Assignment #9: 图论：遍历，及 树算

Updated 1739 GMT+8 Apr 14, 2024

2024 spring, Complied by ==祁轩宇、经济学院==

说明：

1）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。

2）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。

3）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

04081: 树的转换

http://cs101.openjudge.cn/dsapre/04081/

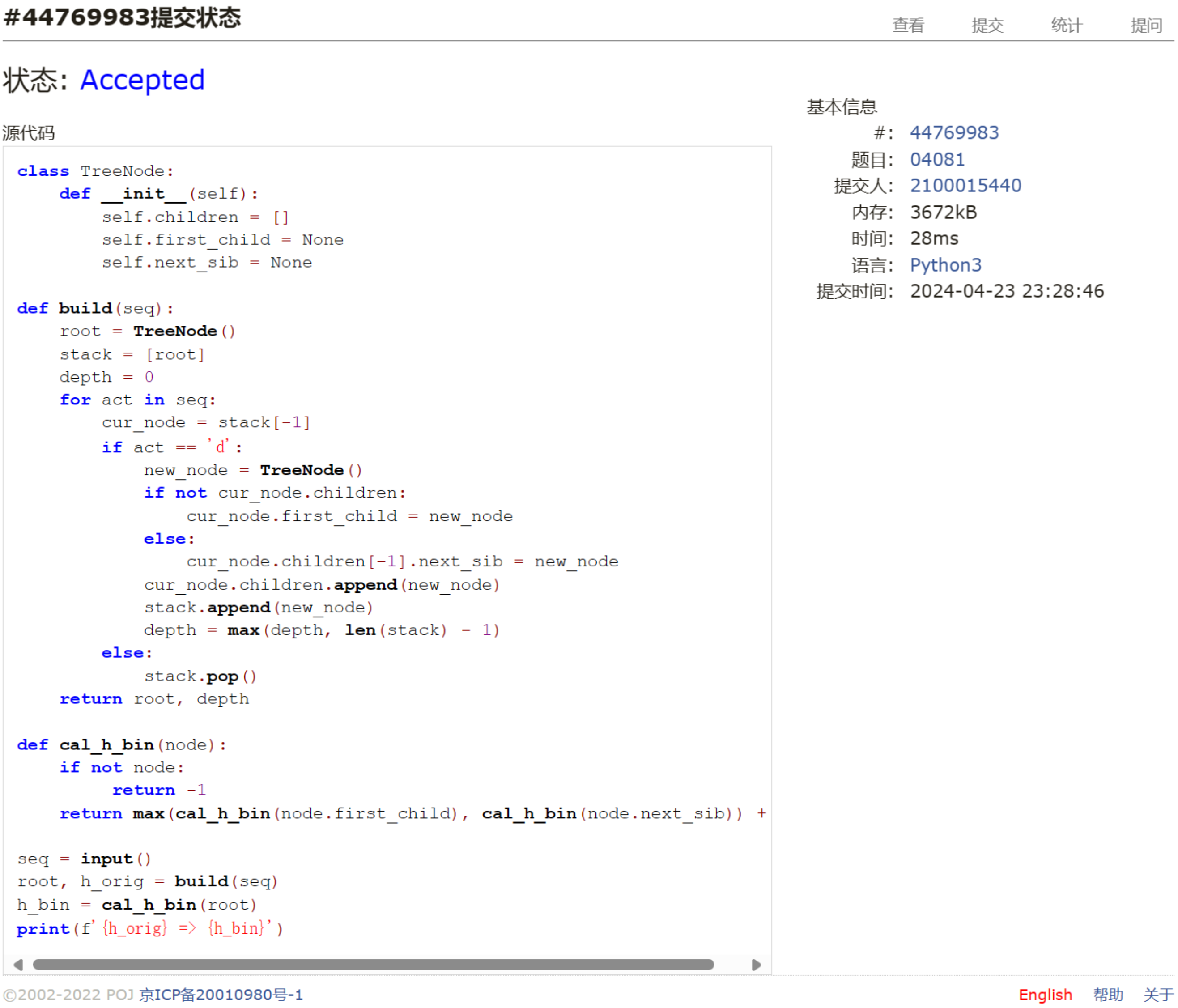
思路：

代码

```
class TreeNode:
    def __init__(self):
        self.children = []
        self.first_child = None
        self.next_sib = None

def build(seq):
    root = TreeNode()
    stack = [root]
    depth = 0
    for act in seq:
        cur_node = stack[-1]
        if act == 'd':
            new_node = TreeNode()
```

```python
            if not cur_node.children:
                cur_node.first_child = new_node
            else:
                cur_node.children[-1].next_sib = new_node
            cur_node.children.append(new_node)
            stack.append(new_node)
            depth = max(depth, len(stack) - 1)
        else:
            stack.pop()
    return root, depth

def cal_h_bin(node):
    if not node:
         return -1
    return max(cal_h_bin(node.first_child), cal_h_bin(node.next_sib)) + 1

