Assignment #9: dfs, bfs, & dp

Updated 2107 GMT+8 Nov 19, 2024

2024 fall, Complied by 徐贤天、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。

1. 题目

18160: 最大连通域面积

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

思路：

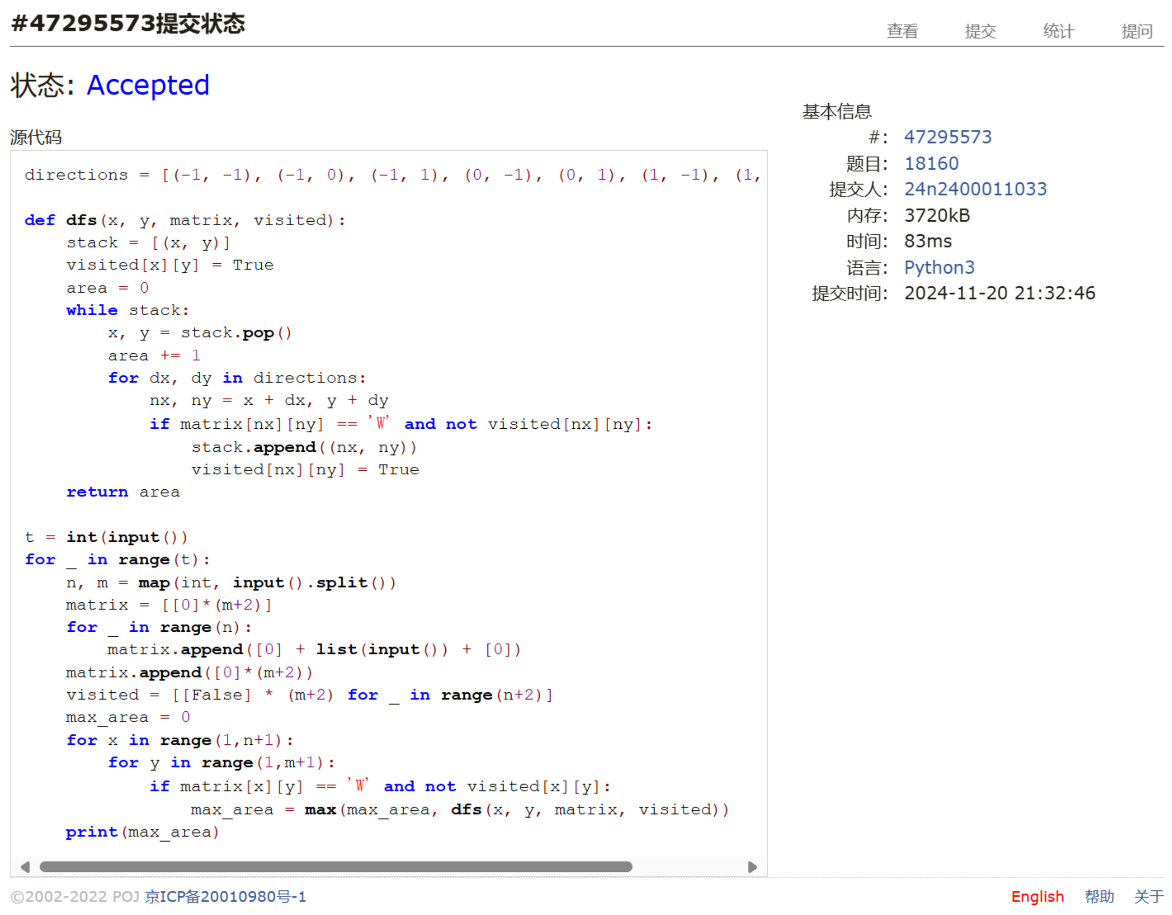
代码：

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

def dfs(x, y, matrix, visited):
    stack = [(x, y)]
    visited[x][y] = True
    area = 0
    while stack:
        x, y = stack.pop()
        area += 1
        for dx, dy in directions:
            nx, ny = x + dx, y + dy
            if matrix[nx][ny] == 'W' and not visited[nx][ny]:
                stack.append((nx, ny))
                visited[nx][ny] = True
    return area

