Оценка больших величин

Оценка больших величин

Оценка больших величин

1 корешок = $10000\$$

Оценка больших величин

1 корешок = $10000\$$

ширина ≈ 35

Оценка больших величин

1 корешок = $10000\$$

ширина ≈ 35

глубина ≈ 15

Оценка больших величин

1 корешок = $10000\$$

ширина ≈ 35

глубина ≈ 15

высота ≈ 35

Оценка больших величин

1 корешок = $10000\$$

ширина ≈ 35

глубина ≈ 15

высота ≈ 35

$$10000 \cdot 35 \cdot 15 \cdot 35 =$$
$$= 183750000$$

Оценка больших величин

1 корешок = $10000\$$

ширина ≈ 35

глубина ≈ 15

высота ≈ 35

$$10000 \cdot 35 \cdot 15 \cdot 35 =$$
$$= 183750000$$

На самом деле:

$$\approx 207000000\$$$

Правила суммирования

Правила суммирования

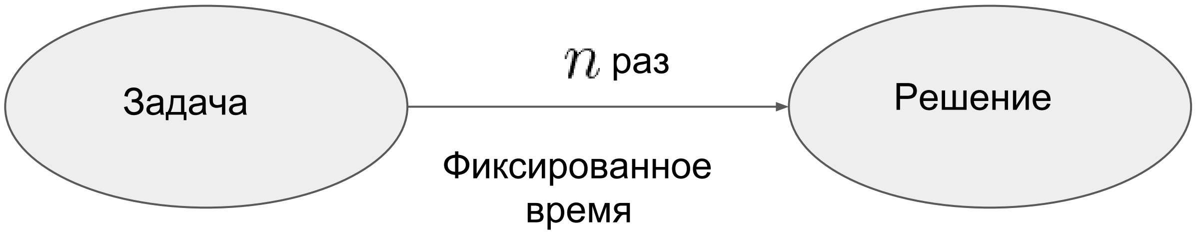
$$\sum_{i=0}^{n} 1 = \underbrace{1 + 1 + \cdots + 1}_{n+1} = n + 1$$

Правила суммирования

$$\sum_{i=0}^{n} 1 = \underbrace{1 + 1 + \cdots + 1}_{n+1} = n + 1$$

Правила суммирования

$$\sum_{i=0}^{n} 1 = \underbrace{1 + 1 + \cdots + 1}_{n+1} = n + 1$$

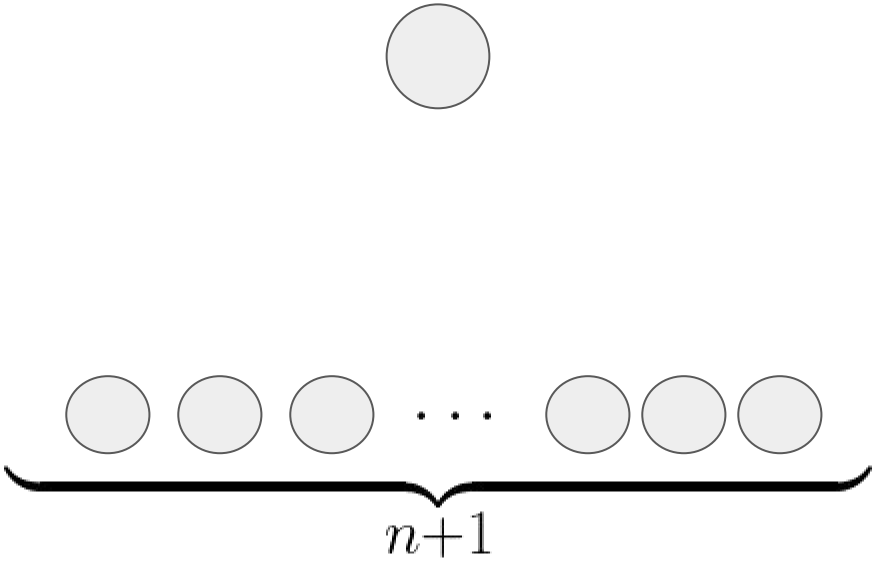


Правила суммирования

$$\sum_{i=0}^{n} n = \underbrace{n + n + \cdots + n}_{n+1} = n \cdot (n+1)$$

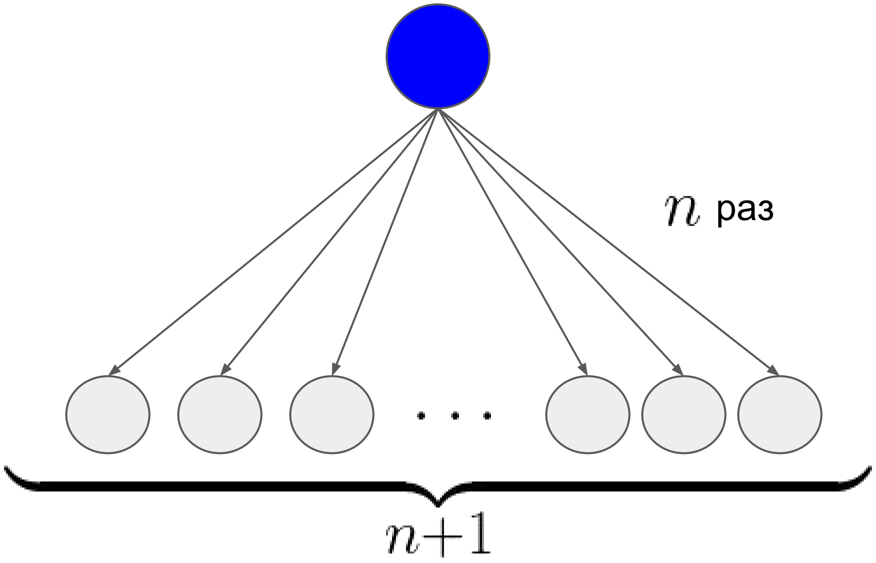
Правила суммирования

$$\sum_{i=0}^{n} n = \underbrace{n + n + \cdots + n}_{n+1} = n \cdot (n+1)$$



Правила суммирования

$$\sum_{i=0}^{n} n = \underbrace{n + n + \cdots + n}_{n+1} = n \cdot (n+1)$$

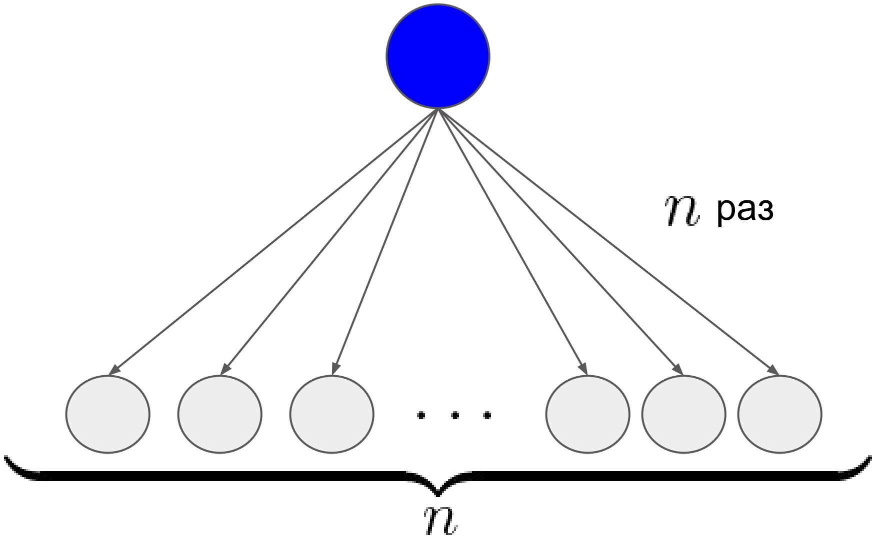


Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$

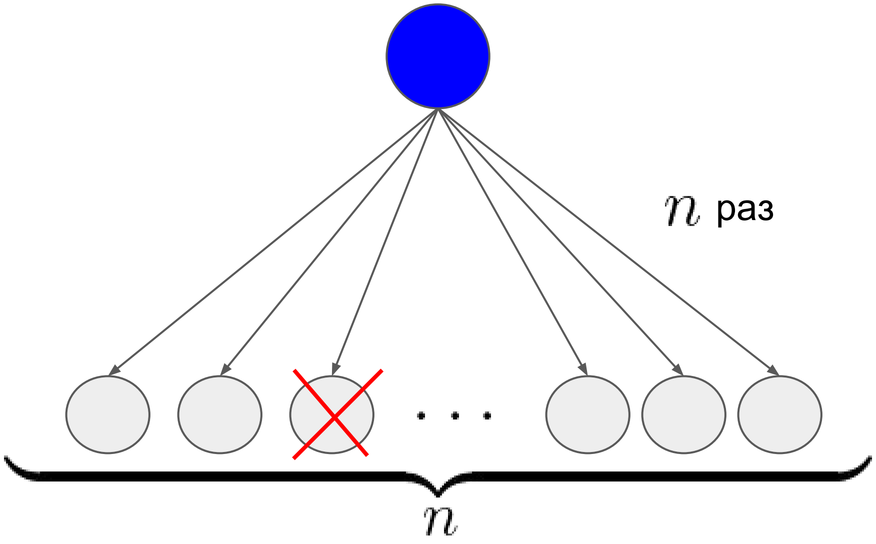
Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$



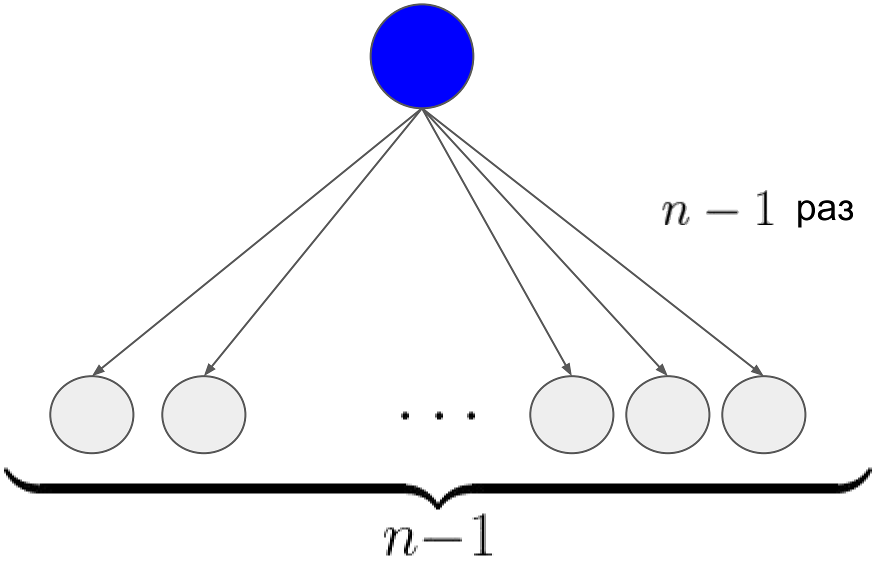
Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$



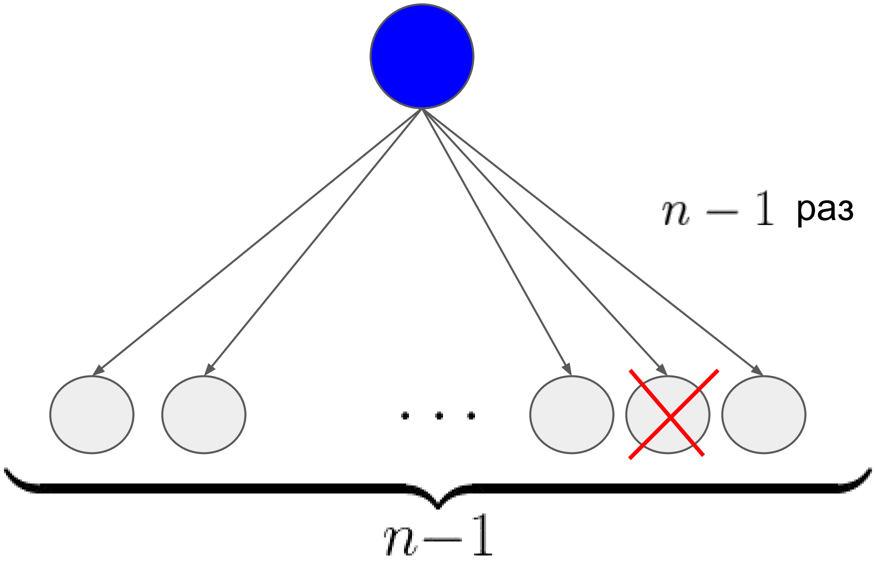
Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$



