Triples
(Решение)

Очевидно произволното взимане на едно, две или три числа би довело до много бавен алгоритъм (макар и верен). Сравнително лесно това може да се избегне като забележим, че ВСЯКО число трябва да бъде използвано рано или късно – следователно можем да сортираме числата и да ги взимаме в сортиран ред (от двете страни на редицата). Това очевидно би било вярно ако взимахме само по едно число. Да видим какво се случва ако взимаме по две: ако числата са положителни и вземем най-голямото число и знаем, че трябва да е съчетано с някое друго, то оптималният отговор е винаги следващото по големина. Ако числата са отрицателни – взимането на най-малкото от тях и съчетаването му с друго би било оптимално при съчетание с отново отрицателно или нула (тъй като произведението им е положително число или нула), при това следващото по абсолютна стойност (максимизиращо произведението). Взимане на тройка числа може да се гледа като взимане на двойка, слепянето им в едно и отново взимане на двойка (слепеното и третото).

До тук доказахме, че сортирането на числата и взимането на такива от двата края на сортирания масив не загубва оптималността на решението. Как обаче да определим кое число ще бъде взето само, кое в двойка и кое – в тройка? Като цяло е възможно да се разработи алчен алгоритъм, разчитащ на много случаи, който би произвел оптималното решение. Поради големия брой случаи, обаче, е далеч по-безопасно да се ползва техниката „динамично програмиране". След сортирането на числата ни интересува единствено по колко вече сме използвали от всяка от страните на масива, като така сме отново в сортиран, по-малък масив и същата задача. Тук състоянието ни е сравнително просто – трябва да пазим само докъде сме стигнали от двете страни – в таблица от вида dyn[1000][1000]. Във всяка позиция трябва да тестваме да вземем по едно число от всяка страна; двойка числа (двете отляво, двете отдясно или едно от ляво и едно отдясно); и аналогично тройка може да бъде взета по 4 начина. Изчерпвайки всички възможни взимания си гарантираме достигане на оптимален отговор, игнорирайки различните случаи.

С това задачата ни ще бъде решена със сложност O(n^2), тъй като във всяко възможно състояние на решението имаме константен брой операции. Трябва да забележим, че типът int не побира резултата, за това през цялото време трябва да извършваме сметките с 64 битов тип.

Автор: Александър Георгиев