### Анализ на решението на задача

### СИМПАТИЧНИ ЧИСЛА

Наивното решение, опитващо всички възможни комбинации от три елемента за всяко i, е твърде бавно. Сложността му е O(N4) и носи 40% точки (файл nice\_40.cpp).

Тъй като стойностите в редицата са малки, можем да вземем масив P, който ни показва дали има определена стойност в редицата преди текущата позиция i. По-точно, P[x] = 1, ако има стойност x в редицата A преди позицията i, в противен случай P[x] = 0. Използвайки това, можем да подобрим нашето решение. Вместо да опитваме всяка възможна комбинация от три елемента, ние опитваме всяка възможна двойка елементи на позиции (j, k) по-малки от i и проверяваме дали има стойност A[i] - A[j] - A[k] в масива преди i. Тази информация е налична в масива P на позиция A[i] - A[j] - A[k]. След обработката на позиция i, задаваме P[A[i]] = 1, ако има такава стойност за някоя двойка (j, k) по-малки от i. Така постигаме сложност O(N3) и 70% точки.

За 100% точки се нуждаем от алгоритъм със сложност O(N2). Вместо да проверяваме дали има стойност A[i] - A[j] - A[k] за всяка двойка (j, k) в масива преди i, можем да проверим за всяка позиция j дали има двойка елементи преди i, такива че сумата им да е равна на A[i] - A[j]. Можем отново да използваме масива P, за да направим това. След обработка на позиция i, за всяка такава двойка (i, j) с j ≤ i записваме в P[A[i] + A[j]] = 1. Използвайки тази оптимизация, получаваме решение, което е достатъчно бързо и което постига пълния брой точки.

*Автор: Пламенка Христова*