

# Задача

**Задача** – это соответствие, определяющее зависимость выхода (слова) от входа (слова).

# Задача

**Задача** – это соответствие, определяющее зависимость выхода (слова) от входа (слова).

$\Sigma$  – алфавит (произвольное конечное множество, элементы которого интерпретируются как символы)

# Задача

**Задача** – это соответствие, определяющее зависимость выхода (слова) от входа (слова).

$\Sigma$  – алфавит (произвольное конечное множество, элементы которого интерпретируются как символы)

$\Sigma^*$  – множество всех слов из букв алфавита  $\Sigma$ .

# Задача

**Задача** – это соответствие, определяющее зависимость выхода (слова) от входа (слова).

$\Sigma$  – алфавит (произвольное конечное множество, элементы которого интерпретируются как символы)

$\Sigma^*$  – множество всех слов из букв алфавита  $\Sigma$ .

$\rho \subset \Sigma^* \times \Sigma^*$  – бинарное отношение, определяющее задачу. Пара  $(x, y) \in \rho$  показывает, что  $y$  является допустимым выходом для входа  $x$

# Алгоритм и программа

**Алгоритм** – это последовательность элементарных операций, обрабатывающая входную строку  $x$  для получения выходной строки  $y$  такой, что  $(x, y) \in \rho$

# Алгоритм и программа

**Алгоритм** – это последовательность элементарных операций, обрабатывающая входную строку  $x$  для получения выходной строки  $y$  такой, что  $(x, y) \in \rho$

Под элементарной операцией в этом курсе мы будем понимать операции, исполняющиеся непосредственно на процессоре: сложение чисел, умножение и т.д.

# Алгоритм и программа

**Алгоритм** – это последовательность элементарных операций, обрабатывающая входную строку  $x$  для получения выходной строки  $y$  такой, что  $(x, y) \in \rho$

Под элементарной операцией в этом курсе мы будем понимать операции, исполняющиеся непосредственно на процессоре: сложение чисел, умножение и т.д.

**Программа** – это алгоритм, выраженный на некотором языке, который может быть транслирован в элементарные операции

# Сложность алгоритма

**Временная сложность алгоритма** – это функция  $f(n)$ ,  $f : \mathbb{N} \rightarrow \mathbb{N}$ , показывающая точную верхнюю границу количества элементарных операций, необходимых для завершения работы алгоритма, в зависимости от количества символов во входе



# Сложность алгоритма

**Временная сложность алгоритма** – это функция  $f(n)$ ,  $f : \mathbb{N} \rightarrow \mathbb{N}$ , показывающая точную верхнюю границу количества элементарных операций, необходимых для завершения работы алгоритма, в зависимости от количества символов во входе

**Емкостная сложность алгоритма** – аналогичная оценка для *дополнительной* памяти, необходимой для анализа входа. Память, используемая для хранения входа, не учитывается.

## Сложность алгоритма

```
var n=Console.ReadLine().Length;

var sum=0;
for (int i=0;i<n;i++)
    for(int j=0;j<2*i;j++)
        sum++;

Console.WriteLine(sum);
```

## Сложность алгоритма

```
var n=Console.ReadLine().Length;

var sum=0;
for (int i=0;i<n;i++)
    for(int j=0;j<2*i;j++)
        sum++;

Console.WriteLine(sum);
```

$$0+2+4+\dots+2(n-1)$$

## Сложность алгоритма

```
var n=Console.ReadLine().Length;

var sum=0;
for (int i=0;i<n;i++)
    for(int j=0;j<2*i;j++)
        sum++;

Console.WriteLine(sum);
```

$$0+2+4+\dots+2(n-1) = n(n-1)$$

## Сложность алгоритма

```
var n=Console.ReadLine().Length;

var sum=0;
for (int i=0;i<n;i++)
    for(int j=0;j<2*i;j++)
        sum++;

Console.WriteLine(sum);
```

$$0+2+4+\dots+2(n-1) = n(n-1)$$

$$f(n) = n(n-1)(2_{++} + 1_{*} + 1_{<}) +$$

## Сложность алгоритма

```
var n=Console.ReadLine().Length;

var sum=0;
for (int i=0;i<n;i++)
    for(int j=0;j<2*i;j++)
        sum++;

Console.WriteLine(sum);
```

$$0+2+4+\dots+2(n-1) = n(n-1)$$

$$f(n) = n(n-1)(2_{++} + 1_{*} + 1_{<}) +$$

$$+ n(1_{=} + 1_{++} + 1_{<}) + 2_{=} + RL + WL$$

## Сложность алгоритма

```
var n=Console.ReadLine().Length;

var sum=0;
for (int i=0;i<n;i++)
    for(int j=0;j<2*i;j++)
        sum++;

Console.WriteLine(sum);
```

