

Assignment #7: bfs、

Updated 0851 GMT+8 Oct 21, 2025

2025 fall, Compiled by 同学的姓名、院系

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本人头像，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

M23555: 节省存储的矩阵乘法

implementation, matrices, <http://cs101.openjudge.cn/practice/23555>

要求用节省内存的方式实现，不能还原矩阵的方式实现。

思路：字典

代码：

```
n,m1,m2=map(int,input().split())
d1,d2={},{}
for _ in range(m1):
    [x,y,z]=list(map(int,input().split()))
    if x not in d1:
        d1[x]=[ ]
    d1[x].append([y,z])
for _ in range(m2):
    [x,y,z]=list(map(int,input().split()))
    if y not in d2:
```

```

        d2[y]=[ ]
    d2[y].append([x,z])
m=sorted(d2.keys())
for i in d1.keys():
    for j in m:
        ans=0
        k=False
        for a in d1[i]:
            for b in d2[j]:
                if a[0]==b[0]:
                    ans+=a[1]*b[1]
                    k=True
        if k:
            print(i,j,ans)

```

代码运行截图 (至少包含有"Accepted")

#50564094提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```

n,m1,m2=map(int,input().split())
d1,d2={},{}
for _ in range(m1):
    [x,y,z]=list(map(int,input().split()))
    if x not in d1:
        d1[x]=[ ]
    d1[x].append([y,z])
for _ in range(m2):
    [x,y,z]=list(map(int,input().split()))
    if y not in d2:
        d2[y]=[ ]
    d2[y].append([x,z])
m=sorted(d2.keys())
for i in d1.keys():
    for j in m:
        ans=0
        k=False
        for a in d1[i]:
            for b in d2[j]:
                if a[0]==b[0]:
                    ans+=a[1]*b[1]
                    k=True
        if k:
            print(i,j,ans)

```

基本信息

#: 50564094
 题目: 23555
 提交人: 24n2400011474
 内存: 3780kB
 时间: 37ms
 语言: Python3
 提交时间: 2025-10-26 11:30:14

M102.二叉树的层序遍历

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

思路：一层一层往下走，更新s

代码:

```
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution:
    def levelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
        l=[]
        s=[root]
        if not root:
            return []
        while s:
            t=[]
            k=[]
            for x in s:
                t.append(x.val)
                if x.left:
                    k.append(x.left)
                if x.right:
                    k.append(x.right)
            l.append(t)
            s=k
        return l
```

代码运行截图 (至少包含有"Accepted")

题目描述 | 通过 x | 题解 | 提交记录

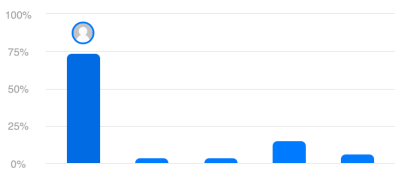
全部提交记录

通过 35 / 35 个通过的测试用例
Ama2ing Tereshkov... 提交于 2025.10.26 17:13

执行用时分布
0 ms | 击败 100.00%

复杂度分析

消耗内存分布
18.06 MB | 击败 87.25%



代码

Python3 智能模式

```
1 # Definition for a binary tree node.
2 # class TreeNode:
3 #     def __init__(self, val=0, left=None, right=None):
4 #         self.val = val
5 #         self.left = left
6 #         self.right = right
7 class Solution:
8     def levelOrder(self, root: Optional[TreeNode]) -> List[List[int]]:
9         l=[]
10        s=[root]
11        if not root:
12            return []
13        while s:
14            t=[]
15            k=[]
16            for x in s:
17                t.append(x.val)
18                if x.left:
19                    k.append(x.left)
20                if x.right:
21                    k.append(x.right)
22            l.append(t)
23            s=k
24        return l
```

M131.分割回文串

dp, backtracking, <https://leetcode.cn/problems/palindrome-partitioning/>

思路：注意i要从后往前遍历，另外这种递归方法好像比找到第一个回文数然后solve后面的高不少

代码:

