

# SPb HSE, 1 курс, осень 2021/22

## Конспект лекций по алгоритмам

Собрано 7 сентября 2021 г. в 00:45

---

### Содержание

<b>1. Асимптотика</b>	<b>1</b>
1.1. $\mathcal{O}$ -обозначения	1
1.2. Рекуррентности и Карацуба	2
1.3. Теоремы о рекуррентных соотношениях	4
1.4. Доказательства по индукции	5
1.5. Числа Фибоначчи	5
1.6. (*) $\mathcal{O}$ -обозначения через пределы	5
1.7. (*) Замена сумм на интегралы	6
1.8. Примеры по теме асимптотики	7
1.9. Сравнение асимптотик	8

## Лекция #1: Асимптотика

3 сентября

1.1.  $\mathcal{O}$ -обозначенияРассмотрим функции  $f, g: \mathbb{N} \rightarrow \mathbb{R}^{>0}$ .**Def 1.1.1.**  $f = \Theta(g) \quad \exists N > 0, C_1 > 0, C_2 > 0: \forall n \geq N, C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n)$ **Def 1.1.2.**  $f = \mathcal{O}(g) \quad \exists N > 0, C > 0: \forall n \geq N, f(n) \leq C \cdot g(n)$ **Def 1.1.3.**  $f = \Omega(g) \quad \exists N > 0, C > 0: \forall n \geq N, f(n) \geq C \cdot g(n)$ **Def 1.1.4.**  $f = o(g) \quad \forall C > 0 \exists N > 0: \forall n \geq N, f(n) \leq C \cdot g(n)$ **Def 1.1.5.**  $f = \omega(g) \quad \forall C > 0 \exists N > 0: \forall n \geq N, f(n) \geq C \cdot g(n)$ Понимание  $\Theta$ : “равны с точностью до константы”, “асимптотически равны”.Понимание  $\mathcal{O}$ : “не больше с точностью до константы”, “асимптотически не больше”Понимание  $o$ : “асимптотически меньше”, “для сколь угодно малой константы не больше”

$\Theta$	$\mathcal{O}$	$\Omega$	$o$	$\omega$
$=$	$\leq$	$\geq$	$<$	$>$

*Замечание 1.1.6.*  $f = \Theta(g) \Leftrightarrow g = \Theta(f)$ *Замечание 1.1.7.*  $f = \mathcal{O}(g), g = \mathcal{O}(f) \Leftrightarrow f = \Theta(g)$ *Замечание 1.1.8.*  $f = \Omega(g) \Leftrightarrow g = \mathcal{O}(f)$ *Замечание 1.1.9.*  $f = \omega(g) \Leftrightarrow g = o(f)$ *Замечание 1.1.10.*  $f = \mathcal{O}(g), g = \mathcal{O}(h) \Rightarrow f = \mathcal{O}(h)$ *Замечание 1.1.11.* Обобщение:  $\forall \beta \in \{\mathcal{O}, o, \Theta, \Omega, \omega\}: f = \beta(g), g = \beta(h) \Rightarrow \boxed{f = \beta(h)}$ *Замечание 1.1.12.*  $\forall C > 0 \quad C \cdot f = \Theta(f)$ 

Докажем для примера 1.1.6.

