Assignment #9: dfs, bfs, & dp

Updated 2107 GMT+8 Nov 19, 2024

2024 fall, Complied by 佟永鑫 元培学院

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。

1. 题目

18160: 最大连通域面积

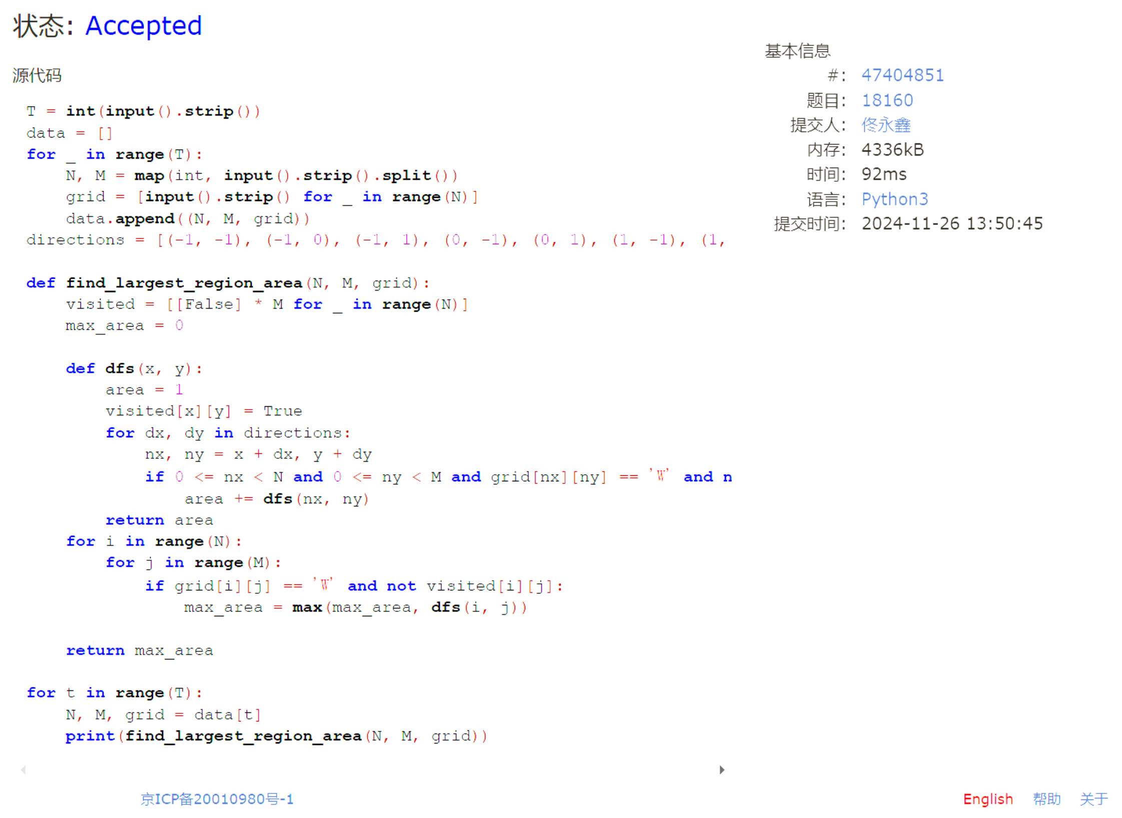
dfs similar, http://cs101.openjudge.cn/practice/18160

