Assignment #5: 链表、栈、队列和归并排序

Updated 1348 GMT+8 Mar 17, 2025

2025 spring, Complied by 任宇桐 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/

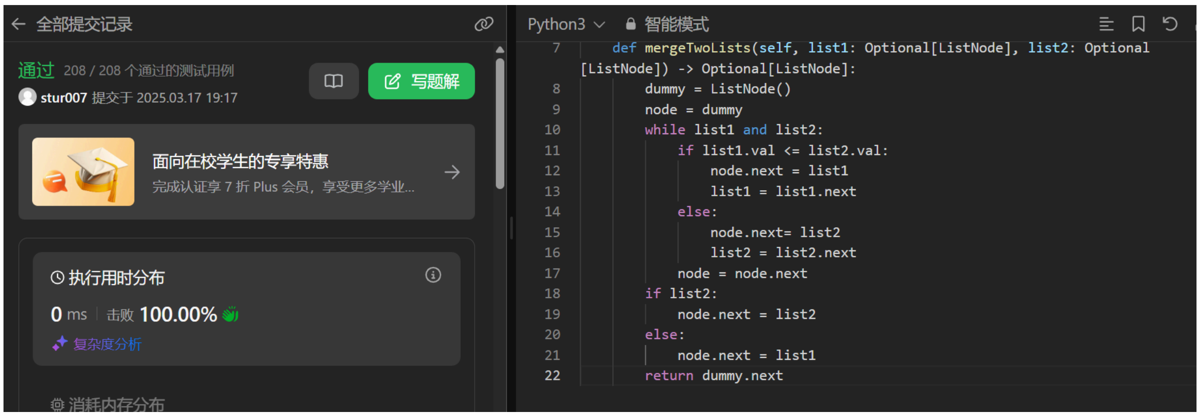
思路：

类似一次归并排序，注意到list1 与list2都有直接为空的情形。

代码：

```python
class Solution:
    def mergeTwoLists(self, list1: Optional[ListNode], list2: Optional[ListNode]) -> Optional[ListNode]:
        dummy = ListNode()
        node = dummy
        while list1 and list2:
            if list1.val <= list2.val:
                node.next = list1
                list1 = list1.next
            else:
                node.next= list2
                list2 = list2.next
            node = node.next
        if list2:
            node.next = list2
        else:
            node.next = list1
        return dummy.next
```

代码运行截图 （至少包含有"Accepted"）



全部提交记录

通过 208 / 208 个通过的测试用例

stur007 提交于 2025.03.17 19:17

写题解

面向在校学生的专享特惠

完成认证享 7 折 Plus 会员，享受更多学业...

执行用时分布

0 ms | 击败 100.00%

复杂度分析

消耗内存分布

Python3 智能模式

```python
    def mergeTwoLists(self, list1: Optional[ListNode], list2: Optional
[ListNode]) -> Optional[ListNode]:
        dummy = ListNode()
        node = dummy
        while list1 and list2:
            if list1.val <= list2.val:
                node.next = list1
                list1 = list1.next
            else:
                node.next= list2
                list2 = list2.next
            node = node.next
        if list2:
            node.next = list2
        else:
            node.next = list1
        return dummy.next
```

LC234.回文链表

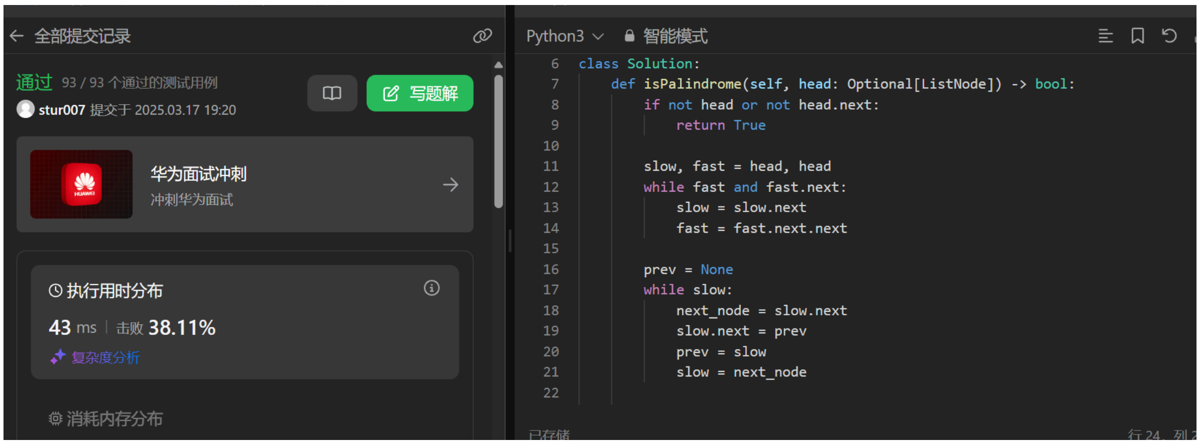
linked list, https://leetcode.cn/problems/palindrome-linked-list/

