Assignment #6: "树"算：Huffman,BinHeap,BST,AVL,DisjointSet

Updated 2214 GMT+8 March 24, 2024

2024 spring, Complied by ==黄源森，工学院==

说明：

1）这次作业内容不简单，耗时长的话直接参考题解。

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：W11

Python编程环境：Spyder IDE 5.2.2

1. 题目

22275: 二叉搜索树的遍历

http://cs101.openjudge.cn/practice/22275/

思路：

代码

```
#
def f(l):
    if len(l)==1:
        return [l[0]]
    x=l[0]
    if x<l[1]:
        return f(l[1:])+[x]
    if x>l[-1]:
        return f(l[1:])+[x]
    for i in range(len(l)):
```

```
        if l[i]>x:
            break
    return f(l[1:i])+f(l[i:])+[x]
n=int(input())
l=list(map(int,input().split()))
print(*f(l))
```

代码运行截图 ==（至少包含有"Accepted"）==

状态: Accepted

源代码

```
def f(l):
    if len(l)==1:
        return [l[0]]
    x=l[0]
    if x<l[1]:
        return f(l[1:])+[x]
    if x>l[-1]:
        return f(l[1:])+[x]
    for i in range(len(l)):
        if l[i]>x:
            break
    return f(l[1:i])+f(l[i:])+[x]
n=int(input())
l=list(map(int,input().split()))
print(*f(l))
```

基本信息

#:	44334807
题目:	22275
提交人:	23n2300011031
内存:	3824kB
时间:	23ms
语言:	Python3
提交时间:	2024-03-22 10:38:12

05455: 二叉搜索树的层次遍历

http://cs101.openjudge.cn/practice/05455/

思路:

代码

```
#
l=list(map(int,input().split()))
class Node:
    def __init__(self,val,left=None,right=None):
        self.val=val
        self.left=left
        self.right=right
class Tree:
    def __init__(self,root):
        self.root=Node(root)
    def insert(self,w):
        cur=self.root
        while 1:
            if w<cur.val:
                if cur.left==None:
                    cur.left=Node(w)
                    break
                else:
```

```
                    cur=cur.left
            elif w>cur.val:
                if cur.right==None:
                    cur.right=Node(w)
                    break
                else:
                    cur=cur.right
            else:
                break
    def dfs(self,a,layer):
        if a==None:
            return
        al[layer].append(a.val)
        self.dfs(a.left,layer+1)
        self.dfs(a.right,layer+1)

s=Tree(l[0])
for u in l[1:]:
    s.insert(u)
al=[[] for _ in range(100)]
s.dfs(s.root,0)
ans=[]
for u in al:
    ans.extend(u)
print(*ans)
```

代码运行截图 ==（至少包含有"Accepted"）==

状态: Accepted

源代码

```
l=list(map(int,input().split()))
class Node:
    def __init__(self,val,left=None,right=None):
        self.val=val
        self.left=left
        self.right=right
class Tree:
    def __init__(self,root):
        self.root=Node(root)
    def insert(self,w):
        cur=self.root
        while 1:
            if w<cur.val:
                if cur.left==None:
                    cur.left=Node(w)
                    break
                else:
                    cur=cur.left
            elif w>cur.val:
                if cur.right==None:
                    cur.right=Node(w)
                    break
                else:
                    cur=cur.right
            else:
                break
    def dfs(self,a,layer):
        if a==None:
            return
        al[layer].append(a.val)
        self.dfs(a.left,layer+1)
        self.dfs(a.right,layer+1)

s=Tree(l[0])
for u in l[1:]:
    s.insert(u)
al=[[] for _ in range(100)]
s.dfs(s.root,0)
ans=[]
for u in al:
    ans.extend(u)
print(*ans)
```

基本信息

#: 44192447
题目: 05455
提交人: 23n2300011031
内存: 3692kB
时间: 22ms
语言: Python3
提交时间: 2024-03-13 10:30:31

04078: 实现堆结构

http://cs101.openjudge.cn/practice/04078/

练习自己写个BinHeap。当然机考时候，如果遇到这样题目，直接import heapq。手搓栈、队列、堆、AVL等，考试前需要搓个遍。

思路:

代码

```
#
class Node:
    def __init__(self,v,p):
        self.val=v
        self.left=None
        self.right=None
        self.par=p
class Heap:
    def __init__(self,v):
        self.root=Node(v,None)
    def insert(self,v):
        cur=self.root
```

```
        if cur==None:
            self.root=Node(v,None)
            return
        while cur.left!=None:
            cur=cur.left
        cur.left=Node(v,cur)
        cur=cur.left
        while cur.par!=None and v<cur.par.val:
            cur.val,cur.par.val=cur.par.val,cur.val
            cur=cur.par
    def re(self):
        print(self.root.val)
        cur=self.root
        while cur.left!=None:
            cur=cur.left
        t=cur.val
        if cur==self.root:
            self.root=self.root.right
            return
        cur.par.left=None

        self.root.val=t
        cur=self.root
        while 1:
            try:
                if cur.val>cur.left.val:
                    cur.val,cur.left.val=cur.left.val,cur.val
                    cur=cur.left
                elif cur.val>cur.right.val:
                    cur.val,cur.right.val=cur.right.val,cur.val
                    cur=cur.right
            except:
                break

n=int(input())
a,b=map(int,input().split())
h=Heap(b)
for _ in range(n-1):
    s=input()
    if len(s)>1:
        a,b=s.split()
        b=int(b)
        h.insert(b)
        #print(h.root.val)
    else:
        h.re()
```

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,v,p):
        self.val=v
        self.left=None
        self.right=None
        self.par=p
class Heap:
    def __init__(self,v):
        self.root=Node(v,None)
    def insert(self,v):
        cur=self.root
        if cur==None:
            self.root=Node(v,None)
            return
        while cur.left!=None:
            cur=cur.left
        cur.left=Node(v,cur)
        cur=cur.left
        while cur.par!=None and v<cur.par.val:
            cur.val,cur.par.val=cur.par.val,cur.val
            cur=cur.par
    def re(self):
        print(self.root.val)
        cur=self.root
        while cur.left!=None:
            cur=cur.left
        t=cur.val
        if cur==self.root:
            self.root=self.root.right
            return
        cur.par.left=None

        self.root.val=t
        cur=self.root
        while 1:
            try:
                if cur.val>cur.left.val:
                    cur.val,cur.left.val=cur.left.val,cur.val
                    cur=cur.left
                elif cur.val>cur.right.val:
                    cur.val,cur.right.val=cur.right.val,cur.val
                    cur=cur.right
            except:
                break

n=int(input())
a,b=map(int,input().split())
h=Heap(b)
for _ in range(n-1):
    s=input()
    if len(s)>1:
        a,b=s.split()
        b=int(b)
        h.insert(b)
        #print(h.root.val)
    else:
        h.re()
```

基本信息

#:	44398450
题目:	04078
提交人:	23n2300011031
内存:	4684kB
时间:	5362ms
语言:	Python3
提交时间:	2024-03-25 18:36:10

22161: 哈夫曼编码树

http://cs101.openjudge.cn/practice/22161/

思路:

代码

```
#
class Node:
    def __init__(self,l,s):
        self.con=l
```

```
        self.s=s
        self.left=None
        self.right=None
    def __lt__(self,u):
        if self.s==u.s:
            return self.con[0]<u.con[0]
        return self.s<u.s
dic={}
def f(x,a):
    if x.left==None:
       dic[x.con[0]]=a
       return
    f(x.left,a+'0')
    f(x.right,a+'1')
n=int(input())
al=[]
for _ in range(n):
    a,b=input().split()
    b=int(b)
    x=Node([a], b)
    al.append(x)
while len(al)>1:
    al.sort(reverse=1)
    a=al.pop()
    b=al.pop()
    c=Node(list(sorted(a.con+b.con)),a.s+b.s)
    c.left=a
    c.right=b
    al.append(c)
t=al[0]
f(t,'')
re={}
for u in dic:
    re[dic[u]]=u
while 1:
    try:
        s=input()
        res=''
        if s.isnumeric():
            i=0
            for j in range(1,len(s)):
                if s[i:j] in re:
                    res+=re[s[i:j]]
                    i=j
            res+=re[s[i:]]
        else:
            for u in s:
                res+=dic[u]
        print(res)
    except:
        break
```

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,l,s):
        self.con=l
        self.s=s
        self.left=None
        self.right=None
    def __lt__(self,u):
        if self.s==u.s:
            return self.con[0]<u.con[0]
        return self.s<u.s
dic={}
def f(x,a):
    if x.left==None:
       dic[x.con[0]]=a
       return
    f(x.left,a+'0')
    f(x.right,a+'1')
n=int(input())
al=[]
for _ in range(n):
    a,b=input().split()
    b=int(b)
    x=Node([a], b)
    al.append(x)
while len(al)>1:
    al.sort(reverse=1)
    a=al.pop()
    b=al.pop()
    c=Node(list(sorted(a.con+b.con)),a.s+b.s)
    c.left=a
    c.right=b
    al.append(c)
t=al[0]
f(t,'')
re={}
for u in dic:
    re[dic[u]]=u
while 1:
    try:
        s=input()
        res=''
        if s.isnumeric():
            i=0
            for j in range(1,len(s)):
                if s[i:j] in re:
                    res+=re[s[i:j]]
                    i=j
            res+=re[s[i:]]
        else:
            for u in s:
                res+=dic[u]
        print(res)
    except:
        break
```