思路：

代码：

```
T = int(input().strip())
data = []
for _ in range(T):
    N, M = map(int, input().strip().split())
    grid = [input().strip() for _ in range(N)]
    data.append((N, M, grid))
directions = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1, 0), (1,
1)]

def find_largest_region_area(N, M, grid):
    visited = [[False] * M for _ in range(N)]
    max_area = 0

    def dfs(x, y):
        area = 1
        visited[x][y] = True
        for dx, dy in directions:
            nx, ny = x + dx, y + dy
            if 0 <= nx < N and 0 <= ny < M and grid[nx][ny] == 'W' and not
visited[nx][ny]:
                area += dfs(nx, ny)
        return area
    for i in range(N):
        for j in range(M):
                if grid[i][j] == 'W' and not visited[i][j]:
                    max_area = max(max_area, dfs(i, j))

    return max_area

for t in range(T):
    N, M, grid = data[t]
    print(find_largest_region_area(N, M, grid))
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```
T = int(input().strip())
data = []
for _ in range(T):
    N, M = map(int, input().strip().split())
    grid = [input().strip() for _ in range(N)]
    data.append((N, M, grid))
directions = [(-1, -1), (-1, 0), (-1, 1), (0, -1), (0, 1), (1, -1), (1,

def find_largest_region_area(N, M, grid):
    visited = [[False] * M for _ in range(N)]
    max_area = 0

    def dfs(x, y):
        area = 1
        visited[x][y] = True
        for dx, dy in directions:
            nx, ny = x + dx, y + dy
            if 0 <= nx < N and 0 <= ny < M and grid[nx][ny] == 'W' and n
                area += dfs(nx, ny)
        return area
    for i in range(N):
        for j in range(M):
            if grid[i][j] == 'W' and not visited[i][j]:
                max_area = max(max_area, dfs(i, j))

    return max_area

for t in range(T):
    N, M, grid = data[t]
    print(find_largest_region_area(N, M, grid))
```

基本信息

#:	47404851
题目:	18160
提交人:	佟永鑫
内存:	4336kB
时间:	92ms
语言:	Python3
提交时间:	2024-11-26 13:50:45

19930: 寻宝

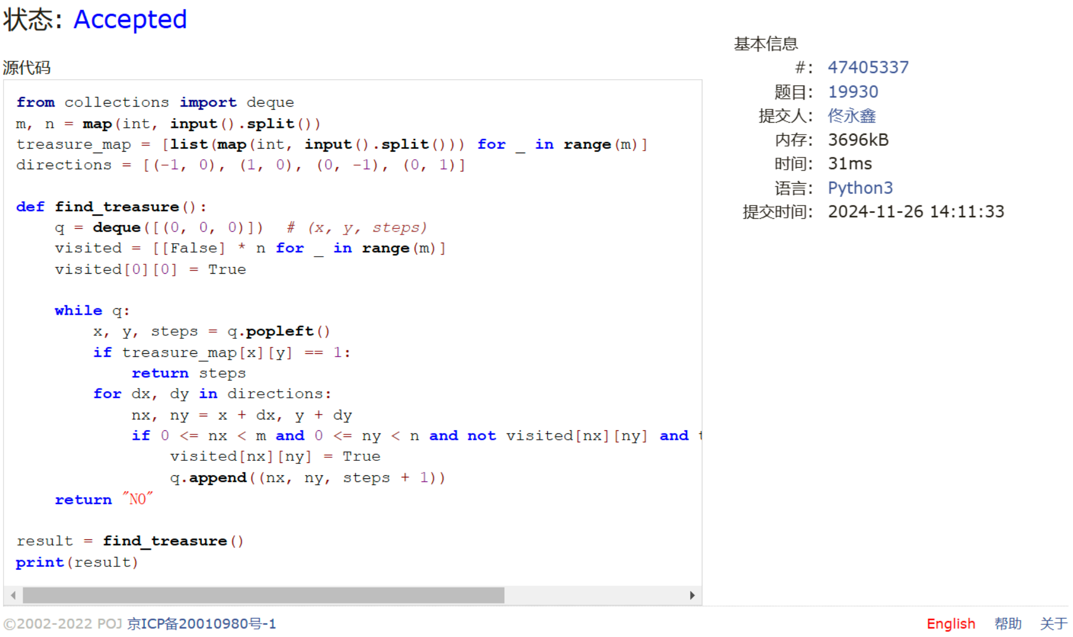
bfs, http://cs101.openjudge.cn/practice/19930

思路:

代码:

```
from collections import deque
m, n = map(int, input().split())
treasure_map = [list(map(int, input().split())) for _ in range(m)]
directions = [(-1, 0), (1, 0), (0, -1), (0, 1)]

def find_treasure():
    q = deque([(0, 0, 0)])  # (x, y, steps)
    visited = [[False] * n for _ in range(m)]
    visited[0][0] = True
    while q:
        x, y, steps = q.popleft()
        if treasure_map[x][y] == 1:
            return steps
        for dx, dy in directions:
            nx, ny = x + dx, y + dy
            if 0 <= nx < m and 0 <= ny < n and not visited[nx][ny] and
treasure_map[nx][ny] != 2:
                visited[nx][ny] = True
                q.append((nx, ny, steps + 1))
    return "NO"

result = find_treasure()
print(result)
```

