

Assignment #5: "树"算：概念、表示、解析、遍历

Updated 2124 GMT+8 March 17, 2024

2024 spring, Compiled by 田济维

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

(python pycharm)

操作系统：macOS Ventura 13.4.1 (c)

Python编程环境：Spyder IDE 5.2.2, PyCharm 2023.1.4 (Professional Edition)

C/C++编程环境：Mac terminal vi (version 9.0.1424), g++/gcc (Apple clang version 14.0.3, clang-1403.0.22.14.1)

1. 题目

27638: 求二叉树的高度和叶子数目

<http://cs101.openjudge.cn/practice/27638/>

思路：

代码

```
1  #
2  class Node:
3      def __init__(self):
```

```

4         self.left = None
5         self.right = None
6     # 求出以node为根结点子树的高度
7     def Treeheight(node):
8         if node == None:
9             return -1
10        else:
11            return max(Treeheight(node.left),Treeheight(node.right))+1
12    # 数出树中结点的个数
13    def Countleaves(node):
14        if node == None:
15            return 0
16        l = node.left
17        r = node.right
18        if l == None and r == None:
19            return 1
20        else:
21            return Countleaves(node.left)+Countleaves(node.right)
22
23    n = int(input())
24    if n == 0:
25        print(-1,0)
26        exit()
27    Nodes = [Node() for i in range(n)]
28    parents = [True for i in range(n)]
29    for i in range(n):
30        l,r = map(int,input().split())
31        if l !=-1:
32            Nodes[i].left = Nodes[l]
33            parents[l]=False
34        if r !=-1:
35            Nodes[i].right = Nodes[r]
36            parents[r]=False
37    s = parents.index(True)
38    print(Treeheight(Nodes[s]),Countleaves(Nodes[s]))

```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
class Node:
    def __init__(self):
        self.left = None
        self.right = None
# 求出以node为根结点的子树的高度
def Treeheight(node):
    if node == None:
        return -1
    else:
        return max(Treeheight(node.left), Treeheight(node.right)) + 1
# 数出树中结点的个数
def Countleaves(node):
    if node == None:
        return 0
    l = node.left
    r = node.right
    if l == None and r == None:
        return 1
    else:
        return Countleaves(node.left) + Countleaves(node.right)

n = int(input())
if n == 0:
    print(-1, 0)
    exit()
Nodes = [Node() for i in range(n)]
parents = [True for i in range(n)]
for i in range(n):
    l, r = map(int, input().split())
    if l != -1:
        Nodes[i].left = Nodes[l]
        parents[l] = False
    if r != -1:
        Nodes[i].right = Nodes[r]
        parents[r] = False
s = parents.index(True)
print(Treeheight(Nodes[s]), Countleaves(Nodes[s]))
```

基本信息

#: 43967973
题目: 27638
提交人: 23n2300011503
内存: 3616kB
时间: 21ms
语言: Python3
提交时间: 2024-02-23 15:28:40

24729: 括号嵌套树

<http://cs101.openjudge.cn/practice/24729/>

思路:

代码

```
1 #
2 # 先构造树
3 class Node:
4     def __init__(self, item):
5         self.key = item
6         self.child = []
7
8 alpha = [chr(i) for i in range(65, 91)]
9
10 s = input()
11 def BuildTree(s):
12     pstack = []
13     for x in s:
14         if x in alpha:
```

```

15         pstack.append(Node(x))
16     elif x == "(":
17         pstack.append(x)
18     elif x == ")":
19         temp = []
20         while pstack[-1]!="(":
21             temp.append(pstack.pop())
22         temp.reverse()
23         pstack.pop()
24         B = pstack[-1]
25         B.child = temp
26     return pstack[0]
27 a = BuildTree(s)
28 def preorder(Tree):
29     if Tree:
30         print(Tree.key,end = "")
31         for x in Tree.child:
32             preorder(x)
33
34 def postorder(Tree):
35     if Tree:
36         for x in Tree.child:
37             postorder(x)
38         print(Tree.key,end = "")
39
40 preorder(a)
41 print("")
42 postorder(a)

```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```

# 先构造树
class Node:
    def __init__(self,item):
        self.key = item
        self.child = []

alpha= [chr(i) for i in range(65,91)]

s = input()
def BuildTree(s):
    pstack = []
    for x in s:
        if x in alpha:
            pstack.append(Node(x))
        elif x == "(":
            pstack.append(x)
        elif x == ")":
            temp = []
            while pstack[-1]!="(":
                temp.append(pstack.pop())
            temp.reverse()
            pstack.pop()
            B = pstack[-1]
            B.child = temp
    return pstack[0]