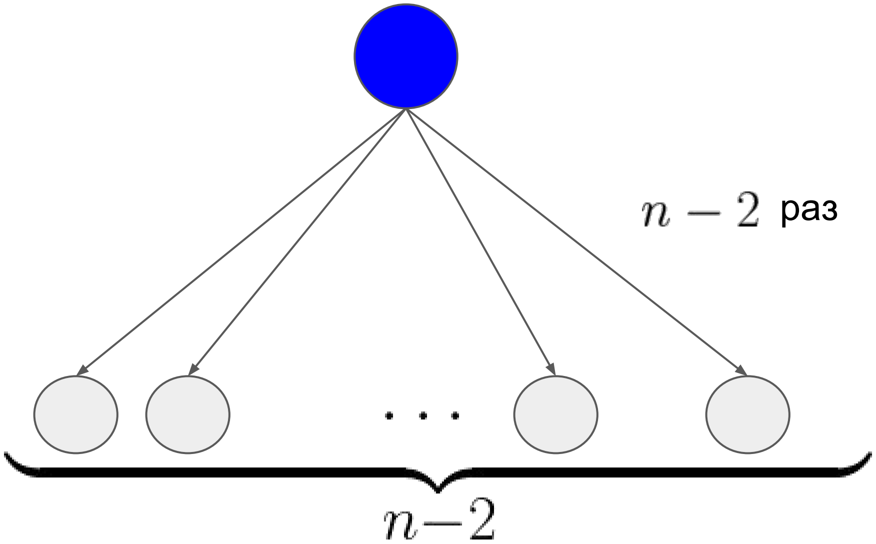
Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$



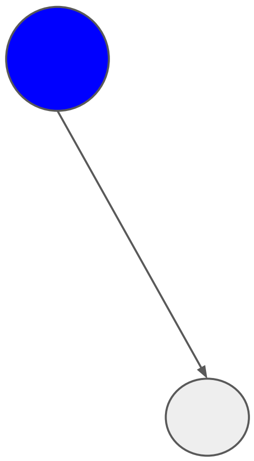
Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$



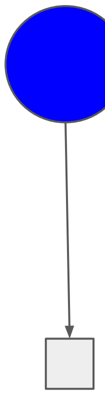
Правила суммирования

$$\sum_{i=0}^{n} i = 1 + 2 + \cdots + n = \frac{n(n-1)}{2}$$



Правила суммирования

$$\sum_{i=0}^{n} \alpha^i = \alpha^0 + \alpha^1 + \cdots + \alpha^n = \frac{\alpha^{n+1} - 1}{\alpha - 1}$$



Перебор пароля при алфавите размера α
и не более n символов длиной

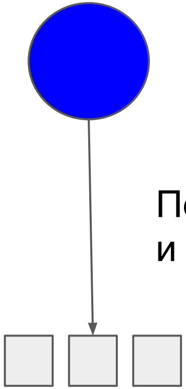
Правила суммирования

$$\sum_{i=0}^{n} \alpha^i = \alpha^0 + \alpha^1 + \dots + \alpha^n = \frac{\alpha^{n+1} - 1}{\alpha - 1}$$

Перебор пароля при алфавите размера α
и не более n символов длиной

Правила суммирования

$$\sum_{i=0}^{n} \alpha^i = \alpha^0 + \alpha^1 + \cdots + \alpha^n = \frac{\alpha^{n+1} - 1}{\alpha - 1}$$



Перебор пароля при алфавите размера α
и не более n символов длиной

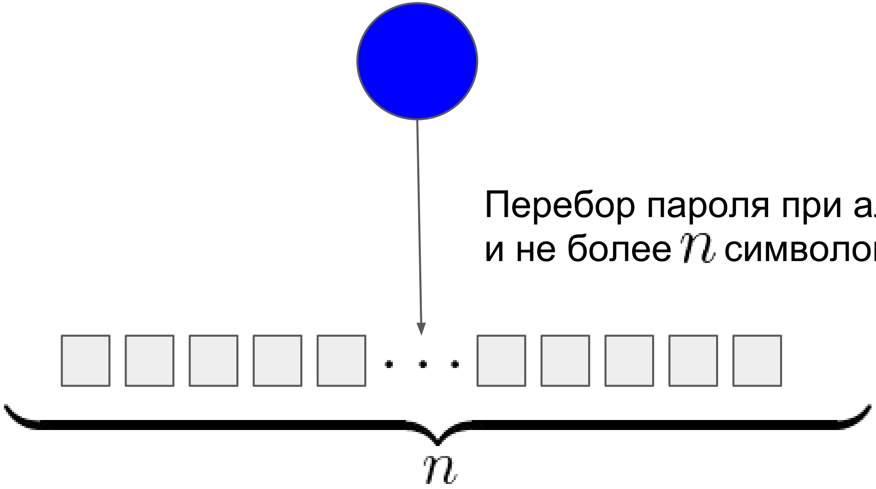
Правила суммирования

$$\sum_{i=0}^{n} \alpha^i = \alpha^0 + \alpha^1 + \cdots + \alpha^n = \frac{\alpha^{n+1} - 1}{\alpha - 1}$$

Перебор пароля при алфавите размера α
и не более n символов длиной

Правила суммирования

$$\sum_{i=0}^{n} \alpha^i = \alpha^0 + \alpha^1 + \cdots + \alpha^n = \frac{\alpha^{n+1} - 1}{\alpha - 1}$$



Перебор пароля при алфавите размера α
и не более n символов длиной

Правила суммирования

$$\sum_{i=1}^{n} \frac{1}{i} = \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n} \approx \ln n + \gamma$$