代码运行截图 ==（至少包含有"Accepted"） ==



状态: Accepted

源代码

```
from collections import deque
m, n = map(int, input().split())
treasure_map = [list(map(int, input().split())) for _ in range(m)]
directions = [(-1, 0), (1, 0), (0, -1), (0, 1)]

def find_treasure():
    q = deque([(0, 0, 0)])  # (x, y, steps)
    visited = [[False] * n for _ in range(m)]
    visited[0][0] = True

    while q:
        x, y, steps = q.popleft()
        if treasure_map[x][y] == 1:
            return steps
        for dx, dy in directions:
            nx, ny = x + dx, y + dy
            if 0 <= nx < m and 0 <= ny < n and not visited[nx][ny] and t
                visited[nx][ny] = True
                q.append((nx, ny, steps + 1))
    return "NO"

result = find_treasure()
print(result)
```

基本信息

#:	47405337
题目:	19930
提交人:	佟永鑫
内存:	3696kB
时间:	31ms
语言:	Python3
提交时间:	2024-11-26 14:11:33

©2002-2022 POJ 京ICP备20010980号-1

04123: 马走日

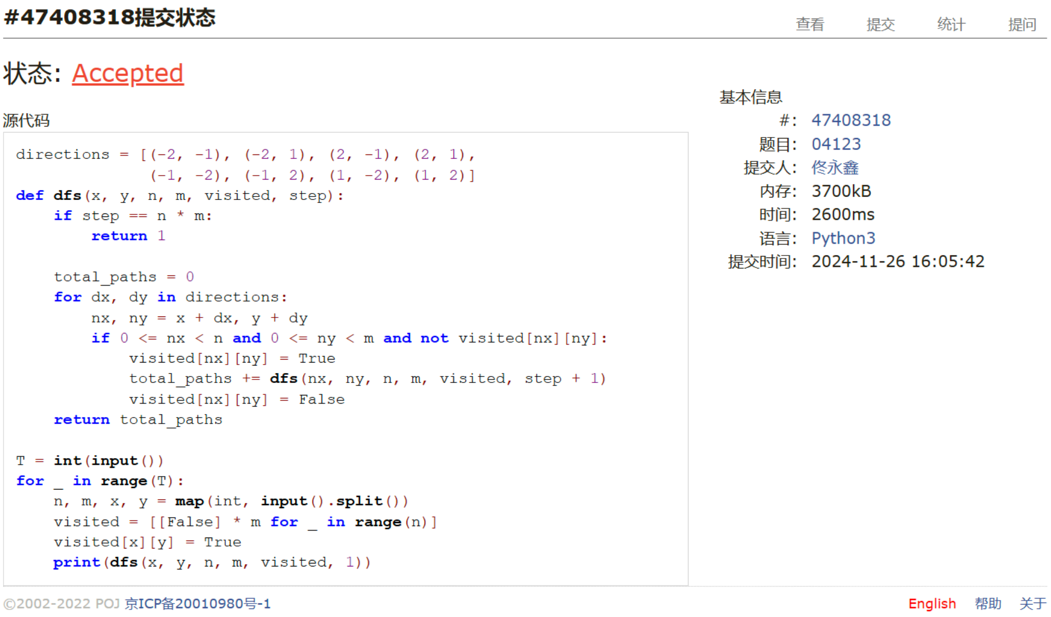
dfs, http://cs101.openjudge.cn/practice/04123

思路：

代码：

```
directions = [(-2, -1), (-2, 1), (2, -1), (2, 1),
              (-1, -2), (-1, 2), (1, -2), (1, 2)]
def dfs(x, y, n, m, visited, step):
    if step == n * m:
        return 1
    total_paths = 0
    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
            visited[nx][ny] = True
            total_paths += dfs(nx, ny, n, m, visited, step + 1)
            visited[nx][ny] = False
    return total_paths

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

代码运行截图 （至少包含有"Accepted"）



#47408318提交状态

状态: Accepted

源代码