请用快慢指针实现。

代码：

```python
class Solution:
    def isPalindrome(self, head: Optional[ListNode]) -> bool:
        if not head or not head.next:
            return True

        slow, fast = head, head
        while fast and fast.next:
            slow = slow.next
            fast = fast.next.next

        prev = None
        while slow:
            next_node = slow.next
            slow.next = prev
            prev = slow
            slow = next_node

        left, right = head, prev
        while right:
            if left.val != right.val:
                return False
            left = left.next
            right = right.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
stur007 提交于 2025.03.17 19:20

43 ms | 击败 38.11%

```python
class Solution:
    def isPalindrome(self, head: Optional[ListNode]) -> bool:
        if not head or not head.next:
            return True

        slow, fast = head, head
        while fast and fast.next:
            slow = slow.next
            fast = fast.next.next

        prev = None
        while slow:
            next_node = slow.next
            slow.next = prev
            prev = slow
            slow = next_node

```

LC1472.设计浏览器历史记录

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

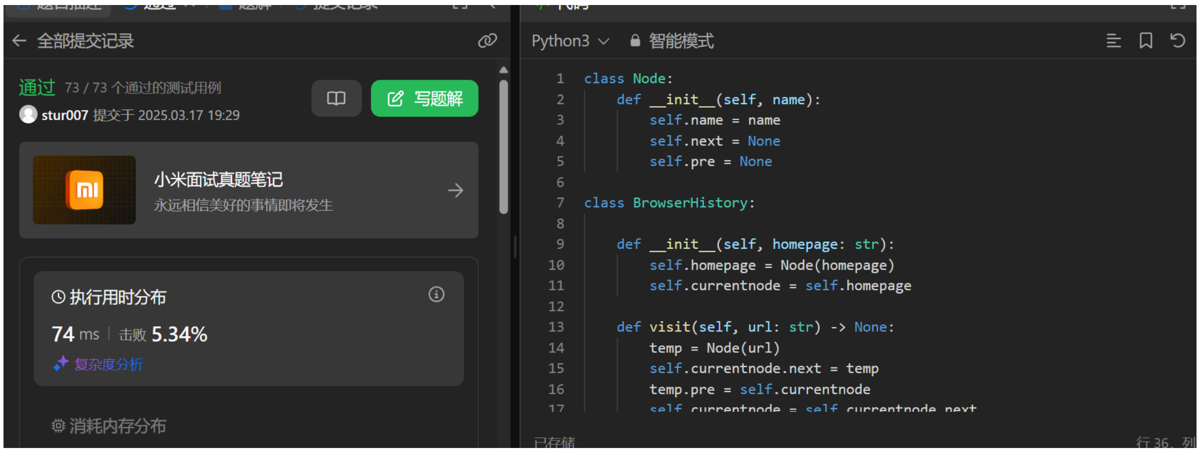
请用双链表实现。

双链表应该比直接使用列表访问索引要慢？

代码：

```python
class Node:
    def __init__(self, name):
        self.name = name
        self.next = None
        self.pre = None

class BrowserHistory:

    def __init__(self, homepage: str):
        self.homepage = Node(homepage)
        self.currentnode = self.homepage

    def visit(self, url: str) -> None:
        temp = Node(url)
        self.currentnode.next = temp
        temp.pre = self.currentnode
        self.currentnode = self.currentnode.next

    def back(self, steps: int) -> str:
        for i in range(steps):
            if self.currentnode.pre:
                self.currentnode = self.currentnode.pre
        return self.currentnode.name