seq = input()
root, h_orig = build(seq)
h_bin = cal_h_bin(root)
print(f'{h_orig} => {h_bin}')
```

代码运行截图 ==（至少包含有"Accepted"）==



状态: Accepted

源代码

```python
class TreeNode:
    def __init__(self):
        self.children = []
        self.first_child = None
        self.next_sib = None

def build(seq):
    root = TreeNode()
    stack = [root]
    depth = 0
    for act in seq:
        cur_node = stack[-1]
        if act == 'd':
            new_node = TreeNode()
            if not cur_node.children:
                cur_node.first_child = new_node
            else:
                cur_node.children[-1].next_sib = new_node
            cur_node.children.append(new_node)
            stack.append(new_node)
            depth = max(depth, len(stack) - 1)
        else:
            stack.pop()
    return root, depth

def cal_h_bin(node):
    if not node:
         return -1
    return max(cal_h_bin(node.first_child), cal_h_bin(node.next_sib)) +

seq = input()
root, h_orig = build(seq)
h_bin = cal_h_bin(root)
print(f'{h_orig} => {h_bin}')
```

基本信息

#: 44769983
题目: 04081
提交人: 2100015440
内存: 3672kB
时间: 28ms
语言: Python3
提交时间: 2024-04-23 23:28:46

