Assignment #9: dfs, bfs, & dp

Updated 2107 GMT+8 Nov 19, 2024

2024 fall, Complied by 熊程宇 物理学院

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

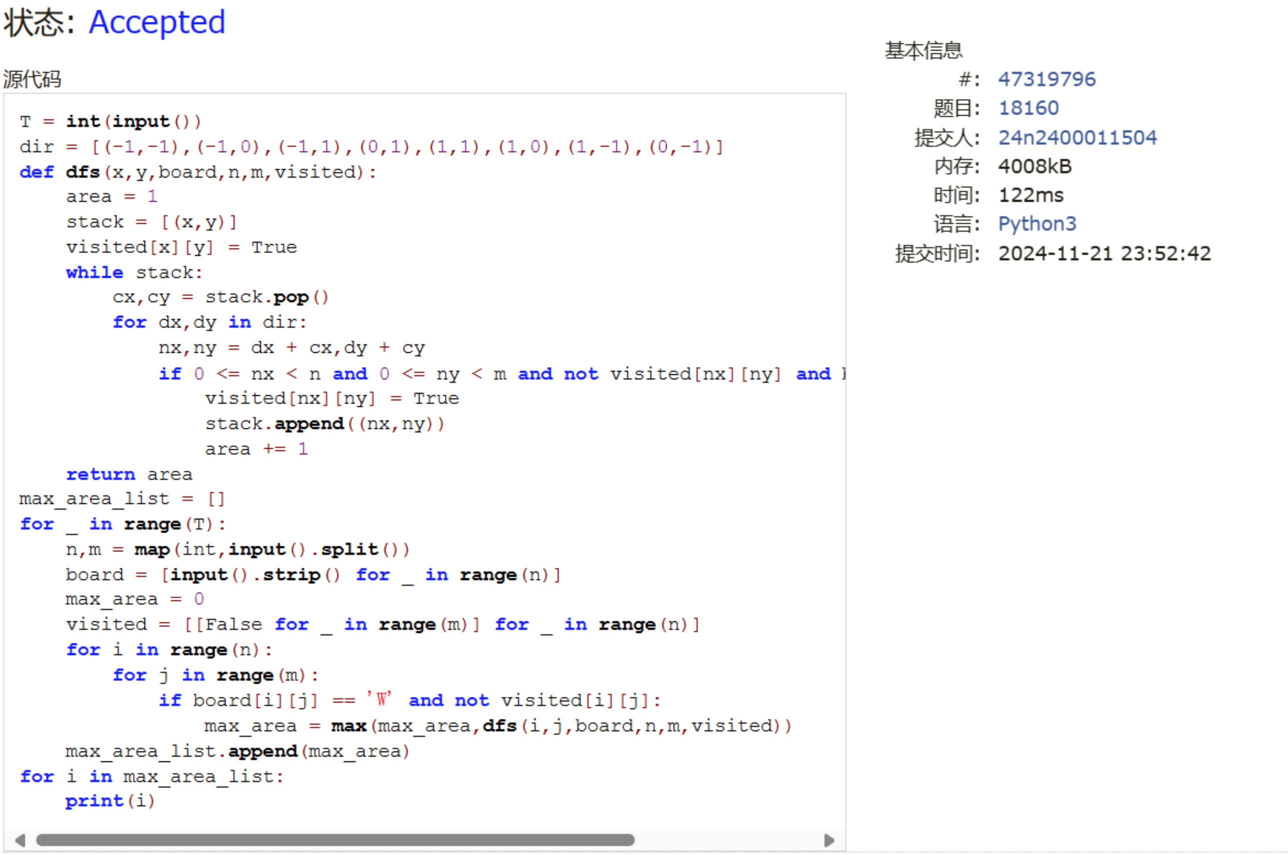
dfs similar, http://cs101.openjudge.cn/practice/18160