$$0+2+4+\dots+2(n-1) = n(n-1)$$

$$f(n) = n(n-1)(2_{++} + 1_{*} + 1_{<}) +$$

$$+ n(1_{=} + 1_{++} + 1_{<}) + 2_{=} + RL + WL$$

$$= k_W n + k_R \log_{10} n(n-1) + 4n^2 - n + 2$$

## O-символика

$$f(n) = o(g(n))$$



## O-символика

$$f(n) = o(g(n)) \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

## O-символика

$$f(n) = o(g(n)) \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

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

## O-символика

$$\begin{array}{ll} f(n) = o(g(n)) & \forall k > 0 \exists n_0 \forall n > n_0 \\ & f(n) < k \cdot g(n) \\ f(n) = O(g(n)) & \exists k > 0 \exists n_0 \forall n > n_0 \\ & f(n) < k \cdot g(n) \end{array}$$

## O-символика

$$f(n) = o(g(n)) \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

$$f(n) = O(g(n)) \quad \exists k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

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

## O-символика

$$\begin{aligned} f(n) = o(g(n)) & \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ & \quad f(n) < k \cdot g(n) \\ f(n) = O(g(n)) & \quad \exists k > 0 \exists n_0 \forall n > n_0 \\ & \quad f(n) < k \cdot g(n) \\ f(n) = \Theta(g(n)) & \quad \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ & \quad k_1 \cdot g(n) < f(n) < k_2 \cdot g(n) \end{aligned}$$

## O-символика

$$\begin{aligned} f(n) = o(g(n)) & \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ & \quad f(n) < k \cdot g(n) \\ f(n) = O(g(n)) & \quad \exists k > 0 \exists n_0 \forall n > n_0 \\ & \quad f(n) < k \cdot g(n) \\ f(n) = \Theta(g(n)) & \quad \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ & \quad k_1 \cdot g(n) < f(n) < k_2 \cdot g(n) \end{aligned} \quad \Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$$

## O-символика

$$f(n) = o(g(n)) \quad \begin{array}{l} \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n) \end{array}$$

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

$$f(n) = O(g(n)) \quad \begin{array}{l} \exists k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n) \end{array}$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty$$

$$f(n) = \Theta(g(n)) \quad \begin{array}{l} \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ k_1 \cdot g(n) < f(n) < k_2 \cdot g(n) \end{array}$$

## O-символика

$$f(n) = o(g(n)) \quad \begin{array}{l} \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n) \end{array}$$

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

$$f(n) = O(g(n)) \quad \begin{array}{l} \exists k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n) \end{array}$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty$$

$$f(n) = \Theta(g(n)) \quad \begin{array}{l} \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ k_1 \cdot g(n) < f(n) < k_2 \cdot g(n) \end{array}$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0$$



## O-символика

$$f(n) = o(g(n)) \quad \begin{array}{l} \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n) \end{array}$$

$$f(n) = O(g(n)) \quad \begin{array}{l} \exists k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n) \end{array}$$

$$f(n) = \Theta(g(n)) \quad \begin{array}{l} \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ k_1 \cdot g(n) < f(n) < k_2 \cdot g(n) \end{array}$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad f(n) \prec g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0$$

## O-символика

$$f(n) = o(g(n)) \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

$$f(n) = O(g(n)) \quad \exists k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

$$f(n) = \Theta(g(n)) \quad \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ k_1 \cdot g(n) < f(n) < k_2 \cdot g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad f(n) \prec g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty \quad f(n) \preceq g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0$$

## O-символика

$$f(n) = o(g(n)) \quad \forall k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

$$f(n) = O(g(n)) \quad \exists k > 0 \exists n_0 \forall n > n_0 \\ f(n) < k \cdot g(n)$$

$$f(n) = \Theta(g(n)) \quad \exists k_1, k_2 > 0 \exists n_0 \forall n > n_0 \\ k_1 \cdot g(n) < f(n) < k_2 \cdot g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad f(n) \prec g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty \quad f(n) \preceq g(n)$$

$$\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0 \quad f(n) \approx g(n)$$

$$f(n) = n(3 + \sin n)$$

$$g(n) = n$$

$$g(n) < f(n) < 5g(n)$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \lim_{n \rightarrow \infty} (3 + \sin n)$$