*Доказательство.*  $C_1 \cdot g(n) \leq f(n) \leq C_2 \cdot g(n) \Rightarrow \frac{1}{C_2} f(n) \leq g(n) \leq \frac{1}{C_1} g(n) \leq f(n)$  ■**Упражнение 1.1.13.**  $f = \mathcal{O}(\Theta(\mathcal{O}(g))) \Rightarrow f = \mathcal{O}(g)$ **Упражнение 1.1.14.**  $f = \Theta(o(\Theta(\mathcal{O}(g)))) \Rightarrow f = o(g)$ **Упражнение 1.1.15.**  $f = \Omega(\omega(\Theta(g))) \Rightarrow f = \omega(g)$ **Упражнение 1.1.16.**  $f = \Omega(\Theta(\mathcal{O}(g))) \Rightarrow f$  может быть любой функцией**Лм 1.1.17.**  $g = o(f) \Rightarrow f \pm g = \Theta(f)$ *Доказательство.*  $g = o(f) \exists N: \forall n \geq N \quad g(n) \leq \frac{1}{2} f(n) \Rightarrow \frac{1}{2} f(n) \leq f(n) \pm g(n) \leq \frac{3}{2} f(n)$  ■**Лм 1.1.18.**  $n^k = o(n^{k+1})$ *Доказательство.*  $\forall C \forall n \geq C \quad n^{k+1} \geq C \cdot n^k$  ■**Лм 1.1.19.**  $P(x)$  – многочлен, тогда  $P(x) = \Theta(x^{\deg P})$  при старшем коэффициенте  $> 0$ .*Доказательство.*  $P(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_k x^k$ . По леммам 1.1.12, 1.1.18 имеем, что все слагаемые кроме  $a_k x^k$  являются  $o(x^{\deg P})$ . Поэтому по лемме 1.1.17 вся сумма является  $\Theta(x^k)$ . ■

## 1.2. Рекуррентности и Карацуба

### • Алгоритм умножения чисел в столбик

Рассмотрим два многочлена  $A(x) = 5 + 4x + 3x^2 + 2x^3 + x^4$  и  $B(x) = 9 + 8x + 7x^2 + 6x^3$ .  
Запишем массивы  $a[] = \{5, 4, 3, 2, 1\}$ ,  $b[] = \{9, 8, 7, 6\}$ .

```
1 for (i = 0; i < an; i++) // an = 5
2   for (j = 0; j < bn; j++) // bn = 4
3     c[i + j] += a[i] * b[j];
```

Мы получили в точности коэффициенты многочлена  $C(x) = A(x)B(x)$ .

Теперь рассмотрим два числа  $A = 12345$  и  $B = 6789$ , запишем те же массивы и сделаем:

```
1 // Перемножаем числа без переносов, как многочлены
2 for (i = 0; i < an; i++) // an = 5
3   for (j = 0; j < bn; j++) // bn = 4
4     c[i + j] += a[i] * b[j];
5 // Делаем переносы, массив c = [45, 76, 94, 100, 70, 40, 19, 6, 0]
6 for (i = 0; i < an + bn; i++)
7   if (c[i] >= 10)
8     c[i + 1] += c[i] / 10, c[i] %= 10;
9 // Массив c = [5, 0, 2, 0, 1, 8, 3, 8, 0], ответ = 83810205
```

Данное умножение работает за  $\Theta(nm)$ , или  $\Theta(n^2)$  в случае  $n = m$ .

*Следствие 1.2.1.* Чтобы умножать длинные числа достаточно уметь умножать многочлены.

Многочлены мы храним, как массив коэффициентов. При программировании умножения, нам важно знать не степень многочлена  $d$ , а длину этого массива  $n = d + 1$ .

### • Алгоритм Карацубы

Чтобы перемножить два многочлена (или два длинных целых числа)  $A(x)$  и  $B(x)$  из  $n$  коэффициентов каждый, разделим их на части по  $k = \frac{n}{2}$  коэффициентов —  $A_1, A_2, B_1, B_2$ .

Заметим, что  $A \cdot B = (A_1 + x^k A_2)(B_1 + x^k B_2) = A_1 B_1 + x^k (A_1 B_2 + A_2 B_1) + x^{2k} A_2 B_2$ .

Если написать рекурсивную функцию умножения, то получим время работы:

$$T_1(n) = 4T_1\left(\frac{n}{2}\right) + \Theta(n)$$

Из последующей теоремы мы сделаем вывод, что  $T_1(n) = \Theta(n^2)$ . Алгоритм можно улучшить, заметив, что  $A_1 B_2 + A_2 B_1 = (A_1 + A_2)(B_1 + B_2) - A_1 B_1 - A_2 B_2$ , где вычитаемые величины уже посчитаны. Итого три умножения вместо четырёх:

$$T_2(n) = 3T_2\left(\frac{n}{2}\right) + \Theta(n)$$

Из последующей теоремы мы сделаем вывод, что  $T_2(n) = \Theta(n^{\log_2 3}) = \Theta(n^{1.585\dots})$ .

