

# Assignment #5: 链表、栈、队列和归并排序

Updated 1348 GMT+8 Mar 17, 2025

2025 spring, Compiled by 王梓航、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]:
        list3 = ListNode()
        head = list3
        while list1 and list2:
            if list1.val <= list2.val:
                head.next = list1
                list1 = list1.next
            else:
                head.next = list2
                list2 = list2.next
            head = head.next
        if list1:
            head.next = list1
```

```
if list2:
    head.next=list2
return list3.next
```

代码运行截图 (至少包含有"Accepted")

通过 208 / 208 个通过的测试用例

 Hungry Northc... 提交于 2025.03.20 15:06



 写题解

## 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:
        if not head or not head.next:
            return True
        s,f = head,head
        while f and f.next:
            s = s.next
            f = f.next.next
        pre = None
        while s:
            cur = s.next
            s.next=pre
            pre = s
            s = cur
        while pre and head:
            if pre.val != head.val:
                return False
            pre,head = pre.next,head.next
        return True
```

代码运行截图 (至少包含有"Accepted")

通过 93 / 93 个通过的测试用例

 Hungry NorthcuttqqE 提交于 2025.03.20 18:42

 官方题解

 写题解

## 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 Node:
    def __init__(self, data):
        self.data = data
        self.next = None
        self.prev = None

class BrowserHistory:

    def __init__(self, homepage: str):
        new_node = Node(homepage)
        self.head = new_node
        self.now = new_node

    def visit(self, url: str) -> None:
        new_node = Node(url)
        self.now.next = new_node
        new_node.prev = self.now
        self.now = new_node

    def back(self, steps: int) -> str:
        cur = self.now
        for i in range(steps):
            if cur.prev:
                cur = cur.prev
            else:
                break
        self.now = cur
        return cur.data

    def forward(self, steps: int) -> str:
        cur = self.now
        for i in range(steps):
            if cur.next:
                cur = cur.next
        self.now = cur
        return cur.data
```

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 infix_to_postfix(expression):
    precedence = {'+': 1, '-': 1, '*': 2, '/': 2}
    stack = []
    output = []

    i = 0
    while i < len(expression):
        char = expression[i]
        if char.isdigit() or char == '.':
            num = ''
            while i < len(expression) and (expression[i].isdigit() or expression[i] == '.'):
                num += expression[i]
                i += 1
            output.append(num)
            continue
        elif char == '(':
            stack.append(char)
        elif char == ')':
            while stack and stack[-1] != '(':
                output.append(stack.pop())
            stack.pop()
        elif char in precedence:
            while stack and stack[-1] in precedence and precedence[stack[-1]] >= precedence[char]:
                output.append(stack.pop())
            stack.append(char)

        i += 1
    while stack:
        output.append(stack.pop())
    return ' '.join(output)

n = int(input())
for _ in range(n):
    print(infix_to_postfix(input()))
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
def infix_to_postfix(expression):
    precedence = {'+': 1, '-': 1, '*': 2, '/': 2}
    stack = []
    output = []

    i = 0
```

基本信息

#: 48646397  
题目: 24591  
提交人: 24n2400011481  
内存: 3712kB  
时间: 35ms  
语言: Python3  
提交时间: 2025-03-20 19:29:43

## 03253: 约瑟夫问题No.2

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

请用队列实现。

代码:

```
from collections import deque
while True:

    n,p,m=map(int,input().split())
    if n==0:
        break
    a = deque(range(1,n+1))
    a.rotate(-p+1)
    b = []
    for _ in range(n):
        a.rotate(-m+1)
        b.append(str(a.popleft()))
    print(','.join(b))
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
from collections import deque
while True:

    n,p,m=map(int,input().split())
    if n==0:
        break
    a = deque(range(1,n+1))
```

基本信息

#: 48646567  
题目: 03253  
提交人: 24n2400011481  
内存: 3632kB  
时间: 32ms  
语言: Python3  
提交时间: 2025-03-20 19:43:54

## 20018: 蚂蚁王国的越野跑

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

思路：利用归并排序处理

代码：

```
def f(a):
    n = len(a)
    if n==1:
        return a,0
    b,count1=f(a[:n//2])
    c,count2=f(a[n//2:])
    i,j=0,0
    d = []
    count = count1+count2
    while i<n//2 and j<n-n//2:
        if b[i]<=c[j]:
            d.append(b[i])
            i+=1
        else:
            count+=n//2-i
            d.append(c[j])
            j+=1
    d+=b[i:]
    d+=c[j:]
    return d,count
n = int(input())
a = [int(input()) for _ in range(n)]
_,m = f(a[::-1])
print(m)
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
def f(a):
    n = len(a)
    if n==1:
        return a,0
    b,count1=f(a[:n//2])
    c,count2=f(a[n//2:])
    i,j=0,0
```

基本信息

#: 48646860  
题目: 20018  
提交人: 24n2400011481  
内存: 11596kB  
时间: 631ms  
语言: Python3  
提交时间: 2025-03-20 20:03:45

## 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。每日选做还在跟进，计划最近额外练习一下leetcode每日一题。