

assignment3

Assignment #3: Stack, DP & Backtracking

2025 fall, Complied by 杨知进 物理学院

说明：

1. 解题与记录：

对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge, Codeforces, LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora <https://typoraio.cn> 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。

2. 提交安排：提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的本人头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。
3. 延迟提交：如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。

请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

1078: Bigram分词

<https://leetcode.cn/problems/occurrences-after-bigram/>

思路：

代码：

```
class Solution:

    def findOcurrences(self, text: str, first: str, second: str) -> List[str]:
        text = text.split()
        outcome = []
        for i in range(len(text)):
```

```

if text[i] == first:

    if i+2 < len(text):

        if text[i+1] == second:

            outcome.append(text[i+2])

return outcome

```

代码运行截图 (至少包含有"Accepted")

The screenshot shows a LeetCode submission page for a Python solution. The code is pasted into the editor, and the output shows "已存储" (Saved). Below the editor, the submission status is "通过" (Passed) with 30/30 test cases passed. There are buttons for "官方题解" (Official Solution) and "写题解" (Write Solution).

```

1 class Solution:
2     def findOccurrences(self, text: str, first: str, second: str) -> List[str]:
3         text = text.split()
4         outcome=[]
5         for i in range(len(text)):
6             if text[i] == first:
7                 if i+2 < len(text):
8                     if text[i+1] == second:
9                         outcome.append(text[i+2])
10        return outcome
11

```

用时：4min左右

283. 移动零

stack, two pointers, <https://leetcode.cn/problems/move-zeroes/>

思路：

代码：

```

class Solution:

def moveZeroes(self, nums: List[int]) -> None:

```

```
count = nums.count(0)

while 0 in nums:

    nums.remove(0)

for i in range(count):

    nums.append(0)
```

代码运行截图 (至少包含有"Accepted")

The screenshot shows the LeetCode Python3 editor interface. The code area contains the provided solution for moving zeros. Below the code, a message says "已存储" (Stored). The submission status at the bottom indicates "通过" (Passed) with "75 / 75 个通过的测试用例" (75 / 75 test cases passed). There are two buttons at the bottom right: "官方题解" (Official Solution) and "写题解" (Write Solution).

```
1 class Solution:
2     def moveZeroes(self, nums: List[int]) -> None:
3         count = nums.count(0)
4         while 0 in nums:
5             nums.remove(0)
6         for i in range(count):
7             nums.append(0)
8
```

用时：4min左右

20.有效的括号

stack, <https://leetcode.cn/problems/valid-parentheses/>

思路：

代码：

```
class Solution:

    def isValid(self, s: str) -> bool:
```

```
stack=[]

left=['(', '{', '[']

right=[')', '}', ']']

for x in list(s):

    if x in left:

        stack.append(x)

    if x in right:

        if (not stack) or stack[-1] != left[right.index(x)]:

            return False

    else:

        stack.pop()

if stack:

    return False

else:

    return True
```

代码运行截图 (至少包含有"Accepted")

Python3 智能模式

```
1 class Solution:
2     def isValid(self, s: str) -> bool:
3         stack=[]
4         left=['(',')','[',']']
5         right=[')','}',']']
6         for x in list(s):
7             if x in left:
8                 stack.append(x)
9             if x in right:
10                if (not stack) or stack[-1] != left[right.index(x)]:
11                    return False
12                else:
13                    stack.pop()
14            if stack:
15                return False
16            else:
17                return True
```

存储中...

题目描述 | 通过 × | 题解 | 提交记录

← 全部提交记录

通过 102 / 102 个通过的测试用例

官方题解

更多题解

10min左右

118.杨辉三角

dp, <https://leetcode.cn/problems/pascals-triangle/>

思路：

代码：

```
class Solution:

    def generate(self, numRows: int) -> List[List[int]]:
        dp = []
        for row in range(numRows):
            s = [1]
```

```

if row > 0:

    for j in range(1, row):

        s.append(dp[row-1][j-1] + dp[row-1][j])

    s.append(1)

dp.append(s)

return dp

```

代码运行截图 (至少包含有"Accepted")

The screenshot shows a Python code editor interface for LeetCode. The code is a solution to generate Pascal's triangle. It uses a dynamic programming approach where each row is built by adding the previous row's elements. The code is as follows:

```

1 class Solution:
2     def generate(self, numRows: int) -> List[List[int]]:
3         dp = []
4         for row in range(numRows):
5             s = [1]
6             if row > 0:
7                 for j in range(1, row):
8                     s.append(dp[row-1][j-1] + dp[row-1][j])
9                 s.append(1)
10            dp.append(s)
11        return dp

```

Below the code, there is a message "已存储" (Saved). At the bottom, the status bar shows "通过 30 / 30 个通过的测试用例" (Passed 30 / 30 test cases), indicating the solution is accepted.

用时：6min左右

46.全排列

backtracking, <https://leetcode.cn/problems/permutations/>

思路：

代码

(至少包含有"Accepted")

78.子集

backtracking, <https://leetcode.cn/problems/subsets/>

思路：

代码

```
class Solution:

    def subsets(self, nums: List[int]) -> List[List[int]]:

        outcome=[[[]]

        for i in nums:

            outcome+=[j + [i] for j in outcome]

        return outcome
```

(至少包含有"Accepted")

Python3 智能模式

```
1 class Solution:
2     def subsets(self, nums: List[int]) -> List[List[int]]:
3         outcome = [[]]
4         for i in nums:
5             outcome += [j + [i] for j in outcome]
6         return outcome
```

已存储

题目描述 | 通过 × | 题解 | 提交记录

← 全部提交记录

通过 10 / 10 个通过的测试用例

官方题解

写题

用时：10min左右

2. 学习总结和个人收获

栈的操作很有意思，跟同时在上的TCS（理论计算机科学基础）联系起来。大一学计概的时候，括号匹配是用的非常暴力的检查办法，而现在看到它首先想到的是栈和上下文无关文法hh。把CFG改成PDA就变成了要写的程序的模样。不过括号匹配比较简单，又找了一道中等的：

856. 括号的分数

中等

相关标签

 相关企业

A文

给定一个平衡括号字符串 s ，按下述规则计算该字符串的分数：

- $()$ 得 1 分。
 - AB 得 $A + B$ 分，其中 A 和 B 是平衡括号字符串。
 - (A) 得 $2 * A$ 分，其中 A 是平衡括号字符串。

Python3 ▾ 智能模式

```
1 class Solution:
2     def scoreOfParentheses(self, s: str) -> int:
3         s=list(s)
4         stack = []
5         for i in range(len(s)):
6             if s[i] == '(':
7                 stack.append('(')
8             else:
9                 for j in range(0,len(stack)):
10                     if stack[len(stack)-1-j]==')':
11                         if j == 0:
12                             stack[-1]=1
13                             break
14                         else:
15                             stack = stack[:len(stack)-1-j] + 2 * [sum(stack[len(stack)-1-j+1:])]
16                             break
17
18         return sum(stack)
```

已存储

题目描述 | 通过 × | 题解 | 提交记录

← 全部提交记录

通过 86 / 86 个通过的测试用例

StrangeI2itchiePAA 提交于 2025.10.08 15:14

官方题解

写题解



字节跳动心动计划

面向短期冲刺字节跳动选手，求职弯道顺利超车



① 执行用时分布

ⓘ

0 ms | 击败 100.00% 🎉

消耗内存分布

17.29 MB | 击败 97.77% 🎉