t = int(input())
for _ in range(t):
    n, m = map(int, input().split())
    matrix = [[0]*(m+2)]
    for _ in range(n):
        matrix.append([0] + list(input()) + [0])
    matrix.append([0]*(m+2))
    visited = [[False] * (m+2) for _ in range(n+2)]
    max_area = 0
    for x in range(1,n+1):
        for y in range(1,m+1):
            if matrix[x][y] == 'W' and not visited[x][y]:
                max_area = max(max_area, dfs(x, y, matrix, visited))
    print(max_area)
```

代码运行截图 （至少包含有"Accepted"）



#47295573提交状态

查看 提交 统计 提问

状态: Accepted

源代码

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

def dfs(x, y, matrix, visited):
    stack = [(x, y)]
    visited[x][y] = True
    area = 0
    while stack:
        x, y = stack.pop()
        area += 1
        for dx, dy in directions:
            nx, ny = x + dx, y + dy
            if matrix[nx][ny] == 'W' and not visited[nx][ny]:
                stack.append((nx, ny))
                visited[nx][ny] = True
    return area

t = int(input())
for _ in range(t):
    n, m = map(int, input().split())
    matrix = [[0]*(m+2)]
    for _ in range(n):
        matrix.append([0] + list(input()) + [0])
    matrix.append([0]*(m+2))
    visited = [[False] * (m+2) for _ in range(n+2)]
    max_area = 0
    for x in range(1,n+1):
        for y in range(1,m+1):
            if matrix[x][y] == 'W' and not visited[x][y]:
                max_area = max(max_area, dfs(x, y, matrix, visited))
    print(max_area)
```

基本信息

#:	47295573
题目:	18160
提交人:	24n2400011033
内存:	3720kB
时间:	83ms
语言:	Python3
提交时间:	2024-11-20 21:32:46

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

19930: 寻宝

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

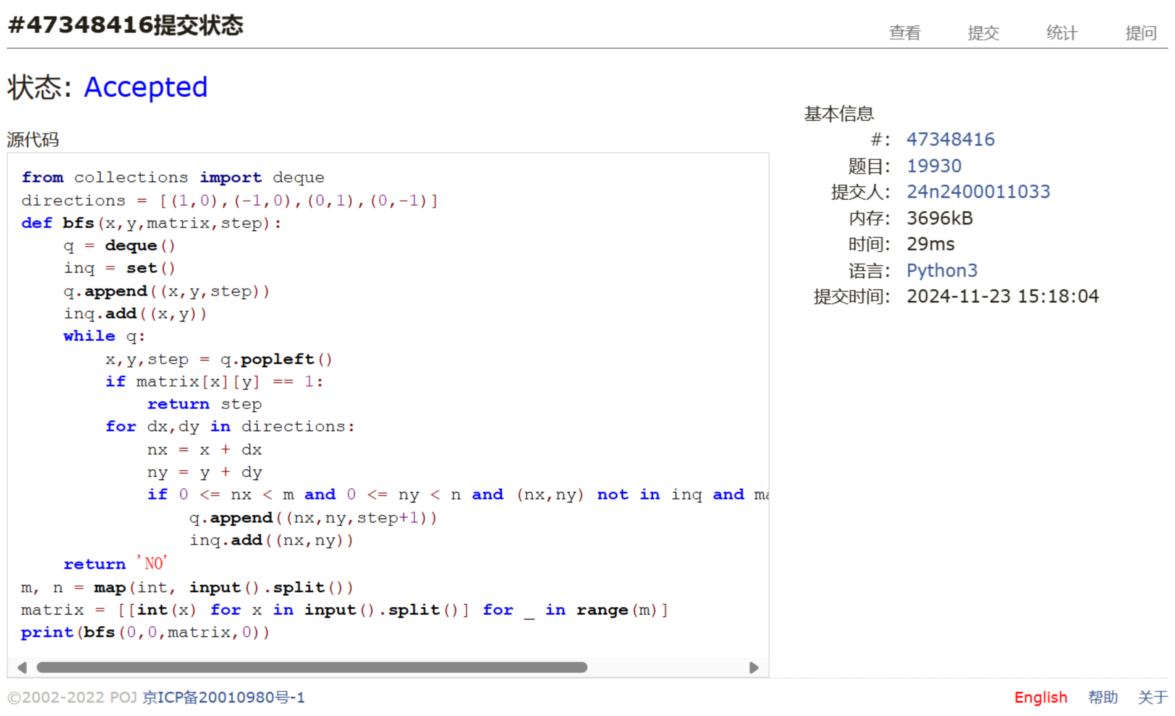
思路：

刚开始在nx和ny处判断是否到1点，结果一直WA，看了群里才发现特殊情况没有考虑

代码：

```
from collections import deque
directions = [(1,0),(-1,0),(0,1),(0,-1)]
def bfs(x,y,matrix,step):
    q = deque()
    inq = set()
    q.append((x,y,step))
    inq.add((x,y))
    while q:
        x,y,step = q.popleft()
        if matrix[x][y] == 1:
            return step
        for dx,dy in directions:
            nx = x + dx
            ny = y + dy
            if 0 <= nx < m and 0 <= ny < n and (nx,ny) not in inq and matrix[nx]
[ny] != 2:
                q.append((nx,ny,step+1))
                inq.add((nx,ny))
    return 'NO'
m, n = map(int, input().split())
matrix = [[int(x) for x in input().split()] for _ in range(m)]
print(bfs(0,0,matrix,0))
```

代码运行截图 == （至少包含有"Accepted"） ==



#47348416提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from collections import deque
directions = [(1,0),(-1,0),(0,1),(0,-1)]
def bfs(x,y,matrix,step):
    q = deque()
    inq = set()
    q.append((x,y,step))
    inq.add((x,y))
    while q:
        x,y,step = q.popleft()
        if matrix[x][y] == 1:
            return step
        for dx,dy in directions:
            nx = x + dx
            ny = y + dy
            if 0 <= nx < m and 0 <= ny < n and (nx,ny) not in inq and ma
                q.append((nx,ny,step+1))
                inq.add((nx,ny))
    return 'NO'
m, n = map(int, input().split())
matrix = [[int(x) for x in input().split()] for _ in range(m)]
print(bfs(0,0,matrix,0))
```

基本信息

#:	47348416
题目:	19930
提交人:	24n2400011033
内存:	3696kB
时间:	29ms
语言:	Python3
提交时间:	2024-11-23 15:18:04

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

04123: 马走日

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

思路：

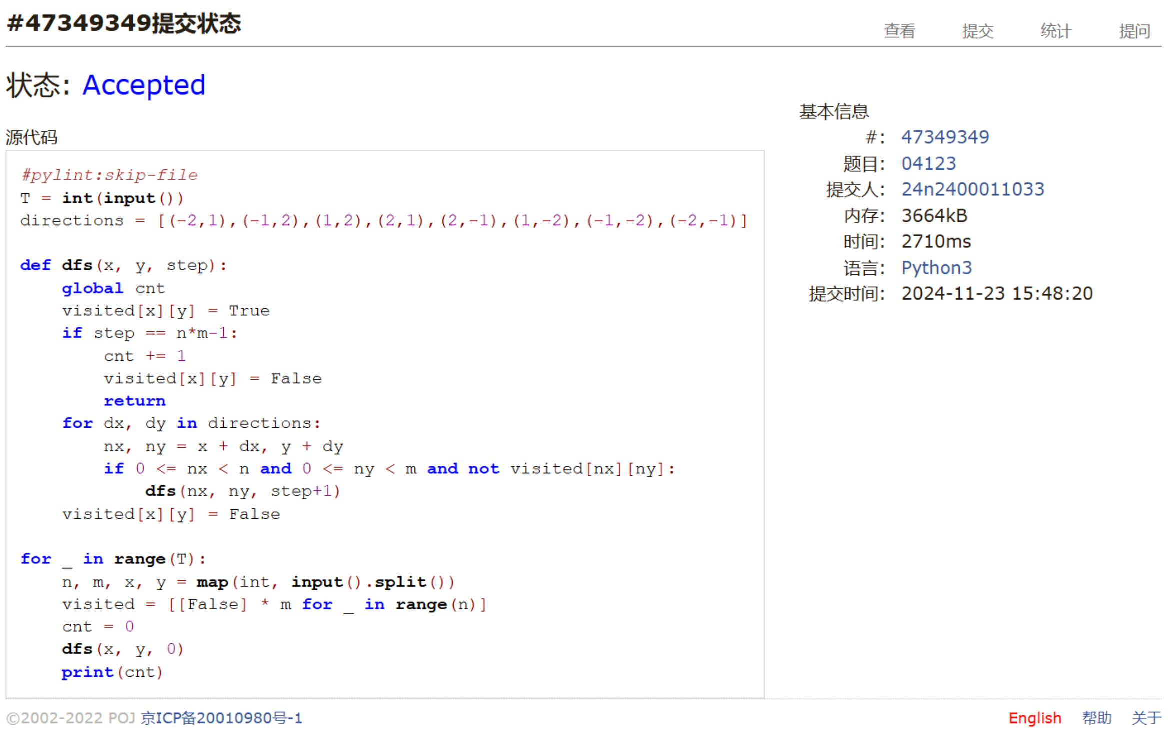
代码：