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

代码运行截图 （至少包含有"Accepted"）



全部提交记录

通过 73 / 73 个通过的测试用例

stur007 提交于 2025.03.17 19:29

写题解

小米面试真题笔记

永远相信美好的事情即将发生

执行用时分布

74 ms | 击败 5.34%

复杂度分析

消耗内存分布

Python3 智能模式

```
class Node:
    def __init__(self, name):
        self.name = name
        self.next = None
        self.pre = None

class BrowserHistory:

    def __init__(self, homepage: str):
        self.homepage = Node(homepage)
        self.currentnode = self.homepage

    def visit(self, url: str) -> None:
        temp = Node(url)
        self.currentnode.next = temp
        temp.pre = self.currentnode
        self.currentnode = self.currentnode.next
```

已存储

24591: 中序表达式转后序表达式

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

思路：

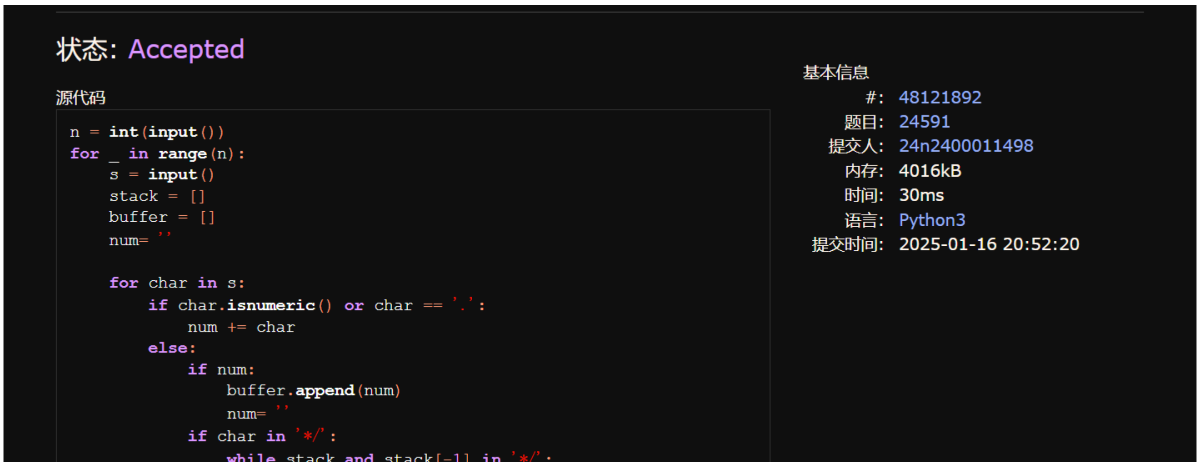
优先级是在比较的过程中确定的，出现后边的运算符才知道应不应该处理这个表达式。

代码：

```
n = int(input())
for _ in range(n):
    s = input()
    stack = []
    buffer = []
    num= ''

    for char in s:
        if char.isnumeric() or char == '.':
            num += char
        else:
            if num:
                buffer.append(num)
                num= ''
            if char in '*/':
                while stack and stack[-1] in '*/':
                    buffer.append(stack.pop())
                stack.append(char)
            elif char in '+-':
                while stack and stack[-1] in '+-*/':
                    buffer.append(stack.pop())
                stack.append(char)
            elif char in '(':
                stack.append(char)
            elif char in ')':
                while stack[-1] in '+-*/':
```

```python
                    buffer.append(stack.pop())
                stack.pop()
    if num:
        buffer.append(num)
    while stack:
        buffer.append(stack.pop())

    print(*buffer, sep = ' ')
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```python
n = int(input())
for _ in range(n):
    s = input()
    stack = []
    buffer = []
    num= ''

    for char in s:
        if char.isnumeric() or char == '.':
            num += char
        else:
            if num:
                buffer.append(num)
                num= ''
            if char in '*/':
                while stack and stack[-1] in '*/':
```

