```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.HashSet;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import java.util.Queue;
import java.util.Set;

class Employee {
	List<Employee> children = new ArrayList<>();
	int id;
	public Employee(int id) {
		this.id = id;
	}
}

public class LCA_KnaryTree {
	public static Employee findLCA (Employee ceo, Employee e1, Employee e2) {
		if (e1 == ceo || e2 == ceo) return null;
		if (e1 == e2) return e1;
		Map<Employee, Employee> map = new HashMap<>();

		boolean b1 = false;
		boolean b2 = false;
		Queue<Employee> queue = new LinkedList<>();
		queue.offer(ceo);
		while (!queue.isEmpty()) {
			int size = queue.size();
			for (int i = 0; i < size; i++) {
				Employee e = queue.poll();
				if (e == e1) {
					b1 = true;
				}

				if (e == e2) {
					b2 = true;
				}

				for (Employee ec : e.children) {
					map.put(ec, e);
					queue.offer(ec);
				}
			}
		}

		if (!b1 || !b2) return null;

		Set<Employee> set = new HashSet<>();

		while (e1 != ceo) {
			set.add(e1);
			e1 = map.get(e1);
		}
```

```
                while (e2 != ceo) {
                        if (set.contains(e2)) return e2;
                        e2 = map.get(e2);
                }
                return null;
        }
}
```