Assignment #4: 排序、栈、队列和树

Updated 0005 GMT+8 March 11, 2024

2024 spring, Complied by Xinjie Song, Phy

说明：

1）The complete process to learn DSA from scratch can be broken into 4 parts:

Learn about Time complexities, learn the basics of individual Data Structures, learn the basics of Algorithms, and practice Problems.

2）请把每个题目解题思路（可选），源码Python, 或者C++（已经在Codeforces/Openjudge上AC），截图（包含Accepted），填写到下面作业模版中（推荐使用 typora https://typoraio.cn ，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

3）提交时候先提交pdf文件，再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。

4）如果不能在截止前提交作业，请写明原因。

编程环境

操作系统：Windows 11 22H2

Python编程环境：PyCharm 2023.2 (Community Edition)

C/C++编程环境：g++ (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0

1. 题目

05902: 双端队列

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

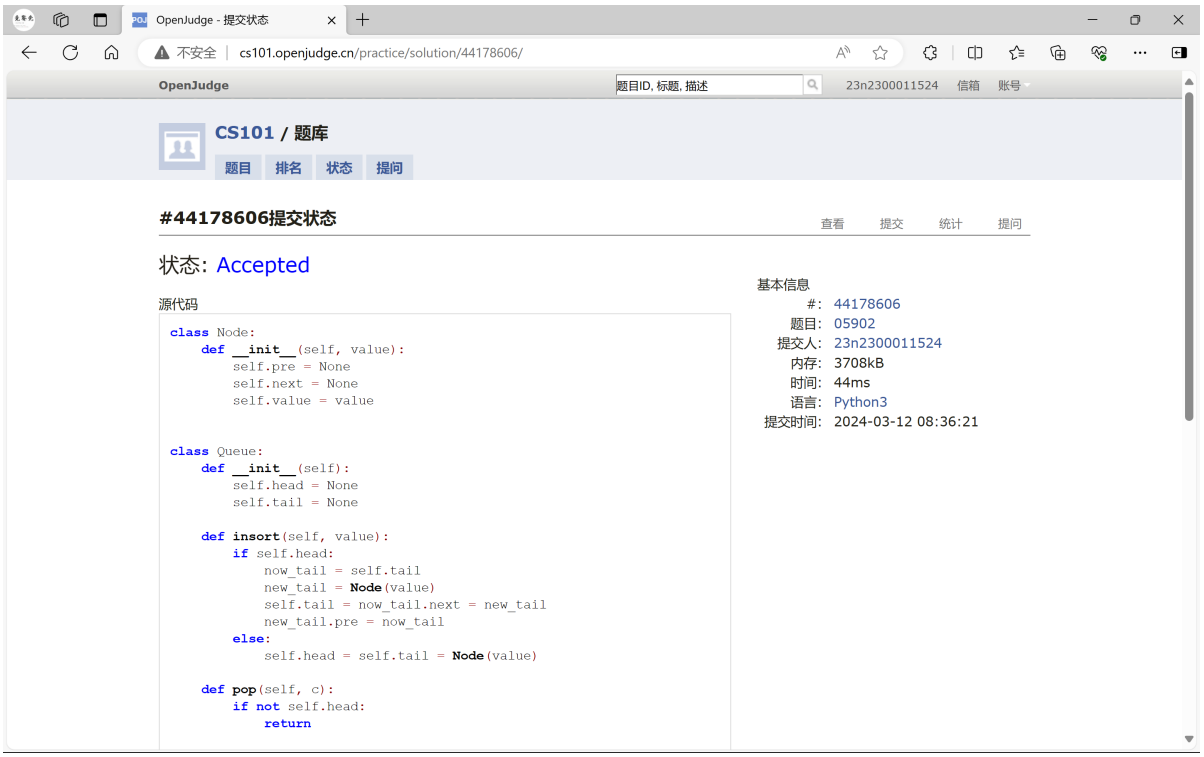
思路：自定义类

代码

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


class Queue:
    def __init__(self):
```

```
        self.head = None
        self.tail = None

    def insort(self, value):
        if self.head:
            now_tail = self.tail
            new_tail = Node(value)
            self.tail = now_tail.next = new_tail
            new_tail.pre = now_tail
        else:
            self.head = self.tail = Node(value)