Данный алгоритм применим и для умножения многочленов, и для умножения чисел.

Псевдокод алгоритма Карацубы для умножения многочленов:

```
1 //  $n = 2^k$ ,  $c(w) = a(w)*b(w)$ 
2 Mul(n, a, b) {
3   if n == 1: return {a[0] * b[0]}
4   a --> a1, a2
5   b --> b1, b2
6   x = Mul(n / 2, a1, b1)
7   y = Mul(n / 2, a2, b2)
8   z = Mul(n / 2, a1 + a2, b1 + b2)
9   // Умножение на  $w^i$  - сдвиг массива на  $i$  вправо
10  return x + y *  $w^n$  + (z - x - y) *  $w^{\{n/2\}}$ ;
11 }
```

Чтобы умножить числа, сперва умножим их как многочлены, затем сделаем переносы.

### 1.3. Теоремы о рекуррентных соотношениях

**Теорема 1.3.1.** *Мастер Теорема* (теорема о простом рекуррентном соотношении)

Пусть  $T(n) = aT(\frac{n}{b}) + f(n)$ , где  $f(n) = n^c$ . При этом  $a > 0, b > 1, c \geq 0$ . Определим глубину рекурсии  $k = \log_b n$ . Тогда верно одно из трёх:

$$\begin{cases} T(n) = \Theta(a^k) = \Theta(n^{\log_b a}) & a > b^c \\ T(n) = \Theta(f(n)) = \Theta(n^c) & a < b^c \\ T(n) = \Theta(k \cdot f(n)) = \Theta(n^c \log n) & a = b^c \end{cases}$$

*Доказательство.* Раскроем рекуррентность:

$$T(n) = f(n) + aT(\frac{n}{b}) = f(n) + af(\frac{n}{b}) + a^2f(\frac{n}{b^2}) + \dots = n^c + a(\frac{n}{b})^c + a^2(\frac{n}{b^2})^c + \dots$$

Тогда  $T(n) = f(n)(1 + \frac{a}{b^c} + (\frac{a}{b^c})^2 + \dots + (\frac{a}{b^c})^k)$ . При этом в сумме  $k + 1$  слагаемых.

Обозначим  $q = \frac{a}{b^c}$  и оценим сумму  $S(q) = 1 + q + \dots + q^k$ .

Если  $q = 1$ , то  $S(q) = k + 1 = \log_b n + 1 = \Theta(\log_b n) \Rightarrow T(n) = \Theta(f(n) \log n)$ .

Если  $q < 1$ , то  $S(q) = \frac{1 - q^{k+1}}{1 - q} = \Theta(1) \Rightarrow T(n) = \Theta(f(n))$ .

Если  $q > 1$ , то  $S(q) = q^k + \frac{q^k - 1}{q - 1} = \Theta(q^k) \Rightarrow T(n) = \Theta(a^k (\frac{n}{b^k})^c) = \Theta(a^k)$ . ■

**Теорема 1.3.2.** *Обобщение Мастер Теоремы*

Мастер Теорема верна и для  $f(n) = n^c \log^d n$

$T(n) = aT(\frac{n}{b}) + n^c \log^d n$ . При  $a > 0, b > 1, c \geq 0, d \geq 0$ .

$$\begin{cases} T(n) = \Theta(a^k) = \Theta(n^{\log_b a}) & a > b^c \\ T(n) = \Theta(f(n)) = \Theta(n^c \log^d n) & a < b^c \\ T(n) = \Theta(k \cdot f(n)) = \Theta(n^c \log^{d+1} n) & a = b^c \end{cases}$$

Без доказательства. ■

**Теорема 1.3.3.** *Об экспоненциальном рекуррентном соотношении*

Пусть  $T(n) = \sum b_i T(n - a_i)$ . При этом  $a_i > 0, b_i > 0, \sum b_i > 1$ .

Тогда  $T(n) = \Theta(\alpha^n)$ , при этом  $\alpha > 1$  и является корнем уравнения  $1 = \sum b_i \alpha^{-a_i}$ , его можно найти бинарным поиском.

