**27638:求二叉树的高度和叶子数目**

def cs27638(n):  
 def k(ni):  
 l = lst[ni][0]  
 if l > -1:  
 ans[l] = ans[ni] + 1  
 k(l)  
 r = lst[ni][1]  
 if r > -1:  
 ans[r] = ans[ni] + 1  
 k(r)  
  
 lst = []  
 ans = [0] \* n  
 f = [False] \* (n + 1)  
 for i in range(n):  
 ai = tuple(map(int, input().split()))  
 lst.append(ai)  
 f[ai[0]] = True  
 f[ai[1]] = True  
 k(f.index(False))  
 print(max(ans), lst.count((-1, -1)))



## 24729:括号嵌套树

def cs24729(s):  
 lst = []  
 ans1, ans2 = "", ""  
 for i in s:  
 if i in "QWERTYUIOPASDFGHJKLZXCVBNM":  
 lst.append(i)  
 ans1 += i  
 elif i == ",":  
 ans2 += lst.pop()  
 elif i == "(":  
 lst.append(i)  
 elif i == ")":  
 while lst[-1] != "(":  
 ans2 += lst.pop()  
 lst.pop()  
 print(ans1)  
 print(ans2)



## 02775:文件结构“图”

def cs02775():  
 flag = False  
 x = 1  
 di = 0  
 print('DATA SET ' + str(x) + ":")  
 print('ROOT')  
 x += 1  
 lst = [[] for i in range(30)]  
 while True:  
 s = input()  
 if s == "#":  
 exit()  
 elif s == "\*":  
 lst[0].sort()  
 for i in lst[0]:  
 print(i)  
 lst = [[] for i in range(30)]  
 flag = True  
 continue  
 if flag:  
 print()  
 print('DATA SET ' + str(x) + ":")  
 print('ROOT')  
 x += 1  
 flag = False  
 if s[0] == "f":  
 lst[di].append(s)  
 elif s[0] == "d":  
 di += 1  
 print("| " \* di + s)  
 elif s == "]":  
 lst[di].sort()  
 for i in lst[di]:  
 print("| " \* di + i)  
 lst[di] = []  
 di -= 1



## 25140:根据后序表达式建立表达式树

def cs25140(n):  
 for i in range(n):  
 s = input()  
 l = len(s)  
 lst = [[-1, -1] for j in range(l)]  
 tmp, ans = [], ""  
 for j in range(l):  
 if 65 <= ord(s[j]) <= 90:  
 lst[j][1] = tmp.pop()  
 lst[j][0] = tmp.pop()  
 tmp.append(j)  
 else:  
 tmp.append(j)  
 tem = [[tmp.pop()]]  
  
 def k(ni):  
 tem.append([])  
 for j in tem[ni]:  
 if lst[j][0] > -1:  
 tem[ni + 1].append(lst[j][0])  
 if lst[j][1] > -1:  
 tem[ni + 1].append(lst[j][1])  
 if tem[ni + 1]:  
 k(ni + 1)  
  
 k(0)  
 for j in range(len(tem)):  
 for k in tem[j]:  
 ans = s[k] + ans  
 print(ans)



## 24750:根据二叉树中后序序列建树

def cs24750():  
 ans = [""]  
  
 def pre(m, p):  
 if len(m) >= 1:  
 a = m.split(p[-1])  
 ans[0] += p[-1]  
 for i in a:  
 pre(i, p[:len(i)])  
 p = p[len(i):]  
  
 pre(input(), input())  
 print(ans[0])



## 22158:根据二叉树前中序序列建树

def cs22158():  
  
 def post(p, m):  
 if len(m) >= 1:  
 a = m.split(p[0])  
 a.reverse()  
 ans[0] = p[0] + ans[0]  
 for i in a:  
 if i:  
 post(p[-len(i):], i)  
 p = p[:-len(i)]  
  
 while True:  
 try:  
 ans = [""]  
 post(input(), input())  
 print(ans[0])  
  
 except:  
 exit()  
  
  
cs22158()



总结：树的问题我习惯于使用链表结构去存储，并以递归的方式处理数据，其优点是代码简洁。建立完整的类的写法我还在跟着题解学习，其优点是思路清晰。这次出现的BUG都是循环里忘记初始化参量导致的，仍需注意。