```java
public class L46 {
    public List<List<Integer>> permute(int[] nums) {
        List<List<Integer>> res = new ArrayList<List<Integer>>();

        if(nums == null || nums.length == 0)
            return res;
        List<Integer> tmp = new ArrayList<>();
        dfs(nums, res, tmp);

        return res;

    }

    public void dfs(int [] nums, List<List<Integer>> res, List<Integer> tmp) {
        if(tmp.size() == nums.length) {
            res.add(new ArrayList<>(tmp));//记住这个，一定不能是res.add(tmp)
            return ;
        }
        //这是个递归的问题，当tmp里面不包含这个数字的时候就添加进去
       for(int i = 0; i < nums.length; i ++) {
            if(!tmp.contains(new Integer(nums[i]))) {
                tmp.add(new Integer(nums[i]));
                dfs(nums, res, tmp);
                tmp.remove(new Integer(nums[i]));
            }
       }
    }

    public static void main(String [] args) {
        int [] nums = new int [] {1, 2, 3};
        new L46().permute(nums);
    }
}
```

46. Permutations

Description | Hints | Submissions | Discuss | Solution

Pick One

Given a collection of **distinct** integers, return all possible permutations.

Example:

```
Input: [1,2,3]
Output:
[
  [1,2,3],
  [1,3,2],
  [2,1,3],
  [2,3,1],
  [3,1,2],
  [3,2,1]
]
```

Seen this question in a real interview before? Yes No