1. (30min)

18160: 最大连通域面积

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

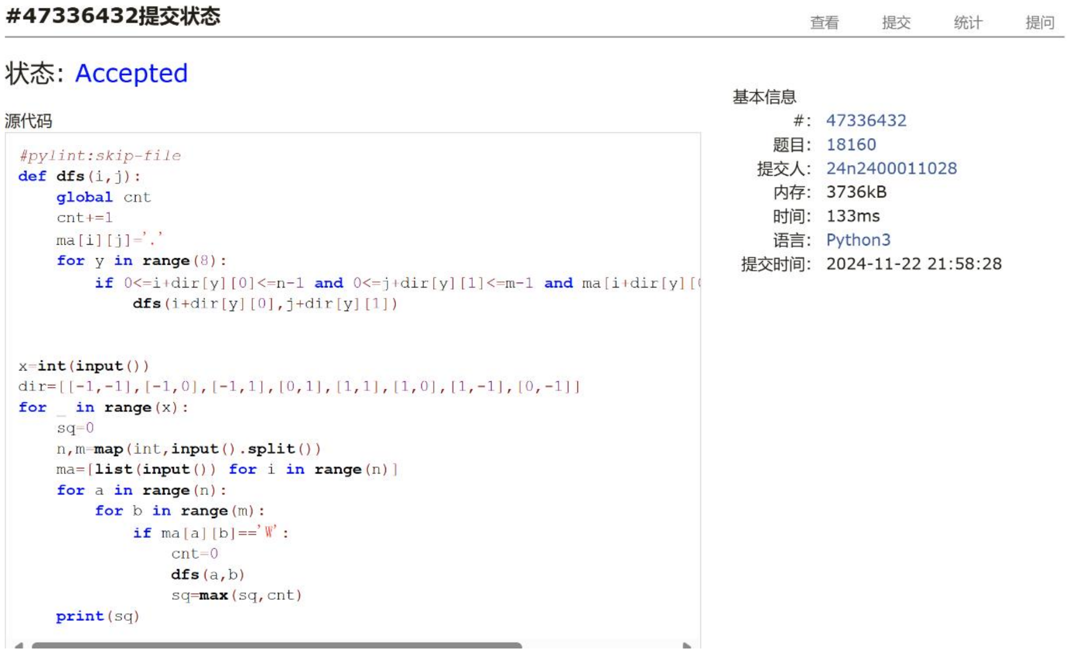
思路：dfs 模板，由于是统计连通域故不需要回溯

代码：

```python
#pylint:skip-file
def dfs(i,j):
    global cnt
    cnt+=1
    ma[i][j]='.'
    for y in range(8):
        if 0<=i+dir[y][0]<=n-1 and 0<=j+dir[y][1]<=m-1 and ma[i+dir[y][0]][j+dir[y][1]]=='W':
            dfs(i+dir[y][0],j+dir[y][1])


x=int(input())
dir=[[-1,-1],[-1,0],[-1,1],[0,1],[1,1],[1,0],[1,-1],[0,-1]]
for _ in range(x):
    sq=0
    n,m=map(int,input().split())
    ma=[list(input()) for i in range(n)]
    for a in range(n):
        for b in range(m):
            if ma[a][b]=='W':
                cnt=0
                dfs(a,b)
                sq=max(sq,cnt)
    print(sq)
```

运行：



#47336432提交状态

状态: Accepted

源代码

```
#pylint:skip-file
def dfs(i,j):
    global cnt
    cnt+=1
    ma[i][j]='.'
    for y in range(8):
        if 0<=i+dir[y][0]<=n-1 and 0<=j+dir[y][1]<=m-1 and ma[i+dir[y][
            dfs(i+dir[y][0],j+dir[y][1])


x=int(input())
dir=[[-1,-1],[-1,0],[-1,1],[0,1],[1,1],[1,0],[1,-1],[0,-1]]
for _ in range(x):
    sq=0
    n,m=map(int,input().split())
    ma=[list(input()) for i in range(n)]
    for a in range(n):
        for b in range(m):
            if ma[a][b]=='W':
                cnt=0
                dfs(a,b)
                sq=max(sq,cnt)
    print(sq)
```