Логарифмы и экспоненты

Логарифмы и экспоненты

$$e^y$$

Логарифмы и экспоненты

$$e^y \qquad \ln x = log_e x$$

Логарифмы и экспоненты

$$e^y \qquad\qquad \ln x = log_e x$$

$$e^n = m \to \ln m = n$$

Логарифмы и экспоненты

$$e^y \qquad \ln x = log_e x$$

$$e^n = m \rightarrow \ln m = n$$

$$e^m = n \leftarrow \ln n = m$$

Логарифмы и экспоненты

$$e^y \qquad\qquad \ln x = log_e x$$

$$e^n = m \rightarrow \ln m = n$$

$$e^m = n \leftarrow \ln n = m$$

Логарифмы и экспоненты

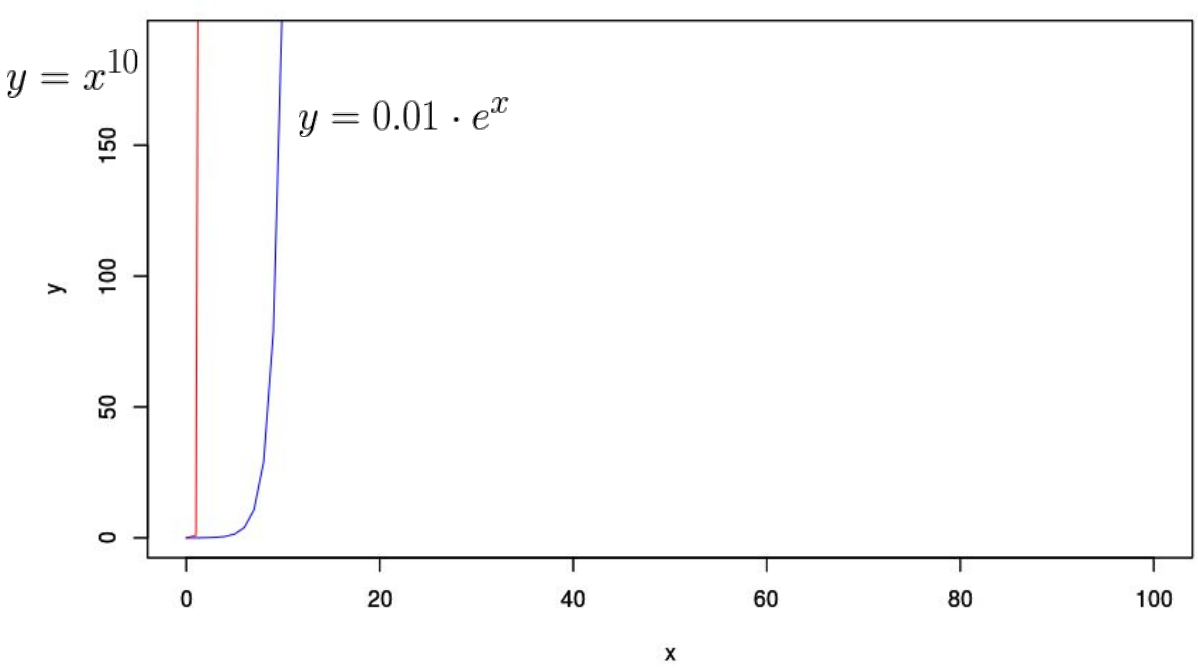
$$e^x \cdot e^y = e^{x+y}$$

Логарифмы и экспоненты

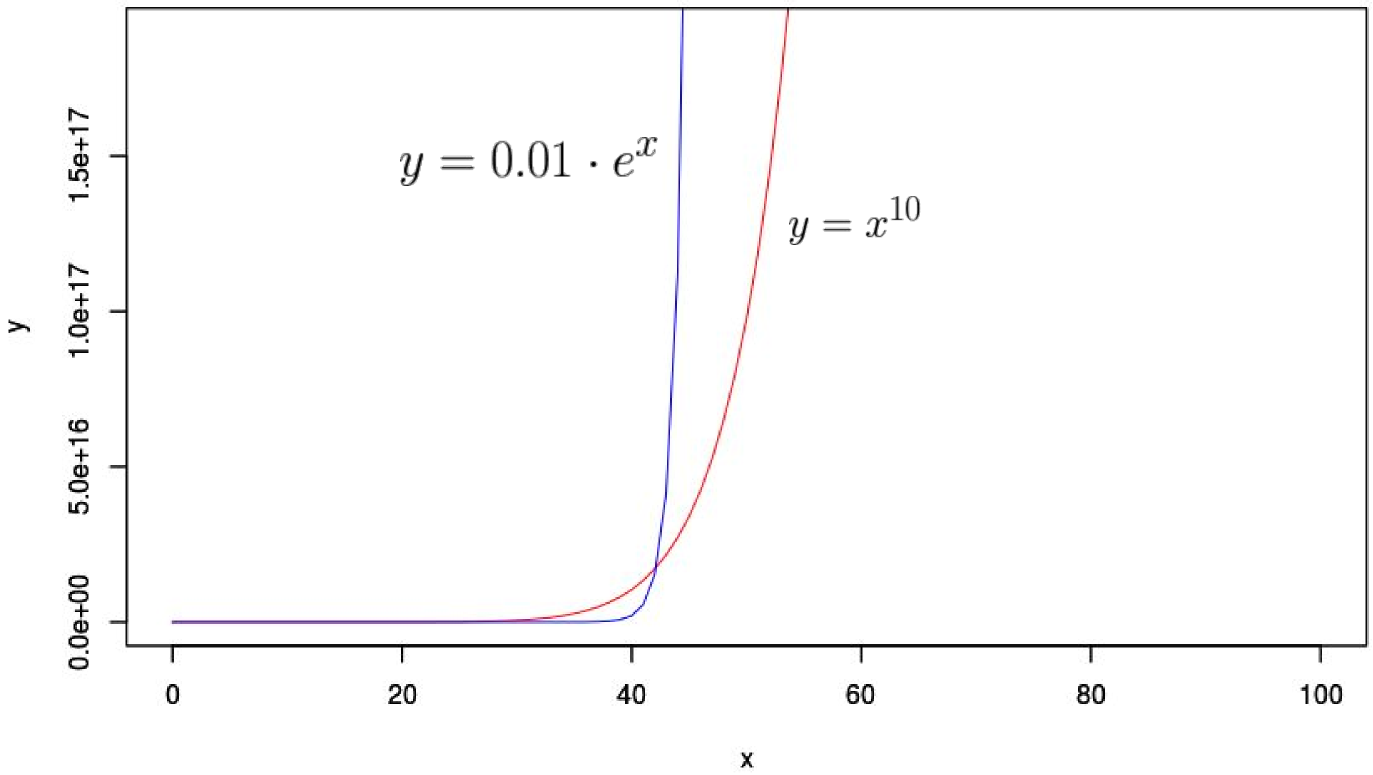
$$e^x \cdot e^y = e^{x+y}$$

$$\ln xy = \ln x + \ln y$$

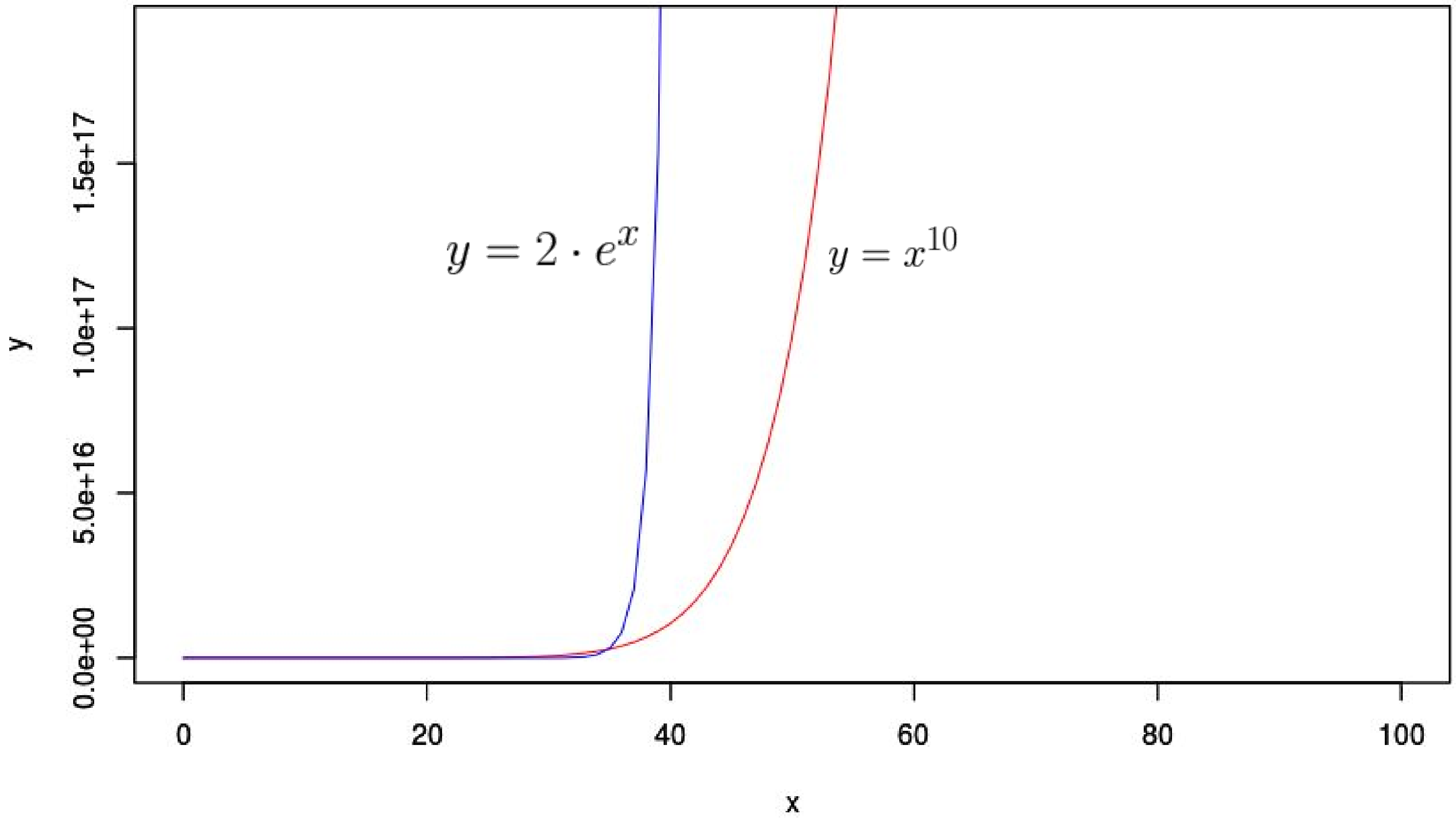
Визуализация лог-лог графиков



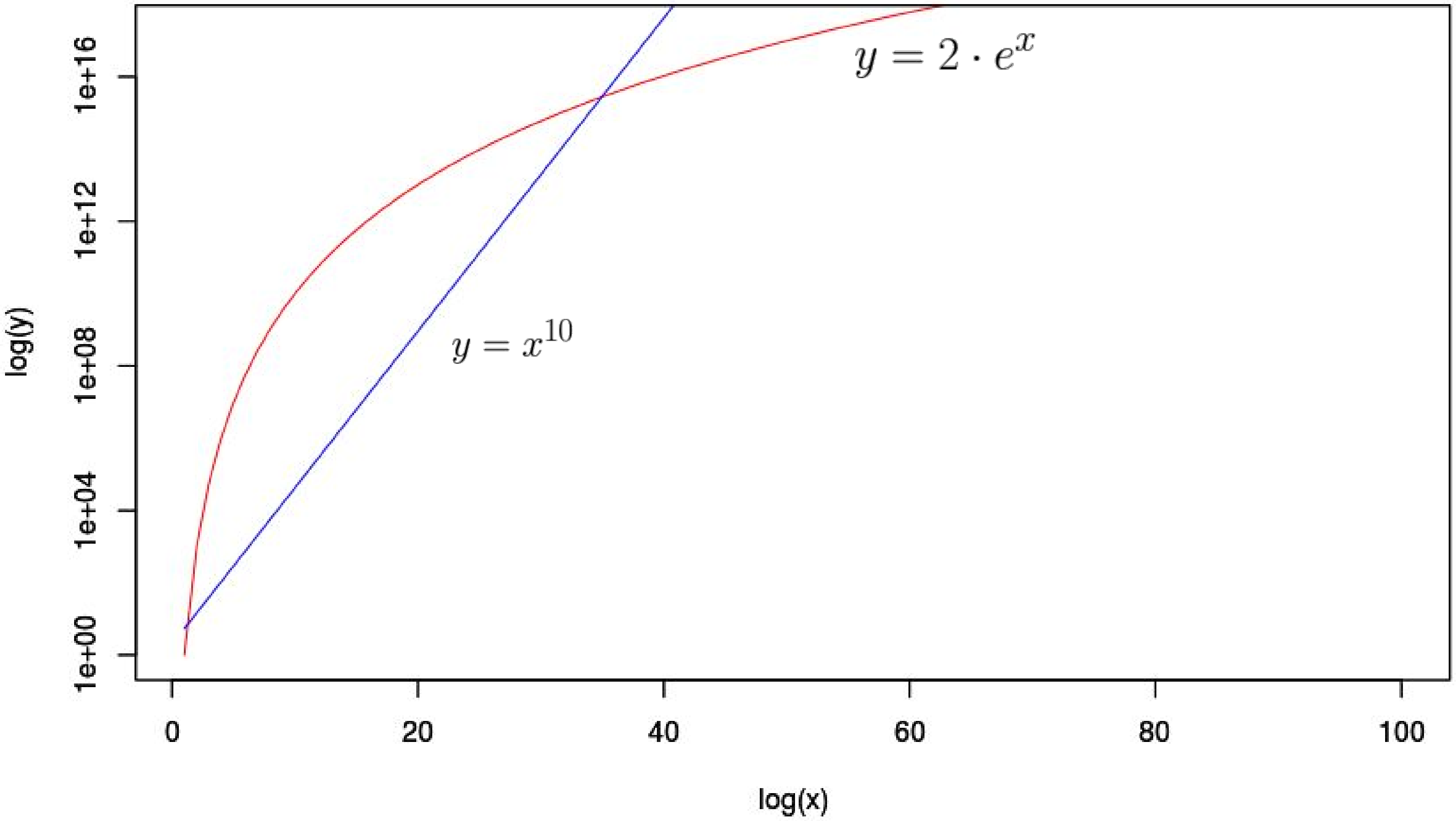
Визуализация лог-лог графиков



Визуализация лог-лог графиков



Визуализация лог-лог графиков



Логарифмирование функций

$$y = ax^n$$

Логарифмирование функций

$$y = ax^n$$

$$\ln y = \ln ax^n$$

Логарифмирование функций

$$y = ax^n$$

$$\ln y = \ln ax^n$$

$$\ln y = \ln a + \ln x^n$$

$$\ln y = \ln a + n \ln x$$

Логарифмирование функций

$$\ln y = \ln a + n \ln x$$

Скорость роста функций

Скорость роста функций

$$f(n)$$

Скорость роста функций

$f(n)$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $g(n)$

Скорость роста функций

$f(n)$ $g(n)$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)}$$

Скорость роста функций

$f(n)$

$g(n)$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)}$$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = C$$

Скорость роста функций

$f(n)$

$g(n)$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)}$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = C$$

Одинаковая скорость роста

Скорость роста функций

$f(n)$ $g(n)$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)}$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = C \qquad \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

Одинаковая скорость роста

Скорость роста функций

$$f(n)$$

$$g(n)$$

$$\lim_{n\to\infty}\frac{f(n)}{g(n)}$$

$$\lim_{n\to\infty}\frac{f(n)}{g(n)} = C$$

Одинаковая скорость роста

$$\lim_{n\to\infty}\frac{f(n)}{g(n)} = 0$$

$$g(n) >> f(n)$$

Скорость роста функций

$$f(n) \qquad\qquad g(n)$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)}$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = C$$

Одинаковая скорость роста

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

$$g(n) >> f(n)$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$f(n) >> g(n)$$

Сравнение скорости роста функций

Сравнение скорости роста функций

1) Константа - $f(n) = C$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k b + C_0$$

$$g(n) = C_1 n^{k'} + C_2 n^{k'-1} + ... + C_{k'} b + C_0$$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$k = k'$$

Одинаковая скорость роста

$$f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k b + C_0$$

$$g(n) = C_1 n^{k'} + C_2 n^{k'-1} + ... + C_{k'} b + C_0$$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k b + C_0$$

$$g(n) = C_1 n^{k'} + C_2 n^{k'-1} + ... + C_{k'} b + C_0$$

$\xrightarrow{k = k'}$ Одинаковая скорость роста

$\xrightarrow{k > k'}$ $f(n) >> g(n)$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k b + C_0$$
$$g(n) = C_1 n^{k'} + C_2 n^{k'-1} + ... + C_{k'} b + C_0$$

$k = k'$ → Одинаковая скорость роста

$k > k'$ → $f(n) >> g(n)$

$k < k'$ → $f(n) << g(n)$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = \log^3 n \cdot n^4 \cdot e^{2n} + n! \cdot n^5$$

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = \log^3 n \cdot n^4 \cdot e^{2n} + n! \cdot n^5$$

Алгоритм действий в случае сложной функции:

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = \log^3 n \cdot n^4 \cdot e^{2n} + n! \cdot n^5$$

Алгоритм действий в случае сложной функции:

1) Выбираем слагаемое в которое как множитель входит наиболее быстро растущая функция

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C \log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + ... + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = \cancel{\log^3 n \cdot n^4 \cdot e^{2n}} + n! \cdot n^5$$

Алгоритм действий в случае сложной функции:

1) Выбираем слагаемое в которое как множитель входит наиболее быстро растущая функция
2) Забываем об остальных слагаемых

Сравнение скорости роста функций

1) Константа - $f(n) = C$
2) Логарифмические - $f(n) = C\log n$
3) Полиномиальные - $f(n) = C_1 n^k + C_2 n^{k-1} + \ldots + C_k n + C_0$
4) Экспоненциальные - $f(n) = a^{cn}, a > 1$
5) Функции с факториалом - $f(n) = n!$

$$f(n) = \cancel{\log^3 n \cdot n^4 \cdot e^{2n}} + n! \cdot n^5$$

Алгоритм действий в случае сложной функции:

1) Выбираем слагаемое в которое как множитель входит наиболее быстро растущая функция
2) Забываем об остальных слагаемых
3) Функция растет быстрее, чем наиболее быстро растущая функция
4) Функция растет медленнее, чем следующая по скорости роста функция из списка

