

# 实验五 Python数据结构与数据模型

---

班级：21计科3

学号：20210302302

姓名：蒋俊杰

Github地址：[https://github.com/jiangjunjie666/python\\_study](https://github.com/jiangjunjie666/python_study)

CodeWars地址：<https://www.codewars.com/users/jiangjunjie666>

---

## 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 game。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 "RRGBRBBB"，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：

```
graph 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:]
```

代码运行结果的文本可以直接粘贴在这里。

### 1.停止逆转我的单词

```
def spinWords(str):  
    words = str.split(' ')  
    new_words = []  
    for word in words:  
        if len(word) >= 5:  
            new_words.append(word[::-1])  
        else:  
            new_words.append(word)  
    return ' '.join(new_words)
```

### 2.发现离群的数

```
def find_outlier(integers):  
    # 分别统计奇数和偶数的个数  
    odd_count = 0  
    even_count = 0  
    last_odd = 0  
    last_even = 0  
  
    for num in integers:  
        if num % 2 == 0:  
            even_count += 1  
            last_even = num  
        else:  
            odd_count += 1  
            last_odd = num  
  
    # 如果已经找到两个奇数或两个偶数，那么最后一个不同类型的数就是离群的N  
    if even_count >= 2 and odd_count == 1:  
        return last_odd  
    elif odd_count >= 2 and even_count == 1:  
        return last_even
```

```
# 如果没有找到离群的N, 那么根据题意返回最后一个元素
return last_even if even_count == 1 else last_odd
```

### 3.检测pangram

```
def is_pangram(sentence):
    # 将句子中的所有字母转换为小写, 以便大小写不相关
    sentence = sentence.lower()

    # 创建一个包含所有字母的集合
    alphabet = set("abcdefghijklmnopqrstuvwxyz")

    # 遍历句子中的字符, 将其添加到一个集合中
    # 一旦集合包含了所有字母, 就返回True
    for char in sentence:
        if char.isalpha():
            alphabet.discard(char)
            if not alphabet:
                return True

    # 如果遍历完成后集合为空, 则句子是pangram, 否则不是
    return not bool(alphabet)
```

### 4.数独

```
import numpy as np

def validate_sudoku(board):
    sudoku = np.array(board)

    def is_valid_unit(unit):
        unique_numbers = np.unique(unit)
        return len(unique_numbers) == 9 and np.all(unique_numbers[unique_numbers
!= 0] == np.arange(1, 10))

    # 验证每一行
    for row in sudoku:
        if not is_valid_unit(row):
            return False

    # 验证每一列
    for col in sudoku.T:
        if not is_valid_unit(col):
            return False

    # 验证每个3x3子网格
    for i in range(0, 9, 3):
```

```

        for j in range(0, 9, 3):
            subgrid = sudoku[i:i+3, j:j+3]
            if not is_valid_unit(subgrid):
                return False

    return True

```

**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？

- 集合是无序的，不支持索引，不允许有重复元素。
- 列表是有序的，可以通过索引访问元素，允许有重复元素。

```

my_set = {1, 2, 3, 3, 4, 5} # 集合不允许重复元素
my_list = [1, 2, 3, 3, 4, 5] # 列表允许重复元素

```

2. 集合（set）类型主要有那些操作？

- 添加元素: add()
- 删除元素: remove() 或 discard()
- 合并集合: union() 或 | 操作符
- 交集: intersection() 或 & 操作符
- 差集: difference() 或 - 操作符
- 子集检查: issubset()
- 超集检查: issuperset()

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将列表重复指定次数，但不能用于嵌套的列表

```

my_list = [1, 2, 3]
repeated_list = my_list * 3 # 重复3次: [1, 2, 3, 1, 2, 3, 1, 2, 3]

nested_list = [[1, 2], [3, 4]]
repeated_nested_list = nested_list * 2 # 错误，不能用于嵌套的列表

```

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 - 从现有列表创建一个新列表:

```
numbers = [1, 2, 3, 4, 5]
squared_numbers = [x**2 for x in numbers]
```

- 集合解析 - 从现有列表或其他可迭代对象创建一个新集合:

```
numbers = [1, 2, 2, 3, 4, 4, 5]
unique_numbers = {x for x in numbers}
```

- 字典解析 - 从现有数据创建一个新字典:

```
fruits = ["apple", "banana", "cherry"]
fruit_lengths = {fruit: len(fruit) for fruit in fruits}
```

- 条件解析 - 根据条件过滤元素:

```
numbers = [1, 2, 3, 4, 5]
even_numbers = [x for x in numbers if x % 2 == 0]
```

## 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。

编程工具的使用：我使用了Python编程语言和Jupyter Notebook作为编程工具。 数据结构：我使用了列表（list）来存储实验数据，并熟练掌握了列表的基本操作，如添加元素、删除元素、修改元素、查找元素等。 程序语言的语法：我掌握了Python编程语言的基本语法，如变量声明、条件语句、循环语句等。 算法：我掌握了冒泡排序算法、插入排序算法、快速排序算法等基本的排序算法，以及列表的逆序操作。 编程技巧：我掌握了列表的基本操作，以及如何使用循环和条件语句实现算法。 编程思想：我掌握了编程的基本思想，如模块化、面向对象编程等