*Доказательство.* Предположим, что  $T(n) = \alpha^n$ , тогда  $\alpha^n = \sum b_i \alpha^{n - a_i} \Leftrightarrow 1 = \sum b_i \alpha^{-a_i} = f(\alpha)$ .

Теперь нам нужно решить уравнение  $f(\alpha) = 1$  для  $\alpha \in [1, +\infty)$ .

Если  $\alpha = 1$ , то  $f(\alpha) = \sum b_i > 1$ , если  $\alpha = +\infty$ , то  $f(\alpha) = 0 < 1$ . Кроме того  $f(\alpha) \searrow [1, +\infty)$ .

Получаем, что на  $[1, +\infty)$  есть единственный корень уравнения  $1 = f(\alpha)$  и его можно найти бинарным поиском.

Мы показали, откуда возникает уравнение  $1 = \sum b_i \alpha^{-a_i}$ . Доказали, что у него  $\exists!$  корень  $\alpha$ .

Теперь докажем по индукции, что  $T(n) = \mathcal{O}(\alpha^n)$  (оценку сверху) и  $T(n) = \Omega(\alpha^n)$  (оценку снизу). Доказательства идентичны, покажем  $T(n) = \mathcal{O}(\alpha^n)$ . База индукции:

$$\exists C: \forall n \in B = [1 - \max_i a_i, 1] \quad T(n) \leq C \alpha^n$$

Переход индукции:

$$T(n) = \sum b_i T(n - a_i) \stackrel{\text{по индукции}}{\leq} C \sum b_i \alpha^{n - a_i} \stackrel{(*)}{=} C \alpha^n$$

(\*) Верно, так как  $\alpha$  – корень уравнения. ■

## 1.4. Доказательства по индукции

### Lm 1.4.1. Доказательство по индукции

Есть простой метод решения рекуррентных соотношений: угадать ответ, доказать его по индукции. Рассмотрим на примере  $T(n) = \max_{x=1..n-1} (T(x) + T(n-x) + x(n-x))$ .

Докажем, что  $T(n) = \mathcal{O}(n^2)$ , для этого достаточно доказать  $T(n) \leq n^2$ :

База:  $T(1) = 1 \leq 1^2$ .

Переход:  $T(n) \leq \max_{x=1..n-1} (x^2 + (n-x)^2 + x(n-x)) \leq \max_{x=1..n-1} (x^2 + (n-x)^2 + 2x(n-x)) = n^2$

### • Примеры по теме рекуррентные соотношения

1.  $T(n) = T(n-1) + T(n-1) + T(n-2)$ .

Угадаем ответ  $2^n$ , проверим по индукции:  $2^n = 2^{n-1} + 2^{n-1} + 2^{n-2}$ .

2.  $T(n) = T(n-3) + T(n-3) \Rightarrow T(n) = 2T(n-3) = 4T(n-6) = \dots = 2^{n/3}$

3.  $T(n) = T(n-1) + T(n-3)$ . Применяем 1.3.3, получаем  $1 = \alpha^{-1} + \alpha^{-3}$ , находим  $\alpha$  бинпоиском, получаем  $\alpha = 1.4655\dots$

## 1.5. Числа Фибоначчи

**Def 1.5.1.**  $f_1 = f_0 = 1, f_i = f_{i-1} + f_{i-2}$ .  $f_n$  -  $n$ -е число Фибоначчи.

### • Оценки снизу и сверху

$f_n = f_{n-1} + f_{n-2}$ , рассмотрим  $g_n = g_{n-1} + g_{n-1}$ ,  $2^n = g_n \geq f_n$ .

$f_n = f_{n-1} + f_{n-2}$ , рассмотрим  $g_n = g_{n-2} + g_{n-2}$ ,  $2^{n/2} = g_n \leq f_n$ .

Воспользуемся 1.3.3, получим  $1 = \alpha^{-1} + \alpha^{-2} \Leftrightarrow \alpha^2 - \alpha - 1 = 0$ , получаем  $\alpha = \frac{\sqrt{5}+1}{2} \approx 1.618$ .

$f_n = \Theta(\alpha^n)$ .

