Assignment #4: 排序、栈、队列和树

Updated 0005 GMT+8 March 11, 2024

2024 spring, Complied by ==张宇帆 心理与认知科学学院==

编程环境

==（请改为同学的操作系统、编程环境等）==

操作系统：Windows 11

Python编程环境：Spyder IDE 5.2.2

1. 题目

05902: 双端队列

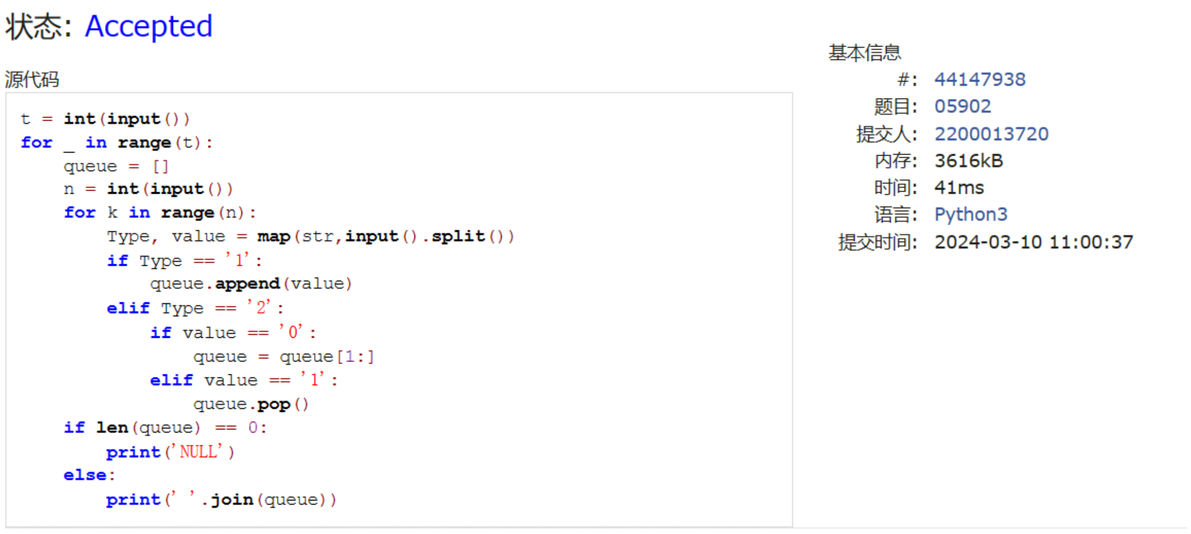
http://cs101.openjudge.cn/practice/05902/

思路：简单的队列实现即可

代码

```python
# 
t = int(input())
for _ in range(t):
    queue = []
    n = int(input())
    for k in range(n):
        Type, value = map(str,input().split())
        if Type == '1':
            queue.append(value)
        elif Type == '2':
            if value == '0':
                queue = queue[1:]
            elif value == '1':
                queue.pop()
    if len(queue) == 0:
        print('NULL')
    else:
        print(' '.join(queue))
```

代码运行截图 ==（至少包含有"Accepted"）==


状态: Accepted

源代码

```
t = int(input())
for _ in range(t):
    queue = []
    n = int(input())
    for k in range(n):
        Type, value = map(str,input().split())
        if Type == '1':
            queue.append(value)
        elif Type == '2':
            if value == '0':
                queue = queue[1:]
            elif value == '1':
                queue.pop()
    if len(queue) == 0:
        print('NULL')
    else:
        print(' '.join(queue))
```

基本信息

#: 44147938
题目: 05902
提交人: 2200013720
内存: 3616kB
时间: 41ms
语言: Python3
提交时间: 2024-03-10 11:00:37

