*# Assignment #9: dfs, bfs, & dp*

Updated 2107 GMT+8 Nov 19, 2024

2024 fall, Complied by <mark>付耀贤，信息管理系</mark>

*## 1. 题目*

*### 18160: 最大连通域面积*

dfs similar, http://cs101.openjudge.cn/practice/18160

思路：

没学过DFS怎么写。向AI学习了写法，觉得思路甚至还没有某些DP难？

代码：

**def** **dfs**(matrix, visited, i, j, n, m):

directions = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1, 1)]

area = 1

visited[i][j] = True

**for** dx, dy **in** directions:

ni, nj = i + dx, j + dy

**if** 0 <= ni < n **and** 0 <= nj < m **and** **not** visited[ni][nj] **and** matrix[ni][nj] == 'W':

area += **dfs**(matrix, visited, ni, nj, n, m)

**return** area

**def** **max\_connected\_component**(matrix, n, m):

visited = [[False] \* m **for** \_ **in** **range**(n)]

max\_area = 0

**for** i **in** **range**(n):

**for** j **in** **range**(m):

**if** matrix[i][j] == 'W' **and** **not** visited[i][j]:

area = **dfs**(matrix, visited, i, j, n, m)

max\_area = **max**(max\_area, area)

**return** max\_area

T = **int**(**input**())

**for** \_ **in** **range**(T):

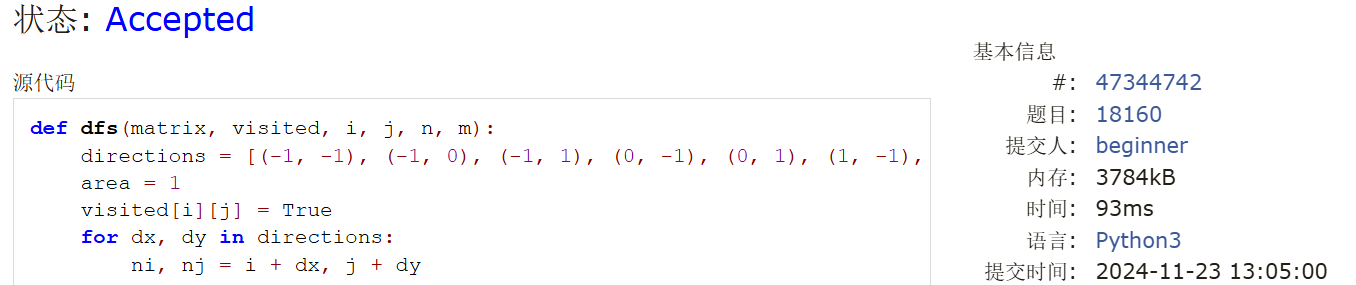
N, M = **map**(int, **input**().**split**())

matrix = [**input**().**strip**() **for** \_ **in** **range**(N)]

result = **max\_connected\_component**(matrix, N, M)

**print**(result)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### 19930: 寻宝*

bfs, http://cs101.openjudge.cn/practice/19930

思路：

尝试用第一题DFS的思路套这一题，失败了，原来求最短路径要用BFS。向AI学！

代码：

**from** collections **import** deque

**def** **bfs**(matrix, m, n):

directions = [(-1, 0), (0, -1), (1, 0), (0, 1)]

queue = **deque**([(0, 0, 0)])

visited = [[False] \* n **for** \_ **in** **range**(m)]

visited[0][0] = True

**while** queue:

i, j, steps = queue.**popleft**()

**if** matrix[i][j] == 1:

**return** steps

**for** dx, dy **in** directions:

ni, nj = i + dx, j + dy

**if** 0 <= ni < m **and** 0 <= nj < n **and** **not** visited[ni][nj] **and** matrix[ni][nj] != 2:

visited[ni][nj] = True

queue.**append**((ni, nj, steps + 1))

**return** "NO"

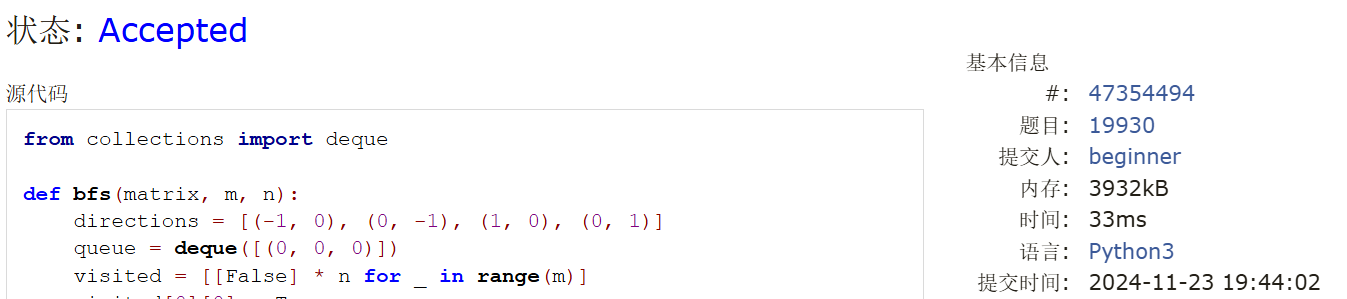
m, n = **map**(int, **input**().**split**())

matrix = [**list**(**map**(int, **input**().**split**())) **for** \_ **in** **range**(m)]

result = **bfs**(matrix, m, n)

**print**(result)

代码运行截图 ==（至少包含有"Accepted"）==



*### 04123: 马走日*

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

思路：

虽然我不能很理解具体的运行步骤，但我主打一个模仿，把第一题的思路改一改就可以用在这一题目上。

代码：

**def** **dfs**(x, y, n, m, visited, steps):

**if** steps == n \* m:

**return** 1

visited[x][y] = True

c = 0

directions = [

(2, 1), (2, -1), (-2, 1), (-2, -1),

(1, 2), (1, -2), (-1, 2), (-1, -2)

]

**for** dx, dy **in** directions:

x1, y1 = x + dx, y + dy

**if** 0 <= x1 < n **and** 0 <= y1 < m **and** **not** visited[x1][y1]:

c += **dfs**(x1, y1, n, m, visited, steps + 1)

visited[x][y] = False

**return** c

**def** **count**(n, m, x, y):

visited = [[False] \* m **for** \_ **in** **range**(n)]

**return** **dfs**(x, y, n, m, visited, 1)

T = **int**(**input**())

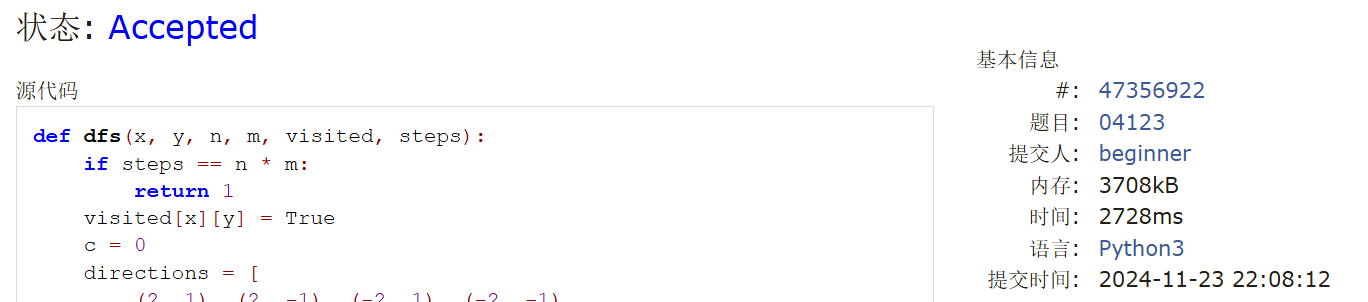
**for** \_ **in** **range**(T):

n, m, x, y = **map**(int, **input**().**split**())

result = **count**(n, m, x, y)

**print**(result)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### sy316: 矩阵最大权值路径*

dfs, https://sunnywhy.com/sfbj/8/1/316

思路：

虽然我知道思路都是一脉相承的，但只要一变复杂，我的思路就木住了…

代码：

def dfs(matrix, x, y, n, m, visited, current\_sum, max\_sum, best\_path, current\_path):  
 if x == n - 1 and y == m - 1:  
 if current\_sum > max\_sum[0]:  
 max\_sum[0] = current\_sum  
 best\_path[:] = current\_path[:]  
 return  
 directions = [  
 (1, 0),  
 (0, 1),  
 (-1, 0),  
 (0, -1)  
 ]  
 for dx, dy in directions:  
 x1, y1 = x + dx, y + dy  
 if 0 <= x1 < n and 0 <= y1 < m and not visited[x1][y1]:  
 visited[x1][y1] = True  
 current\_path.append((x1 + 1, y1 + 1))  
 dfs(matrix, x1, y1, n, m, visited, current\_sum + matrix[x1][y1], max\_sum, best\_path, current\_path)  
 current\_path.pop()  
 visited[x1][y1] = False  
  
def find\_max\_path(matrix, n, m):  
 visited = [[False] \* m for \_ in range(n)]  
 max\_sum = [-float('inf')]  
 best\_path = []  
 visited[0][0] = True  
 dfs(matrix, 0, 0, n, m, visited, matrix[0][0], max\_sum, best\_path, [(1, 1)]) # 从(0, 0)开始  
 return best\_path  
  
n, m = map(int, input().split())  
matrix = [list(map(int, input().split())) for \_ in range(n)]  
result\_path = find\_max\_path(matrix, n, m)  
for x, y in result\_path:  
 print(x, y)

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### LeetCode62.不同路径*

dp, https://leetcode.cn/problems/unique-paths/

思路：

高考组合数数学题。

用DP也不难：一个位置能从上方和左侧到达：dp[i][j] = dp[i - 1][j] + dp[i][j - 1]

代码：

class Solution:

    def uniquePaths(self, m: int, n: int) -> int:

        a = min(m-1, n-1)

        if not a:

            return 1

        else:

            c = 1

            for i in range(a):

                c \*= (m+n-2-i)

            for i in range(a):

                c = c//(i+1)

            return c

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*### sy358: 受到祝福的平方*

dfs, dp, https://sunnywhy.com/sfbj/8/3/539

思路：

做不出来，但看题解能看懂，win！

代码：

def is\_perfect\_square(num):  
 if num<0:  
 return False  
 else:  
 root = int(num\*\*0.5)  
 return root \* root == num  
  
def is\_blessed\_id(A):  
 squares = set()  
 i = 1  
 while i \* i <= 10 \*\* 9:  
 squares.add(i \* i)  
 i += 1  
 digits = list(map(int, str(A)))  
 def dfs(idx):  
 if idx == len(digits):  
 return True  
 num = 0  
 for i in range(idx, len(digits)):  
 num = num \* 10 + digits[i]  
 if num in squares:  
 if dfs(i + 1):  
 return True  
 return False  
 return "Yes" if dfs(0) else "No"  
  
A = int(input())  
print(is\_blessed\_id(A))

代码运行截图 <mark>（至少包含有"Accepted"）</mark>



*## 2. 学习总结和收获*

能在参考经典解法的基础上模仿着写出来一些拐弯比较少的题。但稍微有点复杂时，就有一种不知道怎么把题目内生性的要求嵌入到我已经掌握的思路的感觉…还是理解不动、理解不够。