“力扣”刷题训练1

栈

栈的特点：后进先出（last in first out)

根据此特点，可以总结出一些模板，有助于培养一块块的一环套一环的思路，有助于刷题。

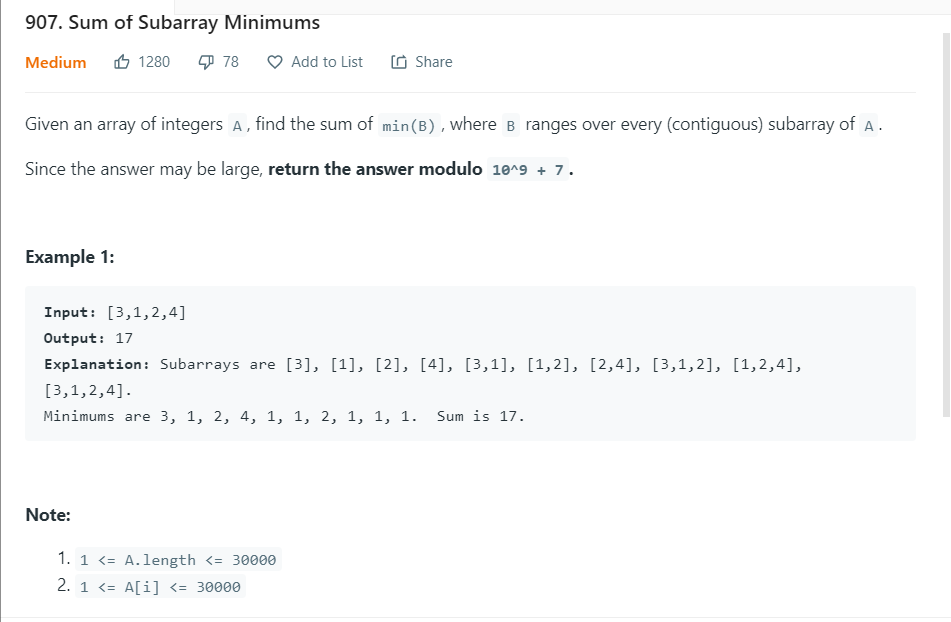
本份讲义主要内容：

1. 编程思路
2. 数据结构栈（stack）的特性解析
3. 小技巧，巧妙代码形式归纳
4. 额外补充练习，代码变式

**“跨度”（spans）题型**

例题1

1. 子数列最小值加和问题



思路：首先，这题是栈的问题，应该去思考栈的特性，利用栈的特性进行解题。

通过率30%多，证明这题不是特别简单。

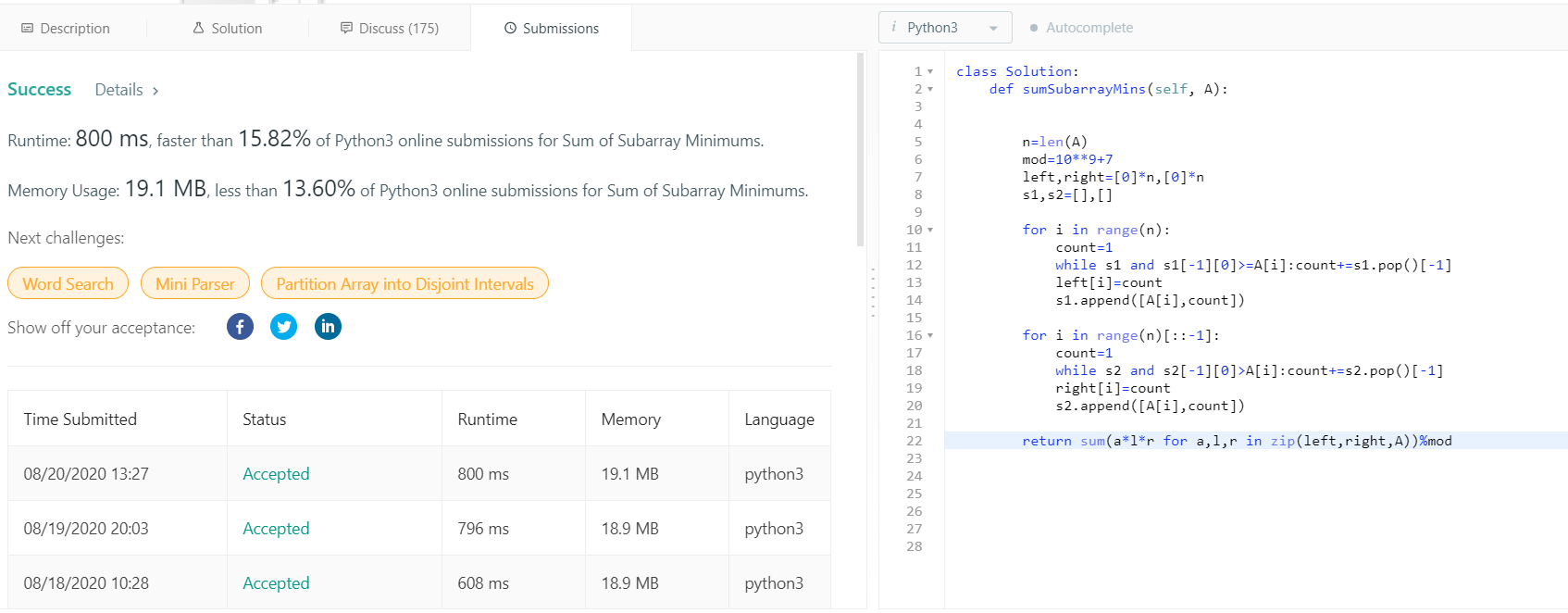
首先个人第一遍做的时候，没有考虑到contiguous,从而导致用了“全组合”代码而做错。

经过交流，得知，这题的思路在于，对每个元素遍历，找到每个元素左边比它大的元素的“跨度”，右边同样，假设左右分别跨度为m,n,那么由基本的数学知识，可以知道，以这个元素为minimum的contiguous subarray一共有m\*n个，那么这个元素在最后的加和过程的贡献是m\*n次。

因此，我们不需要用代码去列出所有的contiguous subarray再去寻找minimum。而要转换思维，利用上述小技巧，找出跨度去相乘，然后直接计算即可。

分享本人难点：无法每次抓住栈的特性写代码。