基本信息

#:	44392879
题目:	22161
提交人:	23n2300011031
内存:	3740kB
时间:	24ms
语言:	Python3
提交时间:	2024-03-25 09:28:42

晴问9.5: 平衡二叉树的建立

https://sunnywhy.com/sfbj/9/5/359

思路:

代码

```
#
class Node:
    def __init__(self,v):
        self.val=v
        self.left=None
        self.right=None
        self.height=1
class avl:
```

```
    def __init__(self):
        self.root=None
    def getheight(self,node):
        if node:
            return node.height
        return 0
    def getbalance(self,node):
        if node:
            return self.getheight(node.left)-self.getheight(node.right)
        return 0
    def _rotate_left(self, z):
        y = z.right
        T2 = y.left
        y.left = z
        z.right = T2
        z.height = 1 + max(self.getheight(z.left), self.getheight(z.right))
        y.height = 1 + max(self.getheight(y.left), self.getheight(y.right))
        return y

    def _rotate_right(self, y):
        x = y.left
        T2 = x.right
        x.right = y
        y.left = T2
        y.height = 1 + max(self.getheight(y.left), self.getheight(y.right))
        x.height = 1 + max(self.getheight(x.left), self.getheight(x.right))
        return x
    def insert(self, value):
        if not self.root:
            self.root = Node(value)
        else:
            self.root = self._insert(value, self.root)

    def _insert(self, value, node):
        if not node:
            return Node(value)
        elif value < node.value:
            node.left = self._insert(value, node.left)
        else:
            node.right = self._insert(value, node.right)

        node.height = 1 + max(self.getheight(node.left),
self.getheight(node.right))

        balance = self._get_balance(node)

        if balance > 1:
            if value < node.left.value: # 树形是 LL
                return self._rotate_right(node)
            else:   # 树形是 LR
                node.left = self._rotate_left(node.left)
                return self._rotate_right(node)

        if balance < -1:
            if value > node.right.value:    # 树形是 RR
                return self._rotate_left(node)
```

```
            else:   # 树形是 RL
                node.right = self._rotate_right(node.right)
                return self._rotate_left(node)

        return node
    def pre(self,node):
        if not node:
            return []
        return self.pre(node.left)+node.val+self.pre(node.right)

n=int(input())
l=list(map(int,input().split()))
t=avl()
for u in l:
    t.insert(u)
print(*t.pre(t.root))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

02524: 宗教信仰

http://cs101.openjudge.cn/practice/02524/

思路:

代码

```
#
def find(x):
    while dic[x]!=x:
        x=dic[x]
    return x
t=1
while 1:
    n,m=map(int, input().split())
    if n==0:
        break
    dic={}
    for i in range(1,n+1):
        dic[i]=i
    for _ in range(m):
        a,b=map(int, input().split())
        dic[find(a)]=dic[find(b)]
    c=0
    for u in dic:
        if u==dic[u]:
            c+=1
    print('Case %d: %d'%(t,c))
    t+=1
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

状态: Accepted

源代码

```
def find(x):
    while dic[x]!=x:
        x=dic[x]
    return x
t=1
while 1:
    n,m=map(int, input().split())
    if n==0:
        break
    dic={}
    for i in range(1,n+1):
        dic[i]=i
    for _ in range(m):
        a,b=map(int, input().split())
        dic[find(a)]=dic[find(b)]
    c=0
    for u in dic:
        if u==dic[u]:
            c+=1
    print('Case %d: %d'%(t,c))
    t+=1
```

基本信息

#:	43880121
题目:	02524
提交人:	23n2300011031
内存:	10160kB
时间:	1327ms
语言:	Python3
提交时间:	2024-02-08 08:34:02

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。==

堆和avl虽然手写了一遍，但是堆的标准列表写法不够熟，avl全部自己写肯定也不容易，但是写一遍能有更深的印象。其实很多把n变成logn的都是类似树的结构，虽然名称不同，但是这种想法会很有启发性。