

# Assignment #9: Huffman, BST & Heap

Updated 1834 GMT+8 Apr 15, 2025

2025 spring, Compiled by 叶靖、信管

## 说明:

### 1. 解题与记录:

对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge，Codeforces，LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora <https://typoraio.cn> 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。

2. \*\*提交安排:\*\* 提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。

3. \*\*延迟提交:\*\* 如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。

请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

## 1. 题目

### LC222.完全二叉树的节点个数

dfs, <https://leetcode.cn/problems/count-complete-tree-nodes/>

思路:

代码:

```
class Solution:
    def countNodes(self, root: Optional[TreeNode]) -> int:
        def getDepth(node):
            depth = 0
            while node:
                depth += 1
                node = node.left
            return depth

        if not root:
            return 0
```

```

left_depth = getDepth(root.left)
right_depth = getDepth(root.right)
if left_depth == right_depth:
    return (1 << left_depth) + self.countNodes(root.right)
else:
    return (1 << right_depth) + self.countNodes(root.left)

```

代码运行截图（至少包含有"Accepted"）



## LC103.二叉树的锯齿形层序遍历

bfs, <https://leetcode.cn/problems/binary-tree-zigzag-level-order-traversal/>

思路：

代码：

```

class Solution:
    def zigzagLevelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
        if not root:
            return []
        result = []
        queue = deque([root])
        left_to_right = True

        while queue:
            level_size = len(queue)
            level_nodes = deque()

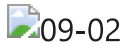
            for _ in range(level_size):
                node = queue.popleft()
                if left_to_right:
                    level_nodes.append(node.val)
                else:
                    level_nodes.appendleft(node.val)

                if node.left:
                    queue.append(node.left)
                if node.right:
                    queue.append(node.right)

            result.append(list(level_nodes))
            left_to_right = not left_to_right
        return result

```

代码运行截图（至少包含有"Accepted"）



## M04080:Huffman编码树

greedy, <http://cs101.openjudge.cn/practice/04080/>

思路:

代码:

```
import heapq

def huffman(codes):
    heapq.heapify(codes) # 将普通列表转成堆结构（默认最小堆）
    sum = 0
    while len(codes) > 1:
        left = heapq.heappop(codes) # 弹出最小元素（堆顶）
        right = heapq.heappop(codes)
        cost = left + right
        sum += cost
        heapq.heappush(codes, cost) # 添加一个元素
    return sum

n = int(input())
codes = list(map(int, input().split()))
print(huffman(codes))
```

代码运行截图（至少包含有"Accepted"）



## M05455: 二叉搜索树的层次遍历

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

思路:

代码:

```
from collections import deque

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

def insert(root, val):
    if root is None:
```

```

        return TreeNode(val)
    if val < root.val:
        root.left = insert(root.left, val)
    elif val > root.val:
        root.right = insert(root.right, val)
    return root

def level_order(root):
    if not root:
        return []
    result = []
    queue = deque([root])
    while queue:
        node = queue.popleft()
        result.append(str(node.val))
        if node.left:
            queue.append(node.left)
        if node.right:
            queue.append(node.right)
    return result

nums = list(map(int, input().split()))
unique_nums = list(dict.fromkeys(nums)) # 去重保序
root = None
for num in unique_nums:
    root = insert(root, num)

print(' '.join(level_order(root)))

```

代码运行截图（至少包含有"Accepted"）



## M04078: 实现堆结构

手搓实现, <http://cs101.openjudge.cn/practice/04078/>

类似的题目是 晴问9.7: 向下调整构建大顶堆, <https://sunnywhy.com/sfbj/9/7>

思路:

代码:

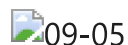
```

n = int(input())
data = []
for _ in range(n):
    procedure = list(map(int, input().split()))
    if procedure[0] == 1:
        data.append(procedure[1])
    elif procedure[0] == 2:

```

```
data.sort()
print(data.pop(0))
```

代码运行截图（至少包含有"Accepted"）



## T22161: 哈夫曼编码树

greedy, <http://cs101.openjudge.cn/practice/22161/>

思路：

代码：

```
import heapq

class Node:
    def __init__(self, weight, chars, left=None, right=None):
        self.weight = weight
        self.chars = chars # 集合
        self.min_char = min(chars)
        self.left = left
        self.right = right

# 用于堆排序：先按权重，再按最小字符排序
def __lt__(self, other):
    if self.weight != other.weight:
        return self.weight < other.weight
    return self.min_char < other.min_char

def build_huffman_tree(char_weights):
    heap = []

    for ch, w in char_weights:
        node = Node(w, {ch})
        heapq.heappush(heap, node)

    while len(heap) > 1:
        node1 = heapq.heappop(heap)
        node2 = heapq.heappop(heap)
        merged = Node(
            node1.weight + node2.weight,
            node1.chars.union(node2.chars),
            left=node1,
            right=node2
        )
        heapq.heappush(heap, merged)

    return heap[0] # 根节点
```

```

def generate_codes(node, prefix="", code_map={}):
    if node.left is None and node.right is None:
        code_map[list(node.chars)[0]] = prefix
    if node.left:
        generate_codes(node.left, prefix + '0', code_map)
    if node.right:
        generate_codes(node.right, prefix + '1', code_map)
    return code_map

def encode(s, code_map):
    return ''.join(code_map[ch] for ch in s)


def decode(bits, root):
    res = ''
    node = root
    for b in bits:
        node = node.left if b == '0' else node.right
        if node.left is None and node.right is None:
            res += list(node.chars)[0]
            node = root
    return res

n = int(input())
char_weights = []
for _ in range(n):
    parts = input().split()
    char_weights.append((parts[0], int(parts[1])))

root = build_huffman_tree(char_weights)
code_map = generate_codes(root)
try:
    while True:
        line = input().strip()
        if not line:
            break
        if set(line).issubset({'0', '1'}):
            print(decode(line, root))
        else:
            print(encode(line, code_map))
except EOFError:
    pass

```

代码运行截图（至少包含有"Accepted"）

09-06

## 2. 学习总结和收获

---

如果发现作业题目相对简单，有否寻找额外的练习题目，如“数算2025spring每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。

对树的题目还是未能掌握，目前进度为一知半解（摇头叹气.gif）。但是我发现我开始掌握到heapq的大致用法了，也算是一小进步！树的题目难以理解，希望我下次功课遇到时不需要ai们问我分析题目。