

Assignment #9: 图论：遍历，及 树算

Updated 1739 GMT+8 Apr 14, 2024

2024 spring, Complied by 陈奕好 工学院

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：macOS Sonoma 14.4 (23E214)

Python编程环境：PyCharm 2023.3.1 (Professional Edition)

1. 题目

04081: 树的转换

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

思路：flag存储的是当前在当下节点之前层数的最大深度，

代码

```
1 def calculate_height(s):
2     flag = [0]*10002 # 标记数组
3     level, pre, post = 0, 0, 0 # 当前层数、最大层数、最大深度
4     for char in s: # 遍历字符串
5         if char == "u": # 如果字符为'u'
6             level -= 1 # 层数减1
7             flag[level] += 1 # 标记数组对应位置加1
8         else: # 如果字符为'd'
9             level += 1 # 层数加1
```

```

10         flag[level] = flag[level-1] + 1 # 标记数组对应位置等于上一层的标记数加1
11         pre = max(level, pre) # 更新最大层数
12         post = max(post, flag[level]) # 更新最大深度
13     return pre, post
14
15
16 s = input() # 读取输入
17 pre, post = calculate_height(s)
18 print(f'{pre} => {post}')
19
20

```

代码运行截图 (至少包含有"Accepted")

#44682631提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

def calculate_height(s):
    flag = [0]*10002 # 标记数组
    level, pre, post = 0, 0, 0 # 当前层数、最大层数、最大深度
    for char in s: # 遍历字符串
        if char == "u": # 如果字符为'u'
            level -= 1 # 层数减1
            flag[level] += 1 # 标记数组对应位置加1
        else: # 如果字符为'd'
            level += 1 # 层数加1
            flag[level] = flag[level-1] + 1 # 标记数组对应位置等于上一层的标记数加1
            pre = max(level, pre) # 更新最大层数
            post = max(post, flag[level]) # 更新最大深度
    return pre, post

s = input() # 读取输入
pre, post = calculate_height(s)
print(f'{pre} => {post}')

```

基本信息

#: 44682631
 题目: 04081
 提交人: 23n2300011030(陈奕好)
 内存: 3684kB
 时间: 26ms
 语言: Python3
 提交时间: 2024-04-17 13:50:22

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

08581: 扩展二叉树

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

思路：一颗满二叉树

代码

```

1 class TreeNode:
2     def __init__(self, x):
3         self.v = x
4         self.l = None

```

```

5         self.r = None
6
7
8     index = 0
9
10
11     def tree_build(pre_order):
12         global index
13         if index >= len(pre_order) or pre_order[index] == ".":
14             index += 1
15             return None
16
17         root = TreeNode(pre_order[index])
18         index += 1
19         root.l = tree_build(pre_order)
20         root.r = tree_build(pre_order)
21         return root
22
23
24     def midOrder(root):
25         if root is None:
26             return ''
27         return midOrder(root.l) + root.v + midOrder(root.r)
28
29
30     def postOrder(root):
31         if root is None:
32             return ''
33         return postOrder(root.l) + postOrder(root.r) + root.v
34
35
36     tree = input()
37     root = tree_build(tree)
38     print(midOrder(root))
39     print(postOrder(root))
40

```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
class TreeNode:
    def __init__(self, x):
        self.v = x
        self.l = None
        self.r = None

index = 0

def tree_build(pre_order):
    global index
    if index >= len(pre_order) or pre_order[index] == ".":
        index += 1
        return None

    root = TreeNode(pre_order[index])
    index += 1
    root.l = tree_build(pre_order)
    root.r = tree_build(pre_order)
    return root

def midOrder(root):
    if root is None:
```

基本信息

#: 44690494
题目: 08581
提交人: 23n2300011030(陈奕好)
内存: 3632kB
时间: 32ms
语言: Python3
提交时间: 2024-04-17 21:21:56

