**ЗАРОВЕ**

Единственото важно съображение е, че сборът на точките от две противоположни стени на зара е винаги 7. Оттук следва, че сумата от точките на двамата играчи ще е равен на броя на заровете K умножен по 7.

В задачата не са дадени ограниченията за точките на първия и втория. Тъй като е изрично подчертано, че тестовете са коректни и максималният брой зарове е 10000, следва, че точките на първия ще са в интервала [1;10000]. Тогава реалните точки на втория ще са до 60000, и понеже може да има променени цифри, то максималната стойност ще е 99999.

При такова малко ограничение на броя зарове, очевидното „наивно“ решение ще работи. Идеята е следната:

Вземаме първата двойка А1 и В1.

Проверяваме за всеки брой зарове *I* (*1 ≤ I ≤ 10000*) дали числото *C = I.7 – А* има същия брой цифри както В. Ако това е така, и разликата в цифрите на В и С е само в една цифра, добавяме *I* в множество M1. В това множество са всички възможни брой зарове, с които може да са играли двамата на първото хвърляне.

Правим същото за A2 и В2 и образуваме множеството М2 и т.н. докато образуваме и последното множество MN. Накрая намираме сечението на всички

М1  M2 … MN.

Важно съображение е следното: Добавя се *I* в множеството M само ако Ai ≥ *I*, защото иначе ще излезе, че при хвърлени *I* зара, сумата от точките Ai е по-малка от броя на заровете. Аналогично, след като намерим възможния реален брой точки на втория играч Тi=*I*\*7-Ai, трябва да проверим и за него дали е в границите [*I*,6.*I*].

При съставяне на тестовете са правени десетки опити, и се оказа, че са достатъчни 10-на хвърляния за да се определи еднозначно търсения брой зарове.

Въпреки всичко, нека теоретично е възможно при 1000 хвърляния това да не може да стане. Тогава са необходими милиони завъртания за да се образуват множествата M. Затова ще използваме следното – след като образуваме М1, следващото множество M2 ще го създадем на базата на него. Естествено е, че ако M1={134,139,156,247}, то за двойката A2 и В2 ще проверим само тези 4 възможности и вероятно ще отпадне поне 1 от тях. Естествено, няма нужда да се обработват всички двойки А и В ако сме стигнали до множество Mi, което е с един елемент – той е решението. Може да се направят и други оптимизации, но при дадените ограничения програмата върви достатъчно бързо и без допълнителни „екстри“.

Във файла zar\_Iliyan.cpp е решението на Илиян Йорданов, което е по-кратко.

*Автор: Павел Петров*