先进行代码分享，再进行解说：



刚刚自己练了手，然后进行了代码的试打，发现自己的思维ok。

解说： 一般用栈，我们是作为一个“比较大小”功能的工具，result\_list往往需要另行设置，然后用于搞出结果，leetcode是习惯于“元组式”的做题方式，在这题中，我们在stack中存放的是配套的（A[i], count），count用于出结果，和叠加，栈的做题技巧往往包含了for,while这样的两层循环，这样就充分利用了栈的特性，对于每个元素，都会和前面的元素或者后面的元素进行比较，while充分凸显栈的好处，

这题的容易出错的小点（自称“小错点”）：mod就是为了防止结果过大，按照格式打就好。

两个for里面必须有一且只有一个是大于等于，因为两边不可计算重复，什么意思呢，拿一个例子思考，比如如果input是[2,0,2],那么代码运行时，如果两个for都打上等于号，以2为例，在左右进行跨度计算的话，左右两个2都互相找到时，其实一次就够，但是两个等号则会重复。

代码小知识点：

For i in range(n)[::-1]: 以这种方式就可以进行简单的反向遍历

Return sum(a\*l\*r for a,l,r in zip(left,right,A))%mod 这个代码很简便，而且三列表相乘加和，用zip这种打包的方式特别方便，可以学习一下流畅的python，多学习一些代码技巧和简便写法。

