title: Язык JavaScript

description: Только сам JavaScript, без браузерных и иных расширений.

slug: js-basic

questionsToAskCount: 20

weight: 1

questions:

-

content: |

Что получится, если сложить `true + false`?

type: single

answers:

- '`"truefalse"`'

- '`0`'

- '`1`'

- '`NaN`'

correctAnswer: 2

-

content: |

После выполнения этого кода -- в каких объектах из списка содержится свойство `name`?

```js

function User() { }

let vasya = new User();

vasya.\_\_proto\_\_.name = "Vasya";

```

type: multi

answers:

- '`vasya.\_\_proto\_\_`'

- '`vasya.prototype`'

- '`User.\_\_proto\_\_`'

- '`User.prototype`'

correctAnswer:

- 0

- 3

-

content: |

Какие варианты подключения скрипта являются корректными с точки зрения современного стандарта HTML?

type: multi

answers:

- '`<script type="text/javascript" src="my.js"></script>`'

- '`<script src="my.js"></script>`'

- '`<script src="my.js"/>`'

- '`<хачу-javascript отсюда="my.js">`'

correctAnswer:

- 0

- 1

-

content: |

Что такое ECMAScript?

type: single

answers:

- Новый язык программирования

- Переработанная реализация JavaScript

- Спецификация языка JavaScript

correctAnswer: 2

-

content: |

Правда ли что `a == b`?

```js

a = [1, 2, 3];

b = [1, 2, 3];

```

type: single

answers:

- Правда

- Неправда

- Как повезёт

correctAnswer: 1

-

content: |

Какие конструкции для циклов есть в JavaScript?

type: single

answers:

- 'Только две: `for` и `while`'

- 'Только одна: `for`'

- 'Три: `for`, `while` и `do...while`'

correctAnswer: 2

-

content: |

Какой оператор из этих выполняет не только математические операции?

type: single

answers:

- '`\*`'

- '`/`'

- '`+`'

- '`-`'

- '`>>>`'

correctAnswer: 2

-

content: |

Сработает ли вызов функции до объявления в этом коде:

```js

sayHi();

function sayHi() {

alert("Hello");

}

```

type: single

answers:

- Да, сработает

- Нет, вызов должен стоять после объявления

correctAnswer: 0

-

content: |

Какие из этих вызовов синтаксически верно сгенерируют исключение?

type: multi

answers:

- '`throw "Ошибка"`'

- '`throw new Error("Ошибка")`'

- '`throw { message: "Ошибка" }`'

- '`throw Error("Ошибка")`'

- Ни один

correctAnswer:

- 0

- 1

- 2

- 3

-

content: |

Сколько параметров можно передать функции?

type: single

answers:

- Ровно столько, сколько указано в определении функции

- Сколько указано в определении функции или меньше

- Сколько указано в определении функции или больше

- Любое количество

correctAnswer: 3

-

content: |

Какие из этих вариантов задают массив из элементов "a", "b"?

type: multi

answers:

- '`let a = new Array("a", "b")`'

- '`let a = { "a", "b" }`'

- '`let a = ( "a", "b" )`'

- '`let a = [ "a", "b" ]`'

- '`let a = "a,b".split(",")`'

correctAnswer:

- 0

- 3

- 4

-

content: |

Какие варианты правильно объявляют функцию `f`, возвращающую сумму двух аргументов?

type: multi

answers:

- '`let f = function(a,b) { return a + b }`'

- '`let f = new Function("a, b", "return a + b")`'

- '`let f = new Function("a", "b", "return a + b")`'

- '`let f = (a, b) => a + b`'

- '`let f = (a, b) => { a + b }`'

- Никакие

correctAnswer:

- 0

- 1

- 2

- 3

-

content: |

Что делает код:

```js

break me;

```

type: single

answers:

- Ломает интерпретатор JavaScript

- Выходит из текущего блока цикла или switch на метку "me"

- Выдает ошибку

correctAnswer: 1

-

content: |

Есть ли разница между выражениями?

```js

!!( a && b )

(a && b)

```

type: single

answers:

- Да

- Нет

- В первом выражении ошибка, что ещё за "!!"??

correctAnswer: 0

-

content: |

Чему равна длина `arr.length` массива `arr`?

```js

let arr = [];

arr[1] = 1;

arr[3] = 33;

```

type: single

answers:

- 0

- 1

- 2

- 3

- 4

- Больше

correctAnswer: 4

-

content: |

Чему равно `i` в конце кода?

```js

for (var i = 0; i < 10; i++) {

console.log(i);

}

// i = ?