基本信息

#:	47336432
题目:	18160
提交人:	24n2400011028
内存:	3736kB
时间:	133ms
语言:	Python3
提交时间:	2024-11-22 21:58:28

2. (30min)

19930: 寻宝

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

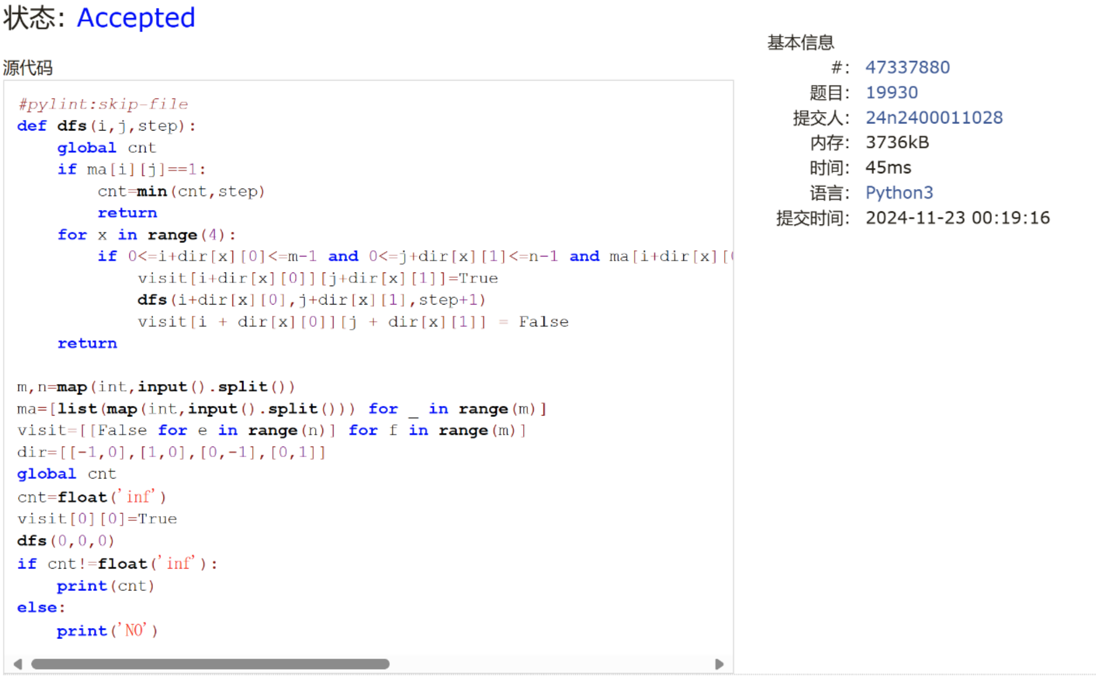
思路：模板，注意回溯

代码：

```
#pylint:skip-file
def dfs(i,j,step):
    global cnt
    if ma[i][j]==1:
        cnt=min(cnt,step)
        return
    for x in range(4):
        if 0<=i+dir[x][0]<=m-1 and
0<=j+dir[x][1]<=n-1 and
ma[i+dir[x][0]][j+dir[x][1]]!=2 and
visit[i+dir[x][0]][j+dir[x][1]]==False:
            visit[i+dir[x][0]][j+dir[x][1]]=True
            dfs(i+dir[x][0],j+dir[x][1],step+1)
            visit[i + dir[x][0]][j + dir[x][1]] =
False
    return
```

```
m,n=map(int,input().split())
ma=[list(map(int,input().split())) for _ in range(m)]
visit=[[False for e in range(n)] for f in range(m)]
dir=[[-1,0],[1,0],[0,-1],[0,1]]
global cnt
cnt=float('inf')
visit[0][0]=True
dfs(0,0,0)
if cnt!=float('inf'):
    print(cnt)
else:
    print('NO')
```

运行:



状态: Accepted

源代码

```
#pylint:skip-file
def dfs(i,j,step):
    global cnt
    if ma[i][j]==1:
        cnt=min(cnt,step)
        return
    for x in range(4):
        if 0<=i+dir[x][0]<=m-1 and 0<=j+dir[x][1]<=n-1 and ma[i+dir[x][
            visit[i+dir[x][0]][j+dir[x][1]]=True
            dfs(i+dir[x][0],j+dir[x][1],step+1)
            visit[i + dir[x][0]][j + dir[x][1]] = False
    return

m,n=map(int,input().split())
ma=[list(map(int,input().split())) for _ in range(m)]
visit=[[False for e in range(n)] for f in range(m)]
dir=[[-1,0],[1,0],[0,-1],[0,1]]
global cnt
cnt=float('inf')
visit[0][0]=True
dfs(0,0,0)
if cnt!=float('inf'):
    print(cnt)
else:
    print('NO')
```

基本信息

#:	47337880
题目:	19930
提交人:	24n2400011028
内存:	3736kB
时间:	45ms
语言:	Python3
提交时间:	2024-11-23 00:19:16

3. (30min)

04123: 马走日

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

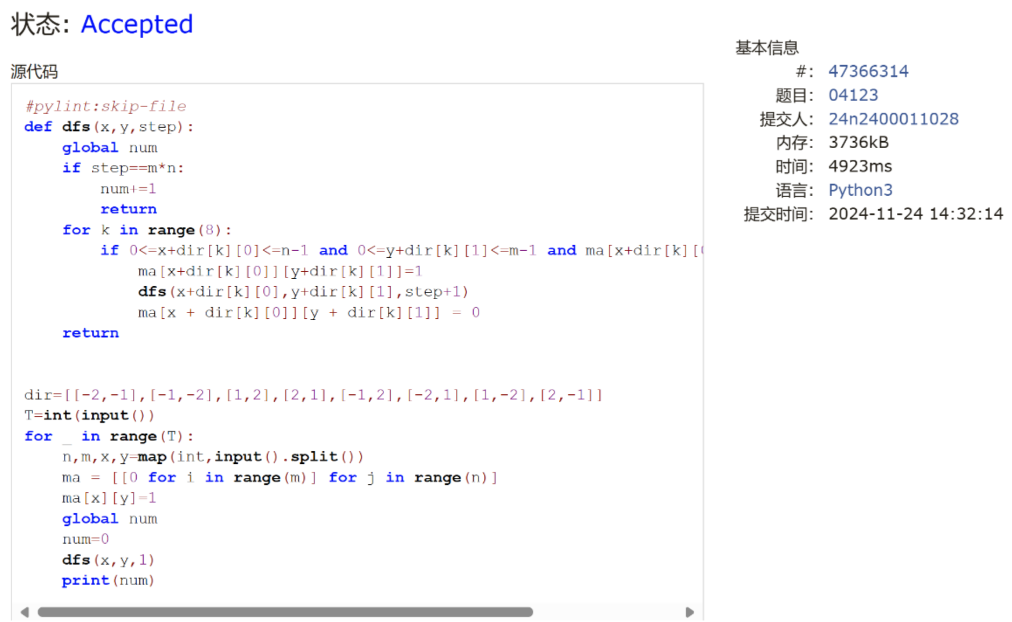
思路：仍然是套模板，用 step 记录走过的格子数，step=m*n 代表遍历了

代码：

```
#pylint:skip-file
def dfs(x,y,step):
    global num
    if step==m*n:
        num+=1
        return
    for k in range(8):
        if 0<=x+dir[k][0]<=n-1 and
0<=y+dir[k][1]<=m-1 and
ma[x+dir[k][0]][y+dir[k][1]]==0:
            ma[x+dir[k][0]][y+dir[k][1]]=1
            dfs(x+dir[k][0],y+dir[k][1],step+1)
            ma[x + dir[k][0]][y + dir[k][1]] = 0
    return


dir=[[-2,-1],[-1,-2],[1,2],[2,1],[-1,2],[-2,1],[1,-
2],[2,-1]]
T=int(input())
for _ in range(T):
    n,m,x,y=map(int,input().split())
    ma = [[0 for i in range(m)] for j in range(n)]
    ma[x][y]=1
    global num
    num=0
    dfs(x,y,1)
    print(num)
```