```
#pylint:skip-file
T = int(input())
directions = [(-2,1),(-1,2),(1,2),(2,1),(2,-1),(1,-2),(-1,-2),(-2,-1)]
```

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

for _ in range(T):
    n, m, x, y = map(int, input().split())
    visited = [[False] * m for _ in range(n)]
    cnt = 0
    dfs(x, y, 0)
    print(cnt)
```

代码运行截图 （至少包含有"Accepted"）



#47349349提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```python
#pylint:skip-file
T = int(input())
directions = [(-2,1),(-1,2),(1,2),(2,1),(2,-1),(1,-2),(-1,-2),(-2,-1)]

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

for _ in range(T):
    n, m, x, y = map(int, input().split())
    visited = [[False] * m for _ in range(n)]
    cnt = 0
    dfs(x, y, 0)
    print(cnt)
```

基本信息

#:	47349349
题目:	04123
提交人:	24n2400011033
内存:	3664kB
时间:	2710ms
语言:	Python3
提交时间:	2024-11-23 15:48:20

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

sy316: 矩阵最大权值路径

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

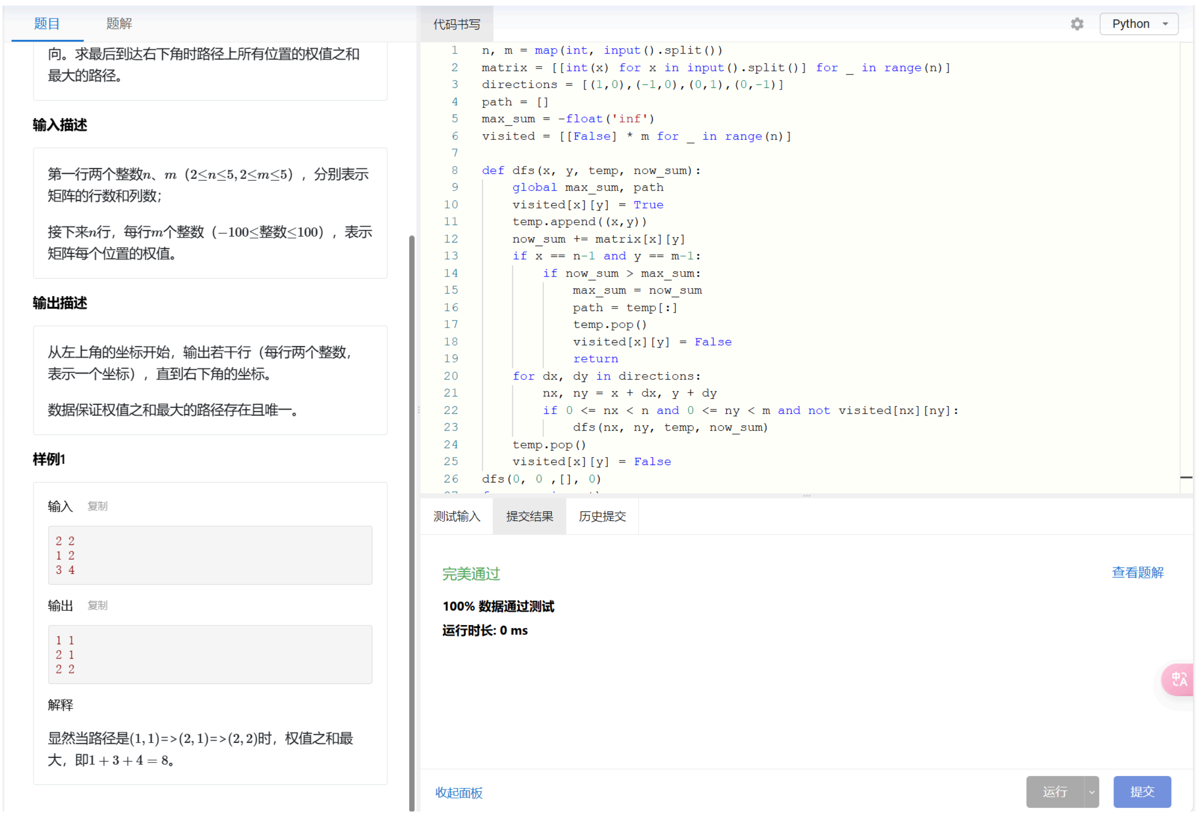
思路：

代码：