```

type: single

answers:

- '`undefined`'

- '`9`'

- '`10`'

- Нет такой переменной после цикла

correctAnswer: 2

-

content: |

Чему равно `i` в конце кода?

```js

for (let i = 0; i < 10; i++) {

console.log(i);

}

// i = ?

```

type: single

answers:

- '`undefined`'

- '`9`'

- '`10`'

- Нет такой переменной после цикла

correctAnswer: 3

-

content: |

Что выведет `sayHi` при вызове через `setTimeout`?

```js

let name = "Вася";

function sayHi() {

alert(name);

}

setTimeout(function() {

let name = "Петя";

sayHi();

}, 1000);

```

type: single

answers:

- Вася

- Петя

- '`undefined`'

- Будет ошибка

correctAnswer: 0

-

content: |

Существует ли такое значение `X`, которое после присваивания `a = X` вызов `alert(a == X)` выдаст `false`?

```js

let a = X;

alert(a == X); // false

```

type: single

answers:

- 'Да, `X` -- это `undefined`'

- 'Да, `X` -- это `null`'

- 'Да, другое'

- 'Нет, не бывает'

correctAnswer: 2

-

content: |

Чему равна переменная `name`?

```js

let name = "пупкин".replace("п", "д")

```

type: single

answers:

- дудкин

- дупкин

- пупкин

- ляпкин-тяпкин

correctAnswer: 1

-

content: |

Каких операторов из этого списка нет в JavaScript?

type: multi

answers:

- '`\*`'

- '`^`'

- '`%`'

- '`#`'

- '`&`'

- '`>>`'

- '`>>>`'

- '`!`'

correctAnswer:

- 3

-

content: |

Чему равно `a + b + c`?

```js

let a = 1;

let b = { toString() {return '1'} };

let c = 1;

```

type: single

answers:

- '`11[object Object]`'

- '`2[object Object]`'

- '`111`'

- '`3`'

correctAnswer: 2

-

content: |

Что делает оператор `===`?

type: single

answers:

- Сравнивает по ссылке, а не по значению

- Сравнивает без приведения типа

- Нет такого оператора

correctAnswer: 1

-

content: |

Что делает оператор `\*\*`?

type: single

answers:

- Возводит в степень

- Умножает число само на себя

- Нет такого оператора

correctAnswer: 0

-

content: |

Какое из этих слов не имеет специального использования в JavaScript, никак не упомянуто в стандарте?

type: single

answers:

- '`this`'

- '`instanceof`'

- '`constructor`'

- '`parent`'

- '`new`'

- Все имеют специальное использование

correctAnswer: 3

-

content: |

Какая арифметическая операция приводит к ошибке в JavaScript?

type: single

answers:

- Деление на ноль

- Умножение числа на строку

- Корень из отрицательного числа

- Никакая из вышеперечисленных

correctAnswer: 3

-

content: |

Чему равно `0 || "" || 2 || undefined || true || falsе`?

type: single

answers:

- '`0`'

- '`""`'

- '`2`'

- '`undefined`'

- '`true`'

- '`false`'

correctAnswer: 2

-

content: |

Чему равно `2 && 1 && null && 0 && undefined`?

type: single

answers:

- '`2`'

- '`1`'

- '`null`'

- '`0`'

- '`undefined`'

- '`false`'

correctAnswer: 2

-

content: |

Чему равно `0 || 1 && 2 || 3`?

type: single

answers:

- '`0`'

- '`1`'

- '`2`'

- '`3`'

- '`true`'

- '`false`'

correctAnswer: 2

-

content: |

Что выведет `alert`?

```js

let arr = [1, 2, 3];

arr.something = 5;

alert(arr.something); // ?

```

type: single

answers:

- '`5`'

- '`undefined`'

- Будет ошибка

correctAnswer: 0

-

content: |

Что выведет `alert`?

```js

let str = "Hello";

str.something = 5;

alert(str.something); // ?

