

Assignment #5: 链表、栈、队列和归并排序

Updated 1348 GMT+8 Mar 17, 2025

2025 spring, Compiled by 同学的姓名、院系

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

LC21.合并两个有序链表

linked list, <https://leetcode.cn/problems/merge-two-sorted-lists/>

思路:

代码:

```
class Solution: def mergeTwoLists(self, list1: Optional[ListNode], list2: Optional[ListNode]) → Optional[ListNode]: dummy = ListNode(-1) current = dummy while list1 and list2: if list1.val < list2.val: current.next = list1 list1 = list1.next else: current.next = list2 list2 = list2.next current = current.next
```

```
if list1:
    current.next = list1
if list2:
    current.next = list2
return dummy.next
```

代码运行截图（至少包含有"Accepted"）



LC234.回文链表

linked list, <https://leetcode.cn/problems/palindrome-linked-list/>

请用快慢指针实现。

代码：

```
class Solution: def isPalindrome(self, head: Optional[ListNode]) → bool: values = [] current = head
```

```
    while current:
        values.append(current.val)
        current = current.next
    return values == values[::-1]
```

代码运行截图（至少包含有"Accepted"）



LC1472.设计浏览器历史记录

doubly-lined list, <https://leetcode.cn/problems/design-browser-history/>

请用双链表实现。

代码：

```
class BrowserHistory:
```

```
    def __init__(self, homepage: str):
        self.history = [homepage]
        self.current = 0

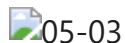
    def visit(self, url: str) -> None:
        self.history = self.history[:self.current + 1]
        self.history.append(url)
        self.current += 1

    def back(self, steps: int) -> str:
        self.current = max(0, self.current - steps)
        return self.history[self.current]

    def forward(self, steps: int) -> str:
```

```
self.current = min(len(self.history) - 1, self.current + steps)
return self.history[self.current]
```

代码运行截图（至少包含有"Accepted"）



24591: 中序表达式转后序表达式

stack, <http://cs101.openjudge.cn/practice/24591/>

思路:

代码:

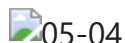
```
def precedence(op): return {'+': 1, '-': 1, '*': 2, '/': 2}.get(op, 0)

def infix(row): numbers = [] output = [] i, n = 0, len(row) while i < n: ch = row[i]

    if ch.isdigit() or ch == '.':
        start = i
        has_dot = ch == '.'
        while i + 1 < n and (row[i + 1].isdigit() or row[i + 1] == '.' and not has_dot):
            if row[i + 1] == '.':
                has_dot = True
            i += 1
        output.append(row[start: i + 1])
    elif ch in '+-*/':
        while numbers and precedence(numbers[-1]) >= precedence(ch):
            output.append(numbers.pop())
        numbers.append(ch)
    elif ch == '(':
        numbers.append(ch)
    elif ch == ')':
        while numbers and numbers[-1] != '(':
            output.append(numbers.pop())
        numbers.pop()
    i += 1
while numbers:
    output.append(numbers.pop())
return output
```

```
lines = int(input()) for _ in range(lines): row = input() print(' '.join(infix(row)))
```

代码运行截图（至少包含有"Accepted"）



03253: 约瑟夫问题No.2

queue, <http://cs101.openjudge.cn/practice/03253/>

请用队列实现。

代码：

```
def sequence(n, p, m):
    elements = list(range(1, n + 1))
    output = []
    index = p - 1

    while elements:
        index = (index + m - 1) % len(elements)
        output.append(elements.pop(index))

    return output

while True:
    try:
        n, p, m = map(int, input().split())
        if n == p == m == 0:
            break
        ans = sequence(n, p, m)
        print(' '.join(map(str, ans)))
    except EOFError:
        break
```

代码运行截图（至少包含有"Accepted"）



20018: 蚂蚁王国的越野跑

merge sort, <http://cs101.openjudge.cn/practice/20018/>

思路：

代码：

```
import bisect
N = int(input())
speeds = []
counts = 0

for _ in range(N):
    ant_speed = int(input())
    # 找出 ant_speed 在 speeds 中的插入位置，直接计算比它小的数量
    index = bisect.bisect_left(speeds, ant_speed)
    counts += index
    bisect.insort_left(speeds, ant_speed)

print(counts)
```

代码运行截图（至少包含有"Accepted"）



2. 学习总结和收获

如果发现作业题目相对简单，有否寻找额外的练习题目，如“数算2025spring每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。

本次作业题目大多是递归、指针类的题目，还看到了计概典型的排队题目。这次作业里面有个特别的题目，网页那题是个很有趣的题目，虽然有点简易版，但让我眼前一亮。