewars.com/users/ttZhang0512

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

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

第一题

```
def spin_words(sentence):
    words = sentence.split()
    re = []
    for word in words:
        if len(word)>=5:
            re.append(word[::-1])
        else:
            re.append(word)
    result = " ".join(re)
    return result
```

第二题

```
def find_outlier(integers):
    o_count = 0
    j_count = 0
    for integer in integers:
        if integer%2==0:
            o_count+=1
            num_o = integer
        else:
            j_count+=1
            num_j = integer
        if o_count>1 and j_count==1:
            result = num_j
            break
        if j_count>1 and o_count==1:
            result = num_o
            break
    return result
```

第三题

```
def is_pangram(s):
    s = s.lower()
    for character in 'abcdefghijklmnopqrstuvwxyz':
        if character not in s:
            return False
    return True
```

第四题

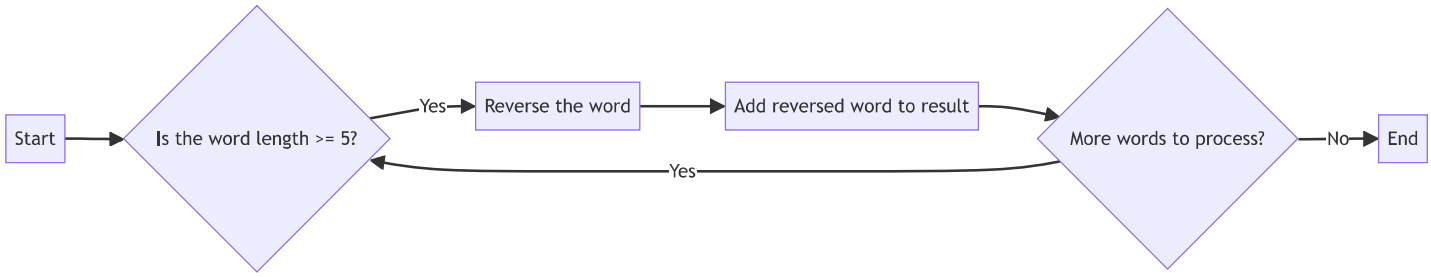
```
def validate_sudoku(board):
    nums = set(range(1,10))
    for i in board:
        if set(i)!=nums:
            return False

    for j in zip(*board):
        if set(j)!=nums:
            return False

    for i in range(0,9,3):
        for j in range(0,9,3):
            if nums!= {(board[x][y])
                for x in range(i,i+3)
                    for y in range(j,j+3)
            }:
                return False
    return True
```

第五题

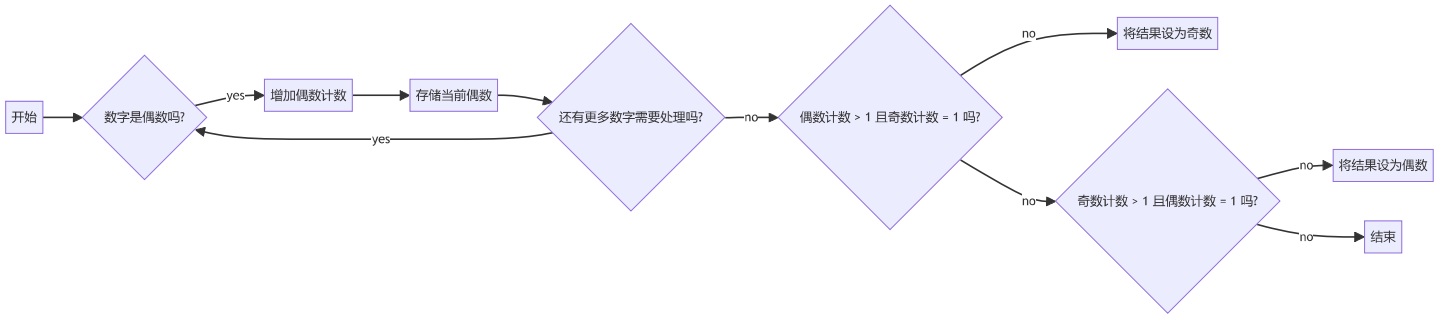
```
def triangle(row):
    reduce = [3**i+1 for i in range(10) if 3**i <= len(row)][::-1]

    COLOR = {'GG': 'G', 'BB': 'B', 'RR': 'R', 'BR': 'G',
            'BG': 'R', 'GB': 'R', 'GR': 'B', 'RG': 'B', 'RB': 'G'}

    for length in reduce:
        while len(row) >= length:
            row = [COLOR[row[i] + row[i+length-1]] for i in range(len(row)-length+1)]

    return row[0]
```

