Assignment #4: 排序、栈、队列和树

Updated 0005 GMT+8 March 11, 2024

2024 spring, Complied by ==祁轩宇、经济学院==

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

==（请改为同学的操作系统、编程环境等）==

操作系统：Windows 11, version 23H2

Python编程环境：VSCode 1.87.1

C/C++编程环境：

1. 题目

05902: 双端队列

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

思路：

代码

```
for i in range(int(input())):
    queue=[]
    command=[]
    n=int(input())
    error=0
    for j in range(n):
        command.append([*map(int,input().split())])
    for j in range(n):
        if command[j][0]==2:
```

```
            if len(queue)==0:
                error=1
                break
            else:
                queue.pop(command[j][1]*(len(queue)-1))
        else:
            k=str(command[j][1])
            queue.append(k)

    if error:
        print('error')
    elif len(queue)==0:
        print('NULL')
    else:
        print(' '.join(queue))
```

代码运行截图 ==（至少包含有"Accepted"）==

#44300163提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
for i in range(int(input())):
    queue=[]
    command=[]
    n=int(input())
    error=0
    for j in range(n):
        command.append([*map(int,input().split())])
    for j in range(n):
        if command[j][0]==2:
            if len(queue)==0:
                error=1
                break
            else:
                queue.pop(command[j][1]*(len(queue)-1))
        else:
            k=str(command[j][1])
            queue.append(k)

    if error:
        print('error')
    elif len(queue)==0:
        print('NULL')
    else:
        print(' '.join(queue))
```

基本信息

#: 44300163
题目: 05902
提交人: 2100015440
内存: 3736kB
时间: 43ms
语言: Python3
提交时间: 2024-03-19 17:00:49

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

02694: 波兰表达式

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

思路：

代码

```
def Polish(l):
    C=['+','-','*','/']
    if len(l)>1:
```

```
        for i in range(len(l)):
            if l[i] in C:
                if (l[i+1] in C) | (l[i+2] in C):
                    continue
                else:
                    cal=str(l[i+1])+str(l[i])+str(l[i+2])
                    l[i]=eval(cal)
                    l.pop(i+2)
                    l.pop(i+1)
                    return Polish(l)
    elif len(l)==1:
        return l[0]

l=input().split()
print(f'{Polish(l):.6f}')
```

代码运行截图 ==（至少包含有"Accepted"）==

#44078909提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def Polish(l):
    C=['+','-','*','/']
    if len(l)>1:
        for i in range(len(l)):
            if l[i] in C:
                if (l[i+1] in C) | (l[i+2] in C):
                    continue
                else:
                    cal=str(l[i+1])+str(l[i])+str(l[i+2])
                    l[i]=eval(cal)
                    l.pop(i+2)
                    l.pop(i+1)
                    return Polish(l)
    elif len(l)==1:
        return l[0]

l=input().split()
print(f'{Polish(l):.6f}')
```

基本信息

#:	44078909
题目:	02694
提交人:	2100015440
内存:	3612kB
时间:	21ms
语言:	Python3
提交时间:	2024-03-05 17:11:34

English 帮助 关于

24591: 中序表达式转后序表达式

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

思路：

代码

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

```
    for char in expression:
        if char.isnumeric() or char == '.':
            number += char
        else:
            if number:
                num = float(number)
                postfix.append(int(num) if num.is_integer() else num)
                number = ''
            if char in '+-*/':
                while stack and stack[-1] in '+-*/' and precedence[char] <=
precedence[stack[-1]]:
                    postfix.append(stack.pop())
                stack.append(char)
            elif char == '(':
                stack.append(char)
            elif char == ')':
                while stack and stack[-1] != '(':
                    postfix.append(stack.pop())
                stack.pop()

    if number:
        num = float(number)
        postfix.append(int(num) if num.is_integer() else num)

    while stack:
        postfix.append(stack.pop())

    return ' '.join(str(x) for x in postfix)

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

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44300852提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 44300852
题目: 24591
提交人: 2100015440
内存: 3684kB
时间: 29ms
语言: Python3
提交时间: 2024-03-19 17:32:19

源代码

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

    for char in expression:
        if char.isnumeric() or char == '.':
            number += char
        else:
            if number:
                num = float(number)
                postfix.append(int(num) if num.is_integer() else num)
                number = ''
            if char in '+-*/':
                while stack and stack[-1] in '+-*/' and precedence[char]
                    postfix.append(stack.pop())
                stack.append(char)
            elif char == '(':
                stack.append(char)
            elif char == ')':
                while stack and stack[-1] != '(':
                    postfix.append(stack.pop())
                stack.pop()

    if number:
        num = float(number)
        postfix.append(int(num) if num.is_integer() else num)

    while stack:
        postfix.append(stack.pop())

    return ' '.join(str(x) for x in postfix)

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

22068: 合法出栈序列

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

思路：

代码