```

基本信息

#: 44303105
 题目: 24729
 提交人: 23n2300011503
 内存: 4024kB
 时间: 25ms
 语言: Python3
 提交时间: 2024-03-19 19:42:56

02775: 文件结构“图”

<http://cs101.openjudge.cn/practice/02775/>

思路:

代码

```
1  #
2  class Node:
3      def __init__(self, key):
4          self.item = key
5          self.left = []
6          self.right = []
7          self.notused = True
8  def Traverse(Tree):
9      result = [Tree.item]
10     if Tree.left:
11         for x in Tree.left:
12             s1 = Traverse(x)
13             for y in s1:
14                 t = "|" + y
15                 result.append(t)
16     if Tree.right:
17         for x in Tree.right:
18             result.append(x.item)
19     return result
20
21
22 s=[]
23 index = 1
24 while True:
25     x = input()
26     if x == "#":
27         break
28     elif x != "*":
29         s.append(x)
30     elif x == "*":
31         print(f"DATA SET {index}:")
32         pstack = []
33         for y in s:
34             if y!="]":
35                 pstack.append(Node(y))
36             elif y == "]":
37                 tf = []
38                 td = []
39                 while pstack:
40                     if (pstack[-1].item)[0] == "f":
41                         tf.append(pstack.pop())
42                 else:
43                     if pstack[-1].notused:
44                         break
```

```

45         else:
46             td.append(pstack.pop())
47             h = pstack[-1]
48             for i in range(len(td)):
49                 h.left.append(td.pop())
50                 tf = sorted(tf, key = lambda x: x.item)
51                 h.right = tf
52                 h.notused = False
53     u = Node("ROOT")
54     for y in pstack:
55         if (y.item)[0] == "d":
56             u.left.append(y)
57         else:
58             u.right.append(y)
59     u.right = sorted(u.right, key = lambda x: x.item)
60     a = Traverse(u)
61     for t in a:
62         print(t)
63     print("")
64     index += 1
65     s.clear()

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
class Node:
    def __init__(self, key):
        self.item = key
        self.left = []
        self.right = []
        self.notused = True
def Traverse(Tree):
    result = [Tree.item]
    if Tree.left:
        for x in Tree.left:
            s1 = Traverse(x)
            for y in s1:
                t = "|" + y
                result.append(t)
    if Tree.right:
        for x in Tree.right:
            result.append(x.item)
    return result

s=[]
index = 1
while True:
    x = input()
    if x == "#":
        break
    elif x != "*":
        s.append(x)
    elif x == "*":
        print(f"DATA SET {index}:")
        pstack = []
        for y in s:
            if y!="]":
                pstack.append(Node(y))
            elif y == "]":
                tf = []
                td = []
                while pstack:
                    if (pstack[-1].item)[0] == "f":
                        tf.append(pstack.pop())
                    else:
                        if pstack[-1].notused:
                            break
                        else:
                            td.append(pstack.pop())
                h = pstack[-1]
                for i in range(len(td)):
                    h.left.append(td.pop())
                tf = sorted(tf, key = lambda x:x.item)
                h.right = tf
                h.notused = False
        u = Node("ROOT")
        for y in pstack:
            if (y.item)[0]=="d":
                u.left.append(y)
            else:
                u.right.append(y)
        u.right = sorted(u.right, key = lambda x :x.item)
        a=Traverse(u)
        for t in a:
            print(t)
        print("")
```

基本信息

#: 44402717
题目: 02775
提交人: 23n2300011503
内存: 3612kB
时间: 24ms
语言: Python3
提交时间: 2024-03-25 23:50:16

25140: 根据后序表达式建立队列表达式

<http://cs101.openjudge.cn/practice/25140/>

思路:

代码

```
1 # from collections import deque
2
```

```

3  # 创建结点类型
4  class TreeNode:
5      def __init__(self,item):
6          self.key = item
7          self.left = None
8          self.right = None
9
10 # 把后序表达式转化为树状图
11 def BuildTree(plist):
12     pstack = [] # 用来模拟运算过程
13     for x in plist:
14         node = TreeNode(x)
15         if x.isupper():
16             node.right = pstack.pop()
17             node.left = pstack.pop()
18             pstack.append(node)
19     return pstack[0]
20
21 # 从树状图中读取列序
22 def listorder(node):
23     que = deque()
24     traversal = []
25     que.append(node)
26     while que:
27         for x in range(len(que)):
28             s = que.popleft()
29             traversal.append(s.key)
30             if s.left:
31                 que.append(s.left)
32             if s.right:
33                 que.append(s.right)
34
35     return traversal
36
37 n = int(input())
38
39 for _ in range(n):
40     string = input()
41     result = listorder(BuildTree(string))
42     result.reverse()
43     print("".join(result))
44
45
46

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
from collections import deque

# 创建结点类型
class TreeNode:
    def __init__(self, item):
        self.key = item
        self.left = None
        self.right = None

# 把后序表达式转化为树状图
def BuildTree(plist):
    pstack = [] # 用来模拟运算过程
    for x in plist:
        node = TreeNode(x)
        if x.isupper():
            node.right = pstack.pop()
            node.left = pstack.pop()
            pstack.append(node)
    return pstack[0]

# 从树状图中读取列序
def listorder(node):
    que = deque()
    traversal = []
    que.append(node)
    while que:
        for x in range(len(que)):
            s = que.popleft()
            traversal.append(s.key)
            if s.left:
                que.append(s.left)
            if s.right:
                que.append(s.right)

    return traversal

n = int(input())

for _ in range(n):
    string = input()
    result = listorder(BuildTree(string))
    result.reverse()
    print("".join(result))
```

基本信息

#: 44303233
题目: 25140
提交人: 23n23000111
内存: 3672kB
时间: 29ms
语言: Python3
提交时间: 2024-03-19

24750: 根据二叉树中后序序列建树

<http://cs101.openjudge.cn/practice/24750/>

思路:

代码

```

1  #
2  infix = input()
3  post = input()
4  def preorder(infix, post):
5      if infix:
6          key = post[-1]
7          l = infix.index(key)
8          print(key, end="")
9          preorder(infix[:l], post[:l])
10         preorder(infix[l+1:], post[l+1:])
11     preorder(infix, post)
12

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```

infix = input()
post = input()
def preorder(infix, post):
    if infix:
        key = post[-1]
        l = infix.index(key)
        print(key, end="")
        preorder(infix[:l], post[:l])
        preorder(infix[l+1:], post[l+1:])
preorder(infix, post)

```

基本信息

#: 43992953
 题目: 24750
 提交人: 23n2300011503
 内存: 4824kB
 时间: 24ms
 语言: Python3
 提交时间: 2024-02-26 21:13:04

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

22158: 根据二叉树前中序序列建树

<http://cs101.openjudge.cn/practice/22158/>

思路:

代码

```

1  def postorder(infix, pre):
2      if infix:
3          key = pre[0]
4          l = infix.index(key)
5
6          postorder(infix[:l], pre[1:l+1])
7          postorder(infix[l+1:], pre[l+1:])
8          print(key, end="")
9
10     while True:
11         try:
12             pre = input()
13             infix = input()
14
15         except EOFError:

```

```
16         break
17     else:
18         postorder(infix, pre)
19     print("")
20
21
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
def postorder(infix, pre):
    if infix:
        key = pre[0]
        l = infix.index(key)

        postorder(infix[:l], pre[1:l+1])
        postorder(infix[l + 1:], pre[l+1:])
        print(key, end=" ")

    while True:
        try:
            pre = input()
            infix = input()

            except EOFError:
                break
            else:
                postorder(infix, pre)
                print("")
```

基本信息

#: 44303480

题目: 22158

提交人: 23n2300011503

内存: 3552kB

时间: 22ms

语言: Python3

提交时间: 2024-03-19 20:02:22

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

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。

最让我难受并且有成就感的 是文件管理系统。