2021 年春季《大学计算机基础》（理科）实验指导书

实验5　自定义数据结构（栈、队列）

说明：

请编程之前认真阅读实验指导。

当在本地运行程序成功、并使用实验指导书给出的输入样例和输出样例检查无误后，再将源代码提交到在线实验平台（OJ 系统，在线评测系统）进行自动测试，以便平台自动记录你的成绩。请确保每道题评测结果为“AC”。

实验平台链接：https://www.myrct.cn/buaaoj/web/dist/index.html

1. 实验目的

(1) 了解Python 自定义数据结构的方式：如何自己定义一个类，并完成“栈”或“队列”所具有的功能。

(2) 理解栈和队列存在的意义，以及它们所代表的思维方式。掌握用列表模拟栈和队列的方法。

(3) 培养思维能力，能够解决思维性题目。

2. 实验任务

实验任务 5-1 有机物不饱和度

- **题目背景：**

不饱和度（英文名称：Degree of unsaturation），又称缺氢指数或者环加双键指数（index of hydrogen deficiency (IHD) or rings plus double bonds），是有机物分子不饱和程度的量化标志，用希腊字母Ω表示。在有机化学中用来帮助画化学结构，在推断有机化合物结构时很有用。不饱和度公式可以帮助使用者确定要画的化合物有多少个环、双键、和叁键，但不能给出环或者双键或者叁键各自的确切数目，而

是环和双键以及两倍叁键（即叁键算2个不饱和度）的数目总和，即Ω=双键数+叁键数×2+环数。最终结构需要借助于核磁共振（NMR）、质谱和红外光谱（IR）以及其他的信息来确认。

- **题目描述：**

现给出一个有机物的化学式（只包含碳、氢、氧、氮、磷、硫中的若干种元素），请你编写Python程序计算该有机物的不饱和度。请注意：

化学式为 $C_xH_yO_zN_kP_mS_n$ 的有机物的不饱和度 $\Omega=\frac{2+2x-y+k+m}{2}$ 。

- **输入格式：**

一个字符串，基本格式为：CxHyOzNkPmSn，表示化学式为$C_xH_yO_zN_kP_mS_n$ 的有机物。任何情况下，原子后面的数字不会省略；保证输入元素的顺序同上。如（乙醇/二甲醚：C2H6O1，苯酚：C6H6O1）。

- **输出格式：**

输出一个**整数**，表示该物质的不饱和度 Ω。

- **输入样例 1：**

```
C8H16O2
```

- **输出样例 1：**

```
1
```

- **输入样例 2：**

```
C7H5O6N3
```

- **输出样例 2：**

```
7
```

- 数据说明：

保证所有输入均为合法化学式。

- **实验指导：**

1. 使用 str.isalpha() 可判断字符/字符串 str 是否全是字母；

2. 使用 str.replace(before, after)，可将字符串内的所有 before 替换为 after；

3. 可使用字典存储化学式。

实验任务 5-2 机器翻译

- **题目背景：**

小安的电脑上安装了一个机器翻译软件，他经常用这个软件来翻译英语文章。

- **题目描述：**

这个翻译软件的原理很简单，它只是从头到尾，依次将每个英文单词用对应的中文含义来替换。对于每个英文单词，软件会先在内存中查找这个单词的中文含义，如果内存中有，软件就会用它进行翻译；如果内存中没有，软件就会在外存中的词典内查找，查出单词的中文含义然后翻译，并将这个单词和译义放入内存，以备后续的查找和翻译。

假设内存中有 M 个单元，每单元能存放一个单词和译义。每当软件将一个新单词存入内存前，如果当前内存中已存入的单词数不超过 M-1，软件会将新单词存入一个未使用的内存单元；若内存中已存入 M 个单词，软件会清空最早进入内存的那个单词，腾出单元来，存放新单词。

给定这篇待译文章，翻译软件需要去外存查找多少次词典？假设在翻译开始前，内存中没有任何单词。

- **输入格式：**

一共两行。第一行为一个正整数 M，代表内存容量。

第二行为待译文章，保证不含换行符。

- **输出格式：**

第一行输出一个正整数，为文章中单词的数目。

第二行输出一个正整数，为软件需要查词典的次数。

- **输入样例 1：**

```
3
I really don't know how many love stories I have left in me, but I want you
to be my last.
```

- **输出样例 1：**

```
21
21
```

- **输入样例 2：**

```
10
If I could rearrange the alphabet, I'll put 'u' and 'i' together.
```

- **输出样例 2：**

```
12
11
```

- **输入样例 3：**

```
20
Don't cry because it is over , smile because it happened.
```

- **输出样例 3：**

```
10
8
```

- **说明/提示：**

本题中的“单词”是满足以下条件的字符串：

每个单词不含空格，单词间至少含有一个空格，同一英文字母的大小写形式视作相同，每个单词以英文字母开头和结尾。

两个单词相同当且仅当其长度相同且对应位置的字符相同。

不同单词之间相互独立，即不能通过已知某些单词的译义来推断出其他单词的译义。

以下是互不相同的单词（以逗号和空格分隔）：

```
seventy, seven, don't, seventy-seven, FIGHTING, i, will, I'll, p.m
```

