Informatyczny kącik olimpijski (110): Liczby

W tym miesiącu omówimy zadanie *Liczby*, które pojawiło się podczas rundy 72. na portalu `codeforces.com`.

W zadaniu należy, mając dane liczby naturalne $a, b, k \leqslant 2 \cdot 10^9$, wyznaczyć liczbę liczb naturalnych w przedziale $[a, b]$, których najmniejszym dzielnikiem, większym od 1, jest k. Zauważmy najpierw, że wystarczy umieć obliczać wyniki dla przedziałów postaci $[1, n]$, ponieważ wynik dla przedziału $[a, b]$ jest różnicą wyników dla przedziałów $[1, b]$ i $[1, a-1]$. Ponadto, jeśli k nie jest liczbą pierwszą, to wynik jest równy 0, ponieważ jeśli liczba jest podzielna przez k, to jest też podzielna przez każdy dzielnik k, a skoro k nie jest pierwsza, to ma dzielnik większy od 1 i mniejszy od k. Załóżmy teraz, że k jest pierwsza. Zadanie można rozwiązać w czasie $O(n \log\log n)$, wyznaczając dla każdej liczby z przedziału $[1, n]$ jej najmniejszy dzielnik, większy od 1, za pomocą sita Eratostenesa. Rozwiązanie to dla $n \leqslant 2 \cdot 10^9$ jest dalece niewystarczające. Zauważmy, że można je łatwo usprawnić. Każda liczba, której najmniejszy dzielnik jest równy k, jest postaci $k \cdot m$, gdzie m nie ma dzielników większych od 1 i mniejszych od k. Wystarczy zatem zliczyć takie m-y z przedziału $\left[1, \left\lfloor \frac{n}{k} \right\rfloor\right]$, które nie mają dzielników pierwszych mniejszych od k. Można to zrobić, iterując kolejne liczby pierwsze mniejsze od k i zaznaczając w tablicy liczby podzielne przez którąś z nich.

Rozwiązanie to wykona dla każdej liczby pierwszej p, mniejszej od k, $\frac{n}{kp}$ operacji, czyli łącznie

$$\frac{n}{k} \sum_{p \in \mathbb{P},\, p<k} \frac{1}{p} = \Theta\left(\frac{n}{k} \log\log k\right).$$

Dla $k \geqslant 100$ rozwiązanie takie jest wystarczające. Jeśli natomiast k jest mniejsze od 100, to liczb pierwszych mniejszych od k jest nie więcej niż 25. Niech teraz A_p oznacza zbiór liczb podzielnych przez k oraz p leżących w przedziale $[1, n]$. Naszym zadaniem jest obliczenie

$$\left| \{x \in [1, n] : k \mid x\} \setminus \left(\bigcup_{p \in \mathbb{P},\, p<k} A_p \right) \right|.$$

Możemy to zrobić, korzystając z zasady włączeń i wyłączeń. W tym celu musimy umieć obliczyć $|A_{p_1} \cap A_{p_2} \cap \ldots \cap A_{p_i}|$. Jest to zbiór liczb podzielnych przez $kp_1p_2 \ldots p_i$, leżących w przedziale $[1, n]$, czyli ma $\left\lfloor \frac{n}{kp_1p_2 \ldots p_i} \right\rfloor$ elementów.

Całe rozwiązanie działa w czasie

$$O\left(\min\left(2^{\pi(k)}, \frac{n}{k} \log\log k\right)\right).$$

Na to zadanie można też spojrzeć inaczej. Niech $F(n, p)$ będzie liczbą liczb w przedziale $[1, n]$, które nie mają dzielników większych od 1 i mniejszych od p. Wykażemy, że

$$F(n, p) = n - \sum_{q \in \mathbb{P},\, q<p} F\left(\left\lfloor \frac{n}{q} \right\rfloor, q\right)$$