```
directions = [(-2, -1), (-2, 1), (2, -1), (2, 1),
              (-1, -2), (-1, 2), (1, -2), (1, 2)]
def dfs(x, y, n, m, visited, step):
    if step == n * m:
        return 1

    total_paths = 0
    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
            visited[nx][ny] = True
            total_paths += dfs(nx, ny, n, m, visited, step + 1)
            visited[nx][ny] = False
    return total_paths

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

基本信息

#:	47408318
题目:	04123
提交人:	佟永鑫
内存:	3700kB
时间:	2600ms
语言:	Python3
提交时间:	2024-11-26 16:05:42

©2002-2022 POJ 京ICP备20010980号-1

sy316: 矩阵最大权值路径

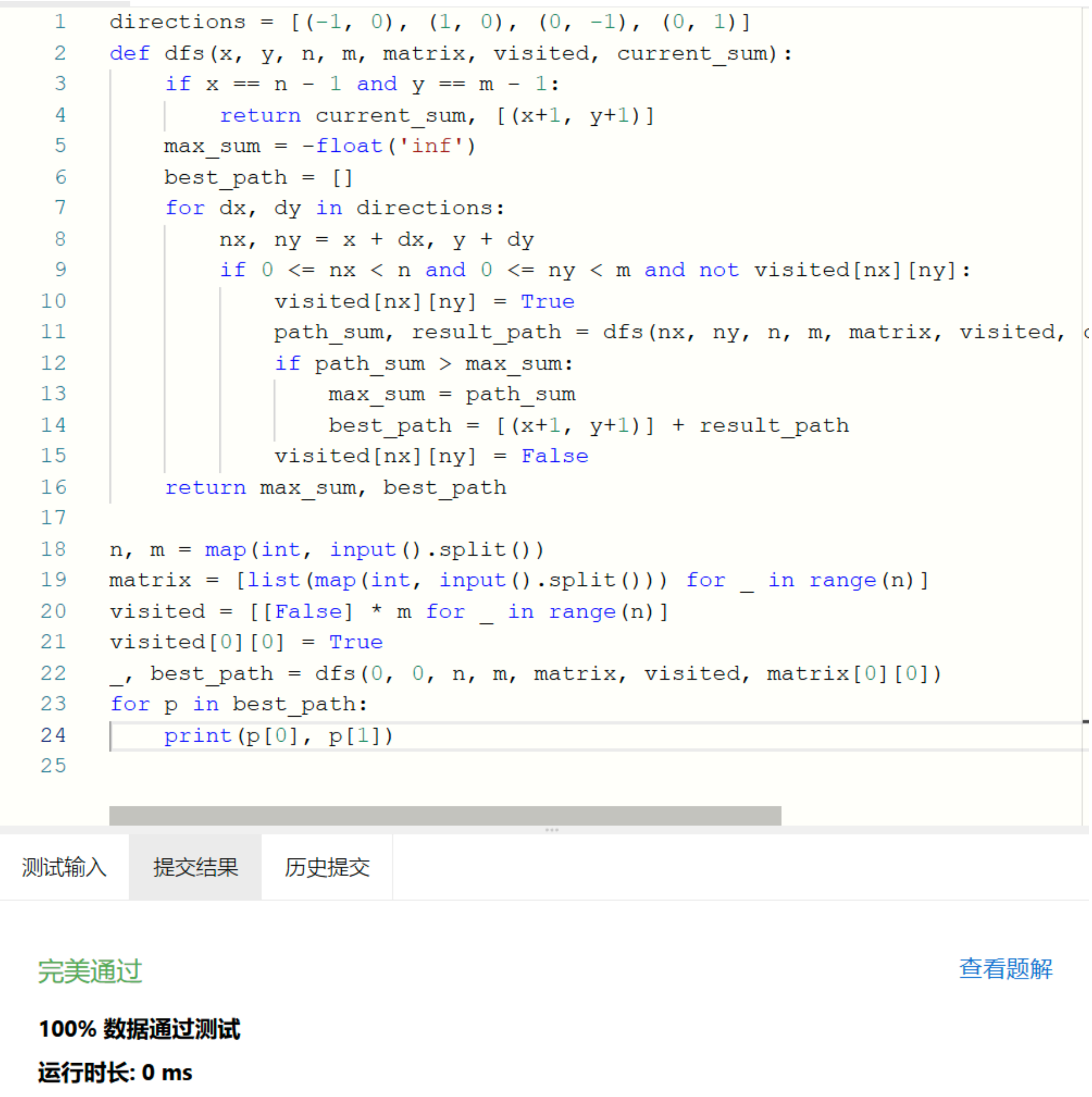
dfs, https://sunnywhy.com/sfbj/8/1/316

思路：

代码：

```
directions = [(-1, 0), (1, 0), (0, -1), (0, 1)]
def dfs(x, y, n, m, matrix, visited, current_sum):
    if x == n - 1 and y == m - 1:
        return current_sum, [(x+1, y+1)]
    max_sum = -float('inf')
    best_path = []
    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
            visited[nx][ny] = True
            path_sum, result_path = dfs(nx, ny, n, m, matrix, visited, current_sum + matrix[nx][ny])
            if path_sum > max_sum:
                max_sum = path_sum
                best_path = [(x+1, y+1)] + result_path
