目 录

1.员工的重要性员工的重要性

给定一个保存员工信息的数据结构，它包含了员工唯一的 id，重要度　和 直系下属的 id。比如，员工 1 是员工 2 的领导，员工 2 是员工 3 的领导。他们相应的重要度为 15, 10, 5。那么员工 1 的数据结构是 [1, 15, [2]]，员工 2 的数据结构是 [2, 10, [3]]，员工 3 的数据结构是 [3, 5, []]。注意虽然员工 3 也是员工 1 的一个下属，但是由于并不是直系下属，因此没有体现在员工 1 的数据结构中。

现在输入一个公司的所有员工信息，以及单个员工 id，返回这个员工和他所有下属的重要度之和。

```
示例 1:

输入: [[1, 5, [2, 3]], [2, 3, []], [3, 3, []]], 1
输出: 11
解释:
员工1自身的重要度是5，他有两个直系下属2和3，而且2和3的重要度均为3。因此员工1的总重要度是 5 + 3 + 3 = 11。
注意:
```

一个员工最多有一个直系领导，但是可以有多个直系下属, 员工数量不超过 2000。

Tips

1. 遍历整个员工列表 employees，找到符合 id 的员工 employee
2. 这个员工 employee 如果没有下属 employee.subordinates.size()==0，重要度就是自己的重要度 employee.importance。
3. 如果这个员工有下属，算出每个下属及下属的重要度。

```
/*
// Definition for Employee.
class Employee {
    public int id;
    public int importance;
    public List<Integer> subordinates;
};
*/

class Solution {
    public int getImportance(List<Employee> employees, int id) {

        for(int i = 0;i < employees.size();i++){
            Employee employee = employees.get(i);
            if(employee.id == id){
                if(employee.subordinates.size() == 0){//没有下属
                    return employee.importance;
                }
                for(int j = 0; j < employee.subordinates.size(); j++){
                    employee.importance += getImportance(employees,employee.
                        subordinates.get(j));
                }
                return employee.importance;
            }
        }
        return 0;
    }
}
```