08581: 扩展二叉树

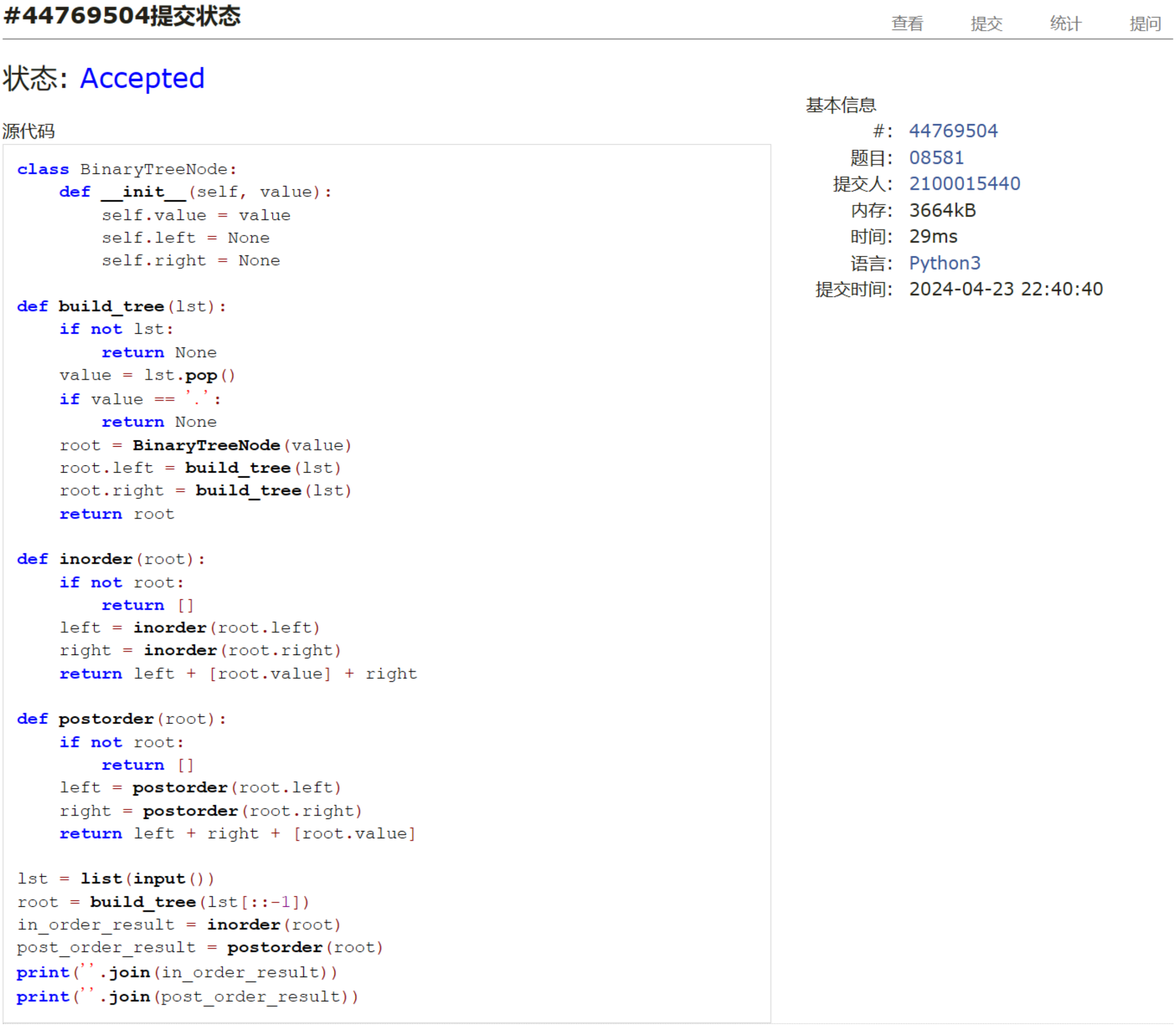
http://cs101.openjudge.cn/dsapre/08581/

思路：

代码

```
class BinaryTreeNode:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

def build_tree(lst):
    if not lst:
        return None
    value = lst.pop()
    if value == '.':
        return None
    root = BinaryTreeNode(value)
    root.left = build_tree(lst)
    root.right = build_tree(lst)
    return root

def inorder(root):
    if not root:
        return []
    left = inorder(root.left)
    right = inorder(root.right)
    return left + [root.value] + right

def postorder(root):
    if not root:
        return []
    left = postorder(root.left)
    right = postorder(root.right)
    return left + right + [root.value]

lst = list(input())
root = build_tree(lst[::-1])
in_order_result = inorder(root)
post_order_result = postorder(root)
print(''.join(in_order_result))
print(''.join(post_order_result))
```

代码运行截图 == （至少包含有"Accepted"） ==



#44769504提交状态

状态: Accepted

源代码

```
class BinaryTreeNode:
    def __init__(self, value):
        self.value = value
        self.left = None
        self.right = None

def build_tree(lst):
    if not lst:
        return None
    value = lst.pop()
    if value == '.':
        return None
    root = BinaryTreeNode(value)
    root.left = build_tree(lst)
    root.right = build_tree(lst)
    return root

def inorder(root):
    if not root:
        return []
    left = inorder(root.left)
    right = inorder(root.right)
    return left + [root.value] + right

def postorder(root):
    if not root:
        return []
    left = postorder(root.left)
    right = postorder(root.right)
    return left + right + [root.value]

lst = list(input())
root = build_tree(lst[::-1])
in_order_result = inorder(root)
post_order_result = postorder(root)
print(''.join(in_order_result))
print(''.join(post_order_result))
```