```

type: single

answers:

- '`5`'

- '`undefined`'

- Будет ошибка

correctAnswer: 2

-

content: |

Что выведет `alert`?

```js

alert(str); // ?

let str = "Hello";

```

type: single

answers:

- '`Hello`'

- '`undefined`'

- Будет ошибка

correctAnswer: 2

-

content: |

Что выведет `alert`?

```js

alert(str); // ?

var str = "Hello";

```

type: single

answers:

- '`Hello`'

- '`undefined`'

- Будет ошибка

correctAnswer: 1

-

content: |

Что выведет этот код?

```js

for (var i = 0; i < 10; i++) {

setTimeout(function() {

alert(i);

}, 100);

}

```

type: single

answers:

- Числа от 0 до 9

- Числа от 0 до 10

- 10 раз число 0

- 10 раз число 10

- 'Ошибка: переменная не определена'

correctAnswer: 3

-

content: |

Что выведет этот код?

```js

for (let i = 0; i < 10; i++) {

setTimeout(function() {

alert(i);

}, 100);

}

```

type: single

answers:

- Числа от 0 до 9

- Числа от 0 до 10

- 10 раз число 0

- 10 раз число 10

- 'Ошибка: переменная не определена'

correctAnswer: 0

-

content: |

Что выведет этот код?

```js

function User() { }

User.prototype = { admin: false };

let user = new User();

alert(user.admin);

```

type: single

answers:

- '`false`'

- '`undefined`'

- '`true`'

correctAnswer: 0

-

content: |

Что выведет этот код?

```js

function User() { }

User.prototype = { admin: false };

let user = new User();

User.prototype = { admin: true };

alert(user.admin);

```

type: single

answers:

- '`false`'

- '`undefined`'

- '`true`'

correctAnswer: 0

-

content: |

Чему равно `arr.length`?

```js

function MyArray() { }

MyArray.prototype = [];

let arr = new MyArray();

arr.push(1, 2, 3);

alert(arr.length);

```

type: single

answers:

- '`0`'

- '`undefined`'

- '`3`'

- 'В этом коде допущена ошибка'

correctAnswer: 2

-

content: |

Какое будет выведено значение?

```js

let x = 5;

alert(x++);

```

type: single

answers:

- '`5`'

- '`6`'

- Другое

correctAnswer: 0

-

content: |

Выберите правильные варианты объявления массива, то есть такие, в результате которых мы получаем массив из двух чисел `1` и `2`.

type: multi

answers:

- '`new Array.prototype.constructor(1, 2)`'

- '`new Array(1, 2)`'

- '`Array(1, 2)`'

- '`[1, 2]`'

- '`1..2`'

- Все варианты правильные

correctAnswer:

- 0

- 1

- 2

- 3

-

content: |

Чему равно это выражение?

```js

[].push(1,2).unshift(3).join()

```

type: single

answers:

- "`3,1`"

- "`1,2,3`"

- "`3,1,2`"

- В коде ошибка

correctAnswer: 3

-

content: |

Какие варианты вызова `try..catch` являются синтаксически верными в JavaScript?

type: multi

answers:

- '`try { ... }` без `catch/finally`'

- '`try { ... } catch { ... }`'

- '`try { ... } finally { ... }`'

- '`try { ... } catch { ... } finally { ... }`'

- 'В JavaScript не поддерживается `try..catch`'

correctAnswer:

- 1

- 2

- 3

-

content: |

Язык JavaScript является подвидом языка Java -- верно?

type: single

answers:

- Да

- Нет

- Наоборот, Java -- подвид JavaScript

correctAnswer: 1

-

content: |

Что выведет этот код?

```js

f.call(f);

function f() {

alert(this);

}

```

type: single

answers:

- '`[object Object]`'

- 'Код функции `f`'

- 'Ошибка: слишком глубокая рекурсия'

- 'Ошибка: переменная `f` не определена'

- другое

correctAnswer: 1

correctAnswerComment: |

Функция `f` определяется при входе в область видимости, так что на первой строке она уже есть.

Далее вызов `f.call(f)` вызывает функцию, передавая ей `f` в качестве `this`, так что выводится строковое представление `f`.

-

content: |

Что выведет этот код?

```js

let f = function g() { return 23; };

alert(typeof g());

```

type: single

answers:

- '`number`'

- '`undefined`'

- '`function`'

- Ошибка

correctAnswer: 3

correctAnswerComment: |

Ошибка, так как это Named Function Expression.

-

content: |

Что выведет этот скрипт?

```js

"use strict";

a = null + undefined;

alert(a);

```

type: single

answers:

- '`null`'

- '`undefined`'

- '`NaN`'

- В коде ошибка

correctAnswer: 3

correctAnswerComment: |

Переменная не определена, в строгом режиме все переменные должны быть определены.

