1. 双倍麻烦

（trouble.cpp/.c）

Time Limit：1 sec， Memory Limit：128 MB

【题目描述】

Hunter 总是对寻找奇怪的数有强烈的欲望，当他处于麻烦中时这种欲望尤为强烈。Hunter 寻找的数有以下性质。如果你将该数右旋（将最后一位数字放到整个数之前），得到的数是原数的两倍。这样的数称为双倍麻烦数（Double Trouble Number）。例如，X = 421052631578947368 是一个双倍麻烦数，因为双倍的 X，2X = 842105263157894736，正好是将 X 右旋得到的。

X 是一个十进制数系统中的双倍麻烦数。任何 p（p ≥ 2）进制数系统中也有很多这样的双倍麻烦数。例如，在二进制数（p = 2）系统中，我们有双倍麻烦数 01 和 0101。注意数前的 0 有必要保留，使在右旋后获得正确的数。

Hunter 似乎对任何进制数的系统中最小的双倍麻烦数感兴趣。例如，在二进制数系统中最小的双倍麻烦数是 01。在十进制数系统中最小的双倍麻烦数是 052631578947368421。现在 Hunter 请你写一个程序帮助他对给定的 p，找出 p 进制数系统中最小的双倍麻烦数。

【输入格式】（trouble.in）

输入 p（2 ≤ p ≤ 200）。

【输出格式】（trouble.out）

输出 p 进制数系统中最小的双倍麻烦数，输出时相邻数字之间以空格分隔。

【输入与输出样例】

输入	输出
2	0 1
10	0 5 2 6 3 1 5 7 8 9 4 7 3 6 8 4 2 1
35	11 23

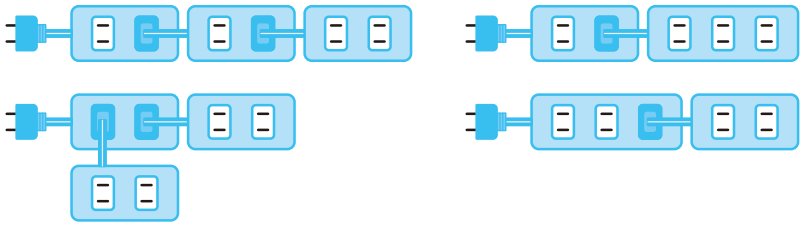
2．插线板
(tap.cpp/c/pas)

Time Limit：1 sec， Memory Limit：128 MB

【问题描述】

对工程师而言，确保电源是最重要的事情。不仅是PC，当智能手机、平板电脑、数码相机等电量不足时，我们也肯定要四处寻找插座。不过，多人共用的时候就必须共享插座，这时插线板就会派上用场。一般的插线板除了有延长线，还会有多个插口。

这里假设有双插口和三插口的插线板。墙壁上只有 1 个插座能用，而需要用电的电器有 n 台，试考虑此时应如何分配插线板。举个例子，当 $n = 4$ 时，如下图所示，有 4 种插线板插线方法（使用同一个插线板时，不考虑插口位置，只考虑插线板的连接方法。另外，要使插线板上最后没有多余的插口）。



$n=4$时

编程求连接 n 个电器，插线板的插线方法有多少种（不考虑电源的功率问题）？

【输入格式】 （tap.in）

输入只有一行，一个整数 n。

【输出格式】 （tap.out）

输出一个整数，即插线方案总数。

【样例输入】

4

【样例输出】

4

【数据范围】

100%的数据保证 1≤n≤42。

3. 骨牌上的点数

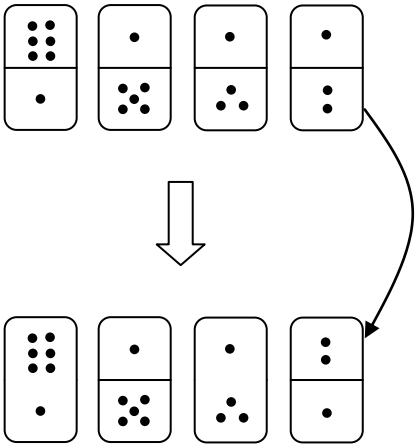
（bones.cpp/c）

Time Limit：1 sec， Memory Limit：128 MB

【题目描述】

有一种 Domino 骨牌是平面的，其正面被分成上下两部分，每一部分的表面或者为空，或者被标上 1 至 6 个点。现有一行骨牌排列在桌面上：

骨牌上部的点数之和为 6+1+1+1=9；骨牌下部点数之和为 1+5+3+2=11。上部和下部的差值是 2。这个差值是两行点数之差的绝对值。每个骨牌都可以上下倒置转换，即上部变为下部，下部变为上部。



现在的任务是，以最少的翻转次数，使得上部和下部之间的差值最小。对于上面这个例子，我们只需要翻转最后一个骨牌，就可以使得上部和下部的差值为 0，所以该例子的答案为 1 次。

【输入格式】（bones.in）

输入的第一行为骨牌数 N ($1 \leqslant N \leqslant 100$)。第二行有 N 个数，为开始时每个骨牌上部的点数。第三行也有 N 个数，为开始时每个骨牌下部的点数。第二行和第三行的数相互对应。

【输出格式】（bones.out）

对于给定的输入，输出通过翻转得到的上部点数和下部点数差值的最小值，以及相应的最少翻转次数。

【输入与输出样例】

输入	输出
4 6　1　1　1 1　5　3　2	0　1