Сравнение роста функций на графиках

Сравнение роста функций на графиках

$$f(n) = 0.1n^3 + 20n$$

Сравнение роста функций на графиках

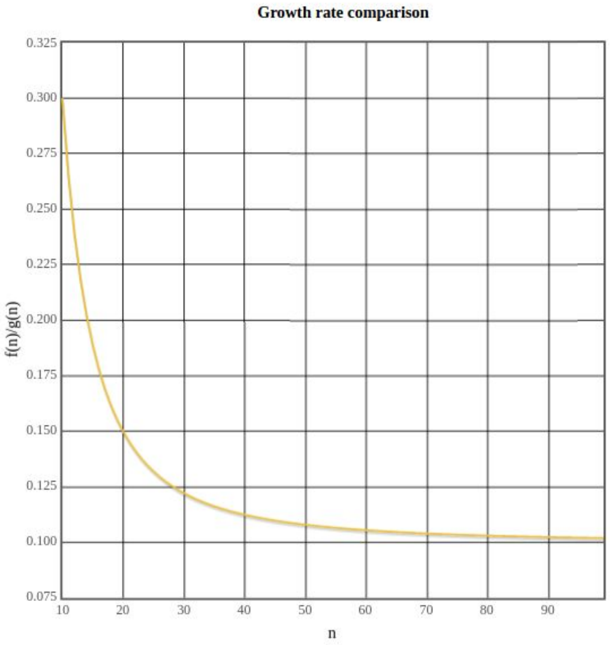
$$f(n) = 0.1n^3 + 20n$$

$$g(n) = n^3$$

Сравнение роста функций на графиках

$$f(n) = 0.1n^3 + 20n$$

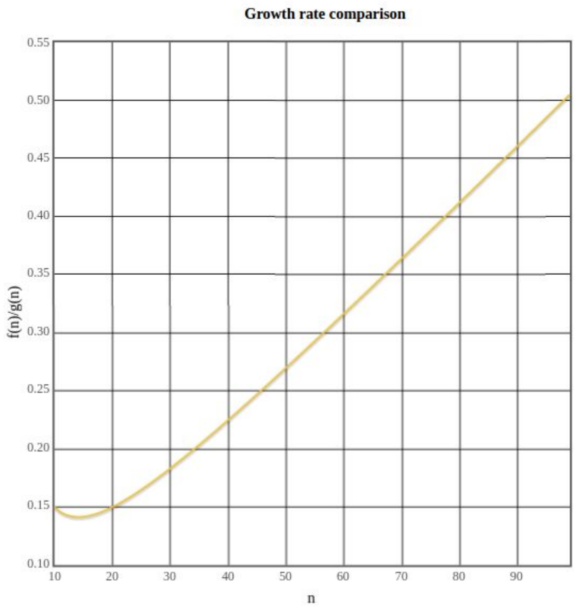
$$g(n) = n^3$$



Сравнение роста функций на графиках

$$f(n) = 0.1n^3 + 20n$$

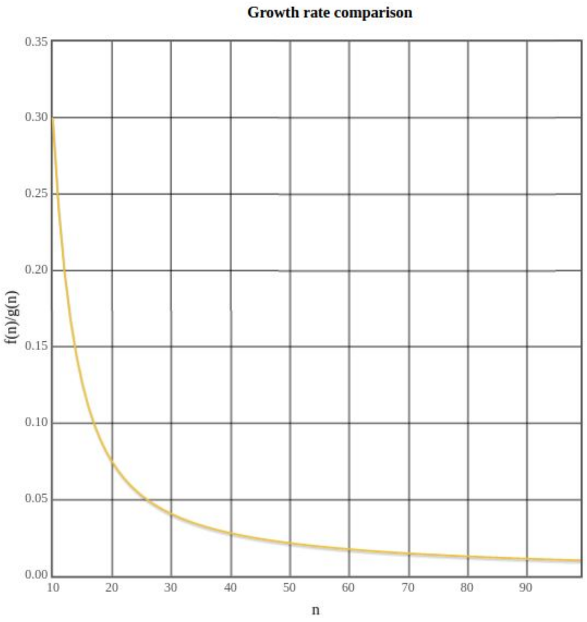
$$g(n) = 20n^2$$



Сравнение роста функций на графиках

$$f(n) = 0.1n^3 + 20n$$

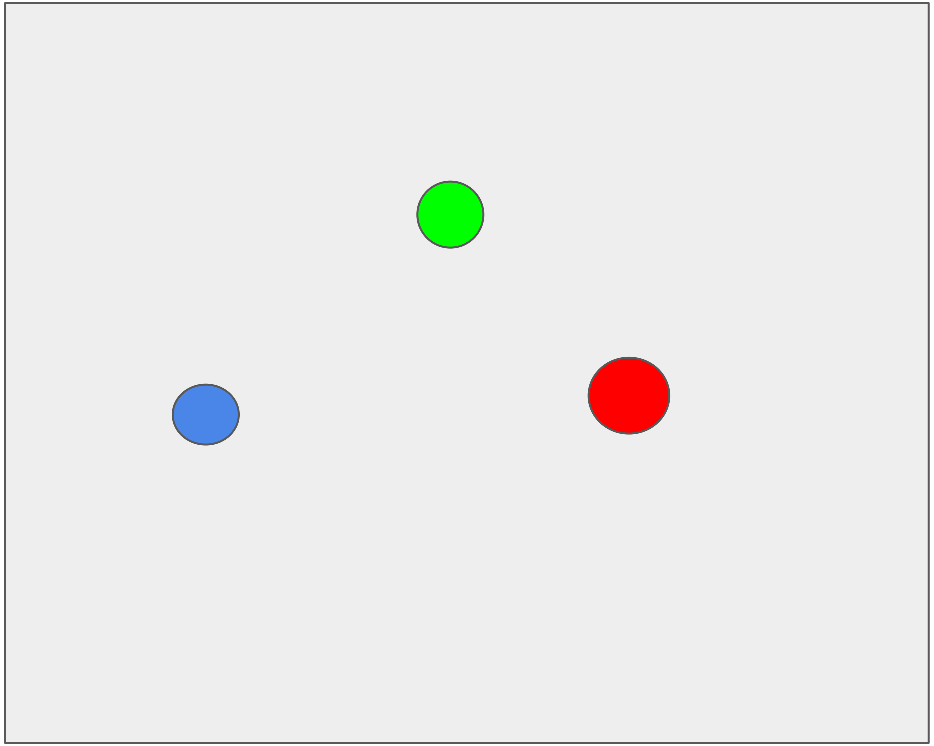
$$g(n) = 0.1n^4$$



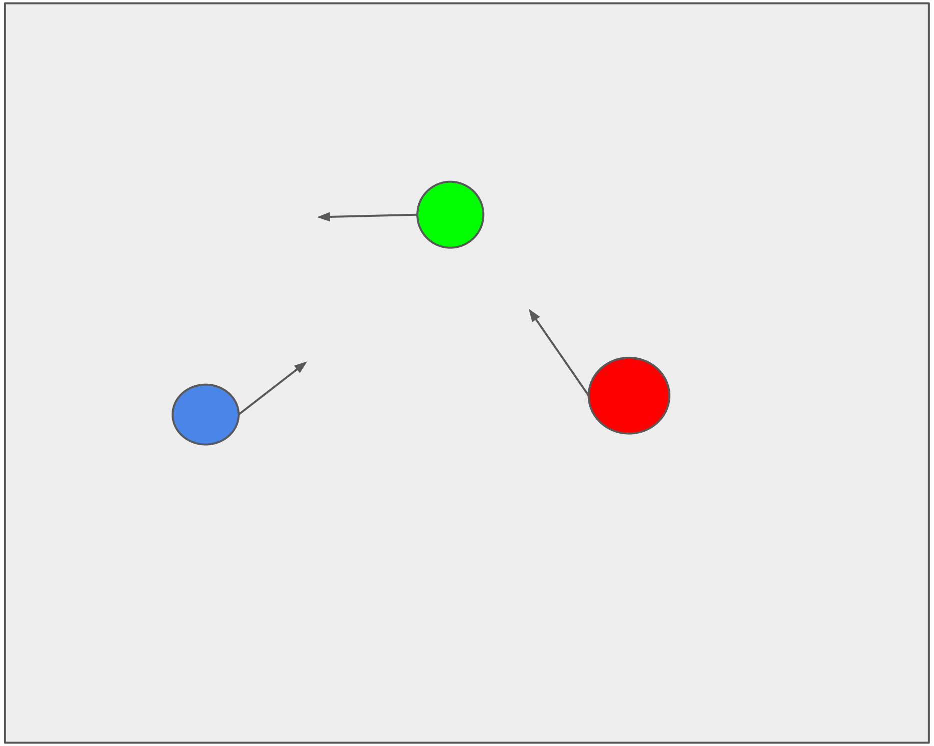
Пример: игра

Пример: игра

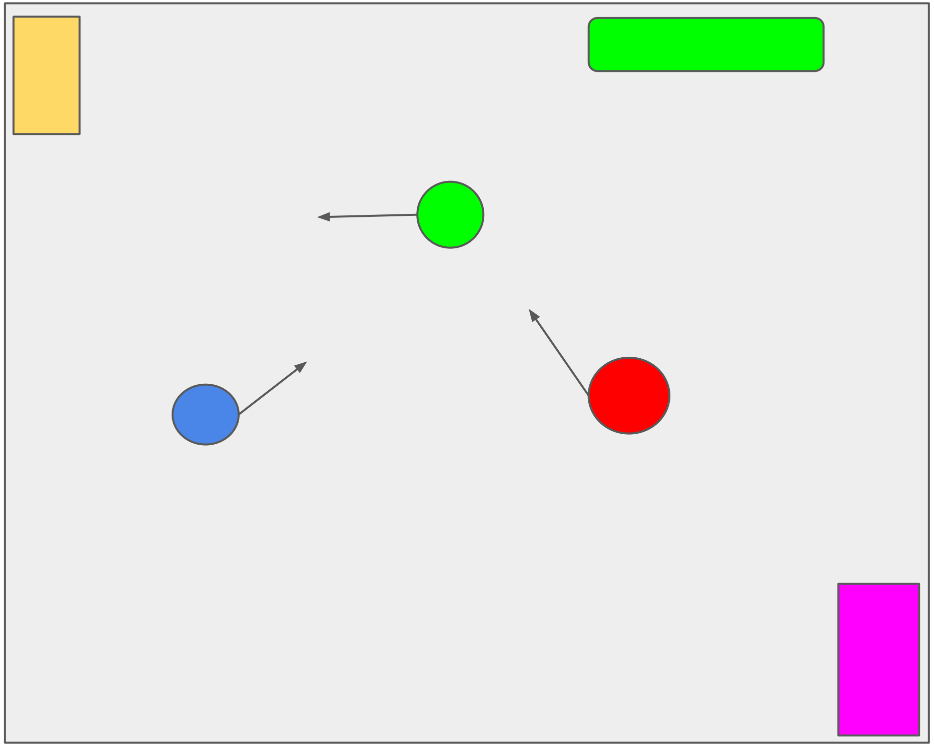
Пример: игра



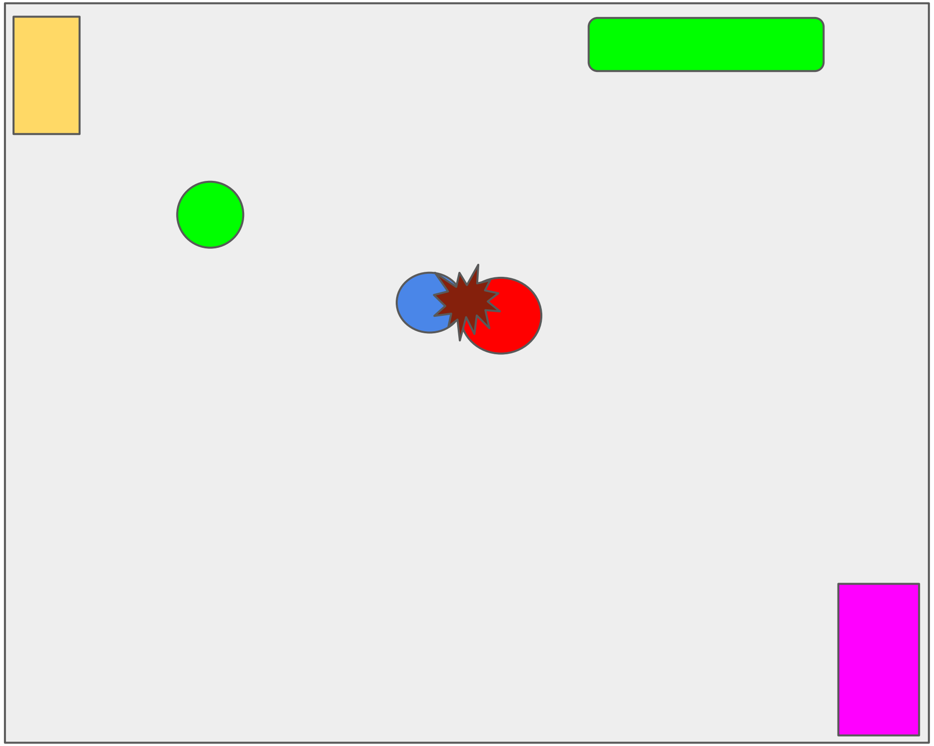
Пример: игра