思路：

代码：

```
T = int(input())
dir = [(-1,-1),(-1,0),(-1,1),(0,1),(1,1),(1,0),(1,-1),(0,-1)]
def dfs(x,y,board,n,m,visited):
    area = 1
    stack = [(x,y)]
    visited[x][y] = True
    while stack:
        cx,cy = stack.pop()
        for dx,dy in dir:
            nx,ny = dx + cx,dy + cy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny] and board[nx][ny]
== 'W':
                visited[nx][ny] = True
                stack.append((nx,ny))
                area += 1
    return area
max_area_list = []
for _ in range(T):
    n,m = map(int,input().split())
    board = [input().strip() for _ in range(n)]
    max_area = 0
    visited = [[False for _ in range(m)] for _ in range(n)]
    for i in range(n):
        for j in range(m):
            if board[i][j] == 'W' and not visited[i][j]:
                max_area = max(max_area,dfs(i,j,board,n,m,visited))
    max_area_list.append(max_area)
for i in max_area_list:
    print(i)
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```
T = int(input())
dir = [(-1,-1),(-1,0),(-1,1),(0,1),(1,1),(1,0),(1,-1),(0,-1)]
def dfs(x,y,board,n,m,visited):
    area = 1
    stack = [(x,y)]
    visited[x][y] = True
    while stack:
        cx,cy = stack.pop()
        for dx,dy in dir:
            nx,ny = dx + cx,dy + cy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny] and 
                visited[nx][ny] = True
                stack.append((nx,ny))
                area += 1
    return area
max_area_list = []
for _ in range(T):
    n,m = map(int,input().split())
    board = [input().strip() for _ in range(n)]
    max_area = 0
    visited = [[False for _ in range(m)] for _ in range(n)]
    for i in range(n):
        for j in range(m):
            if board[i][j] == 'W' and not visited[i][j]:
                max_area = max(max_area,dfs(i,j,board,n,m,visited))
    max_area_list.append(max_area)
for i in max_area_list:
    print(i)
```

基本信息

#:	47319796
题目:	18160
提交人:	24n2400011504
内存:	4008kB
时间:	122ms
语言:	Python3
提交时间:	2024-11-21 23:52:42

19930: 寻宝

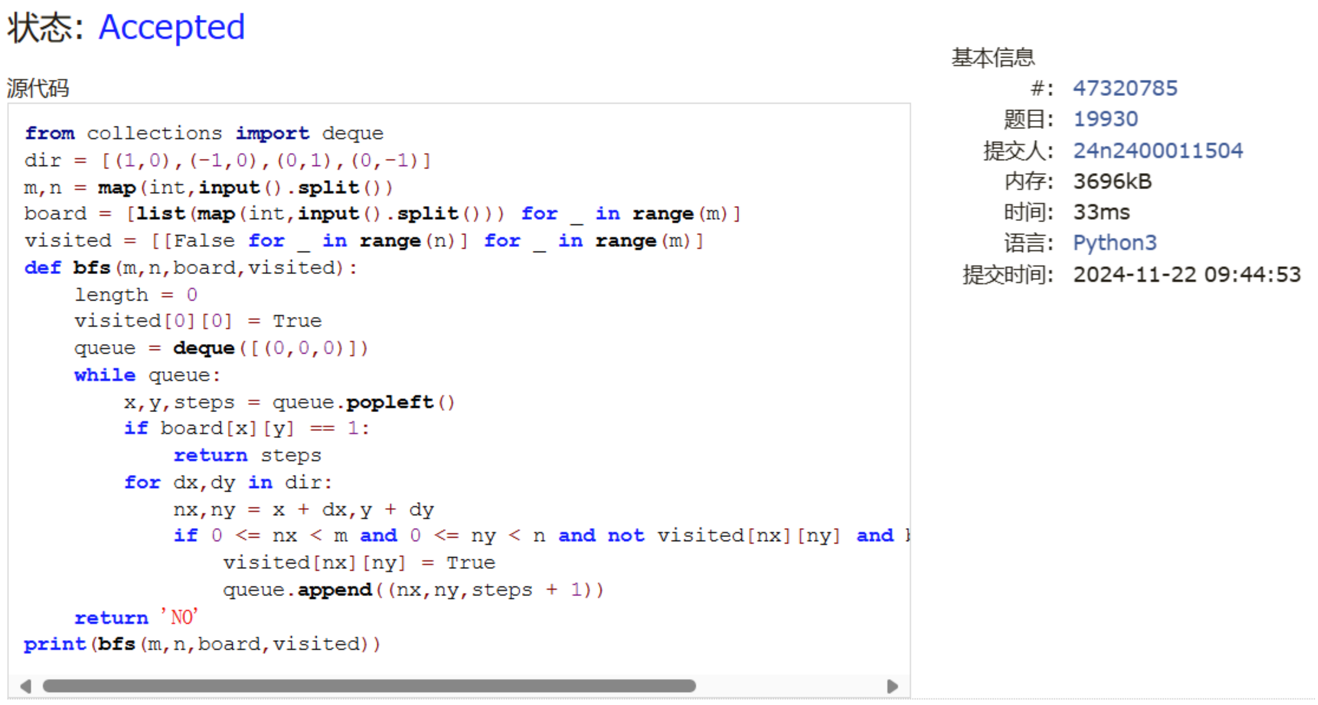
bfs, http://cs101.openjudge.cn/practice/19930