基本信息

#:	44769504
题目:	08581
提交人:	2100015440
内存:	3664kB
时间:	29ms
语言:	Python3
提交时间:	2024-04-23 22:40:40

22067: 快速堆猪

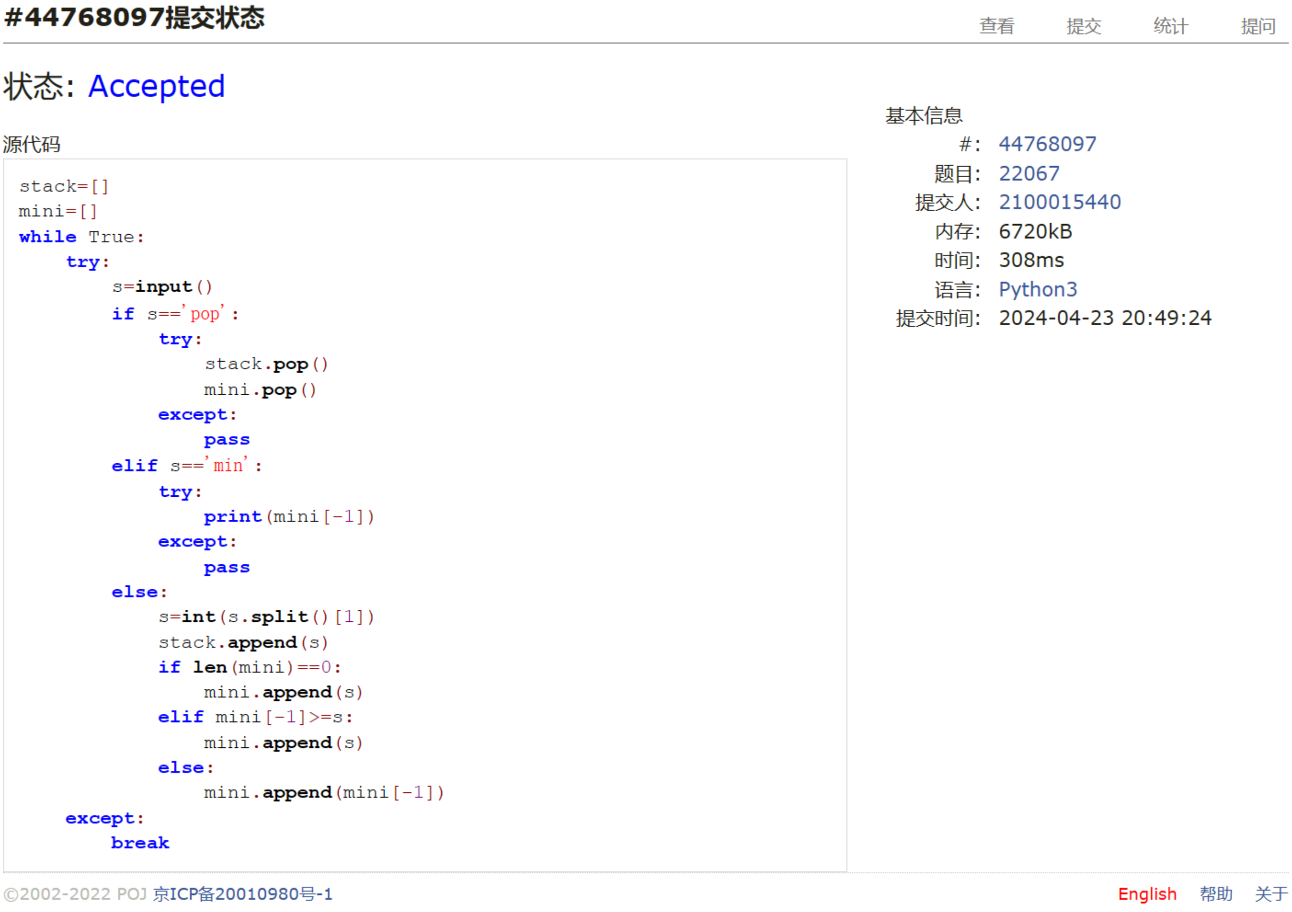
http://cs101.openjudge.cn/practice/22067/