## 1.6. (\*) $\mathcal{O}$ -обозначения через пределы

**Def 1.6.1.**  $f = o(g)$  Определение через предел:  $\lim_{n \rightarrow +\infty} \frac{f(n)}{g(n)} = 0$

**Def 1.6.2.**  $f = \mathcal{O}(g)$  Определение через предел:  $\overline{\lim}_{n \rightarrow +\infty} \frac{f(n)}{g(n)} < \infty$

Здесь необходимо пояснение:  $\overline{\lim}_{n \rightarrow +\infty} f(n) = \lim_{n \rightarrow +\infty} (\sup_{x \in [n..+\infty]} f(x))$ , где  $\sup$  - верхняя грань.

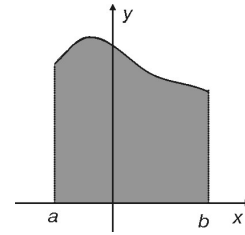
### Lm 1.6.3. Определения $\mathcal{O}$ эквивалентны

*Доказательство.* Вспомним, что речь о положительных функциях  $f$  и  $g$ .

Распишем предел по определению:  $\forall C > 0 \quad \exists N \quad \forall n \geq N \quad \frac{f(n)}{g(n)} \leq C \Leftrightarrow f(n) \leq Cg(n)$ . ■

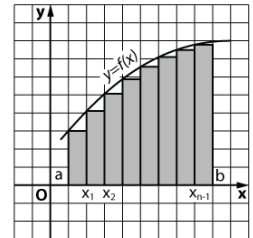
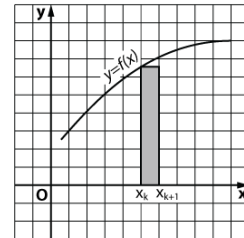
### 1.7. (\*) Замена сумм на интегралы

**Def 1.7.1.** Определённый интеграл  $\int_a^b f(x)dx$  положительной функции  $f(x)$  – площадь под графиком  $f$  на отрезке  $[a..b]$ .



**Lm 1.7.2.**  $\forall f(x) \nearrow [a..a+1] \Rightarrow f(a) \leq \int_a^{a+1} f(x)dx \leq f(a+1)$

**Lm 1.7.3.**  $\forall f(x) \nearrow [a..b+1] \Rightarrow \sum_{i=a}^b f(i) \leq \int_a^{b+1} f(x)dx$



*Доказательство.* Сложили неравенства из 1.7.2 ■

**Lm 1.7.4.**  $\forall f(x) \nearrow [a..b], f > 0 \Rightarrow \int_a^b f(x)dx \leq \sum_{i=a}^b f(i)$

*Доказательство.* Сложили неравенства из 1.7.2, выкинули  $[a-1, a]$  из интеграла. ■

**Теорема 1.7.5.** Замена суммы на интеграл #1

$$\forall f(x) \nearrow [1..∞), f > 0, S(n) = \sum_{i=1}^n f(i), I_1(n) = \int_1^n, I_2(n) = \int_1^{n+1}, I_1(n) = \Theta(I_2(n)) \Rightarrow S(n) = \Theta(I_1(n))$$

*Доказательство.* Из лемм 1.7.3 и 1.7.4 имеем  $I_1(n) \leq S(n) \leq I_2(n)$ .

$$C_1 I_1(n) \leq I_2(n) \leq C_2 I_1(n) \Rightarrow I_1(n) \leq S(n) \leq I_2(n) \leq C_2 I_1(n)$$

**Теорема 1.7.6.** Замена суммы на интеграл #2

$$\forall f(x) \nearrow [a..b], f > 0 \quad \int_a^b f(x)dx \leq \sum_{i=a}^b f(i) \leq f(b) + \int_a^b f(x)dx$$

*Доказательство.* Первое неравенство – лемма 1.7.3. Второе – 1.7.4, применённая к  $\sum_{i=a}^{b-1}$ . ■

*Следствие 1.7.7.* Для убывающих функций два последних факта тоже верны. Во втором ошибкой будет не  $f(b)$ , а  $f(a)$ , которое теперь больше.

• Как считать интегралы?

