

JavaScript

- **сайты**
- мобильные приложения
- десктопные программы
например Atom, VSCode, GitKraken, Hyper
- сервер и вспомогательные инструменты (*node.js*)
Express, Koa, Webpack, Gulp, Grunt



Технологии веб-приложений

```
<!DOCTYPE html>
<html>
<head>
  <title>Page title</title>
  <meta charset="utf-8">
</head>
<body>

<div id="layout">

</div>
</body>
</html>
```

```
html,
body {
  margin: 0;
  padding: 0;
}

#layout {
  bottom: 0;
  box-sizing: border-box;
  left: 0;
  position: absolute;
  right: 0;
  top: 0;
}
```

```
'use strict';

var compose = function(fn1, fn2) {
  return function() {
    fn1(fn2);
  };
};

var either = function(val, ...vars)
  return [...variants].indexOf(val) > -1;
```

HTML

Разметка:
структура элемента,
индексация
в поисковиках,
доступность,
стандартное поведение
(ссылки, формы)

CSS

Внешний вид,
простое поведение
(ховеры, фокусы)

JavaScript

Любое сложное
поведение.
Переопределение
стандартного поведения.
Работа с сетью.



JavaScript –

*язык программирования сценариев (script)
со слабой типизацией и динамическим приведением типов.*



Язык сценариев –

язык системной интеграции. Языки, которые работают
с готовыми программными компонентами.



Движки JavaScript

- MS Internet Explorer 11
- MS Edge
- Gecko (Moz://a)
- Webkit (Safari)
- Blink (Chrome, Chromium, Opera, Yandex...)



JavaScript в браузере

- **DOM** – *Document Object Model*. Работа со страницей: разметка, события
- **BOM** – *Browser Object Model*. работа с браузером: история, хранилища, окно, cookies, таймеры, таймауты
- **HTTP** – работа с сетью
- **Графика <canvas>** – прослойка для создания графики



Синхронное подключение JS

```
<body>

<header></header>

<script src="external.js"></script>

<main>

<aside></aside>

<section></section>

</main>

</body>
```



Асинхронное подключение JS – 1

```
<body>
```

```
  <header></header>
```

```
  <script src="external.js" async></script>
```

```
  <main>
```

```
    <aside></aside>
```

```
    <section></section>
```

```
  </main>
```

```
</body>
```



Асинхронное подключение JS – 2

```
<body>
```

```
  <header></header>
```

```
  <script src="external.js" defer></script>
```

```
  <main>
```

```
    <aside></aside>
```

```
    <section></section>
```

```
  </main>
```

```
</body>
```



Программирование = Алгоритмы + Структуры данных

<https://ru.wikipedia.org/wiki/Программирование>



Алгоритм —

законченный и упорядоченный набор действий, которые нужно предпринять, чтобы достигнуть прогнозируемого результата



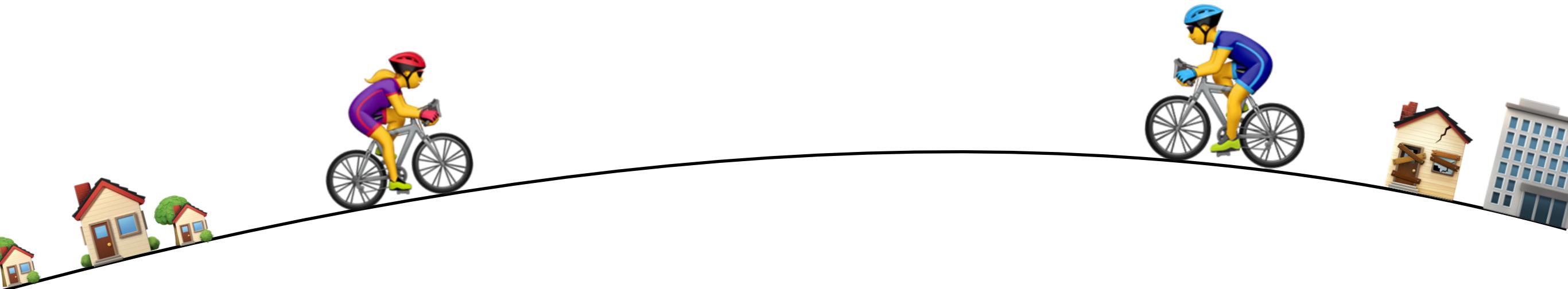
Примеры алгоритмов

- готовка по рецепту
- перемещение по навигатору
- решение квадратного уравнения
- сортировка