思路:

代码:

```
from collections import deque
dir = [(1,0),(-1,0),(0,1),(0,-1)]
m,n = map(int,input().split())
board = [list(map(int,input().split())) for _ in range(m)]
visited = [[False for _ in range(n)] for _ in range(m)]
def bfs(m,n,board,visited):
    length = 0
    visited[0][0] = True
    queue = deque([(0,0,0)])
    while queue:
        x,y,steps = queue.popleft()
        if board[x][y] == 1:
            return steps
        for dx,dy in dir:
            nx,ny = x + dx,y + dy
            if 0 <= nx < m and 0 <= ny < n and not visited[nx][ny] and board[nx][ny]
!= 2:
                visited[nx][ny] = True
                queue.append((nx,ny,steps + 1))
    return 'NO'
print(bfs(m,n,board,visited))
```

代码运行截图 ==（至少包含有"Accepted"） ==



状态: Accepted

源代码

```
from collections import deque
dir = [(1,0),(-1,0),(0,1),(0,-1)]
m,n = map(int,input().split())
board = [list(map(int,input().split())) for _ in range(m)]
visited = [[False for _ in range(n)] for _ in range(m)]
def bfs(m,n,board,visited):
    length = 0
    visited[0][0] = True
    queue = deque([(0,0,0)])
    while queue:
        x,y,steps = queue.popleft()
        if board[x][y] == 1:
            return steps
        for dx,dy in dir:
            nx,ny = x + dx,y + dy
            if 0 <= nx < m and 0 <= ny < n and not visited[nx][ny] and 
                visited[nx][ny] = True
                queue.append((nx,ny,steps + 1))
    return 'NO'
print(bfs(m,n,board,visited))
```