Формула Ньютона-Лейбница:  $\int_a^b f'(x)dx = f(b) - f(a)$

Пример:  $\ln'(n) = \frac{1}{n} \Rightarrow \int_1^n \frac{1}{x}dx = \ln n - \ln 1 = \ln n$

## 1.8. Примеры по теме асимптотики

### • Вложенные циклы for

```

1 #define forn(i, n) for (int i = 0; i < n; i++)
2 int counter = 0, n = 100;
3 forn(i, n)
4     forn(j, i)
5         forn(k, j)
6             forn(l, k)
7                 forn(m, l)
8                     counter++;
9 cout << counter << endl;

```

Чему равен counter? Во-первых, есть точный ответ:  $\binom{n}{5} \approx \frac{n^5}{5!}$ . Во-вторых, мы можем сходно посчитать число циклов и оценить ответ как  $\mathcal{O}(n^5)$ , правда константа  $\frac{1}{120}$  важна, оценка через  $\mathcal{O}$  не даёт полное представление о времени работы.

### • За сколько вычисляется $n$ -е число Фибоначчи?

```

1 f[0] = f[1] = 1;
2 for (int i = 2; i < n; i++)
3     f[i] = f[i - 1] + f[i - 2];

```

Казалось бы за  $\mathcal{O}(n)$ . Но это в предположении, что “+” выполняется за  $\mathcal{O}(1)$ . На самом деле мы знаем, что  $\log f_n = \Theta(n)$ , т.е. складывать нужно числа длины  $n \Rightarrow$  “+” выполняется за  $\Theta(i)$ , а  $n$ -е число Фибоначчи считается за  $\Theta(n^2)$ .

### • Задача из теста про $a^2 + b^2 = N$

```

1 int b = sqrt(N);
2 for (int a = 1; a * a <= N; a++)
3     while (a * a + b * b >= N; b--)
4         ;
5     if (a * a + b * b == N)
6         cnt++;

```

Время работы  $\Theta(N^{1/2})$ , так как в сумме  $b$  уменьшится лишь  $N^{1/2}$  раз. Здесь мы первый раз использовали так называемый “метод двух указателей”.

### • Число делителей числа

```

1 vector<int> divisors[n + 1]; // все делители числа
2 for (int a = 1; a <= n; a++)
3     for (int b = a; b <= n; b += a)
4         divisors[b].push_back(a);

```

За сколько работает программа?

$$\sum_{a=1}^n \left\lceil \frac{n}{a} \right\rceil = \mathcal{O}(n) + \sum_{a=1}^n \frac{n}{a} = \mathcal{O}(n) + n \sum_{a=1}^n \frac{1}{a} \stackrel{1.7.5}{=} \mathcal{O}(n) + n \cdot \Theta\left(\int_1^n \frac{1}{x} dx\right) = \Theta(n \log n)$$



• Сумма гармонического ряда

Докажем более простым способом, что  $\sum_{i=1}^n \frac{1}{i} = \Theta(\log n)$

$$1 + \lfloor \log_2 n \rfloor \geq \frac{1}{1} + \overbrace{\frac{1}{2} + \frac{1}{2}}^1 + \overbrace{\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4}}^1 + \overbrace{\frac{1}{8} + \dots}^{1 \dots} \geq \sum_{k=1}^n \frac{1}{k} = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} + \frac{1}{8} + \dots \geq$$

$$\frac{1}{1} + \underbrace{\frac{1}{2}}_{1/2} + \underbrace{\frac{1}{4} + \frac{1}{4}}_{1/2} + \underbrace{\frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8}}_{1/2} + \dots \geq 1 + \frac{1}{2} \lfloor \log_2 n \rfloor \Rightarrow \sum_{k=1}^n \frac{1}{k} = \Theta(\log n)$$

1.9. Сравнение асимптотик

- Def 1.9.1. *Линейная сложность*  $\mathcal{O}(n)$
- Def 1.9.2. *Квадратичная сложность*  $\mathcal{O}(n^2)$
- Def 1.9.3. *Полиномиальная сложность*  $\exists k > 0: \mathcal{O}(n^k)$
- Def 1.9.4. *Полилогарифм*  $\exists k > 0: \mathcal{O}(\log^k n)$
- Def 1.9.5. *Экспоненциальная сложность*  $\exists c > 0: \mathcal{O}(2^{cn})$

**Теорема 1.9.6.**  $\forall x, y > 0, z > 1 \exists N \forall n > N: \log^x n < n^y < z^n$

*Доказательство.* Сперва докажем первую часть неравенства через вторую.