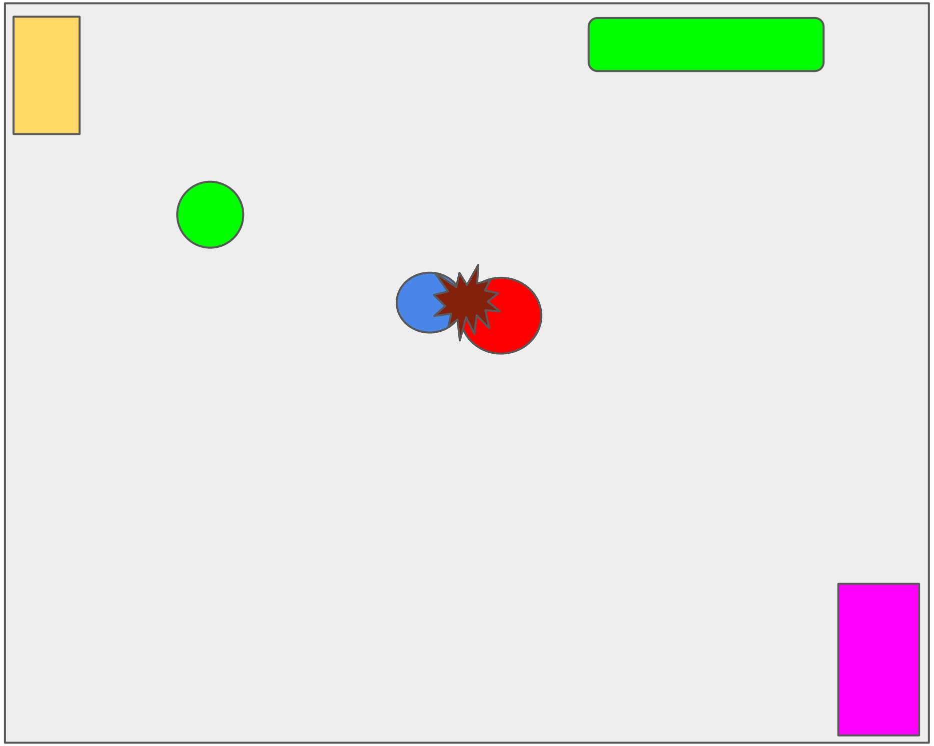
Пример: игра



Пример: игра



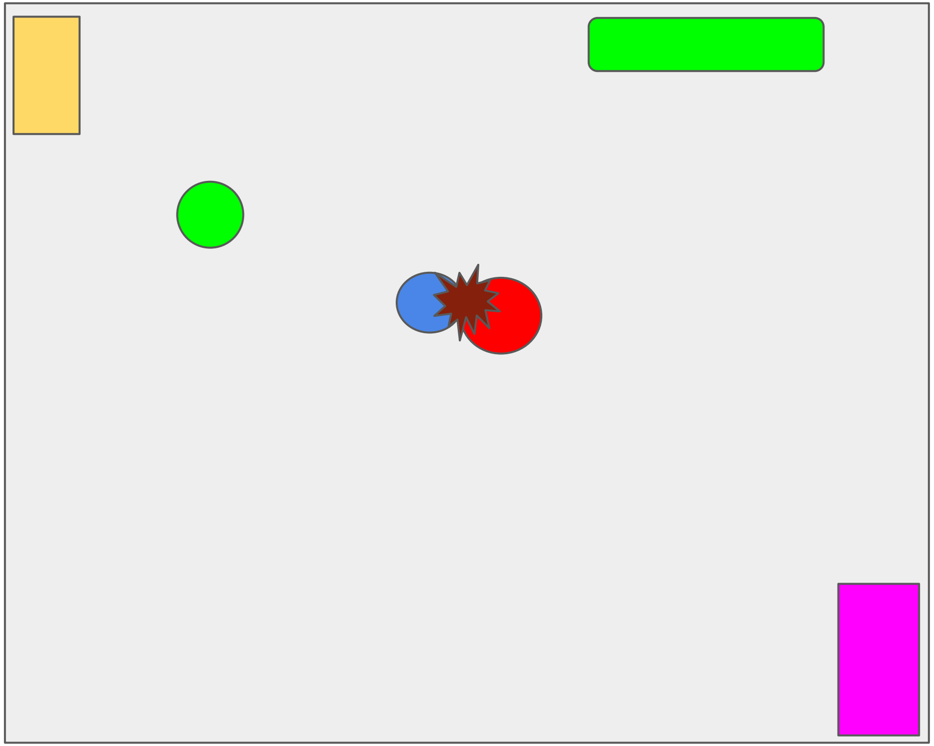
Пример: игра



Алгоритм 1:

Протестировать
каждую пару объектов

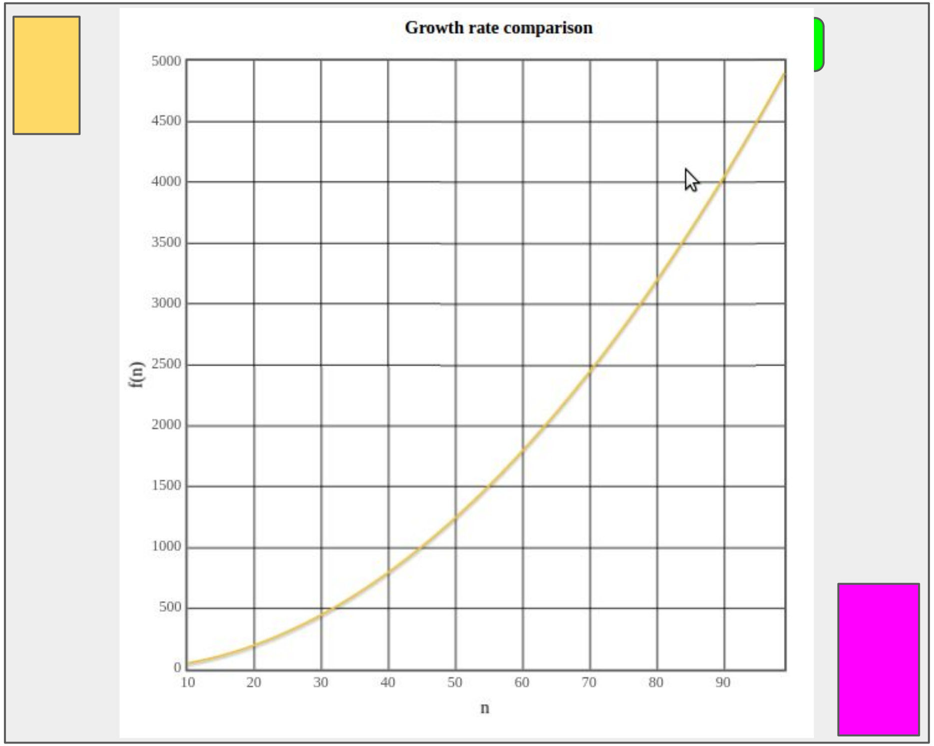
Пример: игра



Алгоритм 1:
Протестировать
каждую пару объектов

$$f(n) \approx \frac{1}{2}n^2$$

Пример: игра

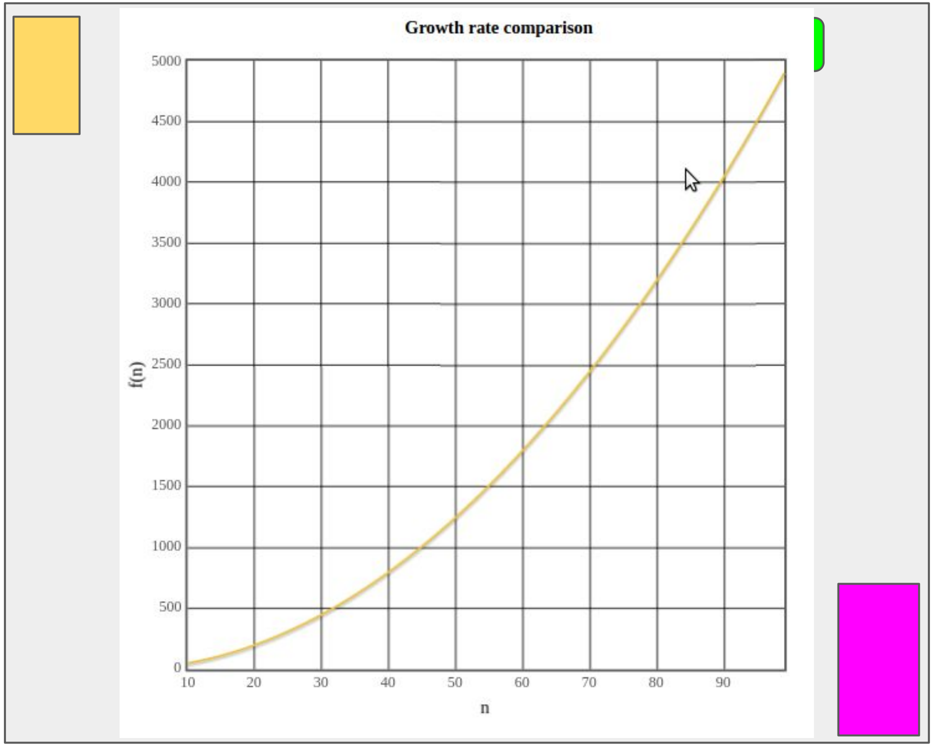


Алгоритм 1:

Протестировать
каждую пару объектов

$$f(n) \approx \frac{1}{2}n^2$$

Пример: игра

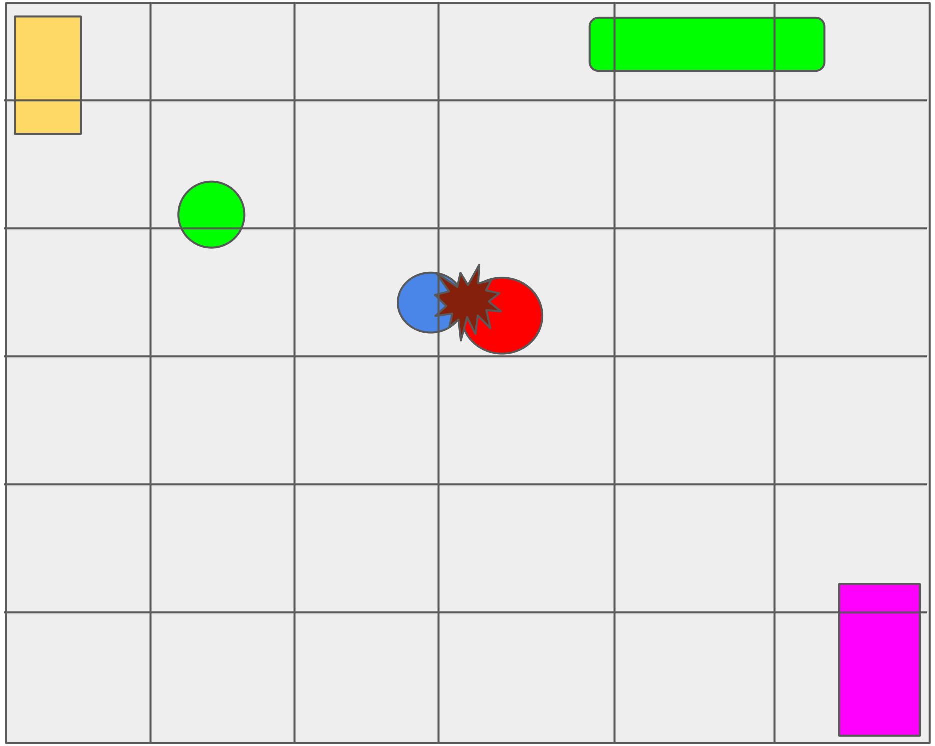


Алгоритм 1:

Протестировать
каждую пару объектов

$$f(n) \approx \frac{1}{2}n^2$$

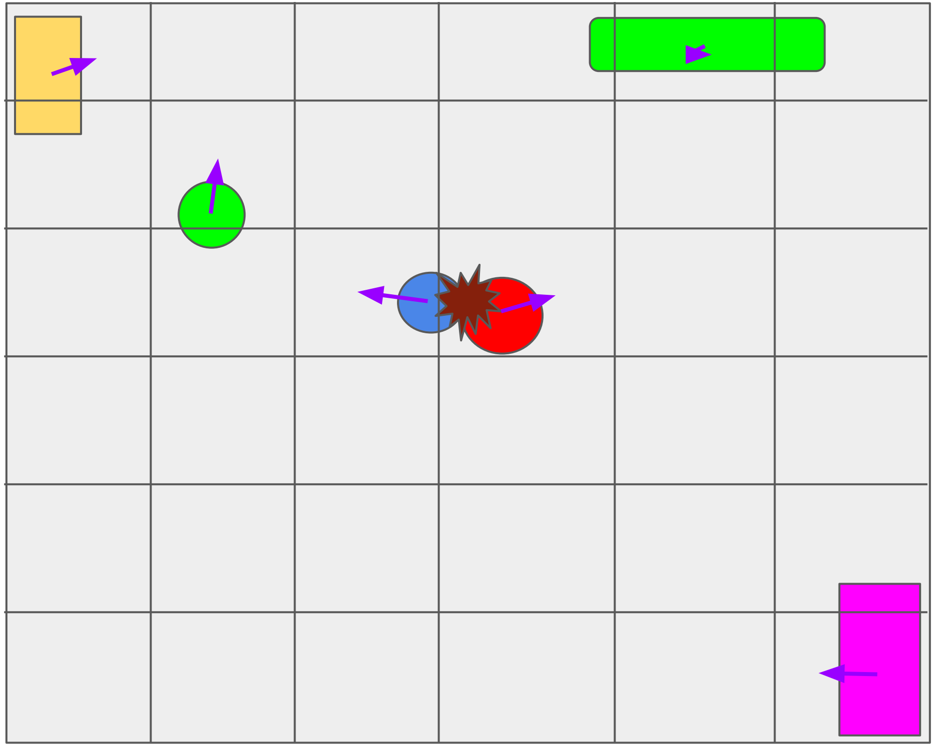
Пример: игра



Алгоритм 2:

Разбить все поле на клетки

Пример: игра

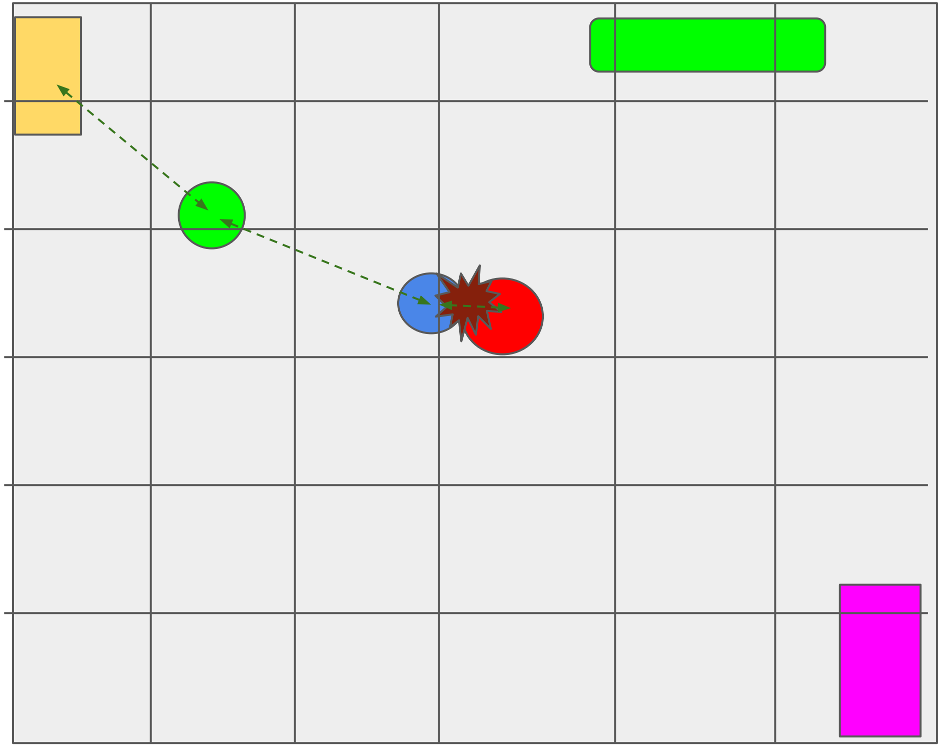


Алгоритм 2:

Разбить все поле на клетки

Каждому объекту назначить свою клетку

Пример: игра



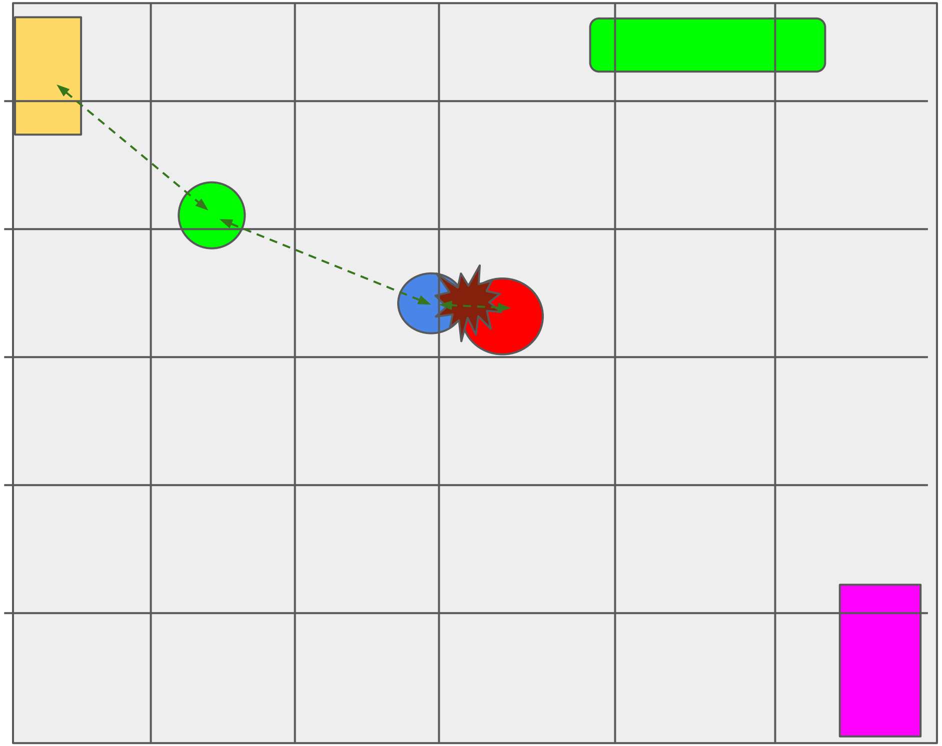
Алгоритм 2:

Разбить все поле на клетки

Каждому объекту назначить свою клетку

Протестировать все объекты внутри одной и соседних клеток

Пример: игра



Алгоритм 2:

Разбить все поле на клетки

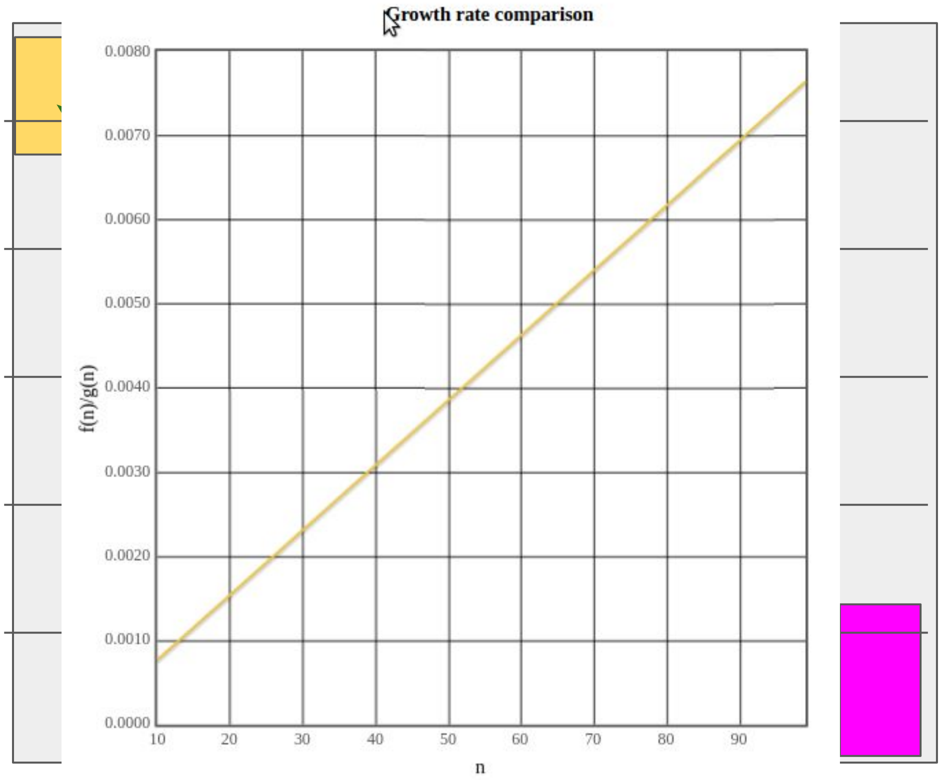
Каждому объекту назначить свою клетку

Протестировать все объекты внутри одной и соседних клеток

$$g(n) \approx 9nm$$

$$m - const$$

Пример: игра



Алгоритм 2:

Разбить все поле на клетки

Каждому объекту назначить свою клетку

Протестировать все объекты внутри одной и соседних клеток

$$g(n) \approx 9nm$$

$$m - const$$

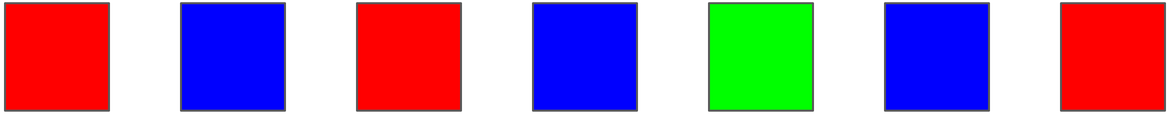
Подсчет времени работы

Подсчет времени работы

Мы хотим пойти в магазин и купить себе некоторое количество одежды. При этом что-то нам не нравится, а что-то нравится и мы можем выразить наше отношение к каждому товару целым числом.

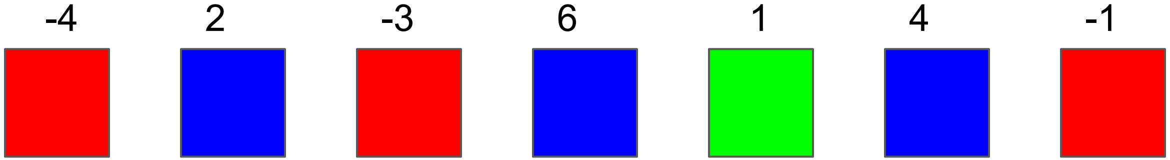
Подсчет времени работы

Мы хотим пойти в магазин и купить себе некоторое количество одежды. При этом что-то нам не нравится, а что-то нравится и мы можем выразить наше отношение к каждому товару целым числом.



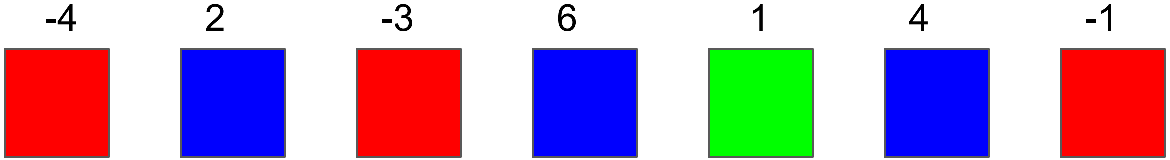
Подсчет времени работы

Мы хотим пойти в магазин и купить себе некоторое количество одежды. При этом что-то нам не нравится, а что-то нравится и мы можем выразить наше отношение к каждому товару целым числом.



Подсчет времени работы

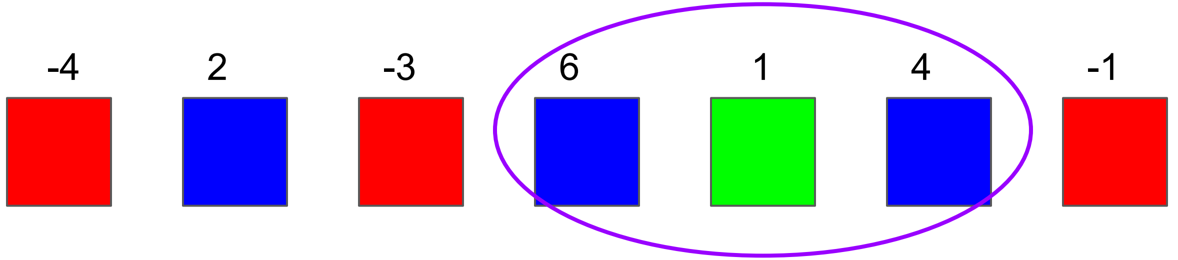
Мы хотим пойти в магазин и купить себе некоторое количество одежды. При этом что-то нам не нравится, а что-то нравится и мы можем выразить наше отношение к каждому товару целым числом.



Мы хотим выбрать непрерывную последовательность товаров, которая максимизировала бы нашу радость от покупки

Подсчет времени работы

Мы хотим пойти в магазин и купить себе некоторое количество одежды. При этом что-то нам не нравится, а что-то нравится и мы можем выразить наше отношение к каждому товару целым числом.



Мы хотим выбрать непрерывную последовательность товаров, которая максимизировала бы нашу радость от покупки

Подсчет времени работы

Алгоритм:

1) Зайти в магазин
2) Подойти к полке
3) Для каждого $i \in [1...n]$
   проверить сколько радости мы получим от покупки $[i...k]$
4) Забрать товар с полки
5) Оплатить
6) Уйти из магазина

Подсчет времени работы

Алгоритм:

1) Зайти в магазин $O(1)$
2) Подойти к полке
3) Для каждого $i \in [1...n]$
   проверить сколько радости мы получим от покупки $[i...k]$
4) Забрать товар с полки
5) Оплатить
6) Уйти из магазина

Подсчет времени работы

Алгоритм:

1) Зайти в магазин $O(1)$
2) Подойти к полке $O(1)$
3) Для каждого $i \in [1...n]$
   проверить сколько радости мы получим от покупки $[i...k]$
4) Забрать товар с полки
5) Оплатить
6) Уйти из магазина

Подсчет времени работы

Алгоритм:

1) Зайти в магазин $O(1)$
2) Подойти к полке $O(1)$
3) Для каждого $i \in [1...n]$ $O(n^2)$

   проверить сколько радости мы получим от покупки $[i \dots k]$
4) Забрать товар с полки $O(n)$
5) Оплатить
6) Уйти из магазина

Подсчет времени работы

Алгоритм:

1) Зайти в магазин $O(1)$
2) Подойти к полке $O(1)$
3) Для каждого $i \in [1...n]$ $O(n^2)$
   проверить сколько радости мы получим от покупки $[i \ldots k]$
4) Забрать товар с полки $O(n)$
5) Оплатить $O(1)$
6) Уйти из магазина $O(1)$

Подсчет времени работы

Алгоритм:

1) Зайти в магазин $O(1)$
2) Подойти к полке $O(1)$
3) Для каждого $i \in [1...n]$ $O(n^2)$
   проверить сколько радости мы получим от покупки $[i \ldots k]$
4) Забрать товар с полки $O(n)$
5) Оплатить $O(1)$
6) Уйти из магазина $O(1)$

$$f(n) = O(1) + O(1) + O(n^2) + O(n) + O(1) + O(1) = O(n^2)$$

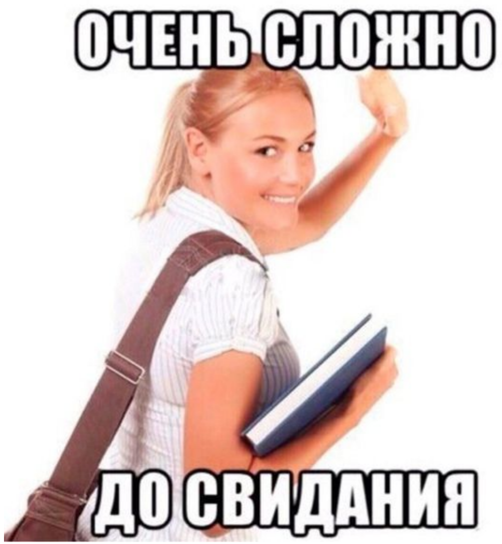
О-нотация

О-нотация

$f(n) = O(g(n))$ - если существуют константы N, c такие, что для $n \geq N \rightarrow f(n) < c \cdot g(n)$

