АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
ЗАРЯД НА БАТЕРИИ

Подзадача 1.

Симулираме изпълнението на първите M операции. За всяка операция обхождаме всички елементи на масива, изчисляваме текущата им стойност и намираме тяхната сума. Времето за работа е *O(N·M)*. Реализацията е в `battery_33.cpp`.

Подзадача 2.

Забелязваме, че ако на дадена стъпка нито един от елементите на масива не се е занулил, то сумата след тази стъпка не е минимална (можем със сигурност да кажем, че на предишната стъпка сумата е била точно с N по-малка). Затова не е нужно да разглеждаме всички операции, а само тези, при които поне един елемент от масива се занулява.

Нека разгледаме N такива операции (тези, при които се занулява първият, вторият, …, последният елемент от масива). За всяка от тях ще обходим всички елементи на масива, ще изчислим текущите им стойности и ще намерим сумата им. Така получаваме алгоритъм с време на работа *O(N·N)*. Програмна реализация в `battery_50.cpp`.

Подзадача 3.

Предварително преброяваме колко пъти всяко от числата от 0 до M−1 се среща в масива (нека означим съответния масив от броячи с `cnt[]`). Сега симулираме изпълнението на първите M операции. Забелязваме, че i-тата операция увеличава сумата на елементите с N (т.е. +1 към всеки елемент), но намалява сумата с `cnt[M-i]* м`. Това се случва, защото елементите на масива, които първоначално са равни на M-i, след i увеличения стават равни на M и се зануляват (т.е. всеки такъв елемент намалява сумата с M). По този начин, знаейки сумата на елементите преди операциите и колко пъти се среща всяка стойност, можем за O(1) да намерим сумата след произволен брой операции по формула. Общото време на работа е *O(N+M)*. Реализация в `battery_72.cpp`.

Подзадача 4.

Комбинираме идеите от подзадачи 2 и 3: т.е. ще разглеждаме само онези операции, след които поне един елемент от масива се е занулил, но вместо директно да изчисляваме сумата на елементите, ще използваме формула.

Може да сортираме всички елементи на масива, така че еднаквите стойности да се окажат една до друга. За всяка такава група от еднакви стойности можем да намерим номера на операцията, при която всички те ще се занулят. Всички елементи със стойности по-малки от текущата също трябва да са се занулили при някоя от по-ранните операции. Тъй като масивът е сортиран, броят на тези елементи също ни е известен. Затова за всяка интересуваща ни операция можем за *O(1)* да намерим сумата след нея. Всички тези суми се изчисляват с едно линейно обхождане на сортирания масив, като общото време за работа е *O(N·logN)* заради предварителното сортиране.

Пример на такова решение е `author.cpp`.

Автор: Кинка Кирилова-Лупанова