Linux发行版的数量

知识点DFS搜索 BFS 搜索并查集
时间限制：1s 空间限制：256MB 限定语言：不限

题目描述：

Linux操作系统有多个发行版，distrowatch.com提供了各个发行版的资料。这些发行版互相存在关联，例如Ubuntu基于 Debian 开发，而Mint又基于Ubuntu开发，那么我们认为Mint同Debian也存在关联。

发行版集是一个或多个相关存在关联的操作系统发行版，集合内不包含没有关联的发行版。

给你一个 n x n 的矩阵 isConnected ，其中 isConnected[i][j] = 1 表示第 i 个发行版和第 j 个发行版直接关联，而 isConnected[i][j] = 0 表示二者不直接相连。

返回最大的发行版集中发行版的数量

输入描述：

第一行输入发行版的总数量N，之后每行表示各发行版间是否直接相关

输出描述：

输出最大的发行版集中发行版的数量

补充说明：

1 <= N <= 200

示例1

输入：

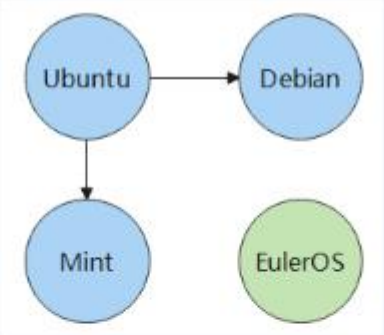
```
4
1 1 0 0
1 1 1 0
0 1 1 0
0 0 0 1
```

输出：

```
3
```

说明：

Debian(1)和Ubuntu(2)相关，Mint(3)和Ubuntu(2)相关，EeulerOS(4)和另外三个都不相关，所以存在两个发行版集，发行版集中发行版的数量分别是3和1，所以输出3



解题思路：

通过回溯求出所有相关联的版本，放在set集合中。输出set长度的最大值。

```
public class Main{

    public static int n;
    public static int[][] ints;
    public static Set<Integer> set;

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        n = sc.nextInt();
```

```
        ints = new int[n][n];
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                ints[i][j] = sc.nextInt();
            }
        }

        Set<Integer> temp = new HashSet<>();       //已经关联的 linux 版本集合
        int max = 0;
        for(int i=0; i<n; i++){
            if(!temp.contains(i)){    //已经关联过的 linux 版本不需要再处理
                set = new HashSet<>();
                handle(i);
                max = Math.max( max, set.size());     //set 的大小代表发行版集中发行版的
数量
                temp.addAll(set);     //处理过的 linux 加入已关联集合
            }
        }

        System.out.println(max);
    }

    /**
     * 找出所有与 linux 相关的版本
     * @param linux
     */
    public static void handle(int linux){

        for(int i=0; i<n; i++){
            if(!set.contains(i) && ints[linux][i] == 1){       //已经关联的版本无需处理
                set.add(i);         //添加到已关联的版本
                handle( i);
            }
        }
    }

}
```