基本信息

#:	48121892
题目:	24591
提交人:	24n2400011498
内存:	4016kB
时间:	30ms
语言:	Python3
提交时间:	2025-01-16 20:52:20

03253: 约瑟夫问题No.2

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

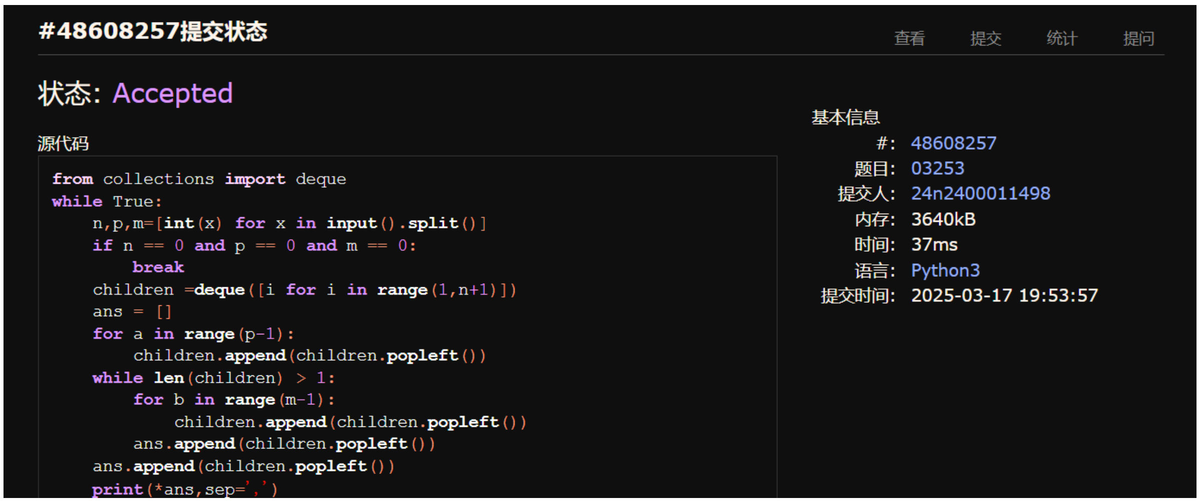
请用队列实现。

似乎使用deque比alist.pop()慢?

代码:

```python
#37ms AC
from collections import deque
while True:
    n,p,m=[int(x) for x in input().split()]
    if n == 0 and p == 0 and m == 0:
        break
    children =deque([i for i in range(1,n+1)])
    ans = []
    for a in range(p-1):
        children.append(children.popleft())
    while len(children) > 1:
        for b in range(m-1):
            children.append(children.popleft())
        ans.append(children.popleft())
    ans.append(children.popleft())
    print(*ans,sep=',')
# 21ms AC
while True:
    n,p,m=[int(x) for x in input().split()]
    if n == 0 and p == 0 and m == 0:
        break
    children = [i for i in range(1,n+1)]
    ans = []
    for a in range(p-1):
        children.append(children.pop(0))
    while len(children) > 1:
        for b in range(m-1):
            children.append(children.pop(0))
        ans.append(children.pop(0))
    ans.append(children.pop(0))
    print(*ans,sep=',')
```

代码运行截图 （至少包含有"Accepted"）



#48608257提交状态

状态: Accepted

源代码

```
from collections import deque
while True:
    n,p,m=[int(x) for x in input().split()]
    if n == 0 and p == 0 and m == 0:
        break
    children =deque([i for i in range(1,n+1)])
    ans = []
    for a in range(p-1):
        children.append(children.popleft())
    while len(children) > 1:
        for b in range(m-1):
            children.append(children.popleft())
        ans.append(children.popleft())
    ans.append(children.popleft())
    print(*ans,sep=',')
```

基本信息

#:	48608257
题目:	03253
提交人:	24n2400011498
内存:	3640kB
时间:	37ms
语言:	Python3
提交时间:	2025-03-17 19:53:57

