组装新的数组

知识点 回溯 数组
时间限制：1s 空间限制：256MB 限定语言：不限

题目描述：

给你一个整数M和数组N,N中的元素为连续整数，要求根据N中的元素组装成新的数组R，组装规则：

1.R中元素总和加起来等于M

2.R中的元素可以从N中重复选取

3.R中的元素最多只能有1个不在N中，且比N中的数字都要小（不能为负数）

请输出：数组R一共有多少组装办法

输入描述：

第一行输入是连续数组N，采用空格分隔
第二行输入数字M

输出描述：

输出的是组装办法数量，int类型

补充说明：

1 <= N.length <= 30
1 <= N.length <= 1000

示例1

输入：

2
5

输出：

1

说明：

只有1种组装办法，就是[2,2,1]

示例2

输入：

2 3
5

输出：

2

说明：

一共2种组装办法，分别是[2,2,1],[2,3]

解题思路：

因为数组是连续的，所以先求出最小值（第一个数）和最大值（最后一个数）
通过回溯法遍历出所有可能性。

```
public class Main{

    public static int max;
    public static int min;
    public static int res;

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        String[] strings = sc.nextLine().split(" ");
        int M = sc.nextInt();

        min = Integer.valueOf(strings[0]);          //数组中的最小数
```

```
            max = Integer.valueOf(strings[strings.length-1]);     //数组中的最大数
            handle( min, M);

            System.out.println(res);
        }

        /**
         * 3 4 5
         * 10
         * 5 5
         * 4 5 1
         * 4 4 2
         * 3 5 2
         * 3 3 4
         * 3 3 3 1
         * @param n         数组中的数字
         * @param sum      M 减去数组中的数字的差值
         */
        public static void handle(int n, int sum){

            if(sum < min){   //剩下的小于最小数就可以组装（符合第三个规则）
                res++;
                return;
            }

            if(n > max){       //大于最大数则返回
                return;
            }

            int i = 0;
            while (true){

                handle(n + 1, sum - i*n);     //因为数组中的数字是连续的，所以只需要+1

                i++;
                if(sum - i*n <= 0){
                    if(sum - i*n == 0){         //完美符合
                        res++;
                    }
                    break;
                }
            }
        }
    }
```