            visited[nx][ny] = False
    return max_sum, best_path

n, m = map(int, input().split())
matrix = [list(map(int, input().split())) for _ in range(n)]
visited = [[False] * m for _ in range(n)]
visited[0][0] = True
_, best_path = dfs(0, 0, n, m, matrix, visited, matrix[0][0])
for p in best_path:
    print(p[0], p[1])
```

代码运行截图 （至少包含有"Accepted"）



```
directions = [(-1, 0), (1, 0), (0, -1), (0, 1)]
def dfs(x, y, n, m, matrix, visited, current_sum):
    if x == n - 1 and y == m - 1:
        return current_sum, [(x+1, y+1)]
    max_sum = -float('inf')
    best_path = []
    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
            visited[nx][ny] = True
            path_sum, result_path = dfs(nx, ny, n, m, matrix, visited,
            if path_sum > max_sum:
                max_sum = path_sum
                best_path = [(x+1, y+1)] + result_path
            visited[nx][ny] = False
    return max_sum, best_path

n, m = map(int, input().split())
matrix = [list(map(int, input().split())) for _ in range(n)]
visited = [[False] * m for _ in range(n)]
visited[0][0] = True
_, best_path = dfs(0, 0, n, m, matrix, visited, matrix[0][0])
for p in best_path:
    print(p[0], p[1])
```