20018: 蚂蚁王国的越野跑

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

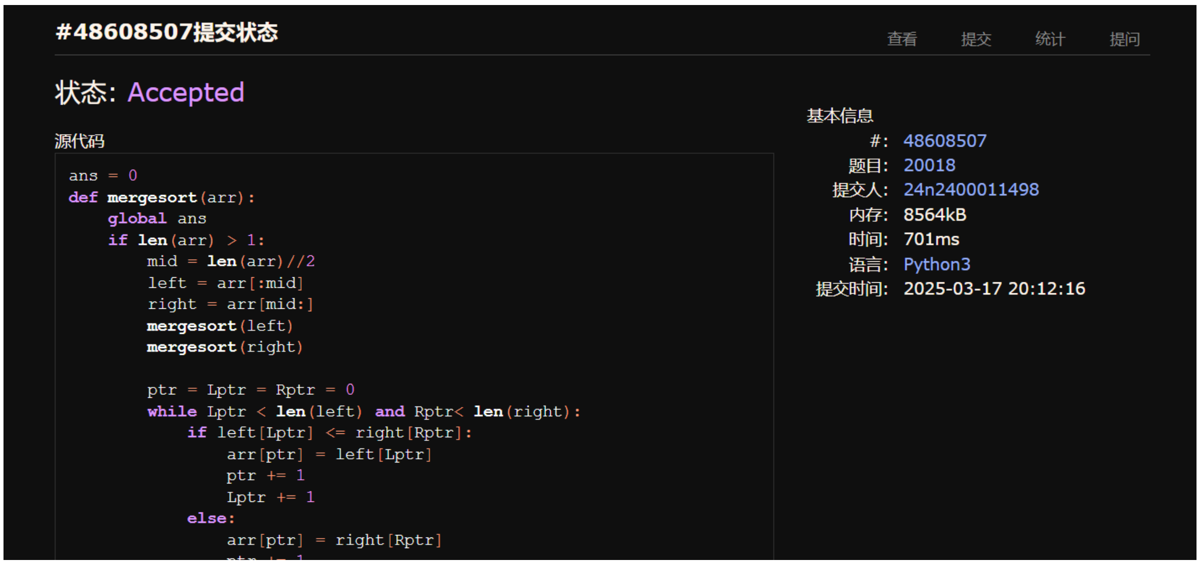
思路：

直接使用归并排序即可，注意先出现的数字在队列的后面。

代码：

```
ans = 0
def mergesort(arr):
    global ans
    if len(arr) > 1:
        mid = len(arr)//2
        left = arr[:mid]
        right = arr[mid:]
        mergesort(left)
        mergesort(right)

        ptr = Lptr = Rptr = 0
        while Lptr < len(left) and Rptr< len(right):
            if left[Lptr] <= right[Rptr]:
                arr[ptr] = left[Lptr]
                ptr += 1
                Lptr += 1
            else:
                arr[ptr] = right[Rptr]
                ptr += 1
                Rptr += 1
                ans += len(left)-Lptr
        while Lptr < len(left):
            arr[ptr] = left[Lptr]
            ptr += 1
            Lptr += 1
        while Rptr<len(right):
            arr[ptr] = right[Rptr]
            ptr += 1
            Rptr += 1
n = int(input())
s = []
for i in range(n):
    s.append(int(input()))
s.reverse()
mergesort(s)
print(ans)
```

代码运行截图 （至少包含有"Accepted"）



#48608507提交状态

状态: Accepted

源代码

```
ans = 0
def mergesort(arr):
    global ans
    if len(arr) > 1:
        mid = len(arr)//2
        left = arr[:mid]
        right = arr[mid:]
        mergesort(left)
        mergesort(right)

        ptr = Lptr = Rptr = 0
        while Lptr < len(left) and Rptr< len(right):
            if left[Lptr] <= right[Rptr]:
                arr[ptr] = left[Lptr]
                ptr += 1
                Lptr += 1
            else:
                arr[ptr] = right[Rptr]
```

基本信息

#:	48608507
题目:	20018
提交人:	24n2400011498
内存:	8564kB
时间:	701ms
语言:	Python3
提交时间:	2025-03-17 20:12:16

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。

感觉这次作业的内容比较简单，基本可以完全独立完成（当然应该也和寒假看了一点点链表有关......）