运行：



状态: Accepted

源代码

```
#pylint:skip-file
def dfs(x,y,step):
    global num
    if step==m*n:
        num+=1
        return
    for k in range(8):
        if 0<=x+dir[k][0]<=n-1 and 0<=y+dir[k][1]<=m-1 and ma[x+dir[k][
            ma[x+dir[k][0]][y+dir[k][1]]=1
            dfs(x+dir[k][0],y+dir[k][1],step+1)
            ma[x + dir[k][0]][y + dir[k][1]] = 0
    return


dir=[[-2,-1],[-1,-2],[1,2],[2,1],[-1,2],[-2,1],[1,-2],[2,-1]]
T=int(input())
for _ in range(T):
    n,m,x,y=map(int,input().split())
    ma = [[0 for i in range(m)] for j in range(n)]
    ma[x][y]=1
    global num
    num=0
    dfs(x,y,1)
    print(num)
```

基本信息

#:	47366314
题目:	04123
提交人:	24n2400011028
内存:	3736kB
时间:	4923ms
语言:	Python3
提交时间:	2024-11-24 14:32:14

4. (40min)

sy316: 矩阵最大权值路径

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

思路: dfs 模板

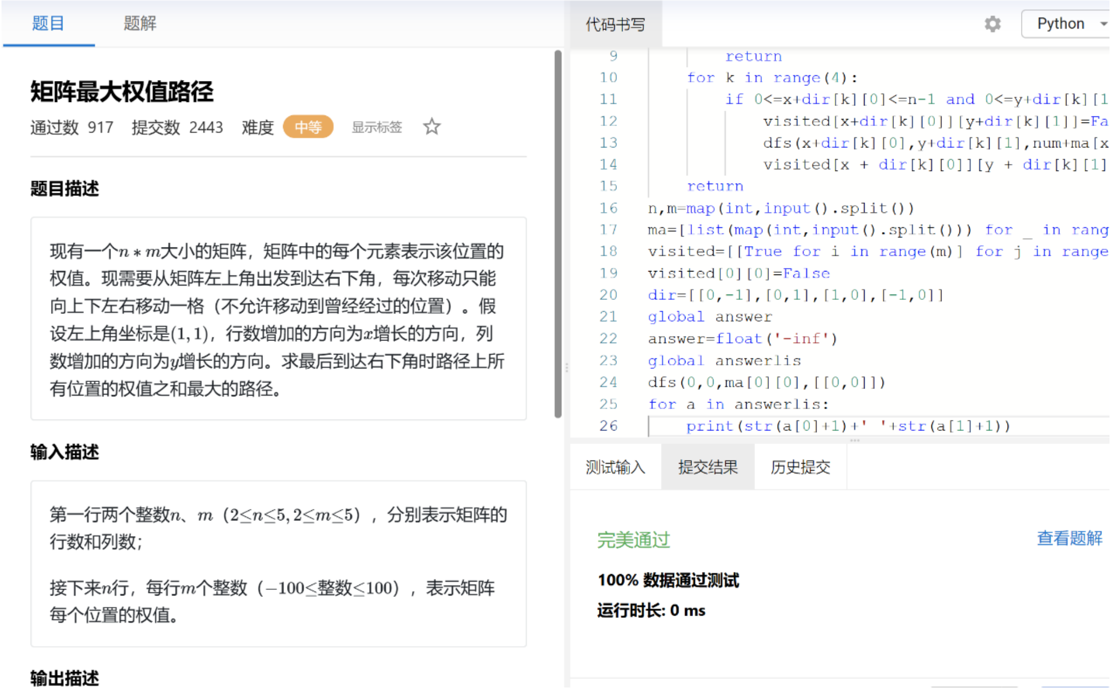
代码:

```
#pylint:skip-file
def dfs(x,y,num,lis):
    global answer
    global answerlis
    if x==n-1 and y==m-1:
        if num>answer:
            answer=num
            answerlis=lis[:]
        return
    for k in range(4):
        if 0<=x+dir[k][0]<=n-1 and
0<=y+dir[k][1]<=m-1 and
visited[x+dir[k][0]][y+dir[k][1]]:
            visited[x+dir[k][0]][y+dir[k][1]]=False

dfs(x+dir[k][0],y+dir[k][1],num+ma[x+dir[k][0]][y+d
ir[k][1]],lis+[[x+dir[k][0],y+dir[k][1]]])
```

```
                visited[x + dir[k][0]][y + dir[k][1]] =
True
     return
n,m=map(int,input().split())
ma=[list(map(int,input().split())) for _ in
range(n)]
visited=[[True for i in range(m)] for j in
range(n)]
visited[0][0]=False
dir=[[0,-1],[0,1],[1,0],[-1,0]]
global answer
answer=float('-inf')
global answerlis
dfs(0,0,ma[0][0],[[0,0]])
for a in answerlis:
     print(str(a[0]+1)+' '+str(a[1]+1))
```