22067: 快速堆猪

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

思路: 三次实现

代码

```
1  """
2  class pig_stack():
3      def __init__(self):
4          self.stack = []
5
6      def push(self, new):
7          self.stack.append(new)
8
9      def pop(self):
10         if self.stack:
11             self.stack.pop()
12
13     def min(self):
14         if len(self.stack):
15             return min(self.stack)
16         return False
17
```

```

18
19 stack1 = pig_stack()
20 while True:
21     try:
22         opt = list(map(str, input().split()))
23         if opt[0] == 'push':
24             stack1.push(int(opt[1]))
25         elif opt[0] == 'pop':
26             stack1.pop()
27         elif opt[0] == 'min':
28             if stack1.min():
29                 print(stack1.min())
30     except EOFError:
31         break
32 """
33
34
35 """
36 stack = []
37 minValue = []
38 while True:
39     try:
40         opt = list(map(str, input().split()))
41         if opt[0] == 'push':
42             x = int(opt[1])
43             if minValue:
44                 if minValue[-1] >= x:
45                     minValue.append(x)
46             else:
47                 minValue.append(x)
48             stack.append(x)
49         elif opt[0] == 'pop':
50             if stack:
51                 if stack[-1] == minValue[-1]:
52                     minValue.pop()
53                 stack.pop()
54         elif opt[0] == 'min':
55             if stack:
56                 print(minValue[-1])
57     except EOFError:
58         break
59 """
60
61
62 stack = []
63 m_list = []
64 while True:
65     try:
66         opt = input().split()
67         if opt[0] == "pop":
68             if stack:
69                 out_ = stack.pop()

```

```
70         if m_list[-1] == out_:
71             m_list.pop()
72             # print(out)
73
74     elif opt[0] == "min":
75         if stack:
76             print(m_list[-1])
77
78     else:
79         in_ = int(opt[1])
80         stack.append(in_)
81         if m_list:
82             if in_ <= m_list[-1]:
83                 m_list.append(in_)
84             else:
85                 m_list.append(in_)
86
87 except EOFError:
88     break
89
90
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

#44690641提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
"""
class pig_stack():
    def __init__(self):
        self.stack = []

    def push(self, new):
        self.stack.append(new)

    def pop(self):
        if self.stack:
            self.stack.pop()

    def min(self):
        if len(self.stack):
            return min(self.stack)
        return False
```

基本信息

#: 44690641
题目: 22067
提交人: 23n2300011030(陈奕好)
内存: 6792kB
时间: 304ms
语言: Python3
提交时间: 2024-04-17 21:38:58

04123: 马走日

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

思路: dfs

代码

```
1 T = int(input())
2 dir = [(2, 1), (1, 2), (-1, 2), (-2, 1), (-2, -1), (-1, -2), (1, -2), (2, -1)]
3
4
5 def valid(x, y, n, m):
6     return 0 <= x < n and 0 <= y < m
7
8
9 def dfs(x, y, n, m, visited, count):
10     if count == n * m: # 看是否自我湮灭
11         return 1
12     total = 0
13     visited[x][y] = True
14     for dx, dy in dir:
15         nx, ny = x + dx, y + dy # 举棋子
16         if valid(nx, ny, n, m) and not visited[nx][ny]:
17             total += dfs(nx, ny, n, m, visited, count + 1) # 放棋子
18     visited[x][y] = False # 回溯
19     return total
20
21
22 for _ in range(T):
23     n, m, x, y = map(int, input().split())
24     visited = [[False]*m for _ in range(n)]
25     print(dfs(x, y, n, m, visited, 1))
26
27
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
T = int(input())
dir = [(2, 1), (1, 2), (-1, 2), (-2, 1), (-2, -1), (-1, -2), (1, -2), (2, -1)]

def valid(x, y, n, m):
    return 0 <= x < n and 0 <= y < m

def dfs(x, y, n, m, visited, count):
    if count == n * m: # 看是否自我湮灭
        return 1
    total = 0
    visited[x][y] = True
    for dx, dy in dir:
        nx, ny = x + dx, y + dy # 举棋子
        if valid(nx, ny, n, m) and not visited[nx][ny]:
            total += dfs(nx, ny, n, m, visited, count + 1) # 放棋子
    visited[x][y] = False # 回溯
    return total

for _ in range(T):
    n, m, x, y = map(int, input().split())
    visited = [[False]*m for _ in range(n)]
    print(dfs(x, y, n, m, visited, 1))
```