栈的意义：为什么要pop掉呢？

经过分析，我惊呆了，leetcode使用栈的形式其实破坏了栈原本的意义，虽然用元素的方式挺巧妙的，但是其实，我发现这题的热门代码完全没有用出栈的精髓。

因此，布置

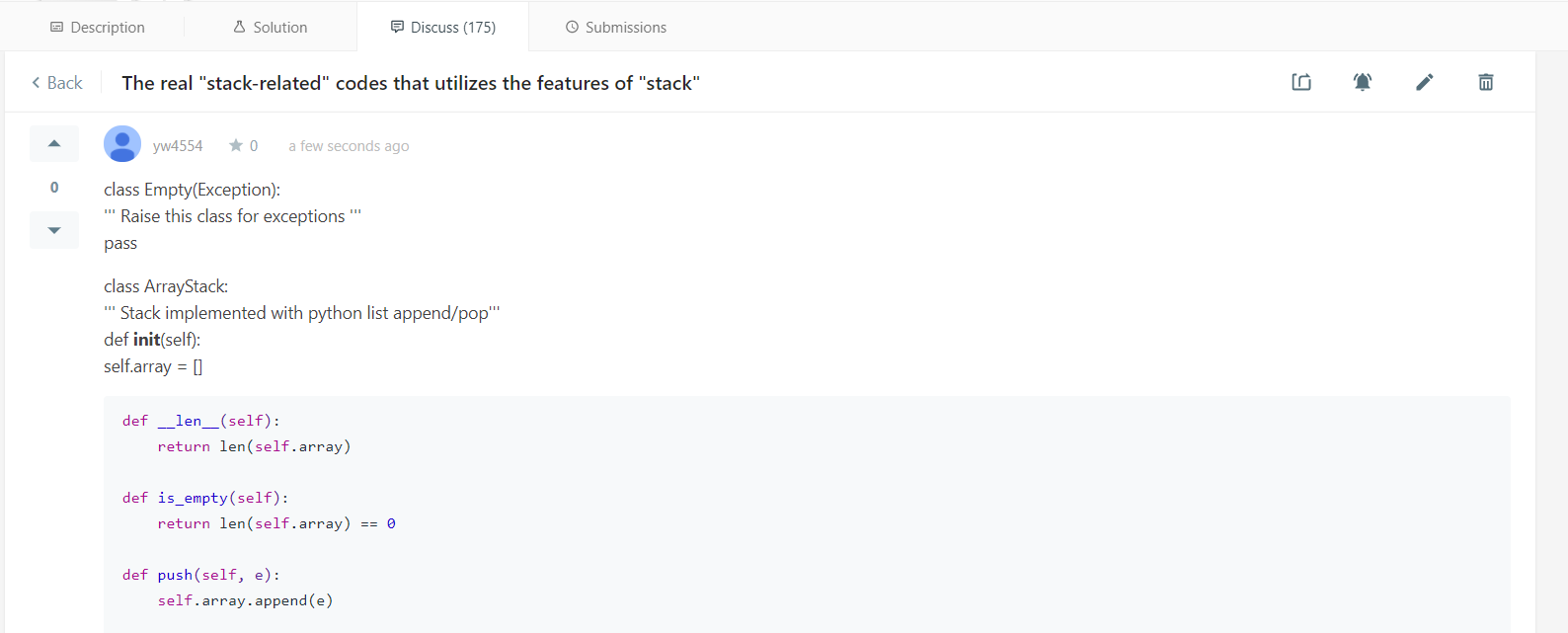
**练习1：**

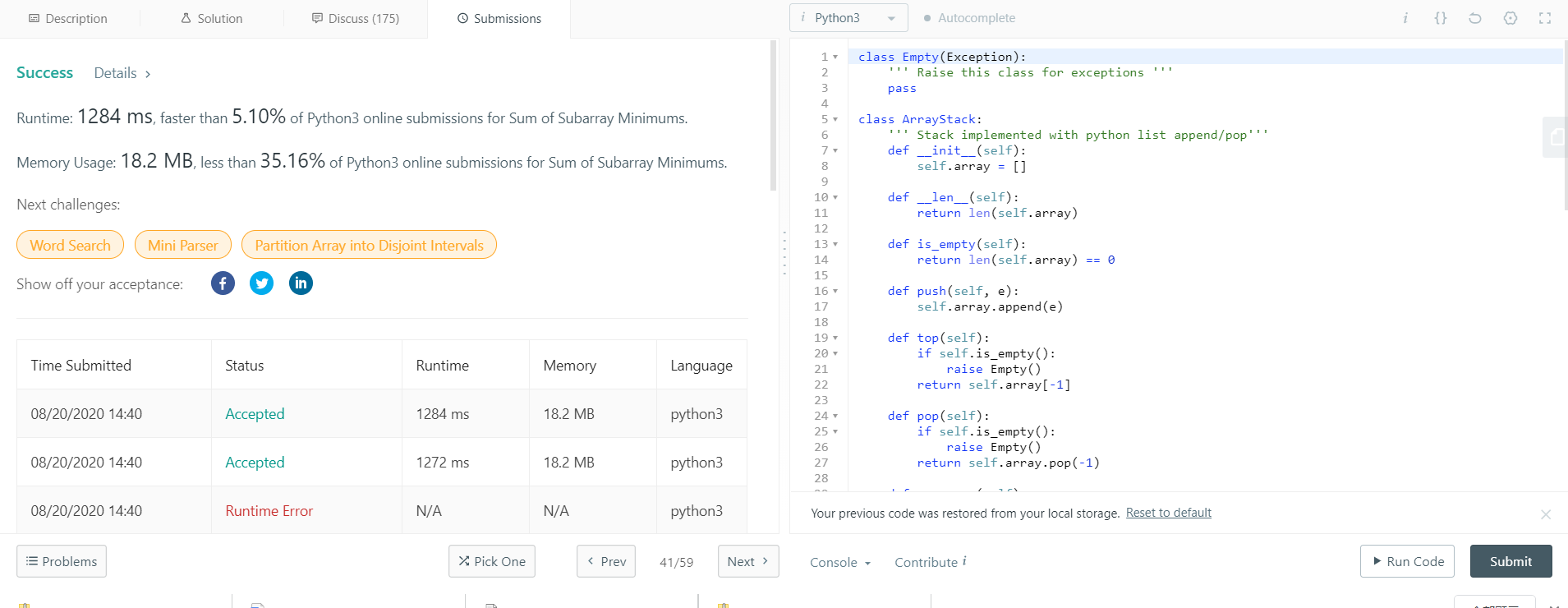
**作业挑战：请使用stack的代码框架改写代码。（文件下载见github）**

**提示：可以先学习例题2，模仿例题2代码进行思考，就感悟出为什么例题1的leetcode热门参考答案并没有利用栈的特性，以及栈的特色代码是怎么样的。