Задача

Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через 3 часа. Первый велосипедист ехал со скоростью 12 км/ч, а второй – 14 км/ч. На каком расстоянии находятся поселки?



Решение задачи с велосипедистами

1. найти расстояние, которое проехал первый велосипедист, умножив его скорость на время



Решение задачи с велосипедистами

1. найти расстояние, которое проехал первый велосипедист, умножив его скорость на время
2. найти расстояние, которое проехал второй велосипедист, умножив его скорость на время



Решение задачи с велосипедистами

1. найти расстояние, которое проехал первый велосипедист, умножив его скорость на время
2. найти расстояние, которое проехал второй велосипедист, умножив его скорость на время
3. сложить полученные расстояния



Решение задачи с велосипедистами

- **закончено**
известно окончательное количество шагов, необходимое для получения решения
- **упорядочено**
невозможно сложить расстояния велосипедистов до того как они будут посчитаны
- **прогнозируемо**
результат выполнения алгоритма предсказуем, мы всегда можем предположить что получится в конце





Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через 3 часа. Первый велосипедист ехал со скоростью 12 км/ч, а второй – 14 км/ч. На каком расстоянии находятся поселки?

Дано

$$t = 3 \text{ ч}$$

$$v_1 = 12 \text{ км/ч}$$

$$v_2 = 14 \text{ км/ч}$$

$$s = ?$$

Решение





Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через 3 часа. Первый велосипедист ехал со скоростью 12 км/ч, а второй – 14 км/ч. На каком расстоянии находятся поселки?

Дано

$$t = 3 \text{ ч}$$

$$v_1 = 12 \text{ км/ч}$$

$$v_2 = 14 \text{ км/ч}$$

$$s = ?$$

Решение

$$s_1 = t \times v_1$$

$$s_1 = 3 \times 12 = 36 \text{ км}$$

$$s_2 = t \times v_2$$

$$s_2 = 3 \times 14 = 42 \text{ км}$$

$$s = s_1 + s_2 = \underline{78 \text{ km}}$$





Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через 3 часа. Первый велосипедист ехал со скоростью 12 км/ч, а второй – 14 км/ч. На каком расстоянии находятся поселки?

Дано

$$t = 3 \text{ ч}$$

$$v_1 = 12 \text{ км/ч}$$

$$v_2 = 14 \text{ км/ч}$$

$$s = ?$$

**числовые
данные**

Решение

$$s_1 = t \times v_1$$

$$s_1 = 3 \times 12 = 36 \text{ км}$$

$$s_2 = t \times v_2$$

$$s_2 = 3 \times 14 = 42 \text{ км}$$

$$s = s_1 + s_2 = \underline{\underline{78 \text{ km}}}$$





Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через 3 часа. Первый велосипедист ехал со скоростью 12 км/ч, а второй – 14 км/ч. На каком расстоянии находятся поселки?