## Оценка сложности

$$f(n) = 4n^2 + k_1 n + k_2 \log n(n-1) + 2$$

## Оценка сложности

$$f(n) = 4n^2 + k_1n + k_2 \log n(n-1) + 2$$

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

## Оценка сложности

$$f(n) = 4n^2 + k_1n + k_2 \log n(n-1) + 2$$

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

$$\lim_{n \rightarrow \infty} \frac{4n^2 + k_1n + k_2 \log n(n-1) + 2}{n^2} =$$

## Оценка сложности

$$f(n) = 4n^2 + k_1n + k_2 \log n(n-1) + 2$$

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

$$\lim_{n \rightarrow \infty} \frac{4n^2 + k_1n + k_2 \log n(n-1) + 2}{n^2} =$$

$$\lim_{n \rightarrow \infty} \frac{4n^2}{n^2} + \lim_{n \rightarrow \infty} \frac{k_1n}{n^2} + \lim_{n \rightarrow \infty} \frac{k_2 \log n(n-1)}{n^2} + \lim_{n \rightarrow \infty} \frac{2}{n^2} =$$



## Оценка сложности

$$f(n) = 4n^2 + k_1n + k_2 \log n(n-1) + 2$$

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

$$\lim_{n \rightarrow \infty} \frac{4n^2 + k_1n + k_2 \log n(n-1) + 2}{n^2} =$$

$$\lim_{n \rightarrow \infty} \frac{4n^2}{n^2} + \lim_{n \rightarrow \infty} \frac{k_1n}{n^2} + \lim_{n \rightarrow \infty} \frac{k_2 \log n(n-1)}{n^2} + \lim_{n \rightarrow \infty} \frac{2}{n^2} = 4$$

## Оценка сложности

$$f(n) = 4n^2 + k_1n + k_2 \log n(n-1) + 2$$

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

$$\lim_{n \rightarrow \infty} \frac{4n^2 + k_1n + k_2 \log n(n-1) + 2}{n^2} =$$

$$\lim_{n \rightarrow \infty} \frac{4n^2}{n^2} + \lim_{n \rightarrow \infty} \frac{k_1n}{n^2} + \lim_{n \rightarrow \infty} \frac{k_2 \log n(n-1)}{n^2} + \lim_{n \rightarrow \infty} \frac{2}{n^2} = 4$$

$$f(n) = \Theta(g(n)) = \Theta(n^2)$$

## Сложность алгоритма

```
var n=int.Parse(Console.ReadLine());

var root=(int)Math.Sqrt(n);
for (int i=2;i<=root;i++)
    if (n % i == 0)
    {
        Console.WriteLine(i);
        return;
    }

Console.WriteLine("N is prime");
```

## Сложность алгоритма

```
var n=int.Parse(Console.ReadLine());

var root=(int)Math.Sqrt(n);
for (int i=2;i<=root;i++)
    if (n % i == 0)
    {
        Console.WriteLine(i);
        return;
    }

Console.WriteLine("N is prime");
```

$$f(n) = \Theta(\sqrt{n})$$

## Сложность алгоритма

```
var n=int.Parse(Console.ReadLine());

var root=(int)Math.Sqrt(n);
for (int i=2;i<=root;i++)
    if (n % i == 0)
    {
        Console.WriteLine(i);
        return;
    }

Console.WriteLine("N is prime");
```

$$f(n) = \Theta(\sqrt{n})$$

$$n = \Theta(10^{|x|})$$

## Сложность алгоритма

```
var n=int.Parse(Console.ReadLine());

var root=(int)Math.Sqrt(n);
for (int i=2;i<=root;i++)
    if (n % i == 0)
    {
        Console.WriteLine(i);
        return;
    }

Console.WriteLine("N is prime");
```

$$f(n) = \Theta(\sqrt{n})$$

$$n = \Theta(10^{|x|})$$

$$f(|x|) = \Theta(\sqrt{10^{|x|}})$$

Алгоритм со сложностью  $f(n)$  называется:

- ▶ Если  $f = \Theta(\log^k n)$ : логарифмическим при  $k = 1$ , полилогарифмическим при  $k > 1$ .
- ▶ Если  $f = \Theta(n)$ : линейным
- ▶ Если  $f = \Theta(n \log^k n)$ : linearithmic при  $k = 1$ , квазилинейным при  $k > 1$
- ▶ Если  $f = \Theta(n^k)$ : полиномиальным, при  $k = 2$  квадратичным.
- ▶ Если  $f = \Theta(2^{n^k})$ : экспоненциальным.