Пусть  $\log n = k$ , тогда  $\log^x n < n^y \Leftrightarrow k^x < 2^{ky} = (2^y)^k = z^k \Leftrightarrow n^y < z^n$  ■

Докажем вторую часть исходного неравенства  $n^y < z^n \Leftrightarrow n < 2^{\frac{1}{y} n \log z}$

Пусть  $n' = \frac{1}{y} n \log z$ , обозначим  $C = 1/(\frac{1}{y} \log z)$ , пусть  $C \leq n'$  (возьмём достаточно большое  $n$ ), тогда  $n^y < z^n \Leftrightarrow n < 2^{\frac{1}{y} n \log z} \Leftrightarrow C \cdot n' < 2^{n'} \Leftrightarrow (n')^2 < 2^{n'}$

Осталось доказать  $n^2 < 2^n$ . Докажем по индукции.

База: для любого значения из интервала  $[10..20)$  верно, так как  $n^2 \in [100..400) < 2^n \in [1024..1048576)$ .

Если  $n$  увеличить в два раза, то  $n^2 \rightarrow 4 \cdot n^2$ , а  $2^n \rightarrow 2^{2n} = 2^n \cdot 2^n \geq 4 \cdot 2^n$  при  $n \geq 2$ .

Значит  $\forall n \geq 2$  если для  $n$  верно, то и для  $2n$  верно.

Переход:  $[10..20) \rightarrow [20..40) \rightarrow [40..80) \rightarrow \dots$  ■

*Следствие 1.9.7.*  $\forall x, y > 0, z > 1: \log^x n = \mathcal{O}(n^y), n^y = \mathcal{O}(z^n)$

*Доказательство.* Возьмём константу 1. ■

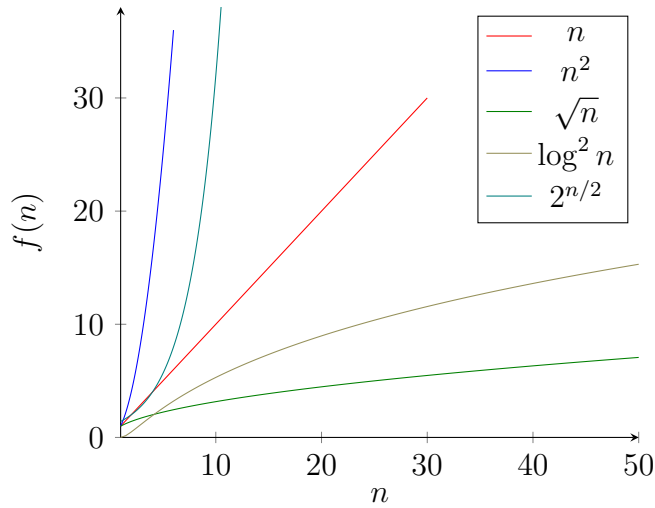
*Следствие 1.9.8.*  $\forall x, y > 0, z > 1: \log^x n = o(n^y), n^y = o(z^n)$

*Доказательство.* Достаточно перейти к чуть меньшим  $y, z$  и воспользоваться теоремой.

$\exists N \forall n \geq N \log^x n < n^{y-\varepsilon} = \frac{1}{n^\varepsilon} n^y, \frac{1}{n^\varepsilon} \xrightarrow{n \rightarrow \infty} 0 \Rightarrow \log^x n = o(n^y)$ .

$\exists N \forall n \geq N n^y < (z - \varepsilon)^n = \frac{1}{(z/(z-\varepsilon))^n} z^n, \frac{1}{(z/(z-\varepsilon))^n} \xrightarrow{n \rightarrow \infty} 0 \Rightarrow n^y = o(z^n)$ . ■

• Посмотрим как ведут себя функции на графике



Заметим, что  $2^{n/2}$ ,  $n^2$  и  $\log^2 n$ ,  $\sqrt{n}$  на бесконечности ведут себя иначе:

