Разбор задачи «Заполните массив»

Автор задачи — В. Гуровиц, автор разбора — В. Каушанский

Несмотря на схожесть формулировок задачи с таким названием в лигах A и B сильно различались по сложности.

В лиге B надо было привести пример хотя бы одной расстановки чисел 2, 3, ..., $n+1$ в массиве $a[1..n]$ так, чтобы $a[i]$ делилось бы на i. Вот один из вариантов:

$$\left(\begin{array}{c|cccccccc} i: & 1 & 2 & 3 & 4 & 5 & 6 & 7 & \cdots & n \\ \hline a[i]: & n+1 & 2 & 3 & 4 & 5 & 6 & 7 & \cdots & n \end{array}\right) \qquad a[i] = \begin{cases} n+1, & \text{если } i = 1; \\ i, & \text{иначе.} \end{cases} \tag{1}$$

В лиге A надо было посчитать количество таких расстановок, что мы сейчас и обсудим. Назовем тривиальной расстановку чисел (1).

Решим задачу динамическим программированием. Пусть $dp[n]$ — ответ для n, то есть количество способов расставить числа 2, 3, ..., $n+1$ в массив $a[1..n]$ указанным образом. Пусть число $n+1$ попало на позицию k (то есть $a[k] = n+1$). Тогда для $a[i] > k$ хотя бы при $i = k, k+1, \ldots, n$, всего $n-k+1$ ячеек. Но среди чисел 2, 3, ..., $n+1$ больше k как раз $n-k+1$, следовательно, все они занимают места с k по n в массиве. Не большие k располагаются тогда в позициях 1, 2, ..., $k-1$. Заметим, что существует единственный способ расположить числа $k+1$, $k+2$, ..., n по позициям $k+1$, ..., n требуемым способом.

$$\left(\begin{array}{c|cccccc} i: & \ldots & k & k+1 & \ldots & n-1 & n \\ \hline a[i]: & \ldots & n+1 & k+1 & \ldots & n-1 & n \end{array}\right)$$

Итак, если $a[k] = n+1$, то числа больше k распределяются по массиву единственным способом, количество же вариантов расположить 2, 3, ..., $k+1$ по первым k позициям по определению равно $dp[k]$.

Надо отметить, что при всем этом $n+1$ обязано делиться на k.

В свете проделанных рассуждений понятна реккурентная формула для $dp[n]$ (суммирование проходит по всем k, на которое делится $n+1$):

$$dp[n] = \sum_{k|(n+1)} dp[k] \tag{2}$$

Считая $dp[1] = 1$, можно вычислить все $dp[i]$, $i \le n$, за $O(n\sqrt{n})$, перебирая для каждого i все его делители (которых, как известно, не более $2\sqrt{i}$). При данных ограничениях такое решение проходит по времени. Написав программу, несложно убедиться, при использовании типа `integer` (`int`) переполнения не возникнет.

Тем не менее, при грамотном переборе делителей оценка на сложность алгоритма снижается до $O(n \log n)$:

```
fillchar(dp, sizeof(dp), 0);
for i := 1 to n do
    for j := 2 to n div i do
        dp[i*j] += dp[i];
```

Количество операций при такой реализации не превышает $\frac{n}{1} + \frac{n}{2} + \cdots + \frac{n}{n} = O(n \log n)$.