

实验三 Python列表

班级： 21计科1

学号： B20210302127

姓名： 刘嘉伟

Github地址： https://github.com/PigeonDuck/python_course

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

实验目的

1. 学习Python的简单使用和列表操作
2. 学习Python中的if语句

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

Python列表操作

完成教材《Python编程从入门到实践》下列章节的练习：

- 第3章 列表简介
 - 第4章 操作列表
 - 第5章 if语句
-

第二部分

在[Codewars网站](#)注册账号，完成下列Kata挑战：

第一题：3和5的倍数 (Multiples of 3 or 5)

难度： 6kyu

如果我们列出所有低于 10 的 3 或 5 倍数的自然数，我们得到 3、5、6 和 9。这些数的总和为 23。完成一个函数，使其返回小于某个整数的所有是3 或 5 的倍数的数的总和。此外，如果数字为负数，则返回 0。

注意：如果一个数同时是3和5的倍数，应该只被算一次。

提示：首先使用列表解析得到一个列表，元素全部是3或者5的倍数。使用sum函数可以获取这个列表所有元素的和。

代码提交地址：<https://www.codewars.com/kata/514b92a657cdc65150000006>

第二题：重复字符的编码器 (Duplicate Encoder)

难度：6kyu

本练习的目的是将一个字符串转换为一个新的字符串，如果新字符串中的每个字符在原字符串中只出现一次，则为"("，如果该字符在原字符串中出现多次，则为")"。在判断一个字符是否是重复的时候，请忽略大写字母。

例如：

```
"din"      => "(((  
"recede"   => "())()  
"Success"  => ")())()  
"(( @"     => "))(("
```

代码提交地址：<https://www.codewars.com/kata/54b42f9314d9229fd6000d9c>

第三题：括号匹配 (Valid Braces)

难度：6kyu

写一个函数，接收一串括号，并确定括号的顺序是否有效。如果字符串是有效的，它应该返回True，如果是无效的，它应该返回False。例如：

```
"(){}[]" => True  
"([{}])" => True  
"{}" => False  
"[]" => False  
"[({})]()" => False
```

提示：python中没有内置堆栈数据结构，可以直接使用list来作为堆栈，其中append方法用于入栈，pop方法可以出栈。

代码提交地址：<https://www.codewars.com/kata/5277c8a221e209d3f6000b56>

第四题：从随机三元组中恢复秘密字符串(Recover a secret string from random triplets)

难度：4kyu

有一个不为你所知的秘密字符串。给出一个随机三个字母的组合的集合，恢复原来的字符串。

这里的三个字母的组合被定义为三个字母的序列，每个字母在给定的字符串中出现在下一个字母之前。"whi"是字符串 "whatisup" 的一个三个字母的组合。

作为一种简化，你可以假设没有一个字母在秘密字符串中出现超过一次。

对于给你的三个字母的组合，除了它们是有效的三个字母的组合以及它们包含足够的信息来推导出原始字符串之外，你可以不做任何假设。特别是，这意味着秘密字符串永远不会包含不出现在给你的三个字母的组合中的字母。

测试用例：

```
secret = "whatisup"
triplets = [
    ['t','u','p'],
    ['w','h','i'],
    ['t','s','u'],
    ['a','t','s'],
    ['h','a','p'],
    ['t','i','s'],
    ['w','h','s']
]
test.assert_equals(recoverSecret(triplets), secret)
```

代码提交地址：<https://www.codewars.com/kata/53f40dff5f9d31b813000774/train/python>

提示：

- 利用集合去掉 `triplets` 中的重复字母，得到字母集合 `letters`，最后的 `secret` 应该由集合中的字母组成，`secret` 长度也等于该集合。

```
letters = {letter for triplet in triplets for letter in triplet }
length = len(letters)
```

- 创建函数 `check_first_letter(triplets, first_letter)`，检测一个字母是不是 `secret` 的首字母，返回 `True` 或者 `False`。
- 创建函数 `remove_first_letter(triplets, first_letter)`，从三元组中去掉首字母，返回新的三元组。
- 遍历字母集合 `letters`，利用上面2个函数得到最后的结果 `secret`。

第五题：去掉喷子的元音 (Disemvowel Trolls)

难度：7kyu

喷子正在攻击你的评论区！处理这种情况的一个常见方法是删除喷子评论中的所有元音(字母：a,e,i,o,u)，以消除威胁。你的任务是写一个函数，接收一个字符串并返回一个去除所有元音的新字符串。例如，字符串 "This website is for losers LOL!" 将变成 "Ths wbst s fr lsrs LL!"。

