

Assignment #9: dfs, bfs, & dp

Updated 2107 GMT+8 Nov 19, 2024

2024 fall, Compiled by 魏佳亮 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: dfs模版

代码:

```
t = int(input())
for _ in range(t):
    n,m = map(int,input().split())
    d = [(-1,-1),(-1,0),(-1,1),(0,-1),(0,1),(1,-1),(1,0),(1,1)]
    def solve(i,j):
        t = [[i,j]]
        ans = 0
        while t:
            x = t.pop()
            if l[x[0]][x[1]] == 'W':
                ans += 1
                for y in d:
                    t.append([x[0]+y[0],x[1]+y[1]])
                    l[x[0]][x[1]] = '.'
        return ans
    k = 0
    l = [['.' for _ in range(m+2)]]+[['.']+list(input())+'.'] for _ in range(n)]+[['.' for _ in range(m+2)]]
    for i in range(1,n+1):
        for j in range(1,m+1):
            if l[i][j] == 'W':
                k = max(k,solve(i,j))
    print(k)
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
t = int(input())
for _ in range(t):
    n,m = map(int,input().split())
    d = [(-1,-1), (-1,0), (-1,1), (0,-1), (0,1), (1,-1), (1,0), (1,1)]
    def solve(i,j):
        t = [[i,j]]
        ans = 0
        while t:
            x = t.pop()
            if l[x[0]][x[1]] == 'W':
                ans += 1
                for y in d:
                    t.append([x[0]+y[0],x[1]+y[1]])
            l[x[0]][x[1]] = '.'
        return ans
    k = 0
    l = [['.' for _ in range(m+2)]]+[['.']+list(input())+'.'] for _ in range(1,n+1)]
    for i in range(1,n+1):
        for j in range(1,m+1):
            if l[i][j] == 'W':
                k = max(k,solve(i,j))
    print(k)
```

基本信息

#: 47277976
题目: 18160
提交人: 2400011474
内存: 3792kB
时间: 115ms
语言: Python3
提交时间: 2024-11-20 10:53:12

©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：利用deque的popleft功能以及最短路径的不可重复性，保证了找到的第一条路径就是最短路径
代码：

```
from collections import deque
m,n = map(int,input().split())
l = [['2']*(n+2)]['2']+input().split()+'2' for _ in range(m)]['2']*(n+2)]
k = [[0 for _ in range(n+2)]] for _ in range(m+2)]
t = deque([[1,1]])
d = [(-1,0), (1,0), (0,-1), (0,1)]
def solve():
    if l[1][1] == '1':
        return 0
    while t:
        x = t.popleft()
        for y in d:
            if l[x[0]+y[0]][x[1]+y[1]] == '0':
                t.append([x[0]+y[0],x[1]+y[1]])
                k[x[0]+y[0]][x[1]+y[1]] = k[x[0]][x[1]]+1
            if l[x[0]+y[0]][x[1]+y[1]] == '1':
                return k[x[0]][x[1]]+1
        l[x[0]][x[1]] = '2'
    return 'NO'
print(solve())
```

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())
l = [['2']*(n+2)]+[['2']+input().split()+['2']] for _ in range(m)+[['2']]
k = [[0 for _ in range(n+2)] for _ in range(m+2)]
t = deque([[1,1]])
d = [(-1,0),(1,0),(0,-1),(0,1)]
def solve():
    if l[1][1] == '1':
        return 0
    while t:
        x = t.popleft()
        for y in d:
            if l[x[0]+y[0]][x[1]+y[1]] == '0':
                t.append([x[0]+y[0],x[1]+y[1]])
                k[x[0]+y[0]][x[1]+y[1]] = k[x[0]][x[1]]+1
            if l[x[0]+y[0]][x[1]+y[1]] == '1':
                return k[x[0]][x[1]]+1
        l[x[0]][x[1]] = '2'
    return 'NO'
print(solve())
```

基本信息

#: 47278870
题目: 19930
提交人: 2400011474
内存: 3696kB
时间: 29ms
语言: Python3
提交时间: 2024-11-20 11:41:48

04123: 马走日

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

思路: dfs模版

代码:

```
t = int(input())
for _ in range(t):
    n,m,x,y = map(int,input().split())
    d = [(-2,-1),(-2,1),(-1,-2),(-1,2),(2,-1),(2,1),(1,-2),(1,2)]
    l = [[False for _ in range(m)] for _ in range(n)]
    l[x][y] = True
    def solve(i,j,l,p):
        if p == m*n:
            return 1
        k = 0
        for x in d:
            if 0<=i+x[0]<n and 0<=j+x[1]<m and not l[i+x[0]][j+x[1]]:
                l[i+x[0]][j+x[1]] = True
                k += solve(i+x[0],j+x[1],l,p+1)
                l[i+x[0]][j+x[1]] = False
        return k
    print(solve(x,y,l,1))
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
t = int(input())
for _ in range(t):
    n,m,x,y = map(int,input().split())
    d = [(-2,-1), (-2,1), (-1,-2), (-1,2), (2,-1), (2,1), (1,-2), (1,2)]
    l = [[False for _ in range(m)] for _ in range(n)]
    l[x][y] = True
    def solve(i,j,l,p):
        if p == m*n:
            return 1
        k = 0
        for x in d:
            if 0<=i+x[0]<n and 0<=j+x[1]<m and not l[i+x[0]][j+x[1]]:
                l[i+x[0]][j+x[1]] = True
                k += solve(i+x[0],j+x[1],l,p+1)
                l[i+x[0]][j+x[1]] = False
        return k
    print(solve(x,y,l,1))
```

基本信息

#: 47279620

题目: 04123

提交人: 2400011474

内存: 3708kB

时间: 3356ms

语言: Python3

提交时间: 2024-11-20 13:02:58

sy316: 矩阵最大权值路径

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

思路: dfs改动一点, 把函数赋值改为操作

代码:

```
n,m = map(int,input().split())
l = [list(map(int,input().split())) for _ in range(n)]
d = [(-1,0), (1,0), (0,-1), (0,1)]
k = [[False for _ in range(m)] for _ in range(n)]
k[0][0] = True
t = [[0,0]]
ans = -float('inf')
u = []
def solve(i,j,k,p,q):
    global u,ans
    for y in d:
        if 0<=i+y[0]<n and 0<=j+y[1]<m and not k[i+y[0]][j+y[1]]:
            k[i+y[0]][j+y[1]] = True
            p.append([i+y[0]+1,j+y[1]+1])
            q += l[i+y[0]][j+y[1]]
            solve(i+y[0],j+y[1],k,p,q)
            k[i+y[0]][j+y[1]] = False
            del p[-1]
            q -= l[i+y[0]][j+y[1]]
    if i == n-1 and j == m-1:
        if q > ans:
            ans = q
            u = [x for x in p]
solve(0,0,k,[1,1],l[0][0])
for x in u:
    print(*x,sep=' ')
```

代码运行截图 (至少包含有"Accepted")

题目

题解

矩阵最大权值路径

通过数 806 提交数 2171 难度 中等 显示标签 ☆

题目描述

现有一个 $n * m$ 大小的矩阵，矩阵中的每个元素表示该位置的权值。现需要从矩阵左上角出发到达右下角，每次移动只能向上下左右移动一格（不允许移动到曾经经过的位置）。假设左上角坐标是(1, 1)，行数增加的方向为 x 增长的方向，列数增加的方向为 y 增长的方向。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 ($2 \leq n \leq 5, 2 \leq m \leq 5$)，分别表示矩阵的行数和列数；

接下来 n 行，每行 m 个整数 ($-100 \leq \text{整数} \leq 100$)，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

代码书写

Python

```
1 n,m = map(int,input().split())
2 l = [list(map(int,input().split())) for _ in range(n)]
3 d = [(-1,0),(1,0),(0,-1),(0,1)]
4 k = [[False for _ in range(m)] for _ in range(n)]
5 k[0][0] = True
6 t = [[0,0]]
7 ans = -float('inf')
8 u = []
9 def solve(i,j,k,p,q):
10     global u,ans
11     for y in d:
12         if 0<=i+y[0]<n and 0<=j+y[1]<m and not k[i+y[0]][j+y[1]]:
13             k[i+y[0]][j+y[1]] = True
14             p.append([i+y[0]+1,j+y[1]+1])
15             q += l[i+y[0]][j+y[1]]
16             solve(i+y[0],j+y[1],k,p,q)
17             k[i+y[0]][j+y[1]] = False
18             del p[-1]
19             q -= l[i+y[0]][j+y[1]]
```

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

LeetCode62.不同路径

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

思路：dp

代码：

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

代码运行截图 (至少包含有"Accepted")

题库

题目描述

通过

题解

提交记录

全部提交记录

通过

官方题解

写题解

提交于 2024.11.20 16

面向在校生的优惠升级方案

完成认证享 1 元/天升级 Plus 会员, ...

执行用时分布

0 ms | 击败 100.00%

复杂度分析

消耗内存分布

16.49 MB | 击败 41.62%

代码

Python3 智能模式

```
1 class Solution:
2     def uniquePaths(self, m: int, n: int) -> int:
3         l = [[1 for _ in range(m)] for _ in range(n)]
4         for i in range(1, n):
5             for j in range(1, m):
6                 l[i][j] = l[i-1][j] + l[i][j-1]
7         return l[n-1][m-1]
```

已存储

测试用例

测试结果

Case 1

Case 2

+

m =

sy358: 受到祝福的平方

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

思路: dp

代码:

```
from math import sqrt
t = input()
l = [False for _ in range(len(t))]
for i in range(len(t)-1):
    if sqrt(int(t[i:])).is_integer() and sqrt(int(t[i:])) != 0:
        l[i] = True
    else:
        for j in range(i+1, len(l)+1):
            if sqrt(int(t[i:j])).is_integer() and sqrt(int(t[i:j])) != 0 and l[j]:
                l[i] = True
                break
if l[0]:
    print('Yes')
else:
    print('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 155 提交数 334 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

代码书写

Python

```
1 from math import sqrt
2 t = input()
3 l = [False for _ in range(len(t))]
4 for i in range(len(t))[:-1]:
5     if sqrt(int(t[i:])).is_integer() and sqrt(int(t[i:])) !=
6         l[i] = True
7     else:
8         for j in range(i+1, len(l)+1):
9             if sqrt(int(t[i:j])).is_integer() and sqrt(int(t
10                 l[i] = True
11                 break
12 if l[0]:
13     print('Yes')
14 else:
15     print('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。

唉，期中过后反而更忙了，10月份刚考完toefl，12月份还有个重要的英语考试，我已经彻底放弃计概了。。。12月份考完再捡起来吧，希望到时候能捡回来，最近dfs和bfs代码量明显增加，都很复杂，也就是能跟上每日选做了（