

题目描述:

地上共有 N 个格子, 你需要跳完地上所有的格子, 但是格子间是有强依赖关系的, 跳完前一个格子后, 后续的格子才会被开启, 格子间的依赖关系由多组 `steps` 数组给出, `steps[0]` 表示前一个格子, `steps[1]` 表示 `steps[0]` 可以开启的格子:

比如 `[0,1]` 表示从跳完第 0 个格子以后第 1 个格子就开启了, 比如 `[2,1]`, `[2,3]` 表示跳完第 2 个格子后第 1 个格子和第 3 个格子就被开启了

请你计算是否能由给出的 `steps` 数组跳完所有的格子, 如果可以输出 `yes`, 否则输出 `no`

说明:

1. 你可以从一个格子跳到任意一个开启的格子
2. 没有前置依赖条件的格子默认就是开启的
3. 如果总数是 N , 则所有的格子编号为 `[0,1,2,3....N-1]` 连续的数组

输入描述:

输入一个整数 N 表示总共有多少个格子, 接着输入多组二维数组 `steps` 表示所有格子之间的依赖关系

输出描述:

如果能按照 `steps` 给定的依赖顺序跳完所有的格子输出 `yes`

否则输出 `no`

补充说明:

$1 \leq N < 500$

`steps[i].length=2`

$0 \leq \text{step}[i][0], \text{step}[i][1] < N$

示例 1

输入:

3

0 1

0 2

输出:

yes

说明:

总共有三个格子 `[0,1,2]`, 跳完 0 个格子后第 1 个格子就开启了, 跳到第 0 个格子后第 2 个格子也被开启了, 按照 `0->1->2` 或者 `0->2->1` 的顺序都可以跳完所有的格子

示例 2

输入:

2

1 0

0 1

输出:

no

说明:

总共有 2 个格子, 第 1 个格子可以开启第 0 格子, 但是第 1 个格子又需要第 0 个格子才能开启, 相互依赖, 因此无法完成

示例 3

输入：

6
0 1
0 2
0 3
0 4
0 5

输出：

yes

说明：

总共有 6 个格子，第 0 个格子可以开启第 1,2,3,4,5 个格子，所以跳完第 0 个格子之后其他格子都被开启了，之后按任何顺序可以跳完剩余的格子

示例 4

输入：

5
4 3
0 4
2 1
3 2

输出：

yes

说明：

跳完第 0 个格子可以开启格子 4，跳完格子 4 可以开启格子 3，跳完格子 3 可以开启格子 2，跳完格子 2 可以开启格子 1，按照 0->4->3->2->1 这样就跳完所有的格子

示例 5

输入：

4
1 2
1 0

输出：

yes

说明：

总共 4 个格子[0,1,2,3]，格子 1 和格子 3 没有前置条件所以默认开启，格子 1 可以开启格子 0 和格子 2，所以跳到格子 1 之后就可以开启所有的格子，因此可以跳完所有格子

```
import sys
```

```
n = int(sys.stdin.readline().strip())
```

```
allblocks = [[i, False] for i in range(n)]
```

```
l = []
```

```
for line in sys.stdin:
```

```
    l.append(list(map(int, line.strip().split())))
```

```
opened = [i for i in range(n)]
```

```
for i in l:
```

```
    if i[1] in opened:
        opened.remove(i[1])

def main():
    if opened == []:
        print('no')
        return
    changed = True
    while changed:
        changed = False
        for i in l:
            if i[0] in opened and i[1] not in opened:
                opened.append(i[1])
                changed = True
    opened.sort()
    if opened == [i for i in range(n)]:
        print('yes')
    else:
        print('no')
main()
```