    def pop(self, c):
        if not self.head:
            return

        if c:
            self.tail = self.tail.pre
            if self.tail:
                self.tail.next = None
            else:
                self.head = None
        else:
            self.head = self.head.next
            if self.head:
                self.head.pre = None
            else:
                self.tail = None

    def __str__(self):
        if not self.head:
            return 'NULL'

        s = [str(self.head.value)]
        t = self.head
        while t.next:
            t = t.next
            s.append(str(t.value))

        return ' '.join(s)


for i in range(int(input())):
    queue = Queue()
    for j in range(int(input())):
        t, c = map(int, input().split())
        if t == 1:
            queue.insort(c)
        else:
            queue.pop(c)
    print(queue)
```

代码运行截图



02694: 波兰表达式

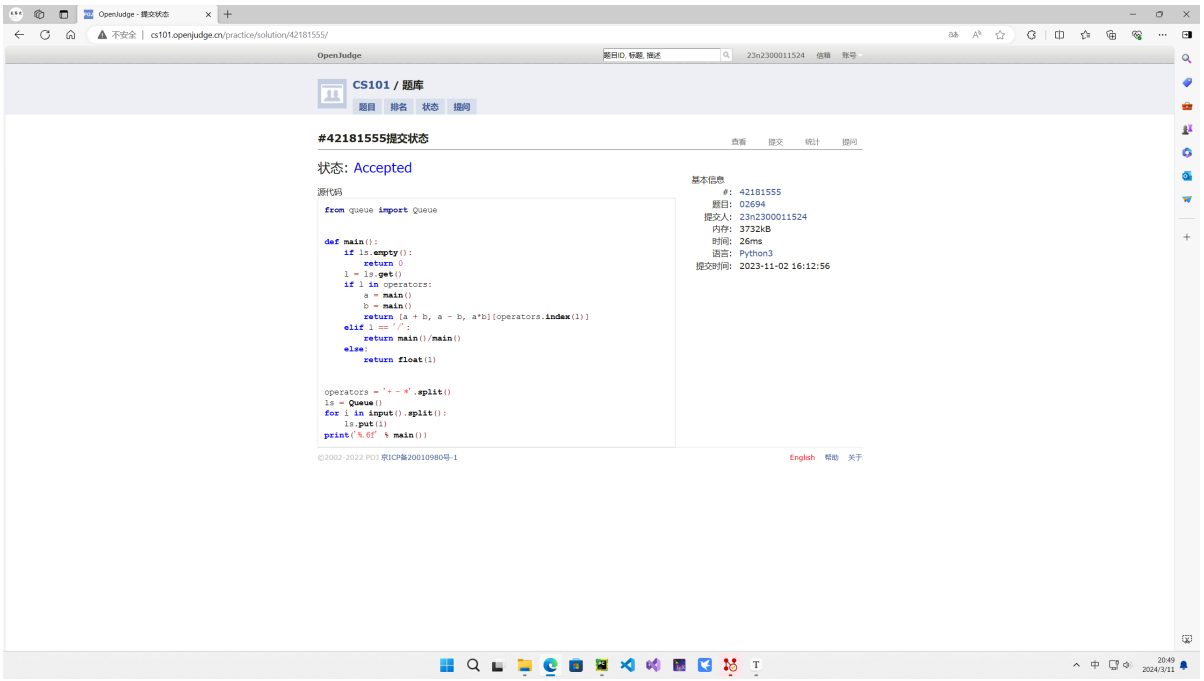
http://cs101.openjudge.cn/practice/02694/

思路：递归

代码

```
from queue import Queue


def main():
    if ls.empty():
        return 0
    l = ls.get()
    if l in operators:
        a = main()
        b = main()
        return [a + b, a - b, a*b][operators.index(l)]
    elif l == '/':
        return main()/main()
    else:
        return float(l)


