Сложность алгоритма (Асимптотика)

T-нотация (количество операций)

Пример

```
sum = 0                      (1)
for (i = 0; i < n; i++)      (2n + 1)
    sum += a[i]              (2n)
print(sum)                   (1)
```

$T(n) = 4n + 3$

O-нотация (какая-то верхняя оценка)

$f(n) = O(g(n))$

$\exists c > 0, N > 0$ что $\forall n > N$ верно $f(n) < c * g(n)$

Ω-нотация (нижняя оценка)

$f(n) = \Omega(g(n))$

$\exists c > 0, N > 0$ что $\forall n > N$ верно $f(n) > c * g(n)$

θ-нотация (оценка и сверху и снизу)

$f(n) = \theta(g(n))$

$\exists c_1, c_2 > 0, N > 0$ что $\forall n > N$ верно $c_1 * g(n) < f(n) < c_2 * g(n)$

Индукция

$T(n) = 2 * T(n/2) = O(n)$

Хотим: $T(n) = O(n) \Rightarrow T(n) < c * n$

База: $T(1) = 1 < c * 1$

Переход: $T(n) = 2 * T(n/2) < 2 * (c * n/2) = c * n$

Мастер-Теорема

$T(n) = a * T(n/b) + n^c$

1) $c < log_b a \Longrightarrow T(n) = O(n^{log_b a})$

2) $c > log_b a \Longrightarrow T(n) = O(n^c)$

3) $c = log_b a \Longrightarrow T(n) = O(n^c \log n)$