运行：



矩阵最大权值路径

通过数 917　提交数 2443　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（$2\le n\le 5, 2\le m\le 5$），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

```
        return
    for k in range(4):
        if 0<=x+dir[k][0]<=n-1 and 0<=y+dir[k][1
            visited[x+dir[k][0]][y+dir[k][1]]=Fa
            dfs(x+dir[k][0],y+dir[k][1],num+ma[x
            visited[x + dir[k][0]][y + dir[k][1]
    return
n,m=map(int,input().split())
ma=[list(map(int,input().split())) for _ in rang
visited=[[True for i in range(m)] for j in range
visited[0][0]=False
dir=[[0,-1],[0,1],[1,0],[-1,0]]
global answer
answer=float('-inf')
global answerlis
dfs(0,0,ma[0][0],[[0,0]])
for a in answerlis:
    print(str(a[0]+1)+' '+str(a[1]+1))
```

完美通过

100% 数据通过测试

运行时长: 0 ms

5. (20min)

LeetCode62.不同路径

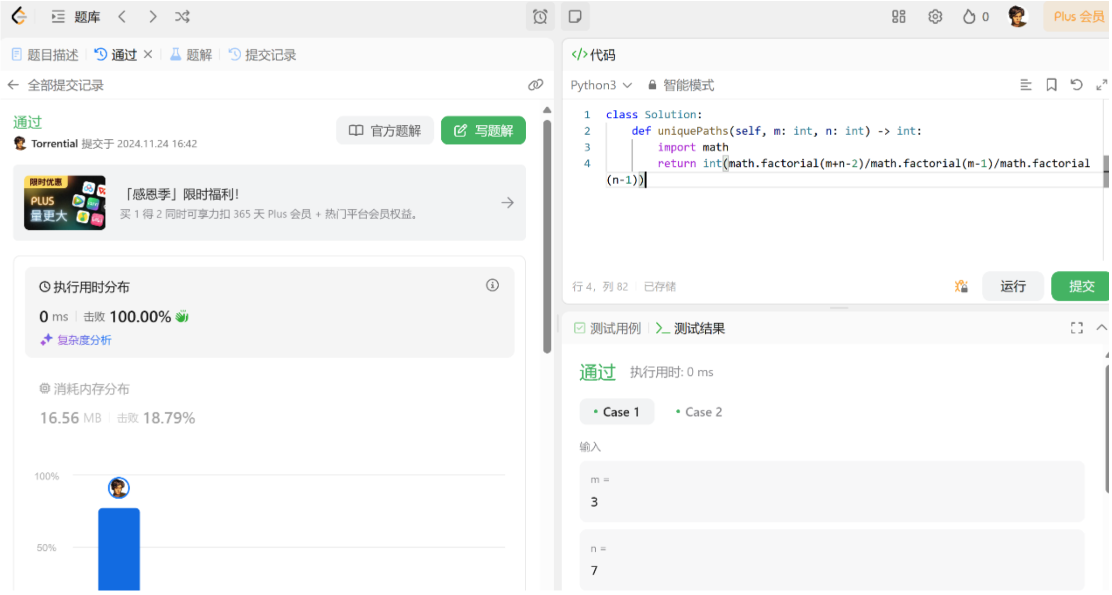
dp, https://leetcode.cn/problems/unique-paths/

思路：显然答案为 C（m-1,m+n-2）

代码：

```
class Solution:
    def uniquePaths(self, m: int, n: int) -> int:
        import math
        return int(math.factorial(m+n-2)/math.factorial(m-1)/math.factorial(n-1))
```

运行：



通过

Torrential 提交于 2024.11.24 16:42

执行用时分布

0 ms | 击败 100.00%

消耗内存分布

16.56 MB | 击败 18.79%

```
class Solution:
    def uniquePaths(self, m: int, n: int) -> int:
        import math
        return int(math.factorial(m+n-2)/math.factorial(m-1)/math.factorial(n-1))
```