完美通过

查看题解

100% 数据通过测试

运行时长: 0 ms

LeetCode62.不同路径

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

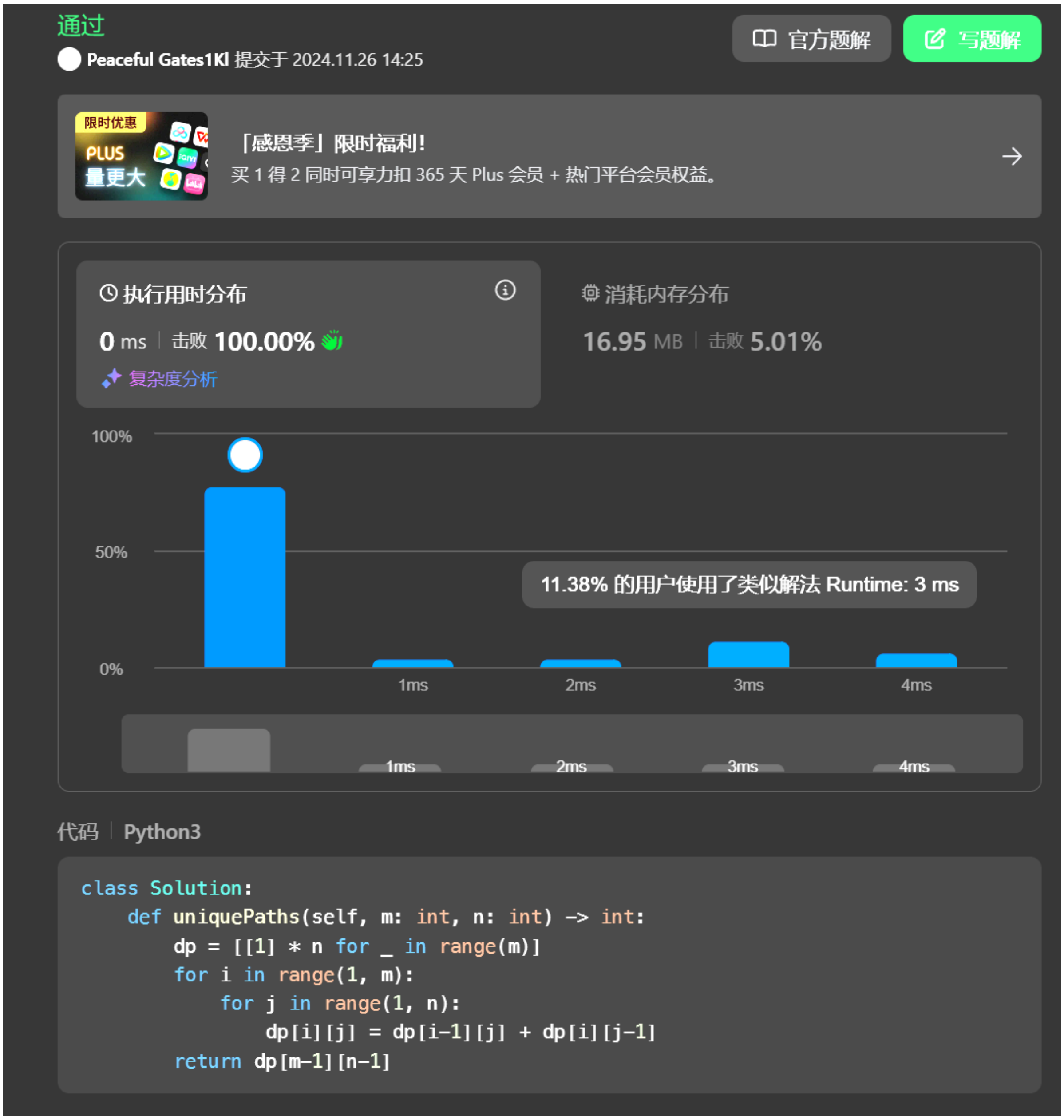
思路：

还可以直接计算组合数(m-1)C(m-1)(n-1)

代码：

```
class Solution:
    def uniquePaths(self, m: int, n: int) -> int:
        dp = [[1] * n for _ in range(m)]
        for i in range(1, m):
            for j in range(1, n):
                dp[i][j] = dp[i-1][j] + dp[i][j-1]
        return dp[m-1][n-1]
```

代码运行截图 （至少包含有"Accepted"）



通过

Peaceful Gates1Kl 提交于 2024.11.26 14:25

执行用时分布

0 ms | 击败 100.00%

复杂度分析

消耗内存分布

16.95 MB | 击败 5.01%

代码 | Python3

```
class Solution:
    def uniquePaths(self, m: int, n: int) -> int:
        dp = [[1] * n for _ in range(m)]
        for i in range(1, m):
            for j in range(1, n):
                dp[i][j] = dp[i-1][j] + dp[i][j-1]
        return dp[m-1][n-1]
```