Дано

$$t = 3 \text{ ч}$$

$$v_1 = 12 \text{ км/ч}$$

$$v_2 = 14 \text{ км/ч}$$

$$s = ?$$

**числовые
данные**

Решение

$$s_1 = t \times v_1$$

$$s_1 = 3 \times 12 = 36 \text{ км}$$

$$s_2 = t \times v_2$$

$$s_2 = 3 \times 14 = 42 \text{ км}$$

$$s = s_1 + s_2 = \underline{\underline{78 \text{ km}}}$$

**математические
операции**



Типы данных

Набор допустимых значений и действия которые с ними можно производить



Типы данных

- Примитивные
 - числа
 - строки (*текст*)
 - логические (булевы значения)
 - служебные (*null*, *undefined*)
- Сложные
 - объекты



– *It's true!*



Числа (*number*)

формат данных для представления числовой информации: целых, дробных чисел, бесконечности и неизвестного числового значения



Числа (*number*)

- записываются без дополнительных символов
- поддерживается работа с целыми и дробными числами, дробная часть отделяется точкой
- существует два дополнительных значения: `Infinity` и `NaN`
- целые числа могут быть записаны в десятеричном или в шестнадцатеричном формате `0xFF`



Строки (*string*)

формат данных для представления текстовой информации



Строки (*string*)

- записываются внутри одинарных или двойных кавычек
`'Hello, I am a string';`
`"Hello, I am also a string";`
- называются строками, потому что переносы, пробелы и отступы являются обычными символами, поэтому весь текст хранится в виде одной длинной последовательности символов
- в случае, если в строке нужно использовать специальный символ, используется экранирование
`'\tHello, I\'m a string\n\n';`



Оператор –

минимальная единица языка, команда. Оператор принимает несколько значений, преобразует их и возвращает новое значение



Операторы

- **унарные**
операторы, которые работают с одним значением
- **бинарные**
операторы, работающие с двумя значениями
- **тернарный**
единственный в языке оператор, принимает на вход три значения



Операторы

- математические
- строковый оператор
- логические операторы
 - операторы сравнения
 - операторы булевой логики
- служебные операторы





Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через 3 часа. Первый велосипедист ехал со скоростью 12 км/ч, а второй – 14 км/ч. На каком расстоянии находятся поселки?

Дано

$$\underline{t} = 3 \text{ ч}$$

$$\underline{v_1} = 12 \text{ км/ч}$$

$$\underline{v_2} = 14 \text{ км/ч}$$

$$\underline{s} = ?$$

Решение

$$s = 3 \times 12 + 3 \times 14 = 78 \text{ km}$$

$$\underline{s_1} = t \times v_1$$

$$s_1 = 3 \times 12 = 36 \text{ км}$$

$$\underline{s_2} = t \times v_2$$

$$s_2 = 3 \times 14 = 42 \text{ км}$$

**поименованные
значения**

$$s = s_1 + s_2 = 78 \text{ km}$$



Переменные —

способ запомнить значения, чтобы использовать их после не вычисляя повторно, а просто обращаясь к значению по имени



Создание переменных

- ключевое слово **var**

```
var speedOfFirstCyclist = 32;  
var cyclistName = "Иван";
```

- в названии можно использовать буквы, цифры (*не в начале*), специальные символы (*не операторы*)

```
var 1value = 32;  
var value1 = 32;  
var one+two = 3;  
var onePlusTwo = 3;  
var $price = 100;
```

- не стоит называть переменные так же, как существующие конструкции языка

```
var var = "var";
```



Название переменных

- английское существительное в именительном падеже для значений. Начинается с маленькой буквы

```
var name = 'Ivan';  
var age = 120;  
var birthYear = 1988;
```

- английский глагол или фраза содержащая английский глагол для функций

```
var doSomethingGreat = function() {};
```



Разделение слов в переменных

- **camelCase**  слова не отделяются пробелами, каждое последующее слово начинается с заглавной буквы
- **snake_case**  вместо пробелов слова разделяются **нижним подчеркиванием**, каждое последующее слово начинается с прописной буквы



Приведение типов 😬



JavaScript –

язык программирования сценариев (*script*)
со слабой типизацией и **динамическим приведением** типов.



