

# Оценка сложности арифметических операций

# Асимптотические обозначения

$f = O(g)$  - функция  $f$  ограничена сверху функцией  $g$ ,

$f = \Omega(g)$  - функция  $f$  ограничена снизу функцией  $g$ ,

$f = \Theta(g)$  - функции  $f, g$  имеют одинаковый порядок роста, запись  $f \approx g$ .

## Свойства

$$O(f + g) = O(\max\{f, g\});$$

$$O(f \cdot g) = O(f \cdot g);$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \begin{cases} 0, \text{ тогда } f = O(g); \\ C \neq 0, \text{ тогда } f = \Theta(g); \\ \infty, \text{ тогда } f = \Omega(g). \end{cases}$$

Формула Стирлинга  $n! \approx \sqrt{2\pi n} \cdot \left(\frac{n}{e}\right)^n.$

## Примеры

$$\log n = O(n), \quad n = \Omega(\log n);$$

$$\log^k n = O(n^m), \quad n^m = \Omega(\log^k n)$$

$$n^k = O(a^n), \quad a^n = \Omega(n^k);$$

$$a^n = O(n!), \quad n! = \Omega(a^n).$$

## Основная лемма.

Если

$$f(1) = d, \quad f(n) = c \cdot f\left(\frac{n}{a}\right) + d \cdot n$$

для  $a, c, d > 0$ , то

$$f(n) = \begin{cases} O(n), & \text{если } a > c, \\ O(n \cdot \log n), & \text{если } a = c, \\ O(n^{\log_a c}), & \text{если } a < c. \end{cases}$$

**Сложность  
арифметических  
операций над целыми  
числами**

# Классификация алгоритмов

1) Полиномиальные алгоритмы

$$f = O(n^m), \quad m = \text{const.}$$

2) Экспоненциальные алгоритмы

$$f = O(a^n), \quad a = \text{const}, a > 1.$$

3) Субэкспоненциальные алгоритмы

$$f = e^{(c+O(1)) \cdot \log^\gamma n \cdot (\log \log n)^{1-\gamma}},$$

$$c = \text{const}, c > 0, \gamma = \text{const}, 0 < \gamma < 1.$$

---

$n = \lceil \log_{\beta} x \rceil + 1 = L_{\beta}(x)$  –  $\beta$ -длина числа  $x$ .

1) Сложность сложения и вычитания  
«СТОЛБИКОМ»

$$f_{sum} = O(n), \quad f_{dif} = O(n).$$

2) Сложность умножения и деления с  
остатком «СТОЛБИКОМ»

$$f_{mult} = O(n^2), \quad f_{div} = O(n^2).$$

---



3) Сложность возведения  $n$ -разрядного числа в степень  $k$ -разрядного числа

$$f_{pow} = O(f_{mult}(n) \cdot k) = O(n^2 k).$$

**Лемма.** Сложность возведения в квадрат  $f_{sq}$ ,  
сложность обращения чисел  $f_{inv}$   
удовлетворяют условиям

$$f_{mult} \approx f_{div} \approx f_{sq} \approx f_{inv}.$$

# Алгоритм Карацубы

$$f_{mult} = O(n^{\log_2 3}).$$

## Алгоритм Евклида

Вычисление наибольшего общего делителя целых чисел  $a$  и  $b > 0$  состоит из следующих этапов: положим  $a_0 = a, a_1 = b$  и выполним деления с остатком  $a_i$  на  $a_{i+1}$ :

$$a_0 = a_1 q_1 + a_2, \quad 0 \leq a_2 < a_1,$$

$$a_1 = a_2 q_2 + a_3, \quad 0 \leq a_3 < a_2,$$

.....

$$a_{k-2} = a_{k-1} q_{k-1} + a_k, \quad 0 \leq a_k < a_{k-1},$$

$$a_{k-1} = a_k q_k.$$

$$a_k = \text{НОД}(a, b).$$

---

**Расширенный алгоритм Евклида** позволяет не только вычислять наибольший общий делитель целых чисел  $a$  и  $b > 0$ , но и представлять его в виде  $\text{НОД}(a, b) = ax + by$  для некоторых  $x, y \in \mathbb{Z}$ .

Значения  $x, y$  находятся в результате обратного прохода этапов алгоритма Евклида, в каждом из которых уравнение разрешается относительно остатка  $a_i$ , который представляется в форме  $a_i = ax_i + by_i$  для некоторых  $x_i, y_i \in \mathbb{Z}$ .

---

Очевидно, что  $x_0 = 1, y_0 = 0, x_1 = 0, y_1 = 1$  и выполняются равенства:

$$a_i = a_{i-2} - a_{i-1}q_{i-1},$$

$$x_i = x_{i-2} - x_{i-1}q_{i-1},$$

$$y_i = y_{i-2} - y_{i-1}q_{i-1}.$$

Отсюда последовательно получаются искомые представления для всех  $a_i$  и, в частности, представление  $\text{НОД}(a, b) = a_k = ax_k + by_k$ .