operators = '+ - *'.split()
ls = Queue()
for i in input().split():
    ls.put(i)
print('%.6f' % main())
```

代码运行截图



24591: 中序表达式转后序表达式

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

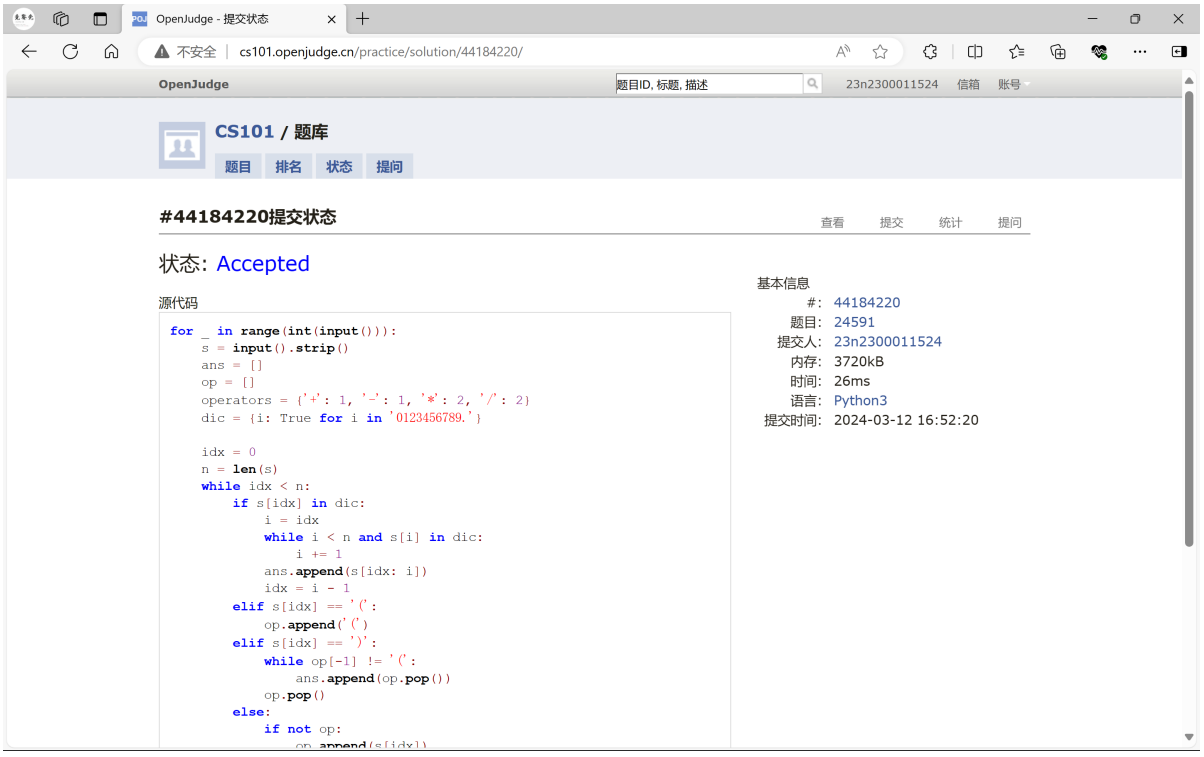
思路：学习了很久终于搞对了

代码

```
for _ in range(int(input())):
    s = input().strip()
    ans = []
    op = []
    operators = {'+': 1, '-': 1, '*': 2, '/': 2}
    dic = {i: True for i in '0123456789.'}

    idx = 0
    n = len(s)
    while idx < n:
        if s[idx] in dic:
            i = idx
            while i < n and s[i] in dic:
                i += 1
            ans.append(s[idx: i])
            idx = i - 1
        elif s[idx] == '(':
            op.append('(')
        elif s[idx] == ')':
            while op[-1] != '(':
                ans.append(op.pop())
```