О-нотация

$f(n) = O(g(n))$ - если существуют константы N, c такие, что для $n \geq N \rightarrow f(n) < c \cdot g(n)$



О-нотация

$f(n) = O(g(n))$ - если существуют константы N, c такие, что для $n \geq N \rightarrow f(n) < c \cdot g(n)$

Это означает, что если для функции $g(n)$ выбрать константу c произвольно, что если мы изобразим на графике одновременно $c \cdot g(n)$ и $f(n)$, то начиная с какой-то точки $c \cdot g(n)$ будет выше, чем $f(n)$

O-нотация: пример

$$3n^2 + 5n + 2 = O(n^2)$$

О-нотация: пример

$$3n^2 + 5n + 2 = O(n^2)$$

$$g(n) = 10n^2$$

О-нотация: пример

$$3n^2 + 5n + 2 = O(n^2)$$

$$g(n) = 10n^2 = 3n^2 + 5n^2 + 2n^2$$

$$n \geq 1$$

$$3n^2 + 5n^2 + 2n^2 \geq 3n^2 + 5n + 2$$

О-нотация: пример

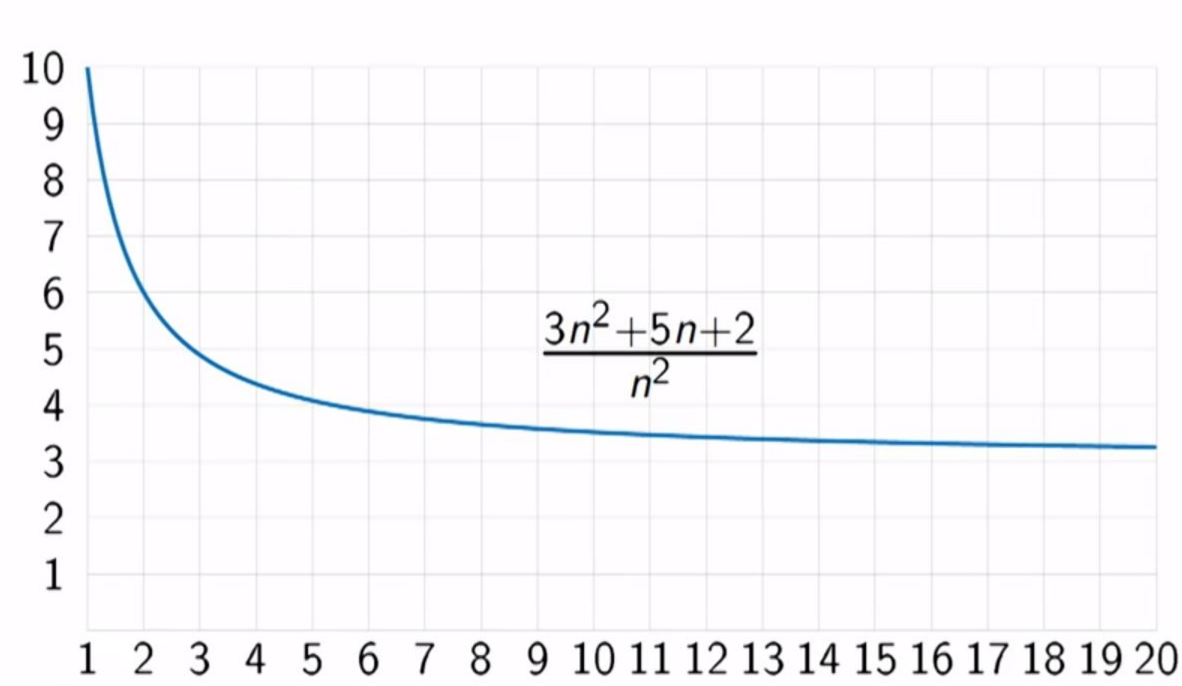
$$3n^2 + 5n + 2 = O(n^2)$$

$$g(n) = 10n^2 = 3n^2 + 5n^2 + 2n^2$$

$$N = 1, c = 10$$

О-нотация: пример

$$3n^2 + 5n + 2 = O(n^2)$$



О-нотация: пример

$$3n^2 + 5n + 2 = O(n^2)$$

$$n + \log_2 n + \sin(n) = O(n)$$

Правила использования О-нотации

Правила использования O-нотации

1) Любой константный множитель может быть отброшен

$$7n^3 = O(n^3), \frac{n^2}{3} = O(n^2)$$

Правила использования О-нотации

2) $f(n) = O(\ldots), f(n) << g(n)$

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

Правила использования О-нотации

2) $f(n) = O(\dots), f(n) << g(n)$

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

$$f(n) = 2 \cdot n = O(n)$$

$$g(n) = n^2 = O(n^2)$$

$$f(n) = O(n^2)$$

Правила использования О-нотации

2) $$f(n) = O(\dots), f(n) << g(n)$$

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

$$f(n) = 2 \cdot n = O(n) \longrightarrow f(n) = O(2^n)$$

Правила использования О-нотации

3) Можно откидывать слагаемые с не самой высокой скоростью роста

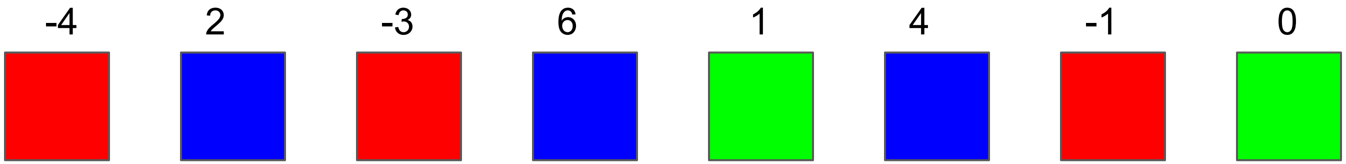
Правила использования О-нотации

3) Можно откидывать слагаемые с не самой высокой скоростью роста

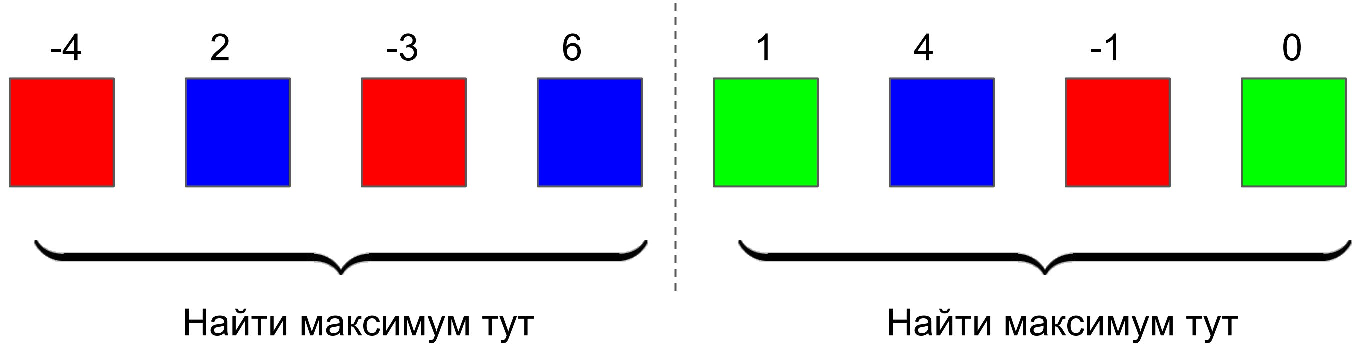
$$n^2 + n = O(n^2)$$

$$2^n + 2n^3 + n = O(2^n)$$

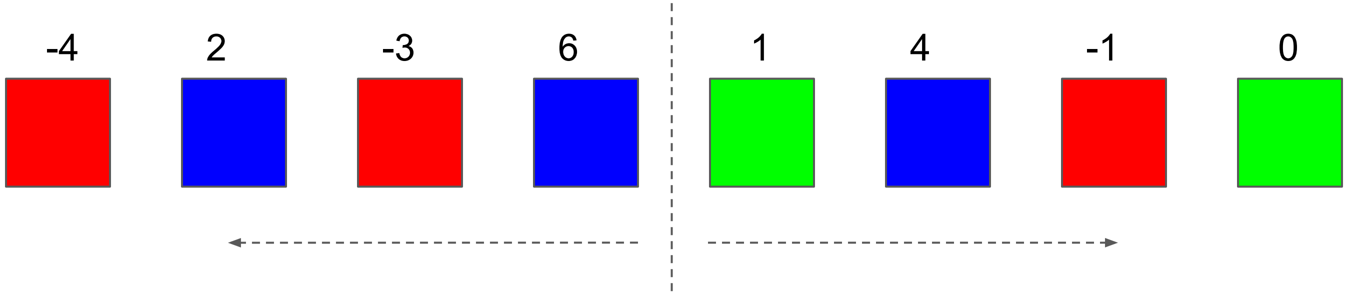
Алгоритмы: разделяй-и-властвуй



Алгоритмы: разделяй-и-властвуй

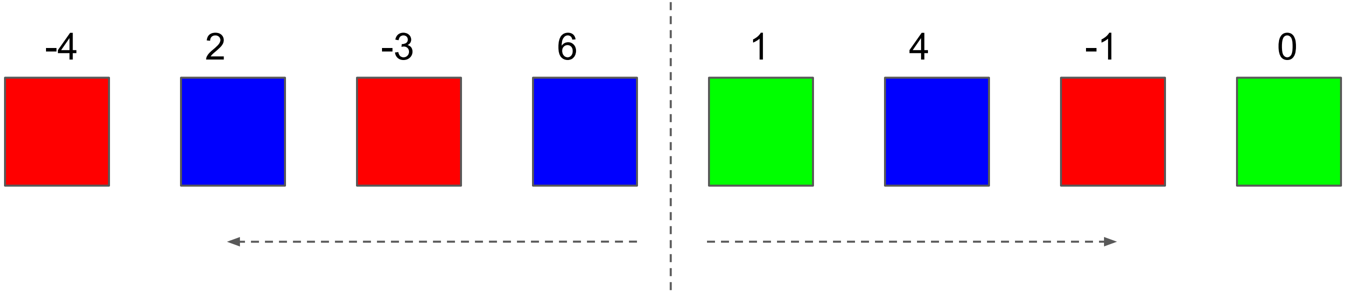


Алгоритмы: разделяй-и-властвуй



Найдем объединение двух подотрезков
с максимальной суммой

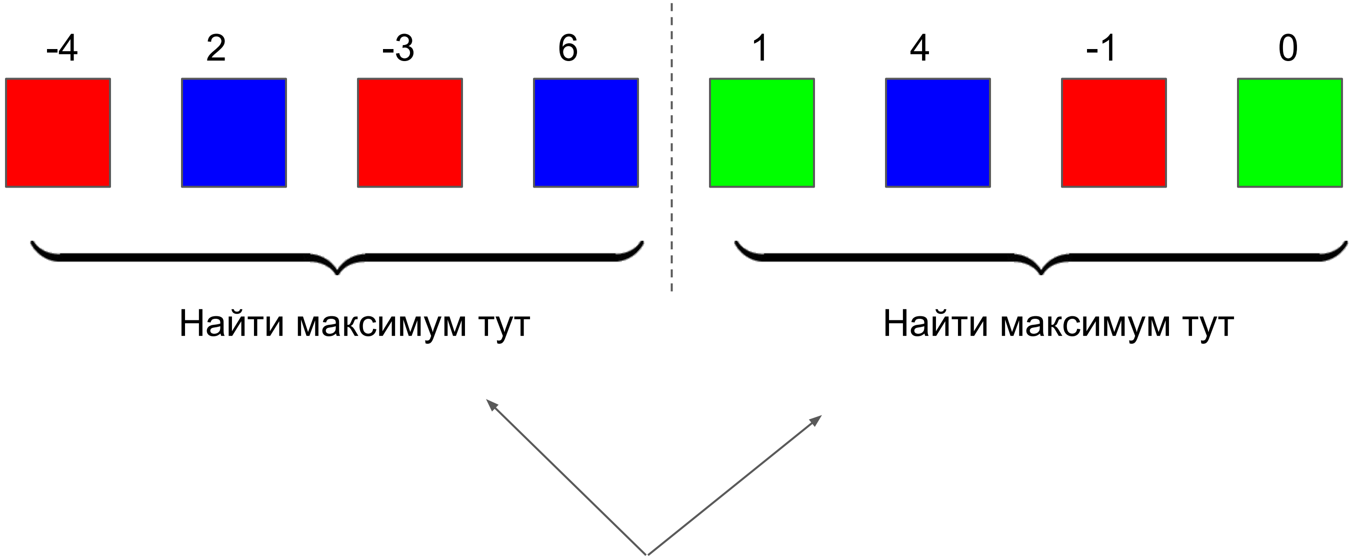
Алгоритмы: разделяй-и-властвуй



Найдем объединение двух подотрезков
с максимальной суммой

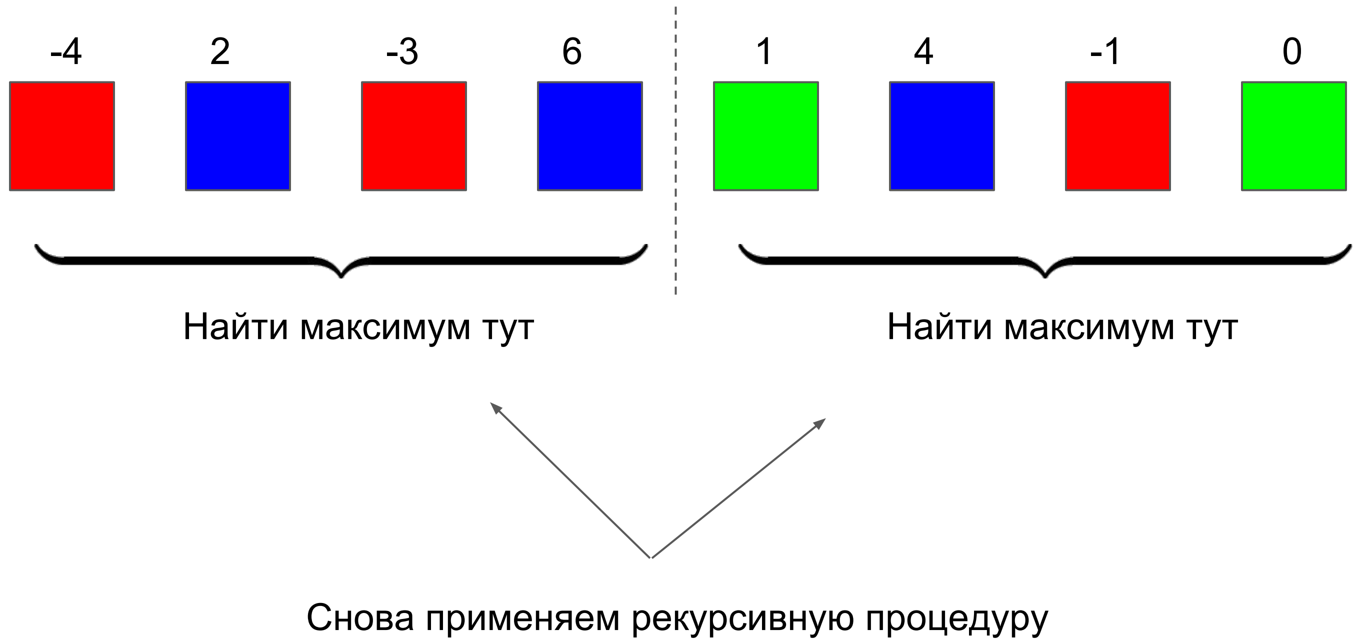