```python
n, m = map(int, input().split())
matrix = [[int(x) for x in input().split()] for _ in range(n)]
directions = [(1,0),(-1,0),(0,1),(0,-1)]
path = []
max_sum = -float('inf')
visited = [[False] * m for _ in range(n)]

def dfs(x, y, temp, now_sum):
    global max_sum, path
    visited[x][y] = True
    temp.append((x,y))
    now_sum += matrix[x][y]
    if x == n-1 and y == m-1:
        if now_sum > max_sum:
            max_sum = now_sum
            path = temp[:]
            temp.pop()
            visited[x][y] = False
            return
    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
            dfs(nx, ny, temp, now_sum)
    temp.pop()
    visited[x][y] = False
dfs(0, 0 ,[], 0)
for x, y in path:
    print(x+1, y+1)
```

代码运行截图 （至少包含有"Accepted"）



向。求最后到达右下角时路径上所有位置的权值之和最大的路径。

输入描述

第一行两个整数n、m（$2 \le n \le 5, 2 \le m \le 5$），分别表示矩阵的行数和列数；

接下来n行，每行m个整数（$-100 \le$整数≤ 100），表示矩阵每个位置的权值。

输出描述

从左上角的坐标开始，输出若干行（每行两个整数，表示一个坐标），直到右下角的坐标。

数据保证权值之和最大的路径存在且唯一。

样例1

输入 复制

```
2 2
1 2
3 4
```

输出 复制

```
1 1
2 1
2 2
```

解释

显然当路径是$(1,1)=>(2,1)=>(2,2)$时，权值之和最大，即$1+3+4=8$。

```python
n, m = map(int, input().split())
matrix = [[int(x) for x in input().split()] for _ in range(n)]
directions = [(1,0),(-1,0),(0,1),(0,-1)]
path = []
max_sum = -float('inf')
visited = [[False] * m for _ in range(n)]

def dfs(x, y, temp, now_sum):
    global max_sum, path
    visited[x][y] = True
    temp.append((x,y))
    now_sum += matrix[x][y]
    if x == n-1 and y == m-1:
        if now_sum > max_sum:
            max_sum = now_sum
            path = temp[:]
            temp.pop()
            visited[x][y] = False
            return
    for dx, dy in directions:
        nx, ny = x + dx, y + dy
        if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny]:
            dfs(nx, ny, temp, now_sum)
    temp.pop()
    visited[x][y] = False
dfs(0, 0 ,[], 0)
```

完美通过

100% 数据通过测试

运行时长: 0 ms

LeetCode62.不同路径

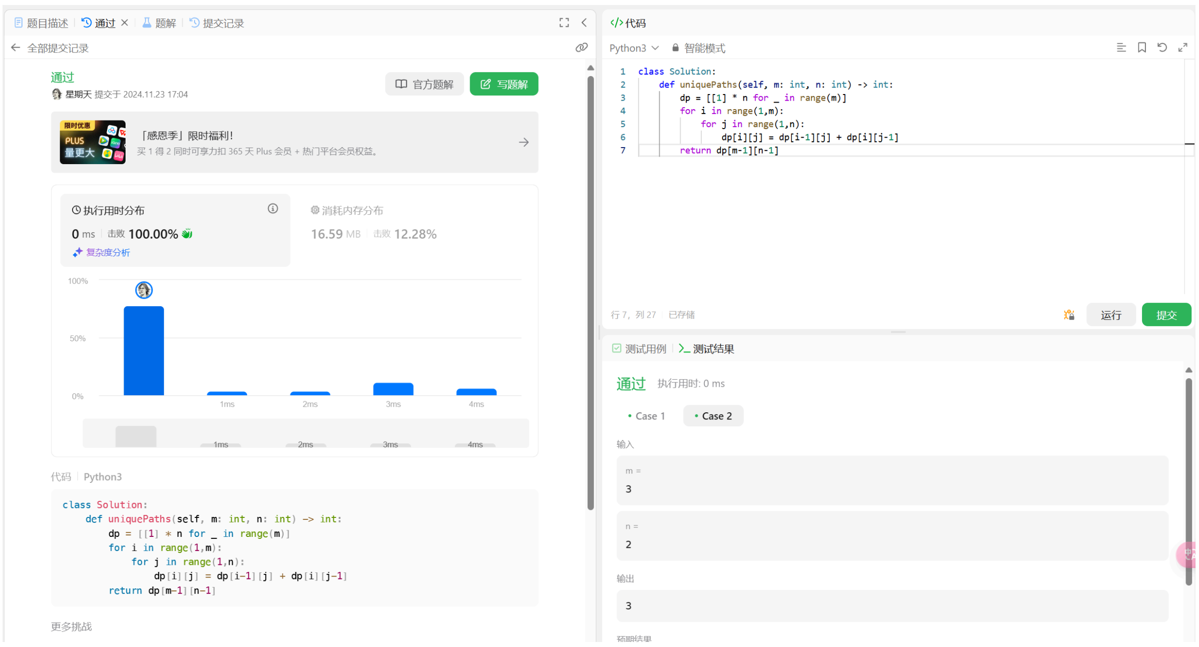
dp, https://leetcode.cn/problems/unique-paths/

思路：

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
星期天 提交于 2024.11.23 17:04

「感恩季」限时福利！
买 1 得 2 同时可享力扣 365 天 Plus 会员 + 热门平台会员权益。

执行用时分布
0 ms | 击败 100.00%
复杂度分析

消耗内存分布
16.59 MB | 击败 12.28%

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]
```