- 第二部分 使用Mermaid绘制程序流程图

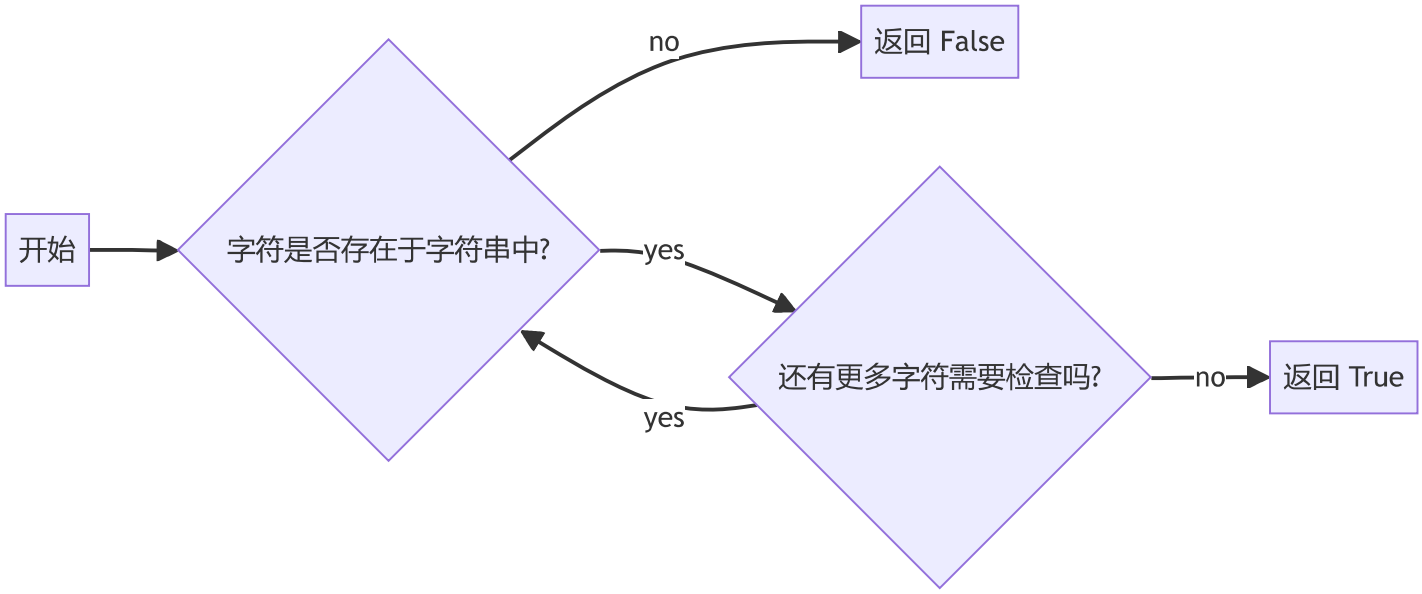
第一题



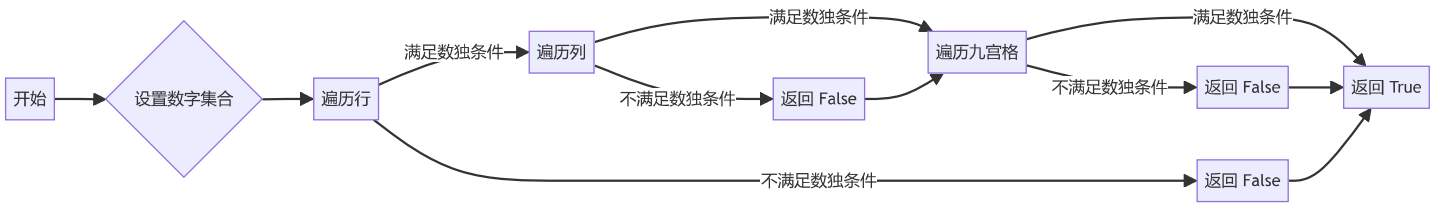
第二题



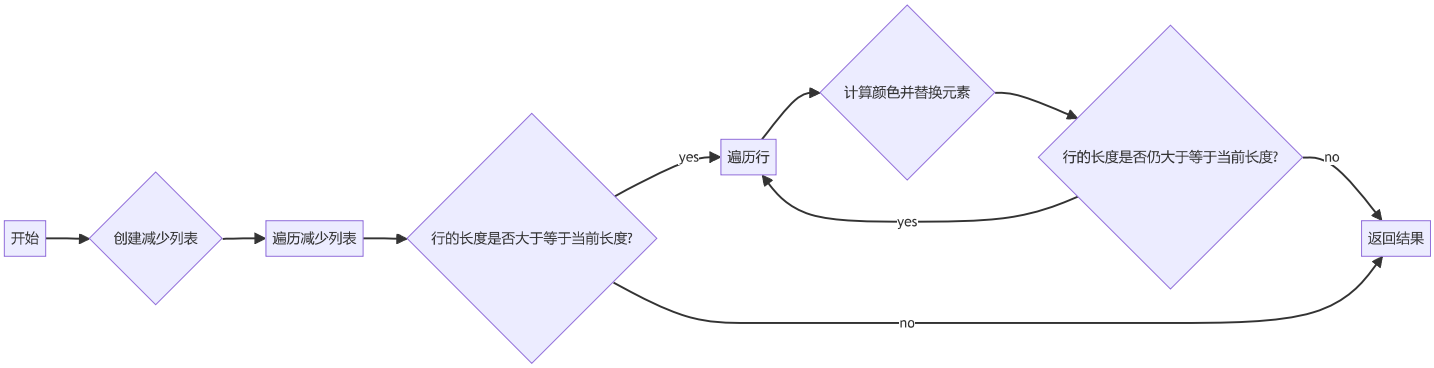
第三题



第四题



第五题



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？
   -- 集合是一种无序的数据集合，其中的元素是唯一的、无序的、不可修改的。可进行交、并、补等集合操作，不可利用索引访问。
   -- 列表（list）类型与集合的区别：列表是有序的数据集合，可以通过索引访问和修改元素。列表的元素可以重复，而集合中的元素是唯一的。
2. 集合（set）类型主要有那些操作？
   添加元素：使用 add() 方法向集合中添加元素。移除元素：使用 remove() 或 discard() 方法从集合中移除元素。集合运算：可以进行交集、并集、差集等操作，如 intersection()、union()、difference() 等。成员判断：使用 in 关键字来判断元素是否在集合中存在。长度获取：使用 len() 函数获取集合的长度。
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。
   使用 * 操作符作用到列表上会产生重复列表的效果。但是不能直接使用 * 操作符作用到嵌套的列表上，因为 * 操作符只是对列表进行简单的复制，对于嵌套列表，复制的只是引用，而不是创建新的嵌套列表。
   例如，如果我们一个列表[1, 2, 3]，并且我们将*操作符应用于它。然而，当我们尝试将*操作符应用于嵌套的列表时，会出现问题。这是因为*操作符只是简单地重复列表中的元素，而不会递归地重复嵌套列表中的元素

```
my_list = [1, 2, 3]
new_list = my_list * 3
print(new_list)  #输出：[1, 2, 3, 1, 2, 3, 1, 2, 3]

nested_list = [[1, 2], [3, 4]]