Приведение типов –

если операнды переданные оператору разного типа, они автоматически приводятся к типу, в котором операцию можно выполнить. Тип, в который будут приведены операнды зависит от оператора



Приведение типов –

- `'10' / '2' === 5`

математический оператор – приведение производится к числу

- `true + 'eee' === 'trueeee'`

строковый оператор – приведение производится к строке

- `'10' + 2 === '102'`

у строкового сложения приоритет над математическим



Явное приведение



Приведение встроенными методами

- приведение к числу

```
parseInt('10', 10) === 10;
```

```
parseFloat('10.2') === 10.2;
```

- приведение к строке

~~10.toString(); Uncaught SyntaxError~~

```
10..toString() === 10;
```

```
10.0.toString() === 10;
```



Приведение операторами

- приведение к числу с помощью оператора знака (унарный плюс или унарный минус)
`+ '10' === 10;`
`- '3' === -3;`
- приведение к строке сложением с пустой строкой
`3 + '' === '3';`
- приведение к булеву значению двойным отрицанием
`!!0 === false;`
`!!1 === true;`



Два велосипедиста одновременно выехали навстречу друг другу из двух поселков и встретились через ... Первый велосипедист ехал со скоростью ..., а второй – ... На каком расстоянии находятся поселки?

Дано

$$\underline{t} = \dots \text{ ч}$$

$$\underline{v_1} = \dots \text{ км/ч}$$

$$\underline{v_2} = \dots \text{ км/ч}$$

$$\underline{s} = ?$$

Решение

$$\underline{s_1} = t \times v_1$$

$$\underline{s_2} = t \times v_2$$

$$s = s_1 + s_2$$



ФУНКЦИИ –
способ использовать код повторно





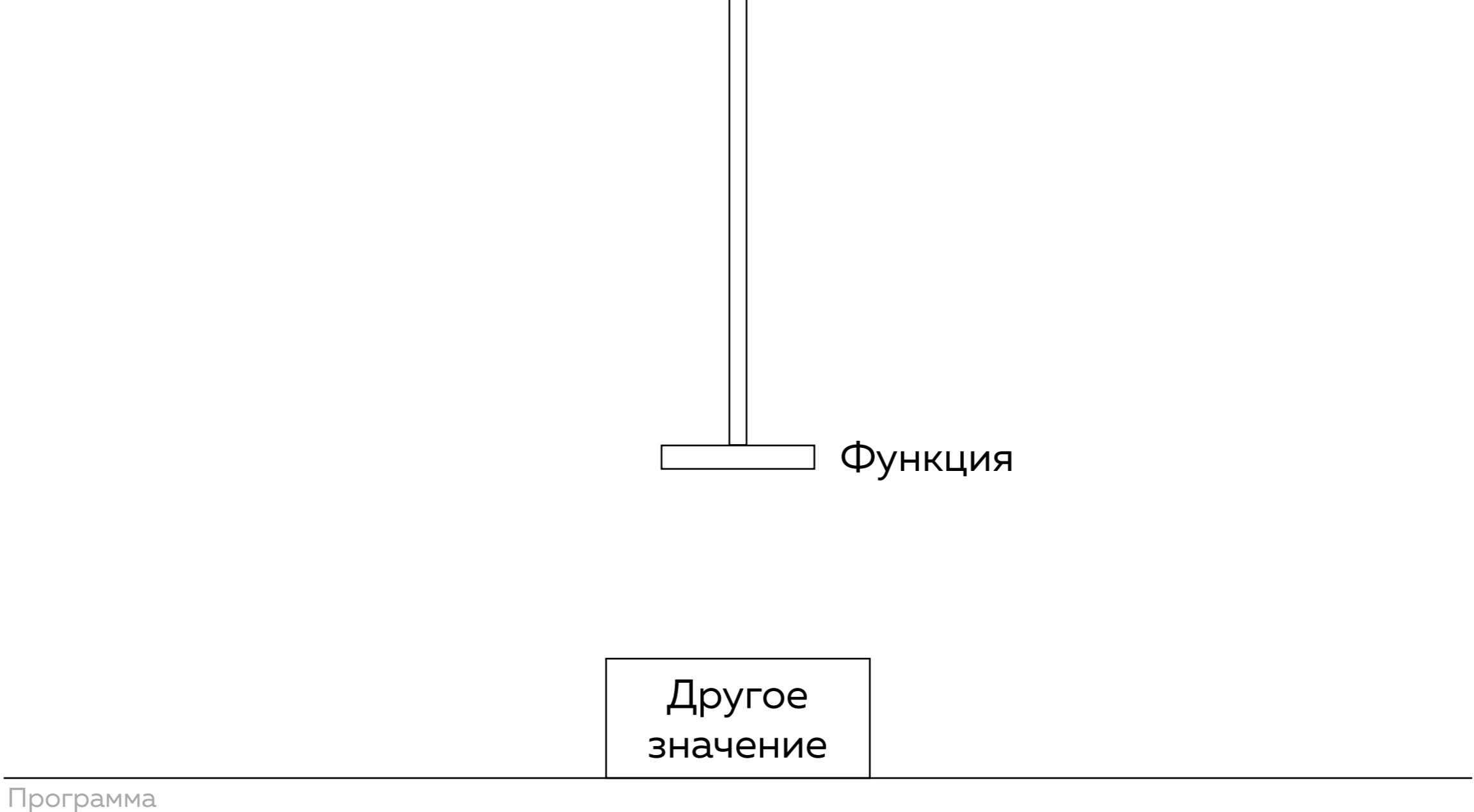


Программа



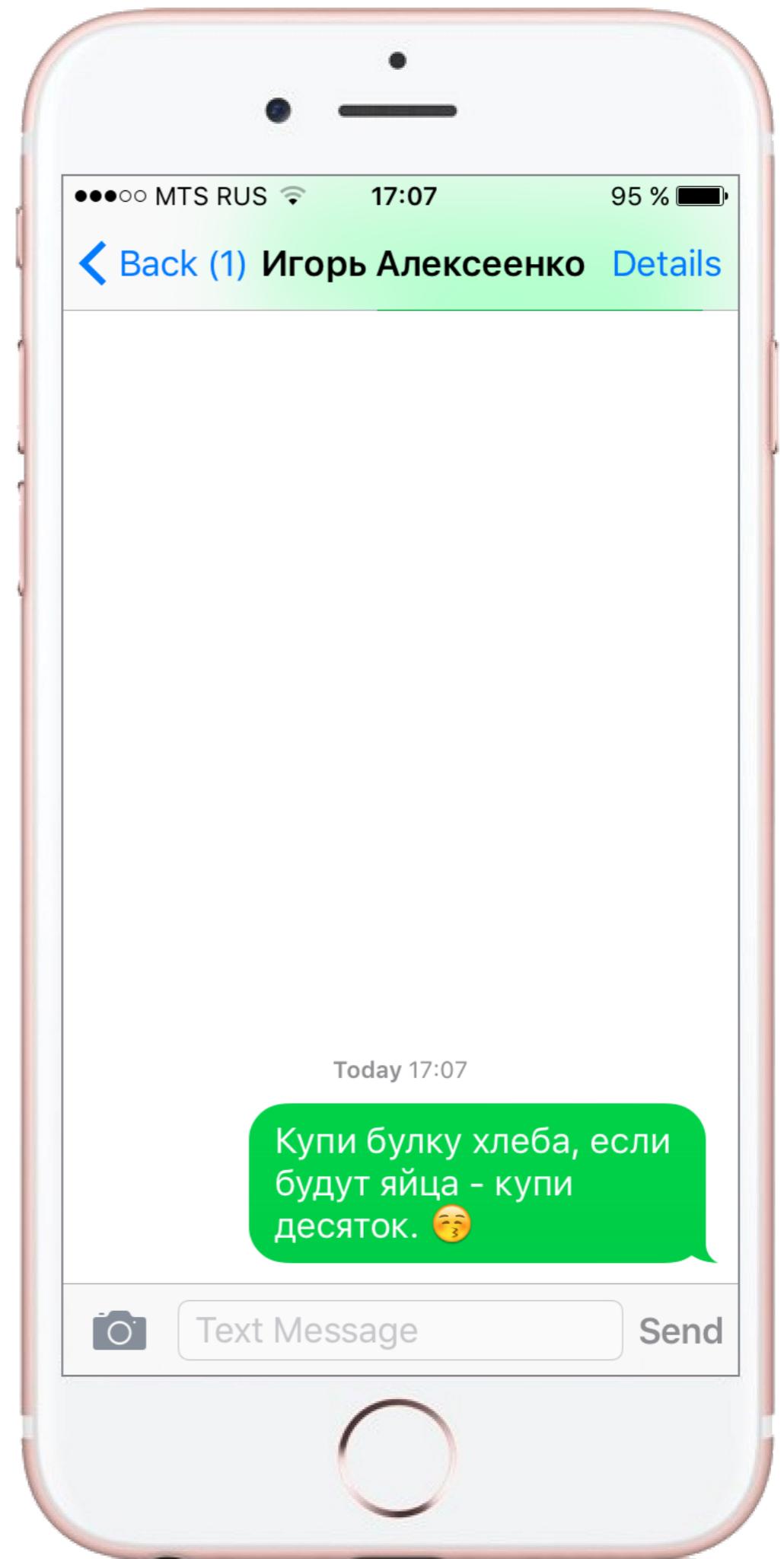
—Хрясь!