sy358: 受到祝福的平方

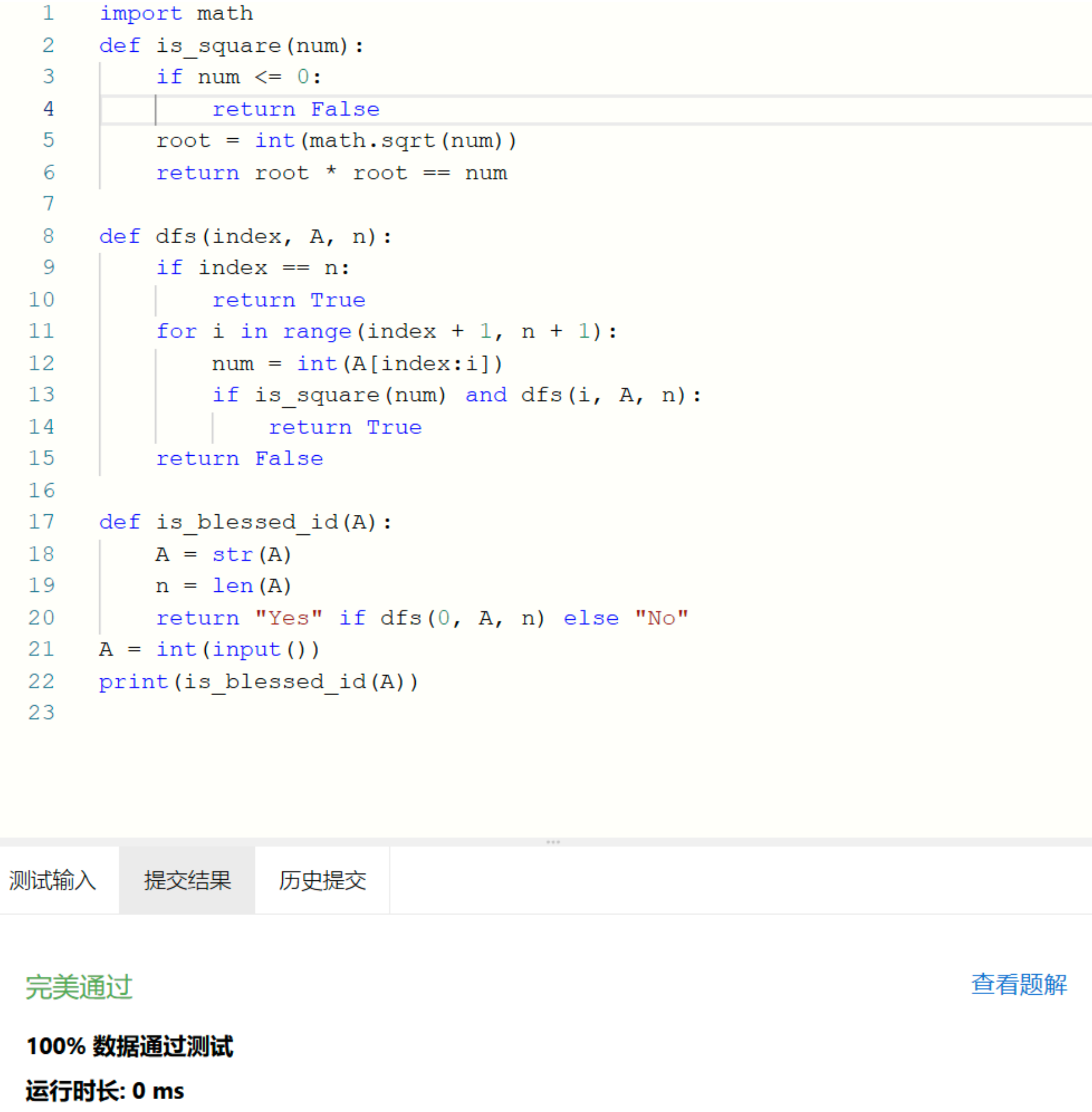
dfs, dp, https://sunnywhy.com/sfbj/8/3/539

思路：

代码：

```
import math
def is_square(num):
    if num <= 0:
        return False
    root = int(math.sqrt(num))
    return root * root == num

def dfs(index, A, n):
    if index == n:
        return True
    for i in range(index + 1, n + 1):
        num = int(A[index:i])
        if is_square(num) and dfs(i, A, n):
            return True
    return False

def is_blessed_id(A):
    A = str(A)
    n = len(A)
    return "Yes" if dfs(0, A, n) else "No"
A = int(input())
print(is_blessed_id(A))
```

代码运行截图 （至少包含有"Accepted"）



```
import math
def is_square(num):
    if num <= 0:
        return False
    root = int(math.sqrt(num))
    return root * root == num

def dfs(index, A, n):
    if index == n:
        return True
    for i in range(index + 1, n + 1):
        num = int(A[index:i])
        if is_square(num) and dfs(i, A, n):
            return True
    return False

def is_blessed_id(A):
    A = str(A)
    n = len(A)
    return "Yes" if dfs(0, A, n) else "No"
A = int(input())
print(is_blessed_id(A))

```

完美通过

100% 数据通过测试

运行时长: 0 ms

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“计概2024fall每日选做”、CF、LeetCode、洛谷等网站题目。

这周生病现在还没好，但好在作业题参考模版不算太难