Java方向编程题答案

day29

[编程题]772-年会抽奖

链接：https://www.nowcoder.com/questionTerminal/610e6c0387a0401fb96675f58cda8559

【题目解析】

无

【解题思路】

该题目是计算中奖率，题目描述中抽奖是无放回的，那么问题就可以转换为排列组合问题。

- 抽奖可能出现的情况作为分母：F
  - n = 1 出现 1 种情况；
  - n = 2 出现 2 种情况；
  - n = 3 出现 6 种情况；
  - n 出现 n! 种情况
- 每个人不可能抽中的情况作为分子：C
  - 假设每个人都中奖，可以认为 n 个元素都在中奖的位置
  - 如果每个人都不中奖，那么只要确保n个元素不在原有中奖位置即可，那么问题转化为如何错排n个元素。
  - 参见错排算法：https://baike.baidu.com/item/错排公式/10978508
- 无法中奖的概率即为：C / F * 100 %

【示例代码】

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        // 输入
        Scanner scan = new Scanner(System.in);
        while(scan.hasNext()){
            int n = scan.nextInt(); // 代表测试数据的组数
            float sum1 = factorial(n);
            float sum2 = count(n);
            //将得到的分子分母进行相除，就可以得到概率了。
            float result1 = (sum2/sum1)*100;
            System.out.println(String.format("%.2f", result1) + "%");
        }

    }
    /**
     * 错排算法
     * @param n
```

```
     * @return
     */
    public static float count(int n) {
        if(n==1){
            return 0;
        }else if(n==2){
            return 1;
        }else{
            return (n-1)*(count(n-1)+count(n-2));
        }

    }
    /**
     * n的阶乘
     * @param num
     * @return
     */
     public static float factorial(int num) {
            float result = 1;
            if(num==0){
                return 1;
            }else if (num > 0) {
                result = num * factorial(num - 1);
            }
            return result;
        }
}
```

[编程题]58542-数字和为sum的方法数

链接：https://www.nowcoder.com/questionTerminal/7f24eb7266ce4b0792ce8721d6259800

【题目解析】

无

【解题思路】

给定一个有n个正整数的数组A和一个整数sum,求选择数组A中部分数字和为sum的方案数。当两种选取方案有一个数字的下标不一样,我们就认为是不同的组成方案。 采用动态规划算法：

```
        dp[i][j]              :代表用前i个数字凑到j最多方案
        dp[i][j] = dp[i-1][j] : 不用第i个数字凑到j的最多方案
        dp[i][j] + = dp[i-1][j-value[i]] : 用第i个数字，只需要看原来凑到j-value[j]的最多方案，并累加
        dp[0] = 1 : 初始化，表示凑到0永远有1中方案
```

【示例代码】

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

```
        int n = scanner.nextInt();// 数组长度为n表示n个数字
        int sum = scanner.nextInt();// 部分元素求和sum
        int[] value = new int[n];//初始化数组
        long[] dp = new long[sum + 1];//动态规划数组
        dp[0] = 1;//index=0的初始化值
        for (int i = 0; i < n; i++) {
            value[i] = scanner.nextInt();
            for (int j = sum; j >= 0; j--) {
                if (j >= value[i]) {
                    dp[j] += dp[j - value[i]];
                }
            }
        }
        System.out.println(dp[sum]);
    }
}
```