思路：

代码

```
stack=[]
mini=[]
while True:
    try:
        s=input()
        if s=='pop':
            try:
                stack.pop()
                mini.pop()
            except:
                pass
        elif s=='min':
            try:
                print(mini[-1])
            except:
                pass
        else:
            s=int(s.split()[1])
            stack.append(s)
            if len(mini)==0:
                mini.append(s)
            elif mini[-1]>=s:
                mini.append(s)
            else:
                mini.append(mini[-1])
    except:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44768097提交状态

状态: Accepted

源代码

```
stack=[]
mini=[]
while True:
    try:
        s=input()
        if s=='pop':
            try:
                stack.pop()
                mini.pop()
            except:
                pass
        elif s=='min':
            try:
                print(mini[-1])
            except:
                pass
        else:
            s=int(s.split()[1])
            stack.append(s)
            if len(mini)==0:
                mini.append(s)
            elif mini[-1]>=s:
                mini.append(s)
            else:
                mini.append(mini[-1])
    except:
        break
```

基本信息

#:	44768097
题目:	22067
提交人:	2100015440
内存:	6720kB
时间:	308ms
语言:	Python3
提交时间:	2024-04-23 20:49:24

04123: 马走日

dfs, http://cs101.openjudge.cn/practice/04123

思路：

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

28046: 词梯

bfs, http://cs101.openjudge.cn/practice/28046/

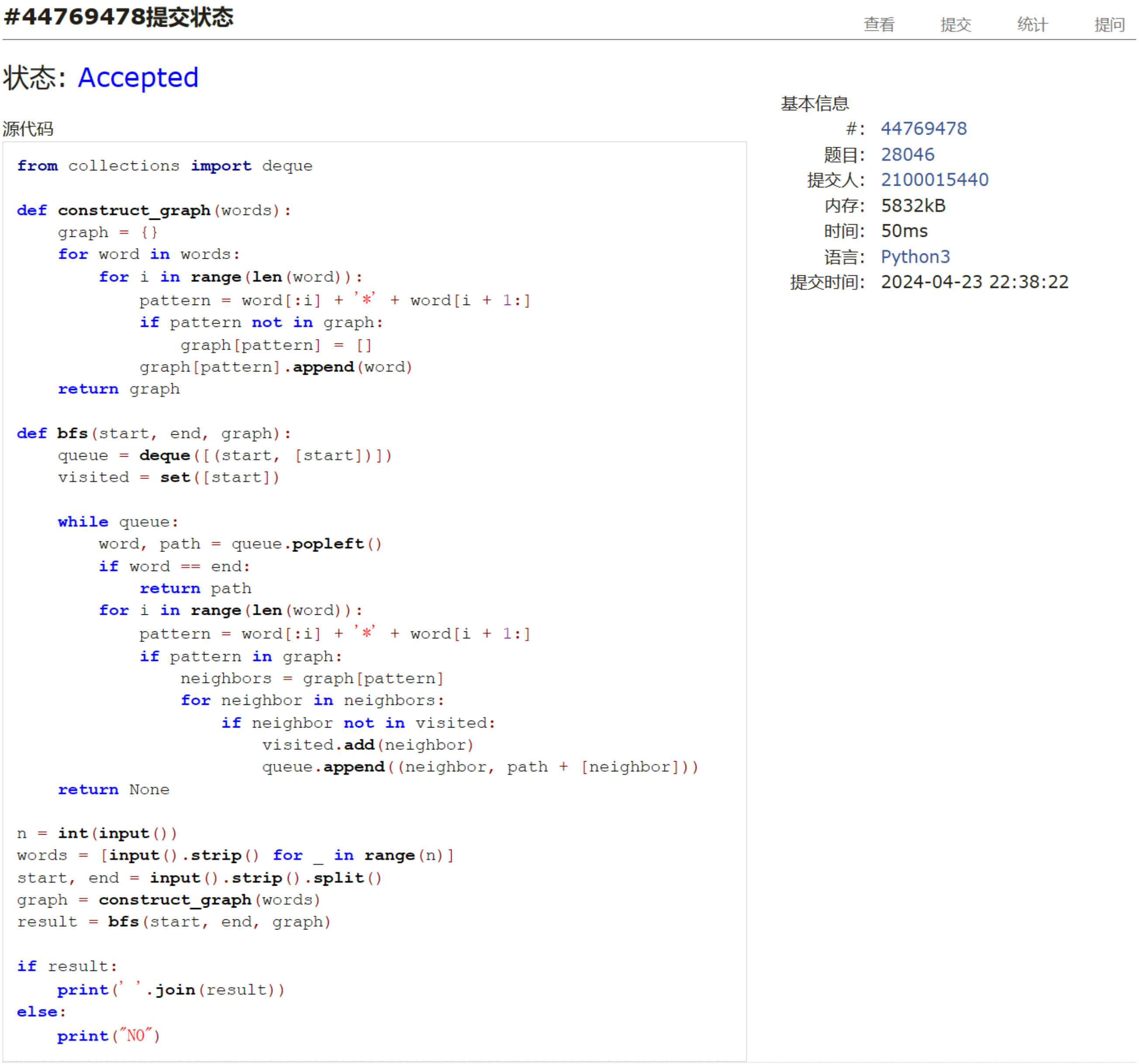
思路：

代码