行 7，列 27 | 已存储

通过 执行用时: 0 ms

Case 1 · Case 2

输入

m =
3

n =
2

输出

3

预期结果

sy358: 受到祝福的平方

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

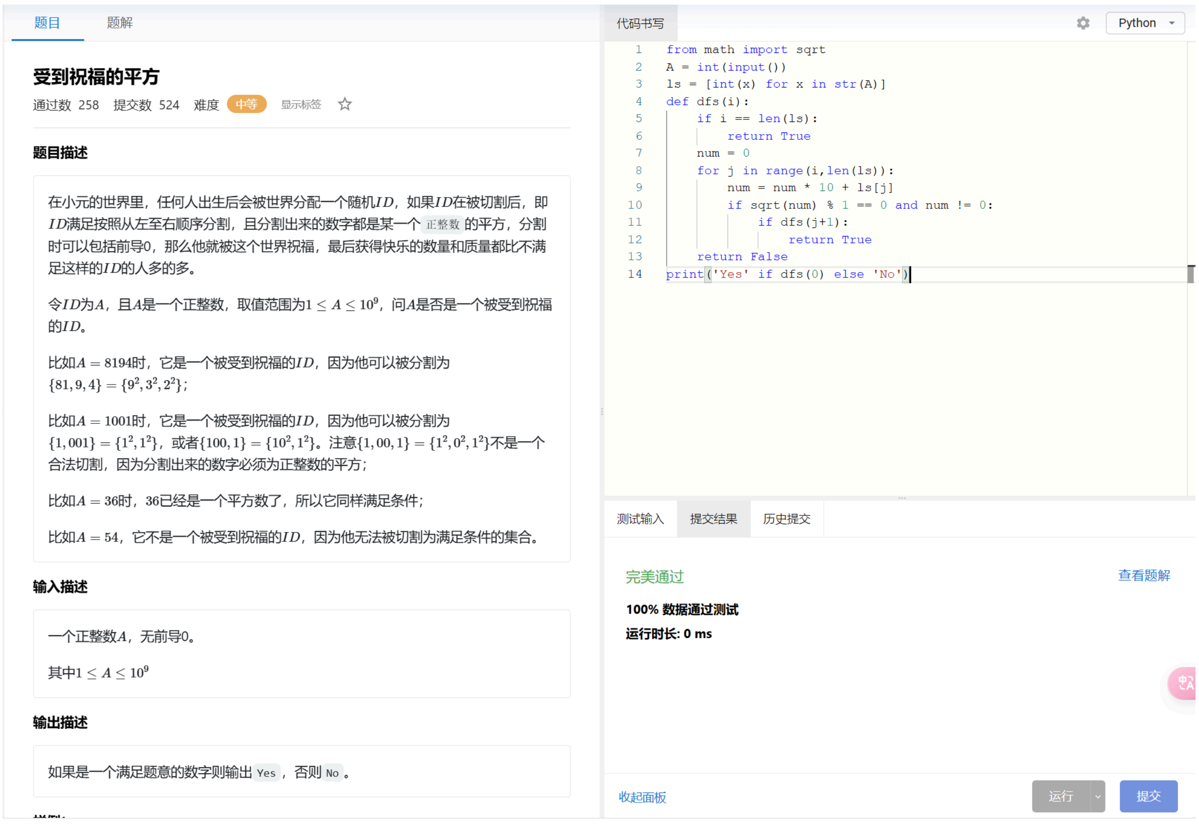
思路：

代码：

```
from math import sqrt
A = int(input())
```

```python
ls = [int(x) for x in str(A)]
def dfs(i):
    if i == len(ls):
        return True
    num = 0
    for j in range(i,len(ls)):
        num = num * 10 + ls[j]
        if sqrt(num) % 1 == 0 and num != 0:
            if dfs(j+1):
                return True
    return False
print('Yes' if dfs(0) else 'No')
```

代码运行截图 （至少包含有"Accepted"）



题目 题解

受到祝福的平方

通过数 258 提交数 524 难度 中等 显示标签

题目描述

在小元的世界里，任何人出生后会被世界分配一个随机ID，如果ID在被切割后，即ID满足按照从左至右顺序分割，且分割出来的数字都是某一个 正整数 的平方，分割时可以包括前导0，那么他就被这个世界祝福，最后获得快乐的数量和质量都比不满足这样的ID的人多的多。

令ID为A，且A是一个正整数，取值范围为$1 \leq A \leq 10^9$，问A是否是一个被受到祝福的ID。

比如$A = 8194$时，它是一个被受到祝福的ID，因为他可以被分割为$\{81, 9, 4\} = \{9^2, 3^2, 2^2\}$；

比如$A = 1001$时，它是一个被受到祝福的ID，因为他可以被分割为$\{1, 001\} = \{1^2, 1^2\}$，或者$\{100, 1\} = \{10^2, 1^2\}$。注意$\{1, 00, 1\} = \{1^2, 0^2, 1^2\}$不是一个合法切割，因为分割出来的数字必须为正整数的平方；

比如$A = 36$时，36已经是一个平方数了，所以它同样满足条件；

比如$A = 54$，它不是一个被受到祝福的ID，因为他无法被切割为满足条件的集合。

输入描述

一个正整数A，无前导0。

其中$1 \leq A \leq 10^9$

输出描述

如果是一个满足题意的数字则输出 `Yes`，否则 `No`。

代码书写 Python

```python
from math import sqrt
A = int(input())
ls = [int(x) for x in str(A)]
def dfs(i):
    if i == len(ls):
        return True
    num = 0
    for j in range(i,len(ls)):
        num = num * 10 + ls[j]
        if sqrt(num) % 1 == 0 and num != 0:
            if dfs(j+1):
                return True
    return False
print('Yes' if dfs(0) else 'No')
```

测试输入 提交结果 历史提交

完美通过 查看题解

100% 数据通过测试

运行时长: 0 ms

收起面板 运行 提交

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。

这次的作业比较友好，基本就是照着模板来的，只是一些细节的地方还是得多加注意。

在矩阵最大权值路径中又一次需要拷贝，这同样是语法上的细节需要注意。

照着讲义上的题目一道一道练下来，对dfs和bfs的掌握更加熟练了。