不能通过已知 i 和 will 推断出 I'll 的译义，也不能通过已知 seventy 和 seven 来推断出 seventy-seven 的译义， p.m 是单词而 p.m. 不是单词。

以下是相同的单词（以逗号和空格分隔）：

```
BrandNewJimZhang, brandnewjimzhang, BRANDNEwJIMzHaNG
```

- 数据范围

  文章长度大于等于 1 小于等于 30000，内存容量 1≤M≤5000

  文章的字符集保证只含英文大小写字母、空格以及 ， ！ . ？ - ’

  空格不属于任一单词，英文字母必属于某一单词，其它字符均可能属于某一单词。

- **实验指导：**

  1. 使用字符串的split()方法将文章以空格分隔开，使用strip(“.-?!’,”)方法去掉单词两边的所有标点，使用lower()方法将字母全部转化为小写形式。

  2. 可以把内存空间看做一个队列，使用 len(lis)函数判断内存是否已满，lis.pop(0) 可弹出队首元素，lis.append(element) 可在队尾添加元素。

实验任务 5-3 hwjj 回消息

- **题目背景**

hwjj 作为《大学计算机基础》的助教组长，每天在微信上向他请教问题的大一同学非常多，请你帮他记录一下他回消息的名单。

- **题目描述：**

最开始时，hwjj 的微信是没有未读消息的。之后在每一个时刻，hwjj 的手机只能处于以下三个状态的一种：接收消息，回复消息，或者什么都不做。这是由于 hwjj

的手机太卡了，若某一时刻手机需要接收消息，则他就无法执行回复消息的操作；只要该时刻可以执行回复消息的操作且当前仍有未回复的消息，hwjj总会尝试挑选最新的消息来回复，在这一时刻回复完毕并把这个聊天框删除，当然这一时刻 hwjj 只能回复一个人的消息。按顺序给出 hwjj 将要接收到的来信人以及相应时刻，请你按照 hwjj 回复他们的时间输出一份名单。

- **输入格式：**

  一共n+1行。第一行为一个正整数 n，表明会有 n 个人给 hwjj 发消息。

  接下来的 n 行，每行用空格隔开的一个整数 b 和一个字符串 s，表示在 b 时刻名字为 s 的人给 hwjj 发了消息 (b<10000)。

- **输出格式：**

  输出共 n 行，每行为用空格隔开的一个整数 c 和一个字符串 s1，表示在 c 时刻名字为 s1 的人收到了 hwjj 的回复。

- **输入样例 1：**

```
5
1 lice
2 ob
3 om
5 rown
8 ack
```

- **输出样例 1：**

```
4 om
6 rown
7 ob
9 ack
10 lice
```

- **输入样例 2：**

```
3
1 hwgg
4 hwdd
5 hwmm
```

- **输出样例 2：**

```
2 hwgg
6 hwmm
7 hwdd
```

- **说明/提示**

保证每个人名最多只会出现一次，且每个时刻最多只有一个人给 hwjj 发消息。

保证输入的名单是按照时间从小到大的顺序。且每个人的名字均只由大小写字母组成，名字长度不超过 10。

- **实验指导：**

1. 本题在处理消息时，会优先处理最新收到的消息，这实质上是一个“栈”结构，接收消息时入栈，回复消息时出栈。

2. 判断列表是否为空可以利用len函数。

实验任务 5-4 小邋遢收拾文件（选做题）

- **题目描述：**

小邋遢可不邋遢，小邋遢有一个能存放 n 个文件的文件夹和一个能存放很多文件的文件柜，从文件柜中取出文件非常麻烦，但如果这个文件在文件夹中，他便可以很方便地取出使用。小邋遢希望最大限度地利用好文件夹的容量，且能够将他近期所需要的文件尽可能地放在文件夹中，因此他采取了“最近最久未使用原则(LRU)”，即当

每次他所需要使用的文件不在文件夹中且文件夹已经放满时，选择文件夹中最久没有使用过的文件放回到文件柜，而将当前需要使用的文件放入文件夹中。最开始文件夹中没有文件，如果小邋遢有 m 个文件，从 1 编号到 m，并且已知他近期需要使用文件的编号序列，该序列长度为 p，请你输出依据LRU原则需要依次被放回到文件柜中的文件编号。

