How Sweet It Is

题意：

输入钱数，判断当前总钱数能买多少份电子游戏，一份电子游戏$50。

若是能买一份，则输出 Sweet!，若是能买得到两份或两份以上，输出 Totally Sweet! 买完游戏之后，余下的钱可在下一次购买中使用。

题解：

用一个计数器记录输入的数据组数，再用一个计数器记录总钱数。每一次输入数据后总钱数增加，对总钱数进行判断。

若（钱数/50）== 1，则输出： Input #当前组数: Sweet!

若（钱数/50）> 1，则输出：Input #当前组数: Totally Sweet!

注：

1、输出的标点符号为英文（包括冒号和感叹号）。

Wheel of Universally Copious Fortune

题解：

在给出的字符串中找出符合的字符串，即长度相同且除字符'_'不同之外相同的所有字符串并统计出个数，特别注意的是每个实例的输出后有一行空行。直接暴力解就可以。

Positively Pentastic

题意：

给你五个整数，有正有负，但是和是正数。随机的放在一个正五边形的角上，现在要使用取反和平衡减法（同时用，先取反再平衡减法）使得正五边形上面的数字都为正数。所谓取反就是取相反数，所谓平衡减法就是找到一个数，用与它相邻的数（即前面一个，后面一个）都减去它。

题解：

这个道题目看上去又臭又长，其实是一道简单的模拟题，只要看懂题目意思，根据题目中给的样例，依葫芦画瓢的模拟即可解题成功。

但是有些细节需要注意一下：找到一个数，如果数是最头或者最尾部不能简单的加一减一，而要进行特判。同时不要太纠结于是否符合条件的判定，因为在找最小数时就可以从 0 开始往下找，如果没找到就说明五个数都符合要求。

总之，这道题目非常非常简单，不要因为描述很长就害怕它。

Lottery Coprimes

题意：

输入一个数字 n 代表共有多少个待检测的彩票字符串，接下来输入 n 个字符串。

若在当前字符串中可以找到一个分割线将字符串分为左右两个字符串（数字），使得这两个数字的最大公因数为 1，就说明这两个数字为 Coprimes，即这个彩票字符串中存在 Coprimes，这时输出这两个 Coprimes，否则输出"Not relative"。

例如："47108"中存在 47、108 为 Coprimes。

题解：

因为只要输入的字符串通过切割字符串成为的两个非空子串的最大公因数为 1，那么输出这两个数字。

所以对于每个字符串可以进行循环，分割线从 1 号位置开始遍历到 size-1 号位置，每次循环中通过 atoi()函数将两个字串转化成两个数字，再求这两个数的最大公因数，最后拿最大公因数与 1 比较。若等于 1 则输出这两个数字，否则继续循环继续向后查找是否存在符合条件的两个数字。直到最后仍没有找到符合条件的两个数字，则输出"Not relative"并结束。

注：

1、输出每个字符串结果之间隔一行。

2、若第二个数字有前导 0, 则删除前导 0 再输出。如："7203217", 则输出"72 3217", 而不是"72 03217"。

Pac Man for your New Phone

题解：

本题很简单，因为起点、终点位置固定，而且 Pac Man 只有两个方向可以选择，那么：

1、先得到第一列和第一行的分数，同时也是最好分数；

2、循环遍历剩余位置，每个位置的最好分数都是它本位的分值加上它前一步的最好分数（即它上面和左边的最好分数中更大的哪个）；

3、最后直接输出终点的最好分数即可。

要注意题目要求的格式，每个测试案例后都要跟一空行，因此最后一个测试案例需要特判，否则会输出两空行。

SierpiMski Triangle

题意：

从某个正方形开始，可以从左上角和右上角删除一半高度和四分之一宽度的矩形，一直重复此步骤，直到生成新的近似 SierpiMski 三角形。

输入：T(有 T 行测试数据)，每一行测试数据输入 k（第几级≤10）。

输出：从"triangle#s:"开始（s 从 1 开始）,输出近似的 k 级三角形的图案，用'X'表示未删除部分，' '表示删除部分，在此过程中要缩放图像（最小的正方形为 2*2 大小），在每个输出的三角形后面加一个空行。