02694: 波兰表达式

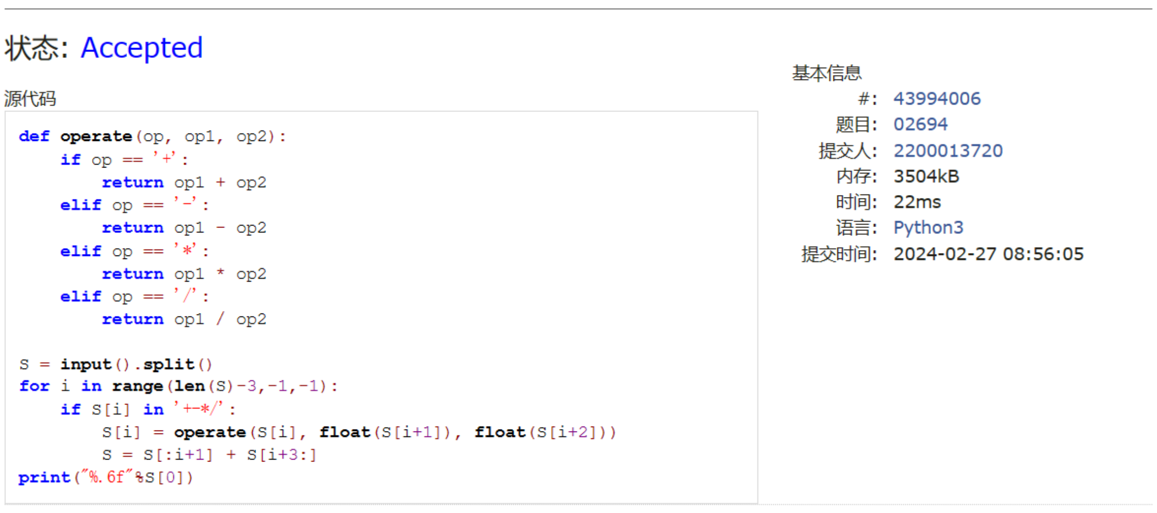
http://cs101.openjudge.cn/practice/02694/

思路：由于不想用栈，于是使用了以前学计概的时候用的思路，即运算符比较作用于排列于其前两位的数字，然后更新列表即可

代码

```
#
def operate(op, op1, op2):
    if op == '+':
        return op1 + op2
    elif op == '-':
        return op1 - op2
    elif op == '*':
        return op1 * op2
    elif op == '/':
        return op1 / op2

s = input().split()
for i in range(len(s)-3,-1,-1):
    if s[i] in '+-*/':
        s[i] = operate(s[i], float(s[i+1]), float(s[i+2]))
        s = s[:i+1] + s[i+3:]
print("%.6f"%s[0])
```

代码运行截图 ==（至少包含有"Accepted"）==


状态: Accepted

源代码

```
def operate(op, op1, op2):
    if op == '+':
        return op1 + op2
    elif op == '-':
        return op1 - op2
    elif op == '*':
        return op1 * op2
    elif op == '/':
        return op1 / op2

s = input().split()
for i in range(len(s)-3,-1,-1):
    if s[i] in '+-*/':
        s[i] = operate(s[i], float(s[i+1]), float(s[i+2]))
        s = s[:i+1] + s[i+3:]
print("%.6f"%s[0])
```

基本信息

#:	43994006
题目:	02694
提交人:	2200013720
内存:	3504kB
时间:	22ms
语言:	Python3
提交时间:	2024-02-27 08:56:05