new_nested_list = nested_list * 2
print(new_nested_list) #输出：[[1, 2], [3, 4], [1, 2], [3, 4]]
```

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（List Comprehension）是一种简洁的方式来创建新的列表，它允许我们使用一行代码来定义一个新的列表，而不需要使用显式的循环。列表解析的语法是在方括号内使用表达式和可选的条件语句。下面的例子，使用列表解析来创建一个包含1到10之间偶数的列表：

```
even_numbers = [x for x in range(1, 11) if x % 2 == 0]
print(even_numbers)  #输出：[2, 4, 6, 8, 10]
```

集合解析（Set Comprehension）与列表解析类似，但是它创建的是一个集合（set）。集合解析的语法是在花括号内使用表达式和可选的条件语句。 下面的例子，使用集合解析来创建一个包含1到10之间奇数的集合：

```
odd_numbers = {x for x in range(1, 11) if x % 2 != 0}
print(odd_numbers)   #输出：{1, 3, 5, 7, 9}
```

字典解析（Dictionary Comprehension）是一种创建新字典的方式，它允许我们使用一行代码来定义一个新的字典，而不需要使用显式的循环。字典解析的语法是在花括号内使用键值对表达式和可选的条件语句。

```
squared_numbers = {x: x**2 for x in range(1, 6)}
print(squared_numbers)      #输出：{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

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。
在这次实验中，我通过学习集合、列表和字典，掌握了不同类型的数据结构在Python中的使用方法。同时，还学习了这些数据结构的常见操作和方法，从而能更加灵活地操作和处理数据。