

Обозначения

$$\sum_{i=1}^n a_i = a_1 + a_2 + \dots + a_n$$

$f(n) = \mathcal{O}(g(n))$ — существуют такие числа a и b , что для всех n : $f(n) \leq a + bg(n)$.

$f(n) = \Theta(g(n))$ — одновременно $f(n) = \mathcal{O}(n)$ и $f(n) = \Omega(n)$.

$$\prod_{i=1}^n a_i = a_1 \cdot a_2 \cdot \dots \cdot a_n$$

$f(n) = \Omega(g(n))$ — то же самое, что и $g(n) = \mathcal{O}(f(n))$.

$a \equiv b \pmod{m}$ — a и b имеют одинаковые остатки при делении на m .

Задачи

Задачи разбиты на блоки, которые отделены друг от друга линиями вида «— — —». На решение задач из каждого блока отводится некоторое время, после которого проводится их разбор. Рекомендуется сдавать задачи именно из текущего блока, если вы, конечно, уже не решили их все.

Задача 1. Предложите алгоритм вычисления НОД двух натуральных чисел за $\mathcal{O}(n^2)$, где n — ограничение на длину чисел в какой-то фиксированной системе счисления.

Разбор. Будем действовать следующим образом:

1. Если оба числа делятся на 2, то $\gcd(a, b) = 2 \cdot \gcd(\frac{a}{2}, \frac{b}{2})$.
2. Если ровно 1 число делится на 2, будем считать, что это a . Тогда $\gcd(a, b) = \gcd(\frac{a}{2}, b)$.
3. В противном случае считаем, что $a \geq b$, и тогда $\gcd(a, b) = \gcd(b, a - b)$.

Заметим, что после действий 1 и 2 хотя бы одно из чисел уменьшается в 2 раза. Также после действия 3 одно из чисел становится четным, следовательно, далее совершается действие 2. За каждые два последовательных действия хотя бы одно из чисел уменьшается в 2 раза, следовательно, мы совершим $\mathcal{O}(n)$ действий, каждое из которых работает за $\mathcal{O}(n)$, из чего и получается требуемое время работы $\mathcal{O}(n^2)$.

Задача 2. Даны n целых чисел, по модулю не превосходящих C . Рассмотрим следующий алгоритм вычисления их НОД.

```
answer = 0
for i = 0 ... n - 1:
    answer = gcd(answer, a[i])
```

Здесь функция `gcd` реализует алгоритм Евклида, использующий операции деления с остатком. Дайте асимптотическую оценку сложности работы данного алгоритма.

Разбор.

Лемма 1. Пусть $a \geq b$. Тогда $a \% b < \frac{a}{2}$.

Доказательство. Пусть $b > \frac{a}{2}$. Тогда очевидно, что $a \% b = a - b < \frac{a}{2}$. Иначе, если $b \leq \frac{a}{2}$, то $a \% b < b \leq \frac{a}{2}$.

Лемма 2. За каждые три последовательных шага алгоритма Евклида каждое из чисел в нем уменьшается хотя бы в 2 раза.

Доказательство. *Случай 1.* $a \geq b$. Тогда на следующем шаге будем находить $\gcd(b, a \% b)$, а потом $\gcd(a \% b, b \% (a \% b))$, притом очевидно, что $b \geq a \% b$. Из леммы 1 следует, что $a \% b \leq \frac{a}{2}$, а также $b \% (a \% b) \leq \frac{b}{2}$. То есть оба числа уменьшились хотя бы в 2 раза после двух шагов. *Случай 2.* $a < b$. Тогда после одного шага перейдем к случаю 1, и за следующие оба числа уменьшатся хотя бы в 2 раза.

Решение задачи. Рассмотрим i -й шаг цикла. Во время него алгоритм Евклида совершал какие-то действия. Мы можем разбить эти действия на последовательные тройки (действия типа 1). Также, возможно, для каждого шага останутся одно или два действия, которые не вошли в тройки. Заметим, что суммарно по всем шагам цикла последовательных троек $\mathcal{O}(\log C)$, поскольку после выполнения тройки ответ уменьшается хотя бы в 2 раза. Следовательно, суммарно будет совершено не более $\mathcal{O}(n + \log C)$ шагов алгоритма Евклида, каждый из которых работает за $\mathcal{O}(1)$.

