硬件产品销售方案

知识点递归数组DFS搜索 回溯
时间限制：1s 空间限制：256MB 限定语言：不限

题目描述：

某公司目前推出了AI开发者套件、AI加速卡、AI加速模块、AI服务器、智能边缘多种硬件产品，每种产品包含若干个型号。现某合作厂商要采购金额为amount元的硬件产品搭建自己的AI基座。假设当前库存有N种产品，每种产品的库存量充足，给定每种产品的价格，记为price（不存在价格相同的产品型号）。请为合作厂商列出所有可能的产品组合。

输入描述：

输入包含采购金额amount和产品价格列表price。第一行为amount，第二行为price。例如：
500
[100, 200, 300, 500]

输出描述：

输出为组合列表。例如：
[[500], [200, 300], [100, 200, 200], [100, 100, 300], [100, 100, 100, 200], [100, 100, 100, 100, 100]]

补充说明：

1. 对于给定输入，产品组合少于150种。输出的组合为一个数组，数组的每个元素也是一个数组，表示一种组合方案。如果给定产品无法组合金额为amount元的方案，那么返回空列表。
2. 两种组合方案，只要存在一种产品的数量不同，那么方案认为是不同的。
3. 每种产品型号价格不相同
4. 1 <= 产品类型数量 <= 30
5. 100 <= 产品价格 <= 20000
6. 100 <= 采购金额 <= 50000

示例1

输入：

500
[100, 200, 300, 500, 500]

输出：

[[100, 100, 100, 100, 100], [100, 100, 100, 200], [100, 100, 300], [100, 200, 200], [200, 300], [500], [500]]

示例2

输入：

100
[100]

输出：

[[100]]

解题思路：

通过回溯求出所有可能的购买情况。
看例题，只统计等于预算的情况，而且集合没有排序的要求。

```
public class Main{

    public static int[] price;   //物品价格数组
    public static int amount;     //预算
    public static List<List<Integer>> resList = new ArrayList<>();

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        amount = sc.nextInt();
        sc.nextLine();
        String[] strings = sc.nextLine().replace("[","")
                                                .replace("]","")
                                                .split(",");


        price = new int[strings.length];
        for(int i=0; i<strings.length; i++){
            price[i] = Integer.valueOf(strings[i].trim());
```

```
        }

        handle(0, 0, new ArrayList<>());

        System.out.println(resList);

    }

    /**
     * 通过递归求出所有的购买情况
     * @param index         物品价格索引
     * @param count         购买物品总价格
     * @param list          购买物品集合
     */
    public static void handle(int index, int count, List<Integer> list){

        if(amount <= count){       //物品总价格大于等于预算总退出
            List<Integer> tempList = new ArrayList<>(); //需要使用 tempList，否则会影响入参中的 list
            tempList.addAll(list);
            if(amount == count){       //只统计等于预算的情况
                resList.add(tempList);
            }
        }else {
            for(int i=index; i<price.length; i++){

                list.add(price[i]);
                handle( i, count + price[i], list);
                list.remove(list.size()-1);
            }
        }
    }

}
```