```
class Solution:
    def partition(self, s: str) -> List[List[str]]:
        n=len(s)
        r=[[0]*n for _ in range(n)]
        for i in range(n)[::-1]:
            for j in range(i,n):
                if s[i]==s[j] and (j-i<2 or r[i+1][j-1]):
                    r[i][j]=1

        ans=[]
        def solve(i,l):
            if i==n:
                ans.append(l)
                return
            for j in range(i,n):
                if r[i][j]:
                    solve(j+1,l+[s[i:j+1]])

        solve(0,[])
        return ans
```

代码运行截图 (至少包含有"Accepted")



题目描述 | 通过 | 题解 | 提交记录

全部提交记录

通过 32 / 32 个通过的测试用例

AmaZing Tereshkov... 提交于 2025.10.26 17:16

写题解

执行用时分布

39 ms | 击败 97.42%

复杂度分析

消耗内存分布

33.79 MB | 击败 66.99%

20%

代码

```
1 class Solution:
2     def partition(self, s: str) -> List[List[str]]:
3         n=len(s)
4         r=[[0]*n for _ in range(n)]
5         for i in range(n)[::-1]:
6             for j in range(i,n):
7                 if s[i]==s[j] and (j-i<2 or r[i+1][j-1]):
8                     r[i][j]=1
9
10        ans=[]
11        def solve(i,l):
12            if i==n:
13                ans.append(l)
14                return
15            for j in range(i,n):
16                if r[i][j]:
17                    solve(j+1,l+[s[i:j+1]])
18        solve(0,[])
19        return ans
```

M200.岛屿数量

dfs, bfs, <https://leetcode.cn/problems/number-of-islands/>

思路: 找到岛屿就把岛屿变成海

代码

```
class Solution:
    def numIslands(self, grid: List[List[str]]) -> int:
```

(至少包含有"Accepted")

1123.最深叶节点的最近公共祖先

思路：开始不会问ai才会的，但是感觉自己应该会

```
# Definition for a binary tree node.
# class TreeNode:
#     def __init__(self, val=0, left=None, right=None):
#         self.val = val
#         self.left = left
#         self.right = right
class Solution:
    def lcaDeepestLeaves(self, root: Optional[TreeNode]) -> Optional[TreeNode]:
```

```
def solve(s):
    if s.left and s.right:
        a,b=solve(s.left),solve(s.right)
        if a[0]==b[0]:
            return [a[0]+1,s]
        elif a[0]>b[0]:
            return [a[0]+1,a[1]]
        else:
            return [b[0]+1,b[1]]
    elif s.left:
        a=solve(s.left)
        return [a[0]+1,a[1]]
    elif s.right:
        b=solve(s.right)
        return [b[0]+1,b[1]]
    else:
        return [0,s]
return solve(root)[1]
```

(至少包含有"Accepted")



M79.单词搜索

回溯, <https://leetcode.cn/problems/word-search/>

思路: 非常正常的dfs

代码:

```
class Solution:
    def exist(self, board: List[List[str]], word: str) -> bool:
        n,m,h=len(board),len(board[0]),len(word)
```

```

r=[[-1,0],[1,0],[0,-1],[0,1]]
def solve(i,j,k,l):
    if k==h:
        return True
    for [dx,dy] in r:
        if 0<=i+dx<n and 0<=j+dy<m and l[i+dx][j+dy] and board[i+dx]
[j+dy]==word[k]:
            l[i+dx][j+dy]=0
            if solve(i+dx,j+dy,k+1,l):
                return True
            l[i+dx][j+dy]=1
    for i in range(n):
        for j in range(m):
            if board[i][j]==word[0]:
                l=[[1]*m for _ in range(n)]
                l[i][j]=0
                if solve(i,j,1,l):
                    return True
    return False

```

代码运行截图 (至少包含有"Accepted")



2. 学习总结和个人收获

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2025fall每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。

上周事情很多，欠下了17道每日选做，幸运的是上周的事最后结果不错，每日选做以及树的相关算法会慢慢补回来的