Задача ОТГАТНИ ЧИСЛОТО

Пояснение към решението

Нека дадената сума във входа е s. Трябва да намерим всички числа x, които имат прост делител p такъв че $x + p = s$.

Класически подход

Понеже търсим x е логично да ориентираме решението си около него. Ясно е че $x < s$ и най-стандартния подход е да обходим всички възможни стойности за x и да проверим кои от тях отговарят на условието. Сега трябва да може за произволно число да проверим дали отговаря на условието.

Нека сега да видим как да проверим дали за произволно число x отговаря на условието. Това има много варианти как да стане.

Едно от най-бавните решения е да намерим всички делители на x и за всеки от тях да проверим дали е просто число и сумата им е равна на s. Една малко по-добра идея е да намерим направо простите делители на x и отново да проверим дали постигаме търсената сума.

Сега за да работи добре класическия подход трябва да се усетим, че няма нужда да разглеждаме всички варианти за прост делител на x. Ако ние знаем x, то е ясно че простия делител p трябва да е равен на разликата $s - x$. Така $p = s - x$ и е достатъчно да проверим, че p е прост делител на x. Тази проверка включва две неща, дали p е делител на x и дали p е просто число. Понеже първото се смята много по бързо от второто е важен редът на двете неща. Така `if(n%p == 0 && isPrime(p))` е много по-добре от `if(isPrime(p) && n%p == 0)`.

Така може да постигнем 80 точки със следния код, като имплементацията на `isPrime` няма нужда да е оптимизирана:

```
for (long long x = 1; x < s; x++) {
    long long p = s-x;
    if (x%p == 0 && isPrime(p)) {
        cout << x << " ";
    }
}
```

Другата гледна точка + наблюдението

Има прекалено много варианти за x и това трудно може да се подобри. Може да пробваме да се насочим към вариантите за простия делител p. За да работи добре този подход има едно основно наблюдение – понеже p дели x, то p дели и тяхната сума. Така възможните стойности за p са простите числа, които са делители на x.

Това ни насочва към това да направим разлагане на прости делители на s. Остава за всеки прост делител p да проверим дали $s-p$ отговаря на условията. Всъщност това винаги е изпълнено понеже вече знаем, че p дели $s-p$.

Последния детайл е сложността на разлагането на прости делители. При линейна имплементация ще се вземат около 92 точки. За това тук ще покажем как се прави бързо разлагане на прости делители, което се надяваме учениците да почнат да разбират.

```
void primeDivs(long long n) {
    for (long long i = 2; i*i <= n; i++) {
        if (n%i != 0) continue;
        cout << i << " ";
        while (n%i == 0) n/=i;
    }
    if (n != 1) cout << n;
}
```

Петър Петров