行 4，列 82 已存储

通过 执行用时: 0 ms

• Case 1 • Case 2

输入

m =
3

n =
7

6. (30min)

sy358：受到祝福的平方

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

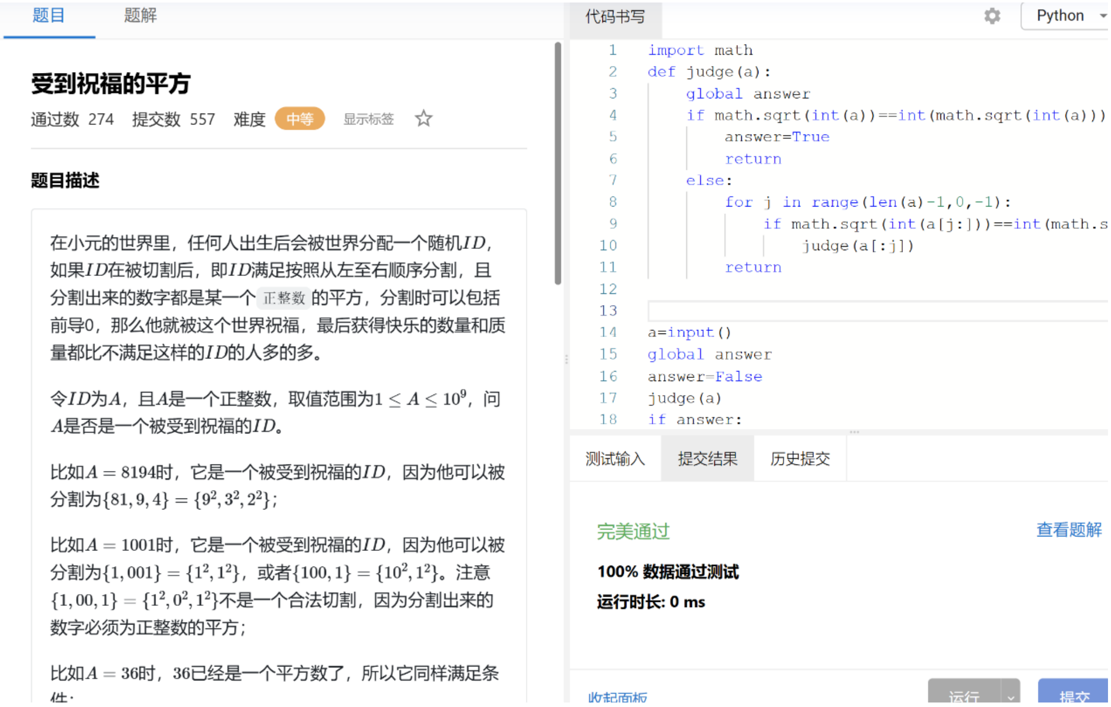
思路：动态规划，用递归，注意考虑有 0 的情况

代码：

```
import math
def judge(a):
    global answer
    if math.sqrt(int(a))==int(math.sqrt(int(a))):
        answer=True
        return
    else:
        for j in range(len(a)-1,0,-1):
            if
math.sqrt(int(a[j:]))==int(math.sqrt(int(a[j:])))
and int(a[j:])>0:
                judge(a[:j])
        return

a=input()
global answer
answer=False
judge(a)
if answer:
    print('Yes')
else:
    print('No')
```

运行：



受到祝福的平方

通过数 274 提交数 557 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$1 \le A \le 10^9$，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；

```
import math
def judge(a):
    global answer
    if math.sqrt(int(a))==int(math.sqrt(int(a)))
        answer=True
        return
    else:
        for j in range(len(a)-1,0,-1):
            if math.sqrt(int(a[j:]))==int(math.s
                judge(a[:j])
        return


a=input()
global answer
answer=False
judge(a)
if answer:
```

完美通过

100% 数据通过测试

运行时长: 0 ms

总结和收获：

1. 学习了 dfs 模板、线段树等内容
2. 在程序调试中对浅拷贝有了更深的理解，比如复制列表时 lis1=lis 就会造成浅拷贝的问题，lis1=lis[:]就不是浅拷贝