```
            op.pop()
        else:
            if not op:
                op.append(s[idx])
            elif op[-1] == '(' or operators[op[-1]] < operators[s[idx]]:
                op.append(s[idx])
            else:
                while op and op[-1] != '(' and operators[op[-1]] >=
operators[s[idx]]:
                    ans.append(op.pop())
                op.append(s[idx])
        idx += 1
    if op:
        op.reverse()
        print(f'{" ".join(ans)} {" ".join(op)}')
    else:
        print(" ".join(ans))
```

代码运行截图



22068: 合法出栈序列

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

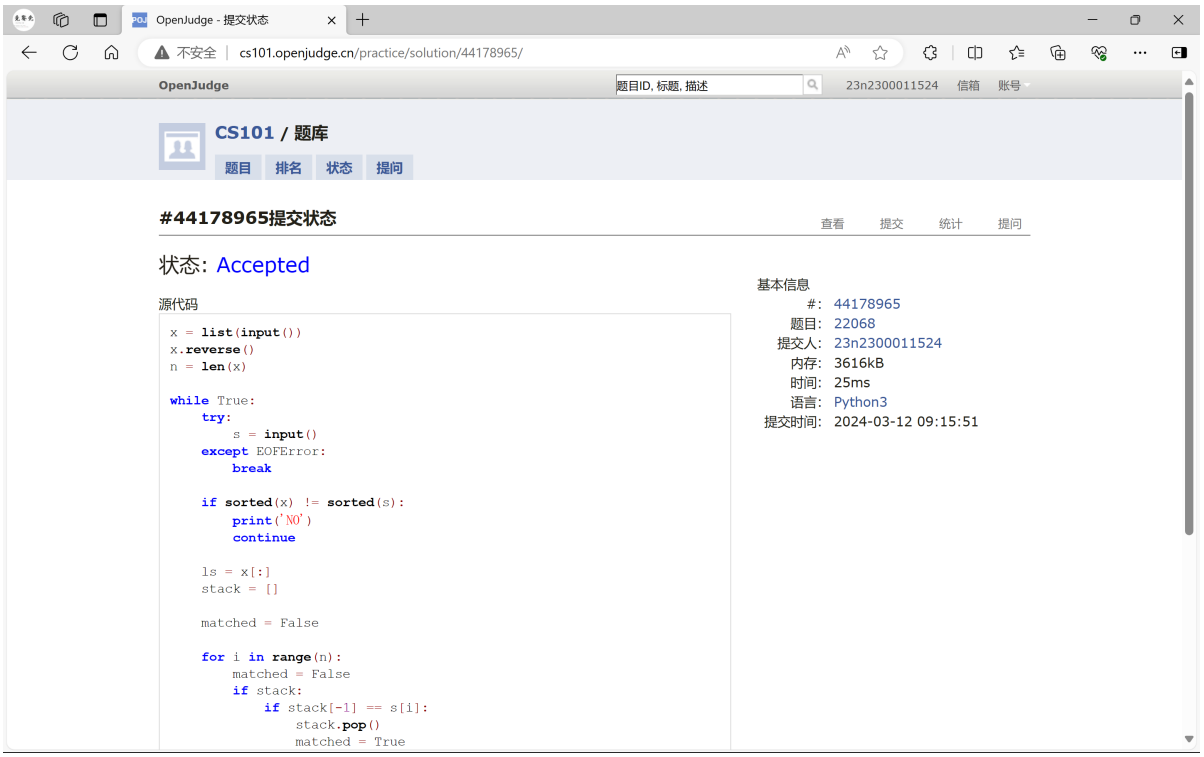
思路：模拟法?

代码

```
x = list(input())
```

```
x.reverse()
n = len(x)

while True:
    try:
        s = input()
    except EOFError:
        break

    if sorted(x) != sorted(s):
        print('NO')
        continue

    ls = x[:]
    stack = []

    matched = False

    for i in range(n):
        matched = False
        if stack:
            if stack[-1] == s[i]:
                stack.pop()
                matched = True
        if matched:
            continue
        while ls:
            if ls[-1] == s[i]:
                ls.pop()
                matched = True
                break
            else:
                stack.append(ls.pop())
        if not matched:
            print('NO')
            break

    if matched:
        print('YES')
```

代码运行截图



06646: 二叉树的深度

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

思路：递归

代码

```
n = int(input())

father = {i: None for i in range(1, n + 1)}
son = {i: [-1, -1] for i in range(1, n + 1)}

for i in range(1, n + 1):
    l, r = map(int, input().split())
    son[i] = [l, r]
    father[l] = father[r] = i


def h(idx):
    ans = 1

    l, r = son[idx]
    if l != -1:
        ans = max(ans, 1 + h(l))
    if r != -1:
        ans = max(ans, 1 + h(r))

    return ans
```

```
for key, value in father.items():
    if not value:
        print(h(key))
        break
```

代码运行截图

02299: Ultra-QuickSort

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

思路：树状数组法超内存了，学习了分治法完成题目

代码