Задача 3. Пусть a, b, c — целые числа. Рассмотрим уравнение $ax + by = c$ относительно целых x, y .

- а. Покажите, что, если c не делится на $\gcd(a, b)$, решений нет.
- б. Покажите, что при $c = \gcd(a, b)$, решение есть.
- в. Покажите, что решение существует тогда и только тогда, когда c делится на $\gcd(a, b)$.
- г. Покажите, что, если существует хотя бы одно решение, существует бесконечно много решений. Опишите их все.

Разбор.

- а.** Так как $\gcd(a, b) | a$ и $\gcd(a, b) | b$, то $\gcd(a, b) | ax + by$. Значит, если решение существует, то $\gcd(a, b) | c$.
- б.** Решение можно явно предъявить, используя расширенный алгоритм Евклида.
- с.** Если c не делится на $\gcd(a, b)$ то по пункту **а** решений нет. Иначе $c = d \cdot \gcd(a, b)$. Пусть (x', y') — решение уравнения $ax + by = \gcd(a, b)$. Домножим x' и y' на d и получим решение исходного уравнения.
- д.** Пусть существует решение $ax_0 + by_0 = c$. Положим $x_t = x_0 + t \cdot b / \gcd(a, b)$, $y_t = y_0 - t \cdot a / \gcd(a, b)$ для некоторого целого t . Эта пара значений также является решением уравнения. Таким образом, решений бесконечно много. Осталось показать, что мы описали все решения. Заметим, что, если есть 2 решения x_1, y_1 и x_2, y_2 , то $a \cdot (x_1 - x_2) + b \cdot (y_1 - y_2) = 0$ (вычли из одного равенства другое). Тогда $a \cdot (x_1 - x_2) = b \cdot (y_2 - y_1)$. Тогда существует целое число t такое, что $(b/\gcd(a, b)) \cdot t = x_1 - x_2$ и $(a/\gcd(a, b)) \cdot t = -(y_1 - y_2)$, следовательно, наш алгоритм, имея одно решение, опишет и другое.

Определение. Числа a и b называются взаимно обратными по модулю m , если $a \cdot b \equiv 1 \pmod{m}$.

Задача 4. Дано число a . Предложите алгоритм, который за $\mathcal{O}(\log m)$ вычислит обратный ему по **не обязательно простому** модулю m или определит, что такого не существует.

Разбор. Мы хотим найти такое число x , что $ax \equiv 1 \pmod{m}$. Иными словами, $ax + my = 1$ для некоторого целого y . Из предыдущей задачи мы знаем, что решение существует тогда и только тогда, когда $\gcd(a, m) = 1$, и в таком случае мы его можем найти.



Задача 5. Докажите, что $\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{n} = \Theta(\log n)$.

Разбор. $\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \dots + \frac{1}{n} = \left(\frac{1}{1}\right) + \left(\frac{1}{2} + \frac{1}{3}\right) + \left(\frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7}\right) + \dots + \left(\dots + \frac{1}{n}\right) \leq 1 \cdot 1 + 2 \cdot \frac{1}{2} + 4 \cdot \frac{1}{4} + \dots \leq \lceil \log_2(n+1) \rceil$

$\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \dots + \frac{1}{n} = \left(\frac{1}{1}\right) + \left(\frac{1}{2} + \frac{1}{3}\right) + \left(\frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7}\right) + \dots + \left(\dots + \frac{1}{n}\right) \geq 1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{4} + 4 \cdot \frac{1}{8} + \dots \geq \lceil \log_2(n+1) \rceil / 2$

Задача 6. Пусть $\tau(n)$ — количество натуральных делителей n . Докажите, что $\sum_{i=1}^n \tau(i) = \mathcal{O}(n \log n)$.

Разбор. Заметим, что 1 — делитель у каждого первого числа, 2 — делитель у каждого второго числа и так далее. Тогда, используя предыдущую задачу, получаем:

$$\sum_{i=1}^n \tau(n) = \left\lfloor \frac{n}{1} \right\rfloor + \left\lfloor \frac{n}{2} \right\rfloor + \dots + \left\lfloor \frac{n}{n} \right\rfloor = \mathcal{O}(n \log n)$$

Определение. Функция Эйлера $\varphi(n)$ — количество натуральных чисел меньших n , взаимно простых с n .