24591: 中序表达式转后序表达式

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

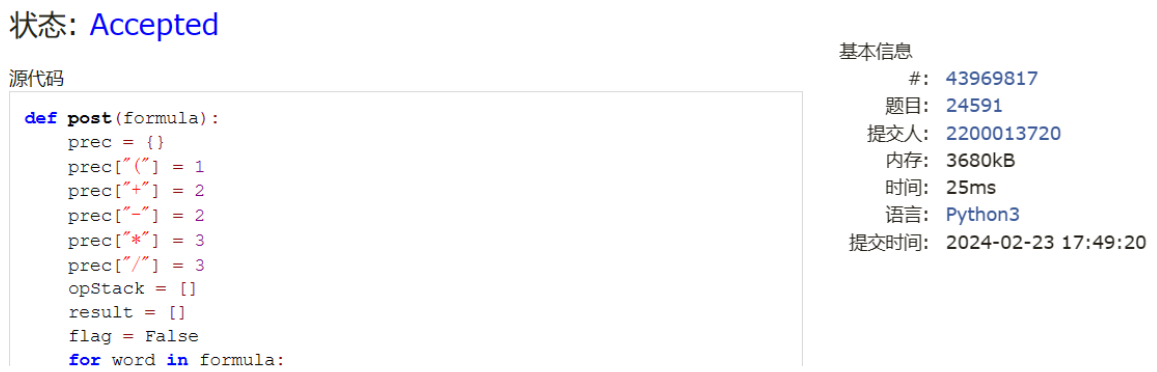
思路：一开始就是简单的用栈去实现，后来发现还得考虑小数和多位数的情况，于是添加了flag作为是否为小数和多位数（即是否直接添加到stack[-1]当中）的判断标识

代码

```
#
def post(formula):
    prec = {}
    prec["("] = 1
    prec["+"] = 2
    prec["-"] = 2
    prec["*"] = 3
    prec["/"] = 3
    opStack = []
    result = []
    flag = False
    for word in formula:
        if word == '(':
            opStack.append(word)
            flag = False
        elif word == ')':
            topword = opStack.pop()
            while not topword == '(':
                result.append(topword)
                topword = opStack.pop()
            flag = False
        elif word == '.':
            flag = True
            result[-1] += word
        elif word in '+-*/':
            while (not opStack == []) and (prec[opStack[-1]] >= prec[word]):
                result.append(opStack.pop())
            opStack.append(word)
            flag = False
```

```python
        elif flag:
            result[-1] += word
        elif word not in '+-*/()':
            result.append(word)
            flag = True
    while not opStack == []:
        result.append(opStack.pop())
    return ' '.join(result)

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

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



状态: Accepted

源代码

```python
def post(formula):
    prec = {}
    prec["("] = 1
    prec["+"] = 2
    prec["-"] = 2
    prec["*"] = 3
    prec["/"] = 3
    opStack = []
    result = []
    flag = False
    for word in formula:
```

基本信息

#:	43969817
题目:	24591
提交人:	2200013720
内存:	3680kB
时间:	25ms
语言:	Python3
提交时间:	2024-02-23 17:49:20

22068: 合法出栈序列

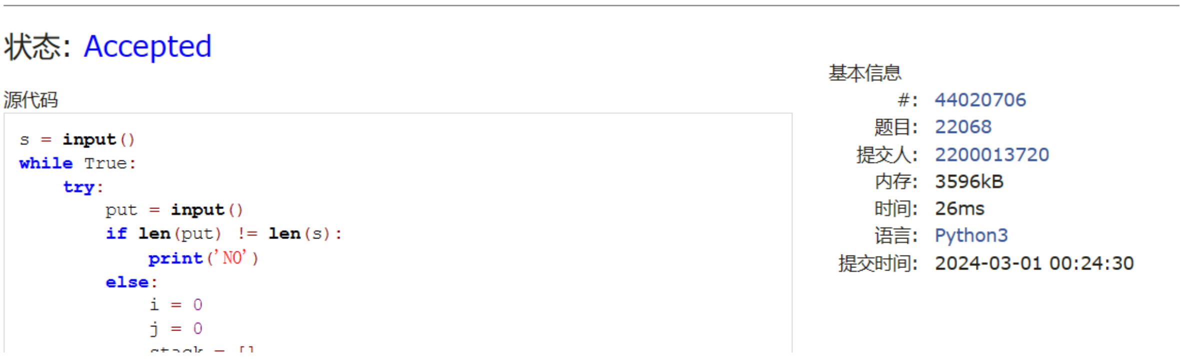
http://cs101.openjudge.cn/practice/22068/

思路：一开始是没有什么好的思路的，后来借鉴了同学发群里询问的代码后才有了思路，简单来说就是在stack中模拟出原始字符串可能的出栈序列，在模拟时与输入字符串相比即可

代码