基本信息

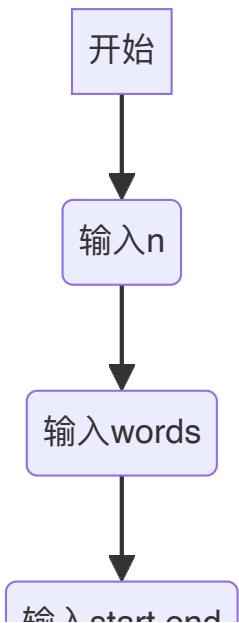
#: 44691587
题目: 04123
提交人: 23n2300011030(陈奕好)
内存: 3688kB
时间: 3710ms
语言: Python3
提交时间: 2024-04-17 23:08:10

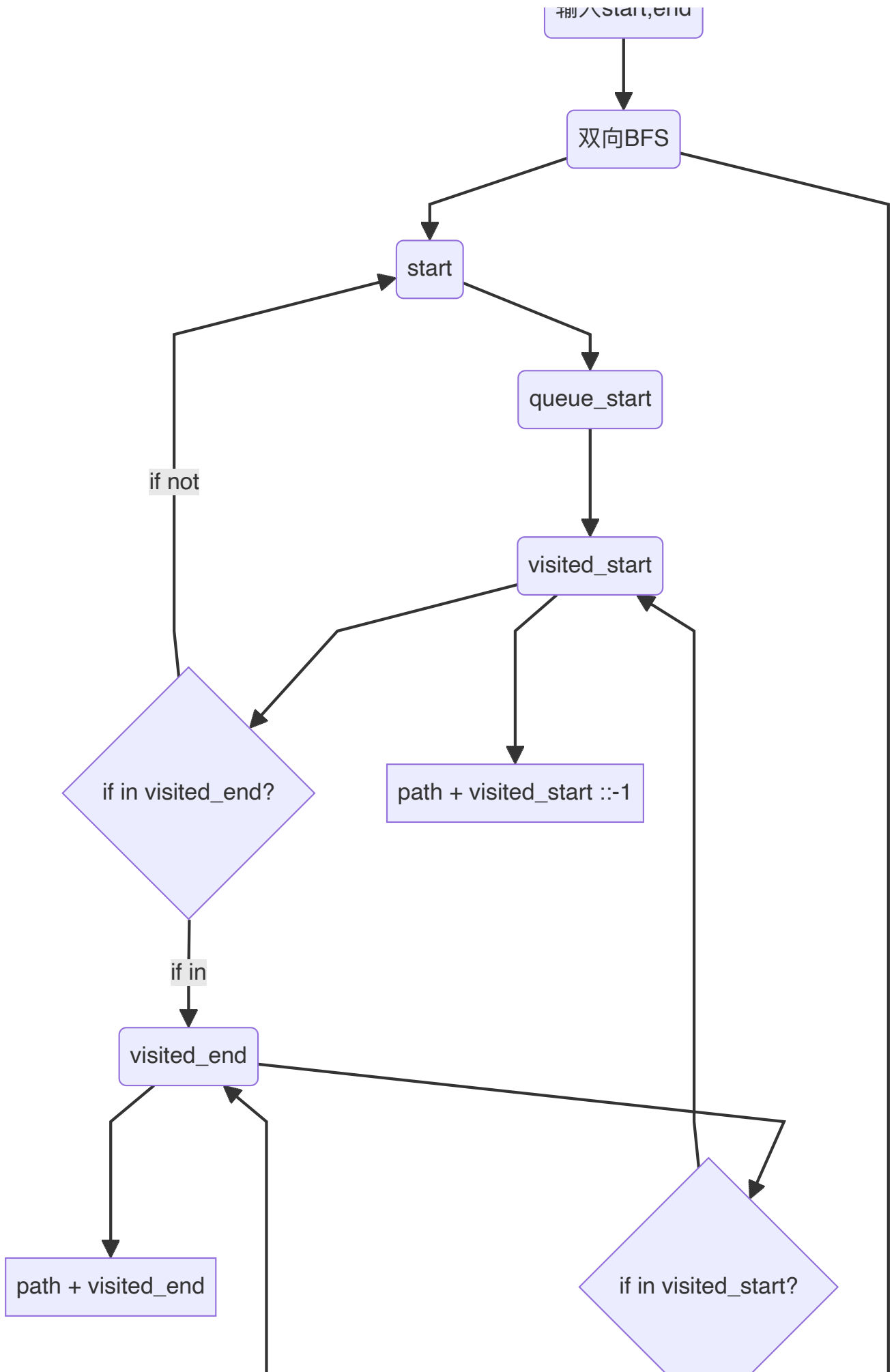
28046: 词梯

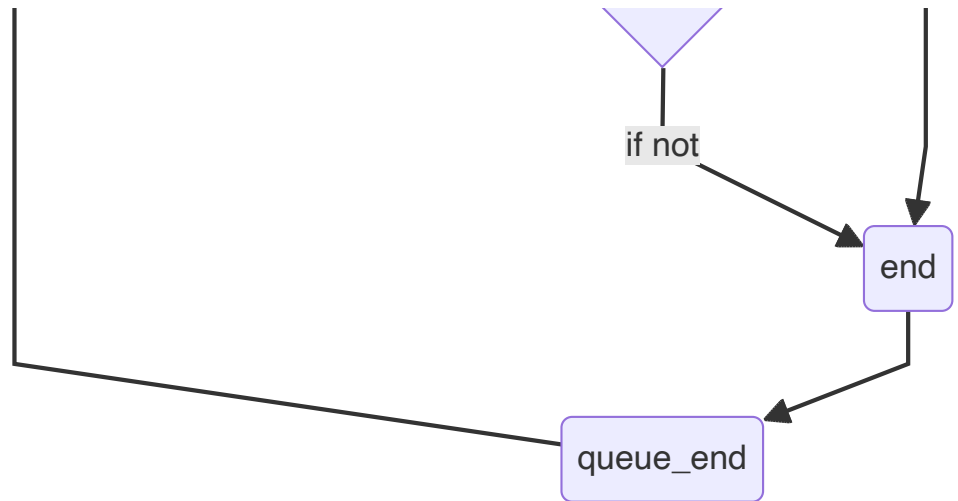
bfs, <http://cs101.openjudge.cn/practice/28046/>

思路：这个代码很直观，就是数据存储时犯了难，在copilot帮助下学习了双向BFS，以及对队列自我湮灭有了更深刻的理解。

流程图







代码

```

1  from collections import defaultdict, deque
2
3
4  def visit_vertex(queue, visited, other_visited, graph):
5      word, path = queue.popleft()
6      for i in range(len(word)):
7          pattern = word[:i] + '_' + word[i + 1:]
8          for next_word in graph[pattern]:
9              if next_word in other_visited:
10                 return path + other_visited[next_word][::-1]
11             if next_word not in visited:
12                 visited[next_word] = path + [next_word]
13                 queue.append((next_word, path + [next_word]))
14
15
16  def word_ladder(words, start, end):
17      graph = defaultdict(list)
18      for word in words:
19          for i in range(len(word)):
20              pattern = word[:i] + '_' + word[i + 1:]
21              graph[pattern].append(word)
22
23      queue_start = deque([(start, [start])])
24      queue_end = deque([(end, [end])])
25      visited_start = {start: [start]}
26      visited_end = {end: [end]}
27
28      while queue_start and queue_end:
29          result = visit_vertex(queue_start, visited_start, visited_end, graph)
30          if result:
31              return ' '.join(result)
32          result = visit_vertex(queue_end, visited_end, visited_start, graph)
33          if result:

```

```

34         return ' '.join(result[::-1])
35
36     return 'NO'
37
38
39 n = int(input())
40 words = [input() for i in range(n)]
41 start, end = input().split()
42 print(word_ladder(words, start, end))
43
44

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

#44692645提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

基本信息

#: 44692645
 题目: 28046
 提交人: 23n2300011030(陈奕好)
 内存: 5528kB
 时间: 38ms
 语言: Python3
 提交时间: 2024-04-18 09:42:43

```

from collections import defaultdict, deque

def visit_vertex(queue, visited, other_visited, graph):
    word, path = queue.popleft()
    for i in range(len(word)):
        pattern = word[:i] + '_' + word[i + 1:]
        for next_word in graph[pattern]:
            if next_word in other_visited:
                return path + other_visited[next_word][::-1]
            if next_word not in visited:
                visited[next_word] = path + [next_word]
                queue.append((next_word, path + [next_word]))

def word_ladder(words, start, end):
    graph = defaultdict(list)
    for word in words:
        for i in range(len(word)):
            pattern = word[:i] + '_' + word[i + 1:]
            graph[pattern].append(word)

    queue_start = deque([(start, [start])])
    queue_end = deque([(end, [end])])

```

28050: 骑士周游

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

思路: get_degree 这个小剪枝很好!

第二版加了个lru_cache, 感觉大一点的数据估计出生点在中间 (? , 还是说受边界的影响较小, 提升并不明显。

代码

```

1 from functools import lru_cache

```

```

2
3 # initializing
4 size = int(input())
5 matrix = [[False]*size for i in range(size)]
6 x, y = map(int, input().split())
7 dir = [(2, 1), (1, 2), (-1, 2), (-2, 1), (-2, -1), (-1, -2), (1, -2), (2, -1)]
8
9
10 def valid(x, y):
11     return 0 <= x < size and 0 <= y < size and not matrix[x][y]
12
13
14 def get_degree(x, y):
15     count = 0
16     for dx, dy in dir:
17         nx, ny = x + dx, y + dy
18         if valid(nx, ny):
19             count += 1
20     return count
21
22
23 @lru_cache(maxsize = 1<<30)
24 def dfs(x, y, count):
25     if count == size**2:
26         return True
27
28     matrix[x][y] = True
29
30     next_moves = [(dx, dy) for dx, dy in dir if valid(x + dx, y + dy)]
31     next_moves.sort(key=lambda move: get_degree(x + move[0], y + move[1]))
32
33     for dx, dy in next_moves:
34         if dfs(x + dx, y + dy, count + 1):
35             return True
36
37     matrix[x][y] = False
38     return False
39
40 if dfs(x, y, 1):
41     print("success")
42 else:
43     print("fail")
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 functools import lru_cache

# initializing
size = int(input())
matrix = [[False]*size for i in range(size)]
x, y = map(int, input().split())
dir = [(2, 1), (1, 2), (-1, 2), (-2, 1), (-2, -1), (-1, -2), (1, -2), (2, -1)]

def valid(x, y):
    return 0 <= x < size and 0 <= y < size and not matrix[x][y]

def get_degree(x, y):
    count = 0
    for dx, dy in dir:
        nx, ny = x + dx, y + dy
        if valid(nx, ny):
            count += 1
    return count

@lru_cache(maxsize = 1<<30)
def dfs(x, y, count):
```

基本信息

#: 44694646
题目: 28050
提交人: 23n2300011030(陈奕好)
内存: 4100kB
时间: 29ms
语言: Python3
提交时间: 2024-04-18 14:28:37

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。

在补每日选坐了，再不补就补不完了。