注意：对于这个Kata来说，y不被认为是元音。代码提交地址：

<https://www.codewars.com/kata/52fba66badcd10859f00097e>

提示：

- 首先使用列表解析得到一个列表，列表中所有不是元音的字母。
- 使用字符串的join方法连结列表中所有的字母，例如：

```
last_name = "lovelace"
letters = [letter for letter in last_name]
print(letters) # ['l', 'o', 'v', 'e', 'l', 'a', 'c', 'e']
name = ''.join(letters) # name = "lovelace"
```


第三部分

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

第一题：3和5的倍数 (Multiples of 3 or 5)

难度：6kyu

如果我们列出所有低于 10 的 3 或 5 倍数的自然数，我们得到 3、5、6 和 9。这些数的总和为 23。完成一个函数，使其返回小于某个整数的所有是 3 或 5 的倍数的数的总和。此外，如果数字为负数，则返回 0。

注意：如果一个数同时是3和5的倍数，应该只被算一次。

提示：首先使用列表解析得到一个列表，元素全部是3或者5的倍数。使用sum函数可以获取这个列表所有元素的和。

代码提交地址：<https://www.codewars.com/kata/514b92a657cdc65150000006>

运行代码如下：

```
def solution(number):
    #pass #直接生成满足条件的列表
    list_number = [x for x in range(0,number) if x%3==0 or x%5==0 ]
    return sum(list_number)
```

第二题：重复字符的编码器 (Duplicate Encoder)

难度：6kyu

本练习的目的是将一个字符串转换为一个新的字符串，如果新字符串中的每个字符在原字符串中只出现一次，则为"("，如果该字符在原字符串中出现多次，则为")"。在判断一个字符是否是重复的时候，请忽略大写字母。

例如：

```
"din"      => "((("
"recede"   => "()()())"
"Success"  => ")()())()"
"(( @"     => "))((("
```

代码提交地址：<https://www.codewars.com/kata/54b42f9314d9229fd6000d9c>

运行代码如下：

```
def duplicate_encode(word):
    #your code here
    #提前处理大小写
    word = word.lower()
    list_num = []
    for i in range(0,len(word)):
        list_num.append('(')

    for i in range(0,len(word)):
        for j in range(0,len(word)):
            if word[j] == word[i] and j!=i :
                list_num[j]=')'
    str1 = ''.join(list_num)
    return str1
```

第三题：括号匹配 (Valid Braces)

难度：6kyu

写一个函数，接收一串括号，并确定括号的顺序是否有效。如果字符串是有效的，它应该返回True，如果是无效的，它应该返回False。例如：

```
"(){}[]" => True
"([{}])" => True
"{}" => False
"[]" => False
"[({})]" => False
```

提示：python中没有内置堆栈数据结构，可以直接使用list来作为堆栈，其中append方法用于入栈，pop方法可以出栈。

代码提交地址 <https://www.codewars.com/kata/5277c8a221e209d3f6000b56>

运行代码如下：

```
def v_b(s):
    bracks = {'(': ')', '{': '}', '[': ']'}
    stack = []
    for x in s:
        if x in bracks.keys():
            stack.append(x)
        else:
            if len(s)==0 or bracks[stack.pop()]!=x:
                return False
    return True
```

第四题：从随机三元组中恢复秘密字符串(Recover a secret string from random triplets)

难度：4kyu

有一个不为你所知的秘密字符串。给出一个随机三个字母的組合的集合，恢复原来的字符串。

这里的三个字母的組合被定义为三个字母的序列，每个字母在给定的字符串中出现在下一个字母之前。"whi"是字符串 "whatisup" 的一个三个字母的組合。

作为一种简化，你可以假设没有一个字母在秘密字符串中出现超过一次。

对于给你的三个字母的組合，除了它们是有效的三个字母的組合以及它们包含足够的信息来推导出原始字符串之外，你可以不做任何假设。特别是，这意味着秘密字符串永远不会包含不出现在给你的三个字母的組合中的字母。

测试用例:

```
secret = "whatisup"
triplets = [
    ['t', 'u', 'p'],
    ['w', 'h', 'i'],
    ['t', 's', 'u'],
    ['a', 't', 's'],
    ['h', 'a', 'p'],
    ['t', 'i', 's'],
    ['w', 'h', 's']
]
test.assertEqual(recoverSecret(triplets), secret)
```