Не все
алгоритмы
линейные



Условные операторы

конструкция языка, выполняющая определенный набор действий только при выполнении некоего логического выражения



Булев (логический) тип данных

результат логических выражений. Имеет только два значения – истина (true) и ложь (false)



Булева логика и логические операторы



– It's true!



Тернарный оператор

оператор возвращающий второе или третье
переданное значение в зависимости от истинности
первого



Тернарный оператор

логическое
выражение

```
var result = true ? 1 : 0;
```

значение,
если true

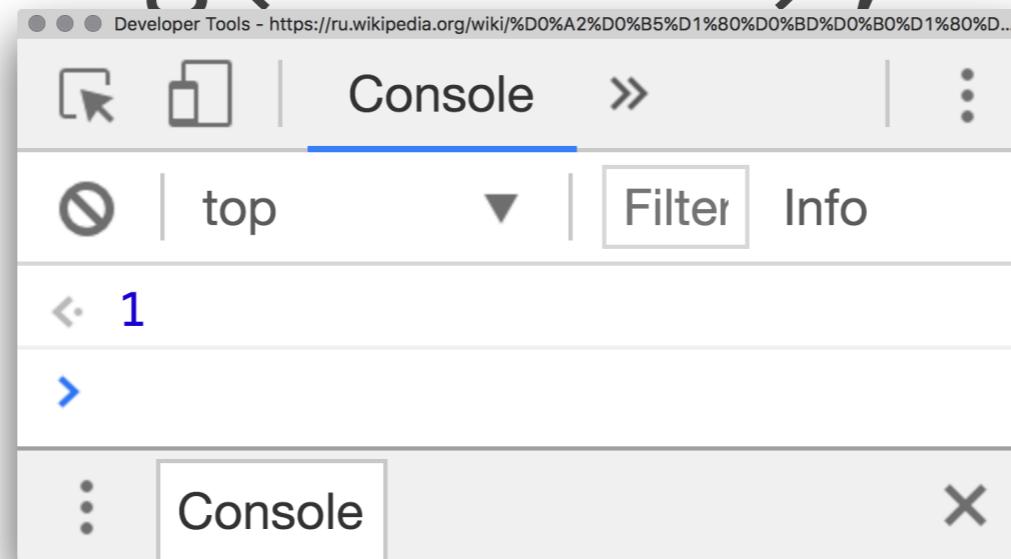
значение,
если false



Тернарный оператор

```
var result = true ? 1 : 0;
```

```
console.log(result);
```



Тернарный оператор

```
var result = false ? 1 : 0;  
console.log(result);
```