```
ls = []


def merge_sort(i, j):
    if j <= i:
        return 0
    mid = (i + j) >> 1
    t = merge_sort(i, mid) + merge_sort(mid + 1, j)

    temp = ls[i: j + 1]
    mid -= i
    l, r = 0, mid + 1
    for idx in range(i, j + 1):
```

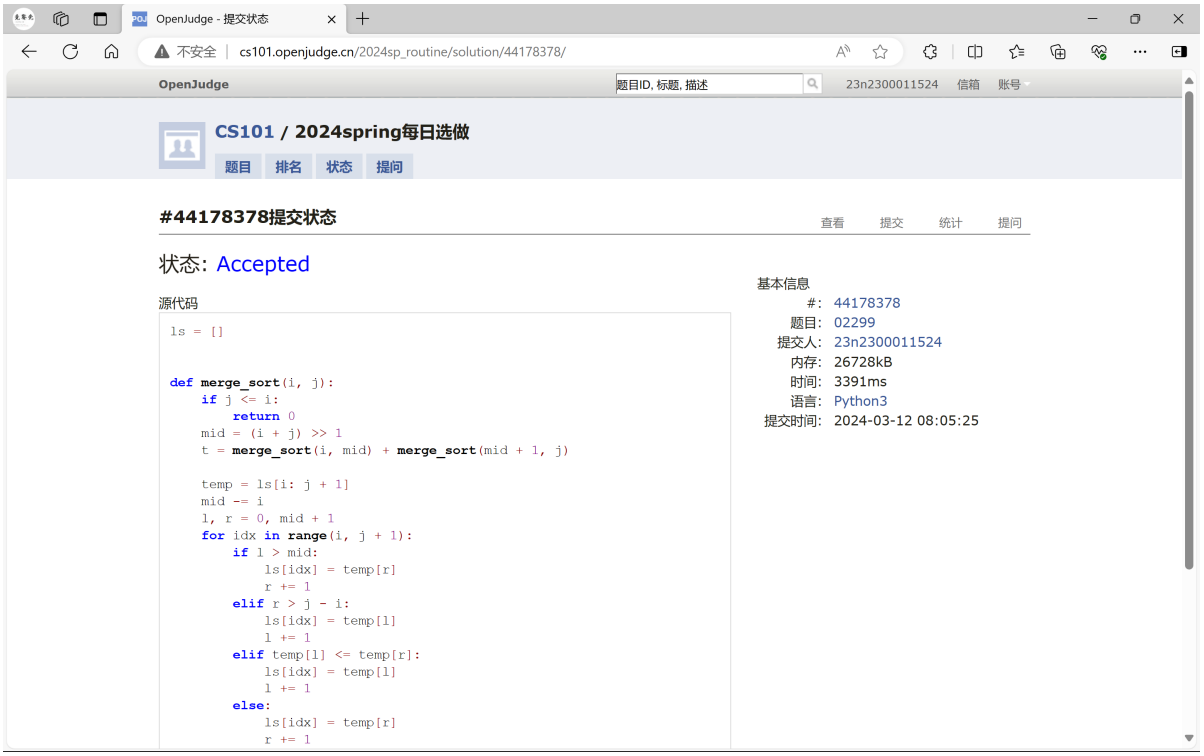
```
        if l > mid:
            ls[idx] = temp[r]
            r += 1
        elif r > j - i:
            ls[idx] = temp[l]
            l += 1
        elif temp[l] <= temp[r]:
            ls[idx] = temp[l]
            l += 1
        else:
            ls[idx] = temp[r]
            r += 1
            t += mid - l + 1

    return t


while True:
    n = int(input())
    if not n:
        break
    ls = [int(input()) for _ in range(n)]
    print(merge_sort(0, n - 1))
```

代码（树状数组）

```
while True:
    ans = 0
    n = int(input())
    if not n:
        break

    ls = sorted([(int(input()), i + 1) for i in range(n)])
    ls = [i[1] for i in ls]

    tr = [0] * (n + 1)
    for i in range(1, n + 1):
        while i <= n:
            tr[i] += 1
            i += i & -i

    for idx in ls:
        j = idx
        while j <= n:
            tr[j] += -1
            j += j & -j

        x = 0
        y = idx - 1

        while y > x:
            ans += tr[y]
            y -= y & -y
```

```
        while x > y:
            ans -= tr[x]
            x -= x & -x

    print(ans)
```

代码运行截图



2. 学习总结和收获

树状数组惨遭内存超出，现学分治依旧遥遥领先。

合法出栈序列简单模拟，树节无树求二叉树深度。

双端队列还是宝宝巴士，波兰表达式仍游刃有余。

中序转后序写了两小时，水平不够还得多家练习！