基本信息

#: 47320785

题目: 19930

提交人: 24n2400011504

内存: 3696kB

时间: 33ms

语言: Python3

提交时间: 2024-11-22 09:44:53

04123: 马走日

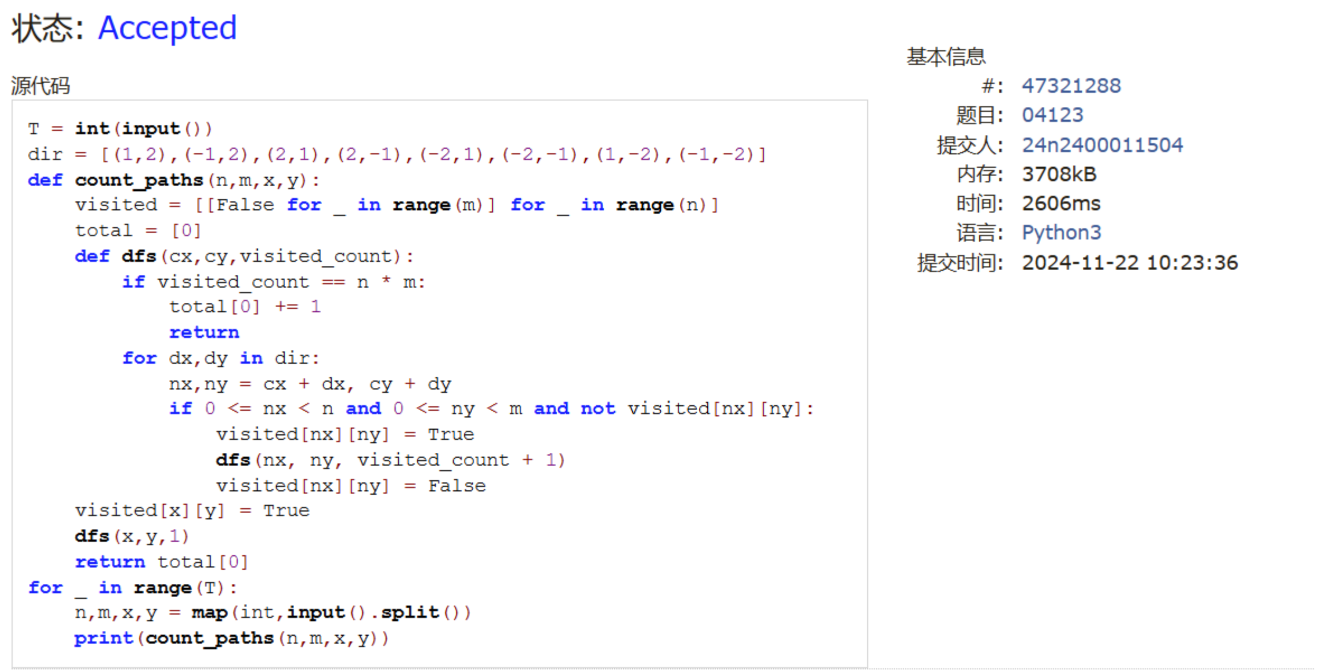
dfs, http://cs101.openjudge.cn/practice/04123

思路:

代码:

```
T = int(input())
dir = [(1,2),(-1,2),(2,1),(2,-1),(-2,1),(-2,-1),(1,-2),(-1,-2)]
def count_paths(n,m,x,y):
    visited = [[False for _ in range(m)] for _ in range(n)]
    total = [0]
    def dfs(cx,cy,visited_count):
        if visited_count == n * m:
            total[0] += 1
            return
        for dx,dy in dir:
            nx,ny = cx + dx, cy + dy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
                visited[nx][ny] = True
                dfs(nx, ny, visited_count + 1)
                visited[nx][ny] = False
    visited[x][y] = True
    dfs(x,y,1)
    return total[0]
for _ in range(T):
    n,m,x,y = map(int,input().split())
    print(count_paths(n,m,x,y))
```

代码运行截图 （至少包含有"Accepted"）



状态: Accepted

源代码

```
T = int(input())
dir = [(1,2),(-1,2),(2,1),(2,-1),(-2,1),(-2,-1),(1,-2),(-1,-2)]
def count_paths(n,m,x,y):
    visited = [[False for _ in range(m)] for _ in range(n)]
    total = [0]
    def dfs(cx,cy,visited_count):
        if visited_count == n * m:
            total[0] += 1
            return
        for dx,dy in dir:
            nx,ny = cx + dx, cy + dy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
                visited[nx][ny] = True
                dfs(nx, ny, visited_count + 1)
                visited[nx][ny] = False
    visited[x][y] = True
    dfs(x,y,1)
    return total[0]
for _ in range(T):
    n,m,x,y = map(int,input().split())
    print(count_paths(n,m,x,y))
```

