АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
Divisor Sequences

Задачата divseq, или divisor sequences беше готина, главно защото позволява голям спектър от състезатели с различен опит и знания да напишат нещо по нея и да хванат различен брой точки в зависимост от това, колко умно решение успеят да направят.

Как да хванем 25
Първите 25 точки би трябвало да са почти безпроблемни за повечето състезатели в тази група. Дори с (до) пет вложени цикъла, определящи числото на всяка една позиция, решението би работело. В него няма какво толкова да дискутираме.

Как да хванем 50
Следващите 25 точки са по-сложни. Ограниченията пак са доста ниски, но не достатъчно ниски за да правим глупости като вложени for цикли. Все пак изчерпване би минало, особено ако го правим хитро (тоест ако предходното число е било X, да не слагаме на текущата позиция Y ако X и Y са несъвместими (тоест не е изпълнено нито X % Y == 0, нито Y % X == 0). И тъй като както казахме би било убийствено трудно да ползваме цикли тук, вместо това ще ползваме рекурсия. Тя е най-мощният похват, когато стане дума за изчерпване (а и не само!), така че съветвам всички състезатели, които не се чувстват достатъчно добре с нея, да я упражняват.

Как да хванем 75
Следващите 25 точки вече са съвсем друго ниво. Ограниченията стават относително големи и всякакви решения, базирани на изчерпване са обречени на неуспех. Какво да правим тогава?
Както и много други пъти в задачи, в които се иска "броят на еди какво си", особено пък по модул, и тук решението е динамично оптимиране. Ако искате да научите повече за динамично оптимиране, можете да погледнете тази статия: http://informatika.bg/info.php?topic=DP1
Забелете рекурсията, която (може би) сте написали за 50 точки. Ако трябва да съставите редица с 10 числа, до сега сте сложили 3, и последното от тях е X, то по колко начина можете да завършите редицата? Независимо от това дали досегашната редица е {A, B, X…}, {B, A, X…}, {A, A, X}, винаги по еднакъв брой начин! Тоест, независимо какви първи нейни два члена сме избрали, броя завършвания е един и същ, ако последното число е сложено на позиция P и е X. Следователно е нужно да изчислим това състояние само веднъж, а не отново и отново, както прави рекурсията.
Така и стейтът на динамичното ни е dyn[P][X], тоест първото измерение е до коя позиция сме стигнали и второто е кое е било последното число, което сме сложили. От всяка позиция трябва да проверим всички възможни продължения (тоест числа, по-малки или равни на K, които или делят или се делят на X). Така това решение е със сложност O(N * K * K).

Как да решим задачата за 100
Горното решение е доста добро. Всъщност е и до голяма степен решението за 100. Това, което не правим добре, е намирането на продължение от даден стейт. Трябва да направим наблюдението, че за произволно число X между 1 и 1000, броят валидни продължения (тоест числа Y, съвместими с X) е сравнително малък. Вярно, има и изключения, разбира се, като например числото 1, което има 1000 продължения, или 2,

което има 500, но за повечето числа не е така. Всъщност, средният брой продължения при K = 1000 е едва 13.
Следователно можем да преизчислим продълженията от всяко число само веднъж, а не да ги търсим отново и отново. Това ни помага да намалим последния множител в горната сложност от K на 13 (амортизирано), правейки сложността ни почти квадратна (вместо кубична, както беше в горния вариант). Самото преизчисление можем да направим и сравнително бързо, но в случая не е нужно. Вместо това можем да ползваме почти тривиалното O(K * K): за всяко число пробваме всяко друго число.

Автор: Александър Георгиев