**

解析：用stack代码框架写的答案里，stack往往记录的是index索引值，索引值一个个pop掉，然后用正在进行遍历的i去减掉栈顶pop出来的索引，得出我们想要求的跨度，这才是栈的特性的巧妙。而原leetcode热门答案实际上是用了元组，虽然也很高级简便，但是没有体现出栈的原汁原味和特点。

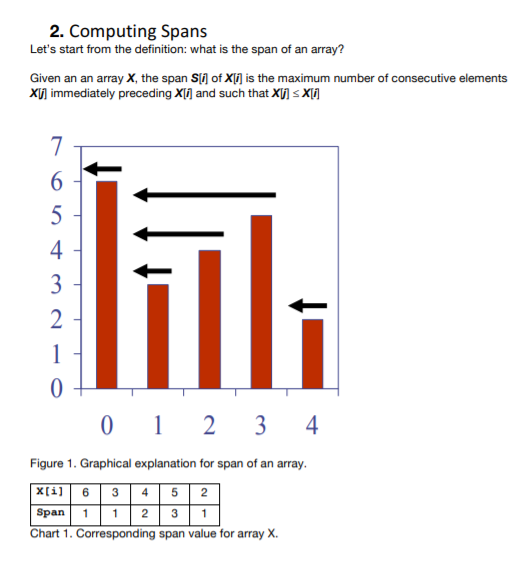
重新回到刚刚未解答的问题：为什么pop掉？因为这样才是利用栈的特性，正确的栈的打开方式，通过栈的后进先出，进行索引值的加减，得到我们想要的“跨度”数值。