基本信息

#:	47321288
题目:	04123
提交人:	24n2400011504
内存:	3708kB
时间:	2606ms
语言:	Python3
提交时间:	2024-11-22 10:23:36

sy316: 矩阵最大权值路径

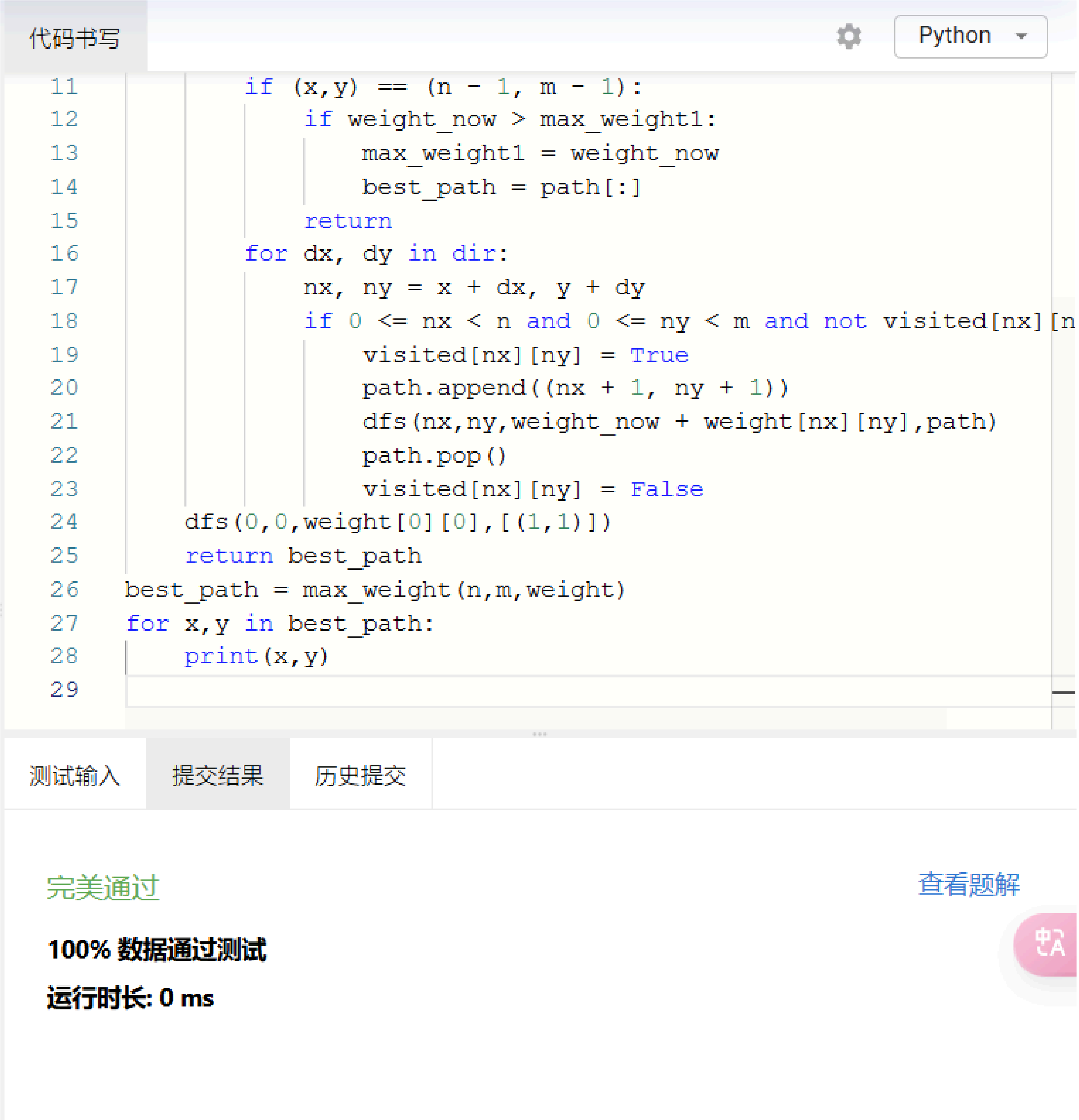
dfs, https://sunnywhy.com/sfbj/8/1/316

思路:

代码:

```
n, m = map(int,input().split())
weight = [list(map(int,input().split())) for _ in range(n)]
dir = dir = [(-1,0),(1,0),(0,1),(0,-1)]
def max_weight(n,m,weight):
    visited = [[False for _ in range(m)] for _ in range(n)]
    max_weight1 = float('-inf')
    visited[0][0] = True
    best_path = []
    def dfs(x,y,weight_now,path):
        nonlocal max_weight1,best_path
        if (x,y) == (n - 1, m - 1):
            if weight_now > max_weight1:
                max_weight1 = weight_now
                best_path = path[:]
            return
        for dx, dy in dir:
            nx, ny = x + dx, y + dy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
                visited[nx][ny] = True
                path.append((nx + 1, ny + 1))
                dfs(nx,ny,weight_now + weight[nx][ny],path)
                path.pop()
                visited[nx][ny] = False
    dfs(0,0,weight[0][0],[(1,1)])
    return best_path
best_path = max_weight(n,m,weight)
for x,y in best_path:
    print(x,y)
```

代码运行截图 （至少包含有"Accepted"）



```python
        if (x,y) == (n - 1, m - 1):
            if weight_now > max_weight1:
                max_weight1 = weight_now
                best_path = path[:]
            return
        for dx, dy in dir:
            nx, ny = x + dx, y + dy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][n
                visited[nx][ny] = True
                path.append((nx + 1, ny + 1))
                dfs(nx,ny,weight_now + weight[nx][ny],path)
                path.pop()
                visited[nx][ny] = False
    dfs(0,0,weight[0][0],[(1,1)])
    return best_path
best_path = max_weight(n,m,weight)
for x,y in best_path:
    print(x,y)
```

完美通过

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/

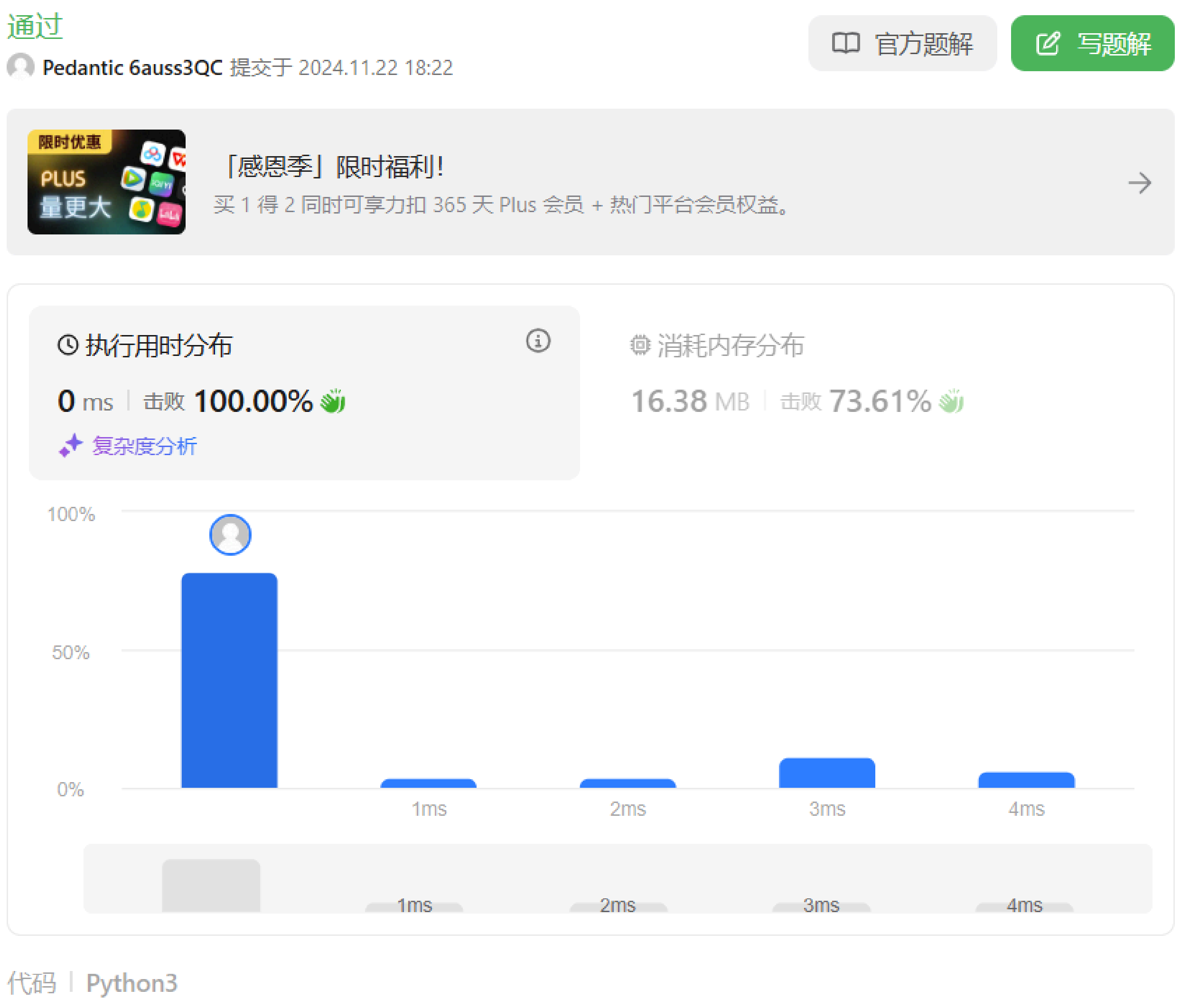
思路:

基础dp

代码:

```
class Solution:
    def uniquePaths(self, m: int, n: int) -> int:
        dp = [[0 for _ in range(n)] for _ in range(m)]
        for i in range(n):
            dp[0][i] = 1
        for i in range(m):
            dp[i][0] = 1
        for i in range(1, m):
            for j in range(1, n):
                dp[i][j] = dp[i][j - 1] + dp[i - 1][j]
        return dp[m - 1][n - 1]
```

代码运行截图 （至少包含有"Accepted"）



通过

Pedantic 6auss3QC 提交于 2024.11.22 18:22

执行用时分布

0 ms | 击败 100.00%

复杂度分析

消耗内存分布

16.38 MB | 击败 73.61%

代码 | Python3

sy358: 受到祝福的平方

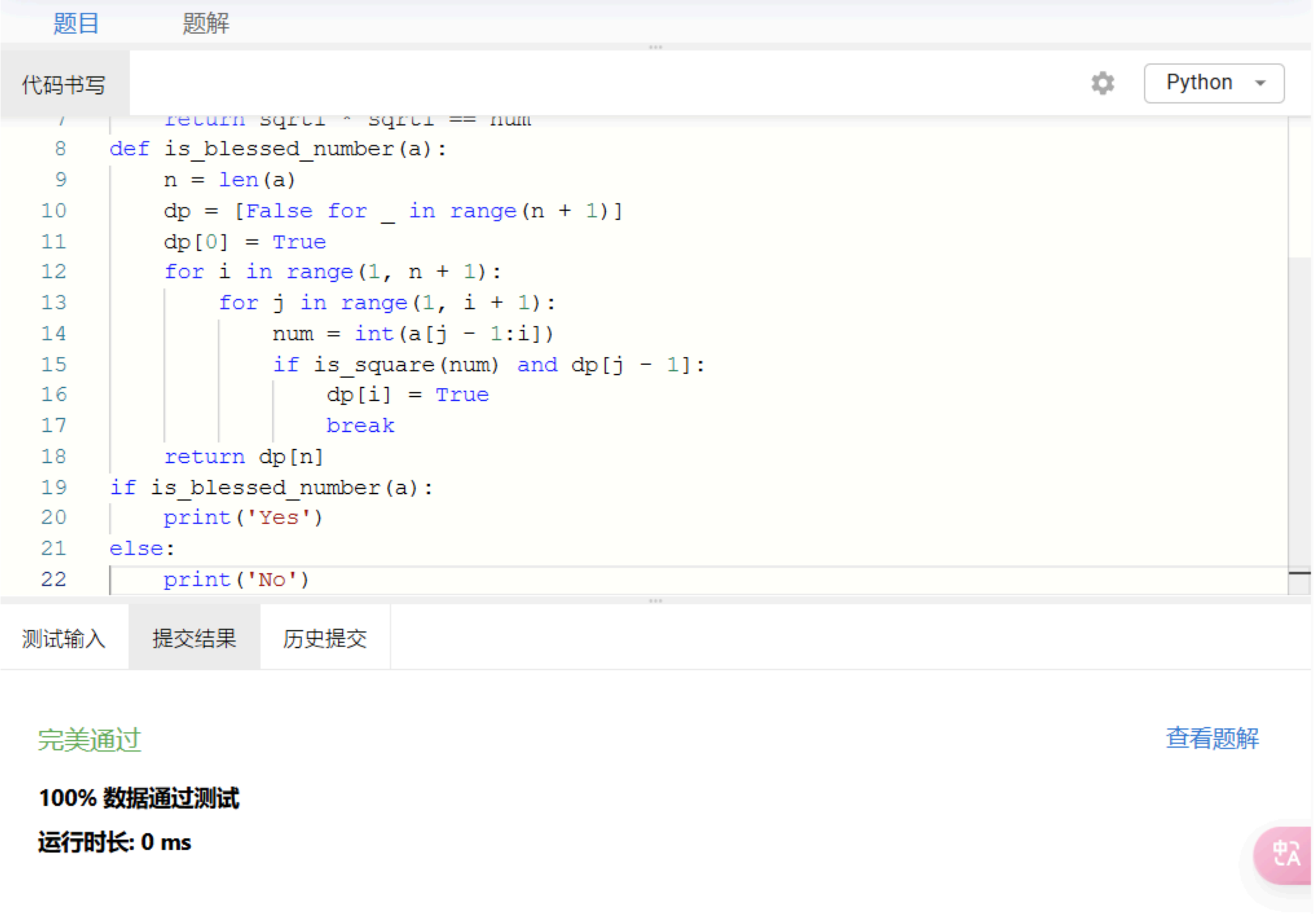
dfs, dp, https://sunnywhy.com/sfbj/8/3/539

思路：

代码：

```python
import math
a = input()
def is_square(num):
    if num == 0:
        return False
    sqrt1 = int(math.sqrt(num))
    return sqrt1 * sqrt1 == num
def is_blessed_number(a):
    n = len(a)
    dp = [False for _ in range(n + 1)]
    dp[0] = True
    for i in range(1, n + 1):
        for j in range(1, i + 1):
            num = int(a[j - 1:i])
            if is_square(num) and dp[j - 1]:
                dp[i] = True
                break
    return dp[n]
if is_blessed_number(a):
    print('Yes')
else:
    print('No')
```

代码运行截图 （至少包含有"Accepted"）



```python
    return sqrt1 * sqrt1 == num
def is_blessed_number(a):
    n = len(a)
    dp = [False for _ in range(n + 1)]
    dp[0] = True
    for i in range(1, n + 1):
        for j in range(1, i + 1):
            num = int(a[j - 1:i])
            if is_square(num) and dp[j - 1]:
                dp[i] = True
                break
    return dp[n]
if is_blessed_number(a):
    print('Yes')
else:
    print('No')
```

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

以为会做八皇后了，结果还是写了个暴力遍历的方法…
在晴问上练了一些dfs和dp的题目。