Задача 7. **а.** Выведите формулу для $\varphi(p^k)$ (p — простое число), которая будет зависеть от p и k .

б. Предложите алгоритм вычисления $\varphi(n)$ для произвольного n за $\mathcal{O}(\sqrt{n})$.

Разбор.

а. Все числа, меньше или равные p^k , взаимно просты с p^k , кроме тех, что делятся на p .

Поэтому $\varphi(p^k) = p^k - \frac{p^k}{p} = p^k - p^{k-1}$.

б. Функция Эйлера мультипликативная, то есть для любых взаимно простых a, b : $\varphi(ab) = \varphi(a)\varphi(b)$. Следовательно, достаточно факторизовать n , после чего задача сводится к пункту **а**.

Задача 8. (*Китайская теорема об остатках*) Дано n взаимно простых чисел a_1, a_2, \dots, a_n ($a_i \leq C$). Дано n неотрицательных чисел r_1, r_2, \dots, r_n . Докажите, что существует единственное число x ($0 \leq x < \prod_{i=1}^n a_i$) такое, что $x \equiv r_i \pmod{a_i}$ для всех $i = 1 \dots n$.

Задача 9. В предположении, что все арифметические операции с числами любой длины выполняются за $\mathcal{O}(1)$, научитесь находить число x из предыдущей задачи за $\mathcal{O}(n \log C)$.

Задача 10. (*Теорема Эйлера*) Докажите, что если k взаимно просто с n , то $k^{\varphi(n)} \equiv 1 \pmod{n}$.

Примечание. Частный случай этой теоремы при простом n называется *малой теоремой Ферма*. Этот случай примечателен тем, что $\varphi(p) = p - 1$ для простого p , отчего теорема принимает совсем простой вид.



Задача 11. Предложите способ ускорить решето Эратосфена, чтобы оно работало за линейное время.

Подсказка. Для каждого числа попробуйте найти минимальный простой делитель.

Задача 12. За $\mathcal{O}(n)$ для всех чисел от 1 до n найдите:

- a. В какой степени минимальный простой делитель входит в его разложение.
- b. Количество его простых делителей.
- c. Количество его делителей.
- d. Сумму его делителей.
- e. Функцию Эйлера от него.

Задача 13. Научитесь вычислять $a \cdot b$ для натуральных a и b , используя только сложение, деление на 2 (в том числе с остатком), а также проверку на равенство 1 за $\mathcal{O}(\log a)$ операций сложения.

Задача 14. Для всех $k \leq n \leq X$ научитесь находить C_n^k по простому модулю $p > X$ за $\mathcal{O}(1)$, с предподсчётом за $\mathcal{O}(n)$.

Задача 15. (Дискретное логарифмирование) $a^x \equiv b \pmod{m}$, a и m взаимнопросты. Найти решение или определить, что его не существует, за время $\mathcal{O}(\sqrt{m} \log m)$.

Подсказка. Представьте x в виде $ky - r$ для $k = \lfloor \sqrt{m} \rfloor$.

— — —

Задача 16. Найти количество натуральных чисел, меньших n , взаимно простых с числом m , за $\mathcal{O}(\sqrt{m})$

Задача 17. Найти сумму НОД по всем подотрезкам массива натуральных чисел, не больших C , за $\mathcal{O}(n \log C)$.

Задача 18. Найти сумму НОД по всем непустым подмножествам массива из n натуральных чисел, не больших C , за $\mathcal{O}(n + C \log C)$. Ответ найдите по модулю $10^9 + 7$.

Задача 19. Дан массив из n натуральных чисел, не больших C . Выпишем gcd по всем непустым подмножествам этого массива. Найти медиану выписанных чисел за $\mathcal{O}(n \cdot C \log C)$.

Задача 20. Назовём натуральное число кубастым, если его можно представить в виде $a^3 \cdot b$ для каких-то натуральных $a > 1, b \geq 1$. Найти количество кубастых чисел, не больших n . (n до 10^{18} , стандартный компьютер со стандартными ограничениями)