$$O(n)$$

Алгоритмы: разделяй-и-властвуй

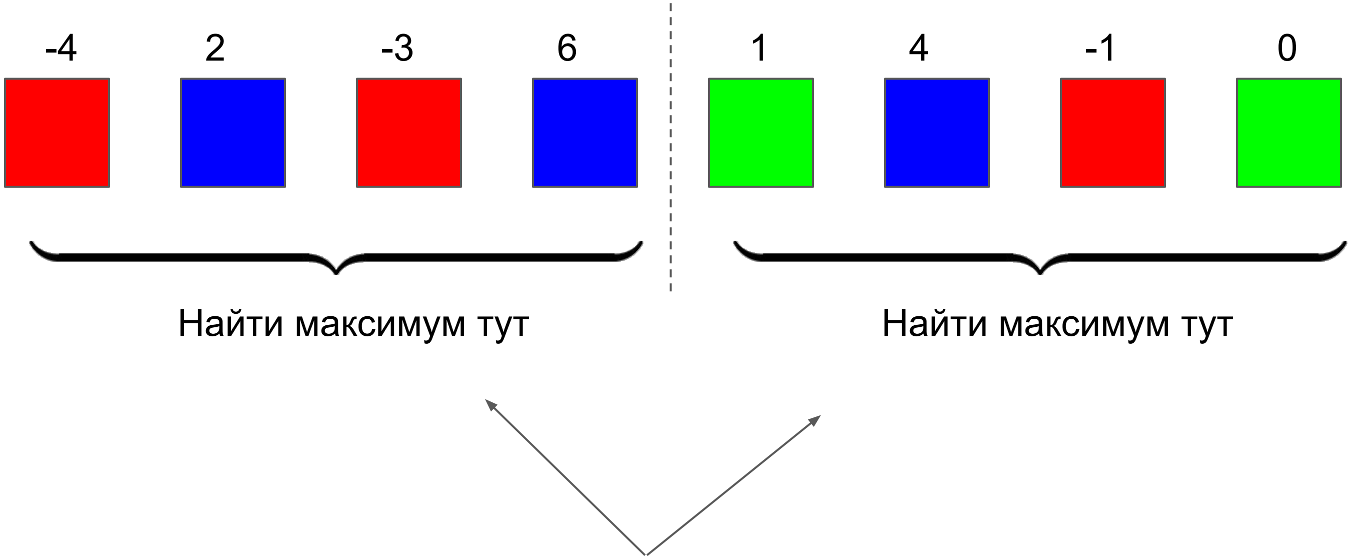


$$O(n) + 2 \cdot O(\frac{n^2}{4}) = O(n^2)$$

Алгоритмы: разделяй-и-властвуй



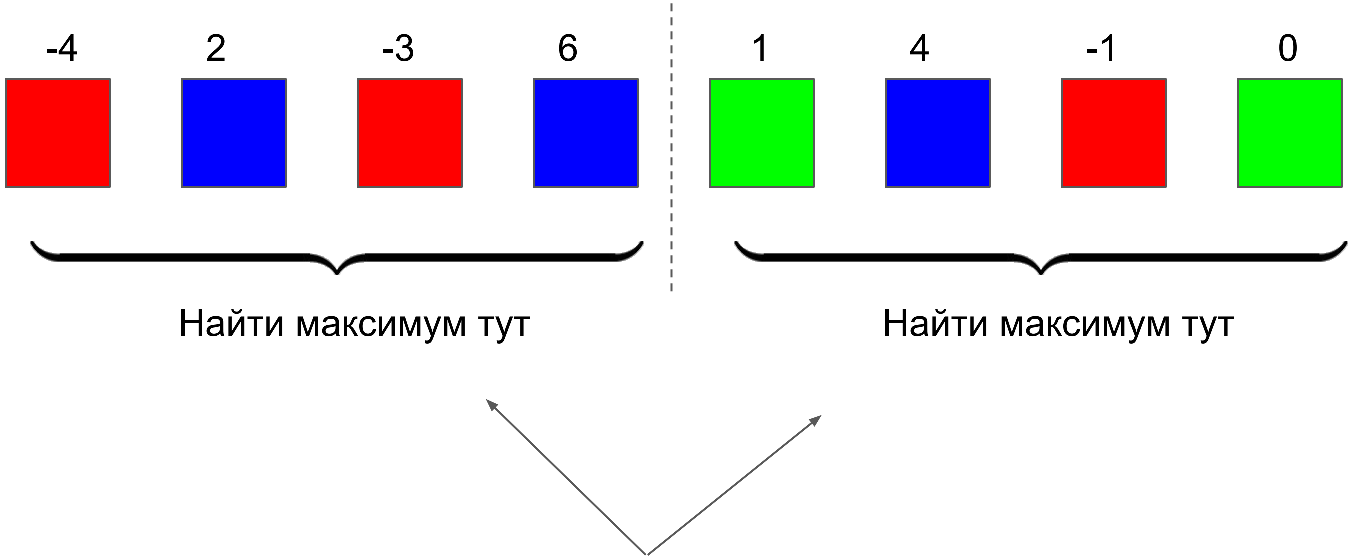
Алгоритмы: разделяй-и-властвуй



Снова применяем рекурсивную процедуру

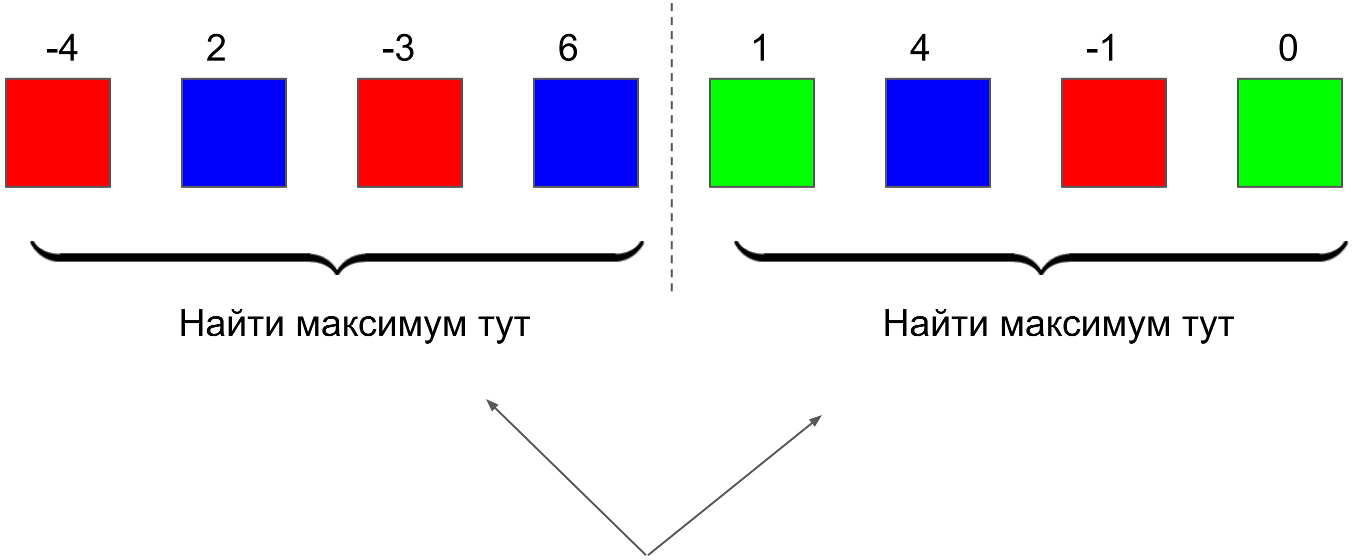
Каждый раз размер подзадачи уменьшается в два раза,
но из одной задачи получаем две

Алгоритмы: разделяй-и-властвуй



$$f(n) = 2f(\frac{n}{2}) + O(n)$$

Алгоритмы: разделяй-и-властвуй



$$f(n) = O(n \log n)$$

Мастер теорема

Мастер теорема

$$f(n) = 2f(\frac{n}{2}) + O(n \log n)$$

Мастер теорема

$$f(n) = 2f(\frac{n}{2}) + O(n \log n)$$

$$f(n) = f(\frac{n}{2}) + O(1)$$

Мастер теорема

$$f(n) = 2f(\frac{n}{2}) + O(n \log n)$$

$$f(n) = f(\frac{n}{2}) + O(1)$$

$$f(n) = 3f(\frac{n}{4}) + O(n^2)$$

Мастер теорема

Пусть время работы алгоритма представимо в виде

$$f(n) = af(\frac{n}{b}) + O(n^d)$$

Мастер теорема

Пусть время работы алгоритма представимо в виде

$$f(n) = af(\frac{n}{b}) + O(n^d)$$

Тогда:

$$d > \log_b a \to f(n) = O(n^d)$$

Мастер теорема

Пусть время работы алгоритма представимо в виде

$$f(n) = af(\frac{n}{b}) + O(n^d)$$

Тогда:

$$d > \log_b a \to f(n) = O(n^d)$$

$$d = \log_b a \to f(n) = O(n^d \log n)$$

Мастер теорема

Пусть время работы алгоритма представимо в виде

$$f(n) = af(\frac{n}{b}) + O(n^d)$$

Тогда:

$$d > \log_b a \to f(n) = O(n^d)$$

$$d = \log_b a \to f(n) = O(n^d \log n)$$

$$d < \log_b a \to f(n) = O(n^{\log_b a})$$

Мастер теорема: пример 1

$$f(n) = 4f(\frac{n}{2}) + O(n)$$

Мастер теорема: пример 1

$$f(n) = 4f(\frac{n}{2}) + O(n)$$

$$a = 4, b = 2, d = 1$$

Мастер теорема: пример 1

$$f(n) = 4f(\frac{n}{2}) + O(n)$$

$$a = 4, b = 2, d = 1$$

$$d < \log_b a \rightarrow f(n) = O(n^{log_b a}) = O(n^2)$$

Мастер теорема: пример 2

$$f(n) = 3f(\frac{n}{2}) + O(n)$$

Мастер теорема: пример 2

$$f(n) = 3f(\frac{n}{2}) + O(n)$$

$$a = 3, b = 2, d = 1$$

Мастер теорема: пример 2

$$f(n) = 3f(\frac{n}{2}) + O(n)$$

$$a = 3, b = 2, d = 1$$

$$d < \log_b a \to f(n) = O(n^{log_b a}) = O(n^{\log_2 3})$$

Мастер теорема: пример 3

$$f(n) = 2f(\frac{n}{2}) + O(n)$$

Мастер теорема: пример 3

$$f(n) = 2f(\frac{n}{2}) + O(n)$$

$$a = 2, b = 2, d = 1$$

Мастер теорема: пример 3

$$f(n) = 2f(\frac{n}{2}) + O(n)$$

$$a = 2, b = 2, d = 1$$

$$d = \log_b a \to f(n) = O(n^d \log n) = O(n \log n)$$