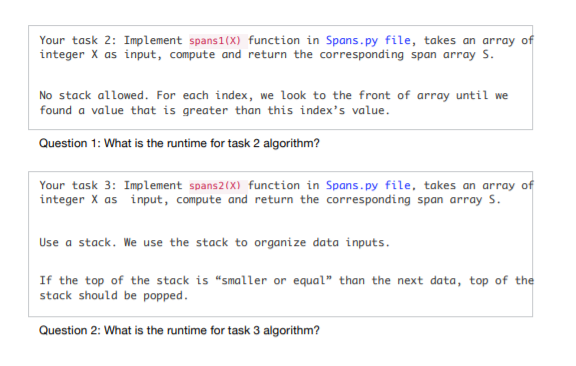


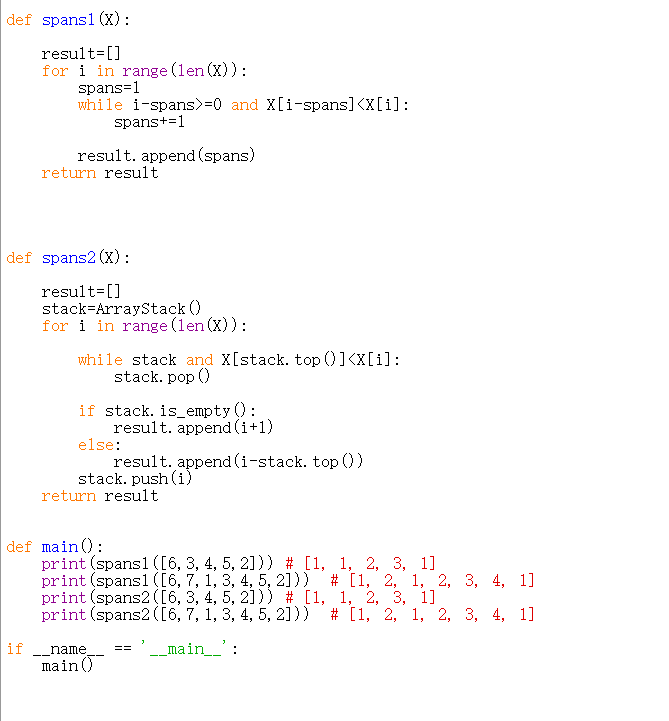


真正用自己的代码，体现栈的特性，提交成功并且在论坛里做分享，是很有成就感的。

例题2

.

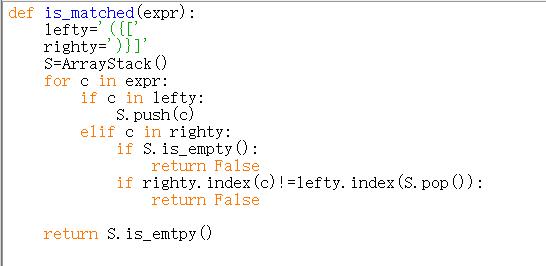




解析：那么这题也是利用栈的特性，正确地使用了符合栈的特性的pop，符合栈后进先出的特点。栈存储的是索引值，而result\_list是另外分开的，用来存我们想要的“跨度”。

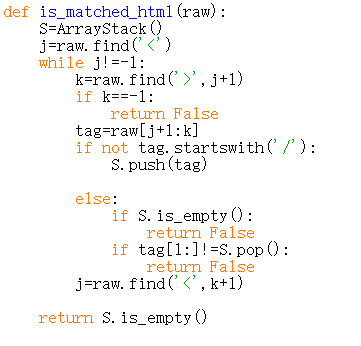
**Parser题型**

基础例子1：分隔符匹配



**解析：利用栈的后进先出特性，这与题型一有所不同，通过将“左边的”符号压进栈，“右边的”去匹配（匹配的同时pop）来操作，最后由是否S.is\_empty()来得出答案（empty时正好全部匹配）**

基础例子2：html匹配



一样的用了栈的特性。

看过这两个基础的匹配代码，可以尝试leetcode的valid parentheses(较为简单，不作为题目放在讲义中）