- **输入格式：**

一共两行。第一行有三个正整数，用空格隔开，分别表示文件夹的容量 n，已有文件的个数 m 和近期需要使用的文件序列的长度 p。

第二行有 p 个数字，数字范围从 1 到 m，用空格隔开，表示近期需要使用的文件的编号顺序。

- **输出格式：**

共一行，为被替换文件的编号，按被替换顺序输出。

- **输入样例 1：**

```
3 6 10
4 3 2 4 3 1 1 3 5 6
```

- **输出样例 1：**

```
2 4 1
```

- **样例解释：**

文件夹的容量为 3，小邋遢一共有 6 个文件，近期需要使用的文件的顺序为 4 3 2 4 3 1 1 3 5 6。使用的前三个文件为 4 3 2，分别放入文件夹中；第四个文件为 4，由于已经放在文件夹中可以直接使用；第五个文件为 3，也存在文件夹中。第六个文件为 1，文件夹中没有且已经放满，则需要选出一个文件进行替换，由于刚使用过 3 和 4，最久没有使用过的文件为 2，因此选择将 2 替换掉，此时文件中的文件为 4 3 1。接下来需要使用 1 和 3，它们已经存在文件夹中；使用文件 5 时，最久没有使用过的文件 4，因此将文件 4 替换为 5，文件夹中文件为 1 3 5。使用文件6时，最久没有使用过的文件为 1，因此将 1 替换为 6。则被替换的文件顺序为 2 4 1。

- **数据范围：**

n≤500，m≤550，p≤100000，保证文件的数量大于文件夹的容量，即m>n。

- **实验指导：**

本题需要使用列表的 pop() 和 remove() 方法。

1.pop()函数用于移除列表中的一个元素（默认最后一个元素），并且返回该元素的值。

2.remove()函数用于移除列表中某个值的第一个匹配项。

实验任务 5-5 合成大西瓜（选做题）

- **题目背景：**

“合成大西瓜”是 2021 年年初走红朋友圈的一款小游戏。由于其玩法简单，随机性强，在不同年龄段都找到了受众。Jerry 同学寒假期间被这款游戏所吸引，但他的技术太差，运气也不好，从来没有合成出过“大西瓜”。现在请聪明的你帮他判断一下，对于某个给定的水果顺序，他是否能合成出大西瓜呢？

- **题目描述：**

现在将合成大西瓜游戏简化至一维情况，在此介绍游戏规则：

1. 游戏中共有 11 种水果，从小到大依次为：葡萄、樱桃、橘子、柠檬、猕猴桃、西红柿、桃子、菠萝、椰子、西瓜、大西瓜，用字符串表示如下：

```
"grape", "cherry", "orange", "lemon", "kiwifruit", "tomato", "peach", "pineapple", "coconut", "watermelon", "WATERMELON"
```

2. 两个相同且相邻的水果自动合成大一级的水果，如两个相邻的葡萄合成一个樱桃，两个相邻的西瓜合成一个大西瓜。

3. 合成大西瓜的容器为一维容器，各种水果无论大小均一字排开；水果从一端随机地按某一顺序逐个地进入容器，相邻的水果合成后位置不变。

4. 进入容器的水果只可能为前 10 种，其出现情况随机，概率分布未知。

5. 当合成出大西瓜时，游戏胜利且立刻结束；若容器已溢出或水果使用完毕但仍未合成出大西瓜，游戏失败；若容器已满，但待放入的水果恰好和容器顶的水果合并，则游戏按规则继续（即瞬间的溢出忽略不计）。参见样例3。

- **输入格式：**

共 M+1 行，第一行为容器的大小 N 和依次进入容器的水果个数 M，二者以空格分开。之后 M 行为依次进入容器的水果名称字符串。

- **输出格式：**

共两行，第一行表示游戏是否胜利，胜利则输出You win!，失败则输出You lose!。第二行为确定胜利或失败时容器中剩余的水果，以空格分隔的字符串形式输出。水果按容器从深至浅的顺序在控制台从左至右输出。

- **输入样例 1：**

```
5 4
coconut
coconut
watermelon
grape
```

- **输出样例 1：**

```
You win!
WATERMELON
```

- **输入样例 2：**

```
5 5
grape
grape
orange
kiwifruit
peach
```

- **输出样例 2：**

```
You lose!
```

```
cherry orange kiwifruit peach
```

- **输入样例3：**

```
2 3
watermelon
coconut
coconut
```

- **输出样例3：**

```
You win!
WATERMELON
```

- **数据范围：**

  对于 70% 的数据，0≤M,N≤120；对于 100% 的数据，0≤M,N≤1500。

- **实验指导：**

  1. 对各类水果进行编码，可以简化水果合成的操作逻辑。

  2. 可使用字典建立水果和编码之间的一一对应关系。

  3. 使用栈表示容器，栈的实现如下：

```
class Stack:
    def __init__(self):
        self.__elems = []

    def isEmpty(self):
        return  self.__elems == []

    def top(self):
        if self.__elems == []:
```

```
            raise Exception('Stack is empty when using top()!')
        else:
            return self.__elems[-1]

    def push(self, elem):
        self.__elems.append(elem)

    def pop(self):
        if self.__elems == []:
            raise Exception('Stack is empty when doing pop()!')
        else:
            return self.__elems.pop()

    def size(self):
        return len(self.__elems)
```

使用方式举例如下：

```
fruit_box = Stack()     # 实例化，建立一个空栈
fruit_box.push(a)       # 压栈（放进一个水果a）
b = fruit.top()         # 查看/得到栈顶元素
c = fruit.pop()         # 弹栈（取出一个水果c）
if fruit.isEmpty():     # 查看栈是否为空
size = fruit.size()     # 查看栈的大小
```