题解：

若输入 k，则正方形的边长为 1<<k,首先将二维数组 g[1<<k][1<<k]全部置为 true，用一个递归函数（用 go 函数）执行左上角和右上角删除一半高度和四分之一宽度的矩形的操作，要删除的点置为 false(用 turnOff 函数)，最后把生成的近似 SierpiMski 三角形输出。

Walking in the Sun

题意：

给出 **s** (0≤s≤50)个圆的圆心坐标和半径，表示阴影部分；**q**(0≤q≤100) 对点(a_i，b_i)的坐标。求出(a_i，b_i)两点之间的最短距离，并且如果两点间路径经过阴影部分，则阴影部分的长度不算，任意两点之间都可以形成路径，没有障碍。

题解：

因为经过阴影部分的长度不算，所以两点间最短距离可能不是两点直接连线所形成的线段长度。我们可以把每个点看作以该点为圆心、半径为 0 的圆，即题目看作求两个指定圆心间的最短距离，且每个圆心之间的距离为 d = 圆心距 - 圆 $A_{半径}$ - 圆 $B_{半径}$。

题目数据范围较小，可以直接用 **Floyd** 算法求出所有最短路。求两个圆心的距离时，要注意两个圆可能重叠，所以要判断是否为负数。

Tautobots and Contradicticons

题意：

n 个机器人，每个机器人要么说的全是真话（Tautobots ），要么说的全是假话（Contradicticons）。然后他们总共说了 m 句话，每句话用三个值描述,"A S X"，代表的含义：A 说 S 是 X。X 表示“T”或者“C”。求有多少种机器人组合使得他们说的这些话不发生矛盾。

题解：

因为机器人只有两种类型，“T”或者“C”，那么考虑用二进制位中的 1 和 0 来分别表示这两种类型。由于 n 比较小，可以直接进行枚举。对于 n 个机器人，总共也就是 2^n 种状态。枚举所有状态（0~2^n-1），对于每个状态（第 i 位上表示第 i 个机器人的类型），都判断一遍这 m 句话中有没有矛盾的，没有矛盾就是可行状态，ans+1。

A Constant Struggle

题意：

给出一个八元一次方程 $C_1X_1 + C_2X_2 + C_3X_3 + C_4X_4 + C_5X_5 + C_6X_6 + C_7X_7 + C_8X_8 = N$ 中的 C_i 和 N 的值，求一共有多少组解（$X_i \geqslant 0$）。 数据范围：$0 \leqslant C_i, N \leqslant 100$

题解：

可以看成有八种物品，每种物品可以拿任意多件，要恰好装满大小为 N 的背包。很容易想到暴力解法，每个 X_i 的可能的取值个数约为 N/C_i 个，8 个 X_i 的可能的取值方式大约有（$N^8/C_1C_2C_3C_4C_5C_6C_7C_8$）种，最坏情况下是 10^{16}，所以采用记忆化搜索。

递归临界条件是正好装满，则解的个数加一，或者超出了 N 以及 取完所有物品后没有到 N，则解的个数返回 0。

Polygon Restoration

题意：

给 n 个点在二维坐标系上的坐标，要求将这 n 个点连接成封闭图形，满足每条边要么垂直要么水平，且任意相邻的边互相垂直，逆时针输出连接顺序。

题解：

任意相邻边一定垂直！(原题面中容易被忽略的条件)

$O(n^2)$预处理任意两点间的位置关系，然后 dfs+剪枝即可。

两个显然的剪枝条件：

1、点 x 连到点 y，不允许有点 z 存在点 x 与点 y 的连线之间。

2、任意相邻边一定垂直。

条件 1：可以直接 O(n)遍历所有点 z，判断是否在 x 和 y 的连线之间。

条件 2：dfs 过程中记录上一条边的方向即可。

输出的时候记得判断一下输出顺序。（因为 dfs 的结果不一定是逆时针）

发现每个点只可能走 4 个方向与他最近的点，因此可以直接预处理每个点 4 个方向最近的点来进一步优化，这样就不用判断是否存在点 z 在点 x 与点 y 之间啦！