代码提交地址: <https://www.codewars.com/kata/53f40dff5f9d31b813000774/train/python>

提示:

- 利用集合去掉triplets中的重复字母, 得到字母集合letters, 最后的secret应该由集合中的字母组成, secret长度也等于该集合。

```
letters = {letter for triplet in triplets for letter in triplet }
length = len(letters)
```

- 创建函数check_first_letter(triplets, first_letter), 检测一个字母是不是secret的首字母, 返回True或者False。
- 创建函数remove_first_letter(triplets, first_letter), 从三元组中去掉首字母, 返回新的三元组。
- 遍历字母集合letters, 利用上面2个函数得到最后的结果secret。

运行代码如下:

第五题: 去掉喷子的元音 (Disemvowel Trolls)

难度: 7kyu

喷子正在攻击你的评论区! 处理这种情况的一个常见方法是删除喷子评论中的所有元音(字母: a,e,i,o,u), 以消除威胁。你的任务是写一个函数, 接收一个字符串并返回一个去除所有元音的新字符串。例如, 字符串 "This website is for losers LOL!" 将变成 "Ths wbst s fr lsrs LL!"。

注意：对于这个Kata来说，y不被认为是元音。代码提交地址：

<https://www.codewars.com/kata/52fba66badcd10859f00097e>

提示：

- 首先使用列表解析得到一个列表，列表中所有不是元音的字母。
- 使用字符串的join方法连结列表中所有的字母，例如：

```
last_name = "lovelace"
letters = [letter for letter in last_name ]
print(letters) # ['l', 'o', 'v', 'e', 'l', 'a', 'c', 'e']
name = ''.join(letters) # name = "lovelace"
```

运行代码如下：

```
def disemvowel(string_):
    letters = [letter for letter in string_]
    new_string = []
    print(letters)
    cnt = 0
    for i in range(0,len(string_)):
        if letters[i]!='i' and letters[i]!='a' and letters[i]!='e' and
letters[i]!='o' and letters[i]!='u' and letters[i]!='I' and letters[i]!='A' and
letters[i]!='E' and letters[i]!='O' and letters[i]!='U' :
            new_string.append(letters[i])
    print(new_string)
    string_ = ''.join(new_string)
    return string_
```

流程图：

第五题流程图：

```
flowchart LR
    A[Start] --> B{i < length of string ? }
    B -->|Yes| C{ letter of i Is vowel? }
    C -->|Yes| G[ append i to letters ]
    G --> F
    C -->|NO| F[i++]
    F --> B
    B ---->|No| E[return strinf of letters]
```

实验考查

Python中的列表可以进行哪些操作？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. Python中的列表可以进行哪些操作？

对列表进行增删改查，例如：append、pop()、delete()、insert()等曹操

2. 哪两种方法可以用来对Python的列表排序？这两种方法有和区别？ sort() 与 sorted()：

sort()会改变列表的顺序，而sorted()会临时进行排序不会真正改变列表元素中的顺序

3. 如何将Python列表逆序打印？

对该列表进行reverse()操作，然后再进行打印

4. Python中的列表执行哪些操作时效率比较高？哪些操作效率比较低？是否有类似的数据结构可以用来替代列表？访问和增加删除元素的时候执行效率比较高，在列表开头和中间插入或者删除元素的时候执行效率比较低 类似的数据结构有数组、集合、链表等来替代列表

5. 阅读《Fluent Python》Chapter 2. An Array of Sequence - Tuples Are Not Just Immutable Lists小节 (p30-p35)。总结该小节的主要内容。

1. 元组是不可变的序列，类似于列表，但不能修改。
2. 元组可以包含任意类型的对象，可以是混合类型的。
3. 元组可以通过逗号分隔的方式创建，也可以使用tuple()函数将其他可迭代对象转换为元组。
4. 元组支持索引和切片操作，可以通过索引访问元素，也可以使用切片获取子元组。
5. 元组可以作为字典的键，而列表不能作为字典的键。
6. 元组可以用于函数的参数和返回值，可以将多个值打包成元组作为函数的返回值，也可以7.将元组解包为多个变量作为函数的参数。
7. 元组可以用于交换变量的值，通过将两个变量的值打包成元组，然后解包给另外两个变量，可以实现变量值的交换。
8. 元组可以用于命名元组（namedtuple），通过给元组的字段命名，可以更加清晰地访问元组的元素。

实验总结

本次实验进一步熟悉了对列表、字典、数组等数据结构的操作，以及如何简单使用它们解决实际问题，本次实验收获颇多