```python
#
s = input()
while True:
    try:
        put = input()
        if len(put) != len(s):
            print('NO')
        else:
            i = 0
            j = 0
            stack = []
            flag = True
            while i<=len(s) and j<len(put):
                if len(stack) == 0:
                    if i == len(s)-1:
                        break
                    else:
                        stack.append(s[i])
                        i += 1
                if put[j] == stack[-1]:
                    stack.pop()
                    j += 1
                elif put[j] in stack or i == len(s):
                    flag = False
                    break
                else:
                    stack.append(s[i])
                    i += 1
            print(['NO','YES'][flag])
    except EOFError:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



状态: Accepted

源代码

```
s = input()
while True:
    try:
        put = input()
        if len(put) != len(s):
            print('NO')
        else:
            i = 0
            j = 0
```

基本信息

#: 44020706
题目: 22068
提交人: 2200013720
内存: 3596kB
时间: 26ms
语言: Python3
提交时间: 2024-03-01 00:24:30

06646: 二叉树的深度

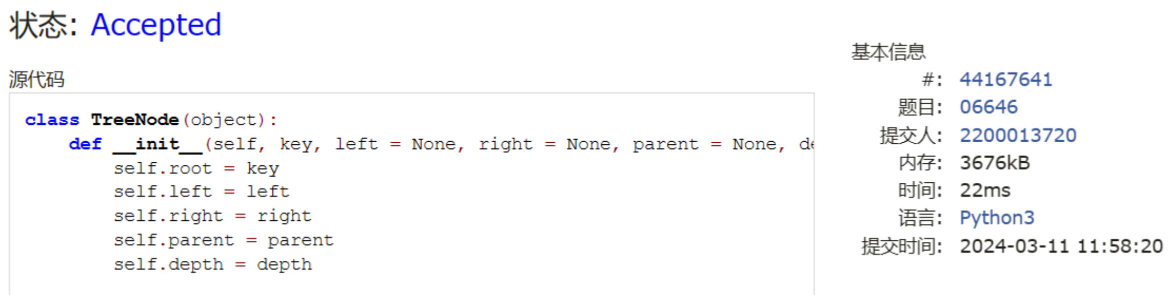
http://cs101.openjudge.cn/practice/06646/

思路：简单的写下树然后写个找深度的函数即可（一开始还担心递归太多TLE的，但是就目前做过的树题目看来好像不太需要担心这方面）

代码

```
#
class TreeNode(object):
    def __init__(self, key, left = None, right = None, parent = None, depth = 1):
        self.root = key
        self.left = left
        self.right = right
        self.parent = parent
        self.depth = depth

    def isleaf(self):
        return self.left == None and self.right == None

def finddepth(treenode, trees):
    if treenode.isleaf():
        treenode.depth = 1
    if not treenode.left == None:
        finddepth(treenode.left, trees)
        treenode.depth = max(treenode.left.depth + 1, treenode.depth)
    if not treenode.right == None:
        finddepth(treenode.right, trees)
        treenode.depth = max(treenode.right.depth + 1, treenode.depth)
    return treenode.depth
n = int(input())
trees = [TreeNode(1) for i in range(n)]
for i in range(n):
    left, right = map(int,input().split())
    if not left == -1:
        trees[i].left = trees[left-1]
        trees[left-1].parent = trees[i]
    if not right == -1:
        trees[i].right = trees[right-1]
        trees[right-1].parent = trees[i]
print(finddepth(trees[0], trees))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==


状态: Accepted

源代码