```
from collections import deque

def construct_graph(words):
    graph = {}
    for word in words:
        for i in range(len(word)):
            pattern = word[:i] + '*' + word[i + 1:]
            if pattern not in graph:
                graph[pattern] = []
            graph[pattern].append(word)
    return graph

def bfs(start, end, graph):
    queue = deque([(start, [start])])
    visited = set([start])

    while queue:
        word, path = queue.popleft()
        if word == end:
            return path
        for i in range(len(word)):
            pattern = word[:i] + '*' + word[i + 1:]
            if pattern in graph:
                neighbors = graph[pattern]
                for neighbor in neighbors:
                    if neighbor not in visited:
                        visited.add(neighbor)
                        queue.append((neighbor, path + [neighbor]))
    return None

n = int(input())
words = [input().strip() for _ in range(n)]
start, end = input().strip().split()
graph = construct_graph(words)
result = bfs(start, end, graph)

if result:
    print(' '.join(result))
```

```
else:
    print("NO")
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44769478提交状态

状态: Accepted

源代码

```
from collections import deque

def construct_graph(words):
    graph = {}
    for word in words:
        for i in range(len(word)):
            pattern = word[:i] + '*' + word[i + 1:]
            if pattern not in graph:
                graph[pattern] = []
            graph[pattern].append(word)
    return graph

def bfs(start, end, graph):
    queue = deque([(start, [start])])
    visited = set([start])

    while queue:
        word, path = queue.popleft()
        if word == end:
            return path
        for i in range(len(word)):
            pattern = word[:i] + '*' + word[i + 1:]
            if pattern in graph:
                neighbors = graph[pattern]
                for neighbor in neighbors:
                    if neighbor not in visited:
                        visited.add(neighbor)
                        queue.append((neighbor, path + [neighbor]))
    return None

n = int(input())
words = [input().strip() for _ in range(n)]
start, end = input().strip().split()
graph = construct_graph(words)
result = bfs(start, end, graph)

if result:
    print(' '.join(result))
else:
    print("NO")
```

基本信息

#:	44769478
题目:	28046
提交人:	2100015440
内存:	5832kB
时间:	50ms
语言:	Python3
提交时间:	2024-04-23 22:38:22

28050: 骑士周游

dfs, http://cs101.openjudge.cn/practice/28050/

思路：

代码

```
#
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

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。==

- 完成了1、2、3、5题，感觉树和图的题目还需要练习，很多题目会想到之前作业相关的题，但是自己写还是有难度。
- 图算法方面，bfs感觉还好，dfs还没完全搞明白。