Istotnie, liczb nie większych od n, których najmniejszym dzielnikiem jest q, jest $F\left(\left\lfloor \frac{n}{q} \right\rfloor, q\right)$, ponieważ każda taka liczba jest postaci $q \cdot m$, gdzie m nie ma dzielników większych od 1 i mniejszych od q, a $q \cdot m \leqslant n$, czyli $m \leqslant \left\lfloor \frac{n}{q} \right\rfloor$. Wszystkich liczb w przedziale $[1, n]$ jest n, a każda z nich albo ma dzielnik pierwszy mniejszy od p, wówczas zostanie odjęta dokładnie raz, dla q będącego jej najmniejszym dzielnikiem pierwszym, albo nie ma i wtedy nie zostanie odjęta. Liczba liczb w przedziale $[1, n]$, których najmniejszym dzielnikiem jest k, jest równa $F\left(\left\lfloor \frac{n}{k} \right\rfloor, k\right)$. Zadanie można rozwiązać, obliczając wartość funkcji F, rekurencyjnie korzystając z tego wzoru. Niech $p_1, p_2, \ldots$ będą kolejnymi liczbami pierwszymi, a $T(n)$ liczbą operacji wykonanych przez algorytm dla liczby pierwszej p_n. Wówczas $T(n) = 1 + \sum_{i=1}^{n-1} T(i)$. Rozwiązaniem tego równania jest $T(n) = \Theta(2^n)$, czyli całe rozwiązanie będzie działało w czasie $O(2^{\pi(k)})$, co jest zdecydowanie za wolne.

Rozwiązanie to można łatwo przyspieszyć. Wystarczy zauważyć, że jeśli $n < p$, to jedyną liczbą w przedziale $[1, n]$ niemającą dzielników większych od 1 i mniejszych od p jest 1, więc $F(n, p) = 1$ i w każdym wywołaniu rekurencyjnym obliczającym wynik dla $n < p$ zwracać $\min(1, n)$ bez dalszych wywołań rekurencyjnych. Wykażemy, że rozwiązanie z tą optymalizacją obliczające $F(n, p)$ wykona co najwyżej $O(n)$ wywołań rekurencyjnych. Rozważmy gałąź w drzewie rekursji, w której algorytm wykonał kolejno dzielenia przez $p_1, p_2, \ldots, p_i$. Oznacza to, że w przedostatnim wywołaniu obliczał $F\left(\left\lfloor \frac{n}{p_1p_2 \ldots p_{i-1}} \right\rfloor, p_{i-1}\right)$. Skoro nie zwrócił wyniku od razu, tylko wykonał następne wykonanie rekurencyjne, to $\left\lfloor \frac{n}{p_1p_2 \ldots p_{i-1}} \right\rfloor \geqslant p_{i-1}$, skąd wynika, że $\frac{n}{p_1p_2 \ldots p_{i-1}} \geqslant p_{i-1}$, czyli $n > p_1p_2 \ldots p_{i-2}p_{i-1}^2$. Algorytm w danym wywołaniu przegląda jedynie liczby pierwsze, mniejsze od poprzedniej, zatem $p_{i-1} > p_i$, skąd otrzymujemy $n > p_1p_2 \ldots p_i$. Liczby p_i są pierwsze, więc dla różnych wywołań rekurencyjnych iloczyny $p_1p_2 \ldots p_i$ będą parami różne, więc liczba tych wywołań nie przekroczy n.

Prezentowane rozwiązanie potrzebuje listy liczb pierwszych mniejszych od $\min(n, p)$. Można ją, oczywiście, wyznaczyć, używając sita Eratostenesa, w czasie $O(\min(n, p) \log(\min(n, p)))$. Ostatecznie, całe rozwiązanie obliczające $F\left(\left\lfloor \frac{n}{p} \right\rfloor, p\right)$ działa w czasie

$$O\left(\min\left(2^{\pi(k)}, \frac{n}{k}\right)\right).$$

Konrad PALUSZEK