```
class TreeNode(object):
    def __init__(self, key, left = None, right = None, parent = None, de
        self.root = key
        self.left = left
        self.right = right
        self.parent = parent
        self.depth = depth
```

基本信息

#:	44167641
题目:	06646
提交人:	2200013720
内存:	3676kB
时间:	22ms
语言:	Python3
提交时间:	2024-03-11 11:58:20

02299: Ultra-QuickSort

http://cs101.openjudge.cn/practice/02299/

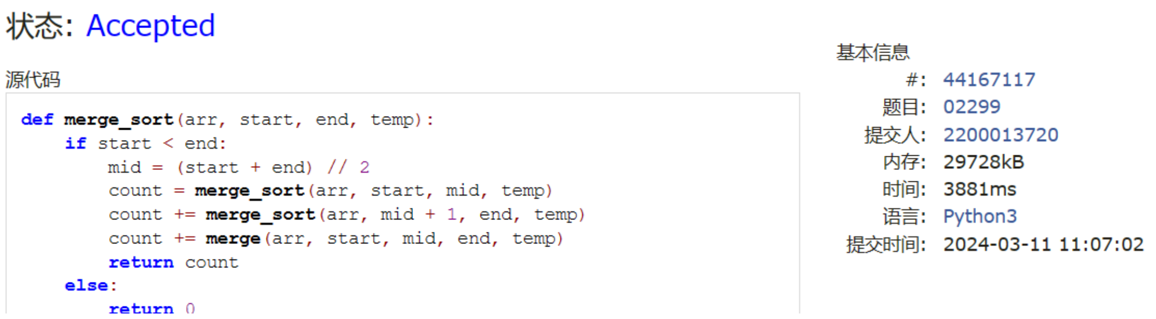
思路：一开始是直接求逆序数，后来TLE了想不出怎么搞，询问了GPT后给出了归并才想起来（他打出merge_sort的时候就恍然大悟了）

代码

```
#
def merge_sort(arr, start, end, temp):
    if start < end:
        mid = (start + end) // 2
        count = merge_sort(arr, start, mid, temp)
        count += merge_sort(arr, mid + 1, end, temp)
        count += merge(arr, start, mid, end, temp)
        return count
    else:
        return 0
```

```python
def merge(arr, start, mid, end, temp):
    count = 0
    i = start
    j = mid + 1
    idx = start
    while i <= mid and j <= end:
        if arr[i] <= arr[j]:
            temp[idx] = arr[i]
            i += 1
        else:
            temp[idx] = arr[j]
            j += 1
            count += mid - i + 1
        idx += 1
    while i <= mid:
        temp[idx] = arr[i]
        i += 1
        idx += 1

    while j <= end:
        temp[idx] = arr[j]
        j += 1
        idx += 1

    for i in range(start, end + 1):
        arr[i] = temp[i]

    return count



n = int(input())
while n != 0:
    num = []
    dp = [0]*n
    for i in range(n):
        num.append(int(input()))
    print(merge_sort(num, 0, n-1, [0]*n))
    n = int(input())
```

代码运行截图 == （AC代码截图，至少包含有"Accepted"） ==



状态: Accepted

源代码

```python
def merge_sort(arr, start, end, temp):
    if start < end:
        mid = (start + end) // 2
        count = merge_sort(arr, start, mid, temp)
        count += merge_sort(arr, mid + 1, end, temp)
        count += merge(arr, start, mid, end, temp)
        return count
    else:
        return 0
```

基本信息

#:	44167117
题目:	02299
提交人:	2200013720
内存:	29728kB
时间:	3881ms
语言:	Python3
提交时间:	2024-03-11 11:07:02

2. 学习总结和收获

==如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。==

最近在浅浅地做一点CF上的dp和greedy的一些题目（1100+），树的题目没做多少，目前就还在执行上一个作业说的“好好再练练dp”，不过刚好最近每日练习和作业已经有树了，所以也可以慢慢跟着走（只是最近的重心就不在这了）。

还有比较难受的一点就是，我没想到这才第四周，我的其他课程就已经开始给我上难度了(┬┬﹏┬┬)，所以这周末确实没有花太多时间在数算上，至少没有以前那么多了，不过之后会努力调整的。

最后就是用心地搭了下自己的github，把自己做过的dp、greedy等题目放了上去，然后也放有最近十分折磨我的课程的一些内容（机器学习方面），打算记录下自己学习的经历，总之慢慢用心学吧！