```
def validout(l0, l1):
    stack = []
    i = 0
    if len(l0)!=len(l1):
        return False
    else:
        for j in l0:
            stack.append(j)
            while stack and stack[-1] == l1[i]:
                stack.pop()
```

```
                    i += 1
            return not stack

    X = input()
    l0 = []
    for i in range(len(X)):
        l0.append(X[i])

    while True:
        try:
            xl = input()
            l1 = []
            for i in range(len(xl)):
                l1.append(xl[i])
            if validout(l0, l1):
                print('YES')
            else:
                print('NO')
        except EOFError:
            break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44302012提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 44302012
题目: 22068
提交人: 2100015440
内存: 3672kB
时间: 25ms
语言: Python3
提交时间: 2024-03-19 18:40:58

源代码

```
def validout(l0, l1):
    stack = []
    i = 0
    if len(l0)!=len(l1):
        return False
    else:
        for j in l0:
            stack.append(j)
            while stack and stack[-1] == l1[i]:
                stack.pop()
                i += 1
        return not stack

X = input()
l0 = []
for i in range(len(X)):
    l0.append(X[i])

while True:
    try:
        xl = input()
        l1 = []
        for i in range(len(xl)):
            l1.append(xl[i])
        if validout(l0, l1):
            print('YES')
        else:
            print('NO')
    except EOFError:
        break
```

06646: 二叉树的深度

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

思路：

代码

```
class TreeNode:
    def __init__(self):
        self.left = None
        self.right = None

def tree_depth(node):
    if node is None:
        return 0
    left_depth = tree_depth(node.left)
    right_depth = tree_depth(node.right)
    return max(left_depth, right_depth) + 1

n = int(input())  # 读取节点数量
nodes = [TreeNode() for _ in range(n)]

for i in range(n):
    left_index, right_index = map(int, input().split())
    if left_index != -1:
        nodes[i].left = nodes[left_index-1]
    if right_index != -1:
        nodes[i].right = nodes[right_index-1]

root = nodes[0]
depth = tree_depth(root)
print(depth)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44302153提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class TreeNode:
    def __init__(self):
        self.left = None
        self.right = None

def tree_depth(node):
    if node is None:
        return 0
    left_depth = tree_depth(node.left)
    right_depth = tree_depth(node.right)
    return max(left_depth, right_depth) + 1

n = int(input())  # 读取节点数量
nodes = [TreeNode() for _ in range(n)]

for i in range(n):
    left_index, right_index = map(int, input().split())
    if left_index != -1:
        nodes[i].left = nodes[left_index-1]
    if right_index != -1:
        nodes[i].right = nodes[right_index-1]

root = nodes[0]
depth = tree_depth(root)
print(depth)
```

基本信息

#: 44302153
题目: 06646
提交人: 2100015440
内存: 3624kB
时间: 24ms
语言: Python3
提交时间: 2024-03-19 18:49:14

02299: Ultra-QuickSort

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

思路：

代码

```
from bisect import *

while True:
    n = int(input())
    if n == 0:
        break
    a = []
    rev = 0
    for _ in range(n):
        num = int(input())
        rev += bisect_left(a, num)
        insort_left(a, num)
    ans = n * (n - 1) // 2 - rev
    print(ans)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44302382提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from bisect import *

while True:
    n = int(input())
    if n == 0:
        break
    a = []
    rev = 0
    for _ in range(n):
        num = int(input())
        rev += bisect_left(a, num)
        insort_left(a, num)
    ans = n * (n - 1) // 2 - rev
    print(ans)
```

基本信息

#: 44302382
题目: 02299
提交人: 2100015440
内存: 21524kB
时间: 28468ms
语言: Python3
提交时间: 2024-03-19 19:02:25

2. 学习总结和收获

题目对我来说比较难，中序表达式转后序表达式对着讲义捋了一遍，合法出栈序列和Ultra-QuickSort没有想明白，最后看群里的解法清楚了一些。