-

content: |

Что выведет этот код?

```js

let y = 1;

let x = y = 2;

alert(x);

```

type: single

answers:

- '`1`'

- '`2`'

- '`NaN`'

- '`undefined`'

- Будет ошибка

correctAnswer: 1

correctAnswerComment: |

Двойное присваивание выполняется справа налево. То есть сначала `y` станет `2`, а потом `x = y`.

-

content: |

Что выведет этот код?

```js

f.call(null);

function f() {

alert(this);

}

```

type: single

answers:

- '`null`'

- '`undefined`'

- В коде ошибка

- Другое

correctAnswer: 0

correctAnswerComment: |

В строгом режиме `call` передаёт в качестве `this` именно то, что указано.

-

content: |

Чему будет равен `this`?

```js

let user = {

sayHi: function() {

alert(this);

}

};

(user.sayBye = user.sayHi)();

```

type: single

answers:

- '`null`'

- '`undefined`'

- 'Объекту `user`'

- В коде ошибка

correctAnswer: 1

correctAnswerComment: |

Контекст вызова теряется при операциях с методом объекта.

-

content: |

Чему равна сумма `[] + 1 + 2`?

type: single

answers:

- '`3`'

- '`NaN`'

- '`undefined`'

- '`"12"`'

- '`12`'

- Другое

correctAnswer: 3

correctAnswerComment: |

Первым делом объект `[]` превращается в примитив. У него нет `valueOf` (с примитивным результатом), так что вызывается `toString` и возвращает список элементов через запятую, т.е. пустую строку.

Получается сложение `"" + 1 + 2`. Далее, так как один из аргументов -- строка, то оператор `"+"` преобразует второй тоже к строке, и в итоге получится строка `"12"`.

-

content: |

Верно ли, что `null == undefined`?

type: single

answers:

- Да

- Нет

correctAnswer: 0

correctAnswerComment: |

Преобразования типов здесь не происходит.

Оператор `==` работает с `null/undefined` без преобразования, все варианты сравнений прописаны в спецификации.

-

content: |

Что выведет этот код?

```js

if (function f(){}) {

alert(typeof f);

}

```

type: single

answers:

- '`undefined`'

- '`function`'

- '`null`'

- '`object`'

- В коде ошибка

correctAnswer: 0

correctAnswerComment: |

Функция задана как Named Function Expression, поэтому её имя не видно снаружи.

-

content: |

Чему равен результат вызова в этом примере?

```js

function f() {

let a = 5;

return new Function('b', 'return a + b');

}

alert(f()(1));

```

type: single

answers:

- '`1`'

- '`6`'

- '`undefined`'

- '`NaN`'

- Будет ошибка

correctAnswer: 4

correctAnswerComment: |

Ошибка, так как функция, созданная через `new Function`, не имеет доступа к текущему замыканию.

-

content: |

Объявлена функция:

```js

function F() {}

```

Верно ли, что `F instanceof Function`?

type: single

answers:

- Да

- Нет

correctAnswer: 0

-

content: |

Объявлена функция:

```js

function F() {}

```

Чем является `F.prototype`?

type: single

answers:

- 'Обычным объектом'

- 'Функцией'

- 'Равен `undefined`'

correctAnswer: 0

-

content: |

Что выведет этот код?

```js

function F() { return F; }

alert(new F() instanceof F);

alert(new F() instanceof Function);

```

type: single

answers:

- '`false`, `false`'

- '`false`, `true`'

- '`true`, `false`'

- '`true`, `true`'

correctAnswer: 1

correctAnswerComment: |

Если функция, запущенная через `new`, возвращает объект (не примитив!), то именно этот объект служит результатом, вместо `this`. Так что результатом `new F` является сама функция `F`.

Поэтому получается, что первый `alert` проверяет: `F instanceof F`. Это неверно, т.к. `a instanceof B` проверяет, был ли объект `a` создан конструктором `B` (с учетом прототипов, но здесь это неважно). Функция `F` не была создана собой же, а является объектом встроенного класса `Function`, поэтому `false`.

Следующая строка идентична такой проверке: `F instanceof Function`. Это верно.

-

content: |

Чему равен `typeof null` в режиме строгом режиме?

type: single

answers:

- '`null`'

- '`undefined`'

- '`object`'

- '`string`'

correctAnswer: 2

correctAnswerComment: |

Это особенность спецификации JavaScript. Значением оператора `typeof null` является строка `"object"`.

-

content: |

Что выведет этот код?

```js

alert("1"[0]);

```

type: single

answers:

- '`0`'

- '`1`'

- '`2`'

- '`undefined`'

- В коде ошибка

correctAnswer: 1

correctAnswerComment: |

Первый символ строки `"1"` -- это `"1"`.

-

content: |

Что выведет этот код?

```js

alert(20e-1['toString'](2));

```

type: single

answers:

- '`2`'

- '`10`'

- '`20`'

- '`NaN`'

- В коде ошибка

correctAnswer: 1

correctAnswerComment: |

<ol>

<li>Запись числа `20e-1` означает `20`, сдвинутое на 1 знак после запятой, т.е. `2`:

```js

//+ run

alert(20e-1); // 2

```

</li>

<li>У каждого числа есть метод `toString(radix)`, который преобразует число в строку, используя `radix` как основание системы счисления. В этом коде `radix = 2`. Так что возвращается `2` в двоичной системе: `"10"`.

</li>

</ol>

-

content: |

Что выведет этот код?

```js

alert(+"Infinity");

```

type: single

answers:

- '`Infinity`'

- '`NaN`'

- '`0`'

- Будет ошибка

correctAnswer: 0

correctAnswerComment: |

`Infinity` -- числовое значение (бесконечность).

-

content: |

Что выведет этот код?

```js

let a = (1,5 - 1) \* 2;

alert(a);

```

type: single

answers:

- '`0.999999999`'

- '`1`'

- '`0.5`'

- '`8`'

- '`-0.5`'

- '`4`'

correctAnswer: 3

correctAnswerComment: |

Оператор "запятая", вычисляемый в скобках, возвращает последнее вычисленное значение. То есть, `1` игнорируется и возвращается `4` (`5 - 1`).

Затем идёт умножение на два, в результате -- `8`.

-

content: |

Что выведет этот код? Посмотрите на него очень внимательно, в этом вопросе есть подвох.

```js

let a = [1, 2]

(function() { alert(a) })()

```

type: single

answers:

- '`[object Array]`'

- '`[object Object]`'

- '`1,2`'

- Будет ошибка

correctAnswer: 3

correctAnswerComment: |

Пропущена точка с запятой после `let a = [1, 2]`.

Браузер автоматически не вставляет её перед скобкой, поэтому интерпретирует это как:

```js

let a = [1, 2](function() { alert(a) })()

```

...То есть, пытается вызвать массив `[1, 2]` как функцию, что, естественно, не выходит. Отсюда и ошибка.

-

content: |

Верно ли сравнение: `"ёжик" > "яблоко"`?

type: single

answers:

- 'Да'

- 'Нет'

- Зависит от локальных настроек браузера

correctAnswer: 0

correctAnswerComment: |

В JavaScript строки сравниваются посимвольно. Символы соотносятся как их коды.

В кодировке Unicode код буквы `"ё"` больше кода буквы `"я"`. Поэтому `"ёжик"` больше.

-

content: |

Какой результат будет у выражения ниже?

```js

null + {0:1}[0] + [,[1],][1][0]

```

type: single

answers:

- '`0`'

- '`1`'

- '`2`'

- '`undefined`'

- '`NaN`'

correctAnswer: 2

correctAnswerComment: |

Подробнее вычисления:

<ol>

<li>`null` превращается в `0`</li>

<li>`{0:1}` -- это объект, у которого ключ `0` имеет значение `1`

```js

{0:1}[0] == 1

```

</li>

<li>

Второе слагаемое `[,[1],]` -- это массив с 3-мя элементами. Элемент с индексом `1` -- это массив `[1]`. Берём от него первый элемент:

```js

[,[1],][1][0] == 1

```

</li>

</ol>

-

content: |

Что выведет выражение ниже?

```js

let a = new Array(1,2), b = new Array(3);

alert(a[0] + b[0]);

```

type: single

answers:

- '`1`'

- '`4`'

- '`undefined`'

- '`NaN`'

correctAnswer: 3

correctAnswerComment: |

Дело в том, что `new Array(1,2)` создаёт массив из элементов `[1,2]`, а вот `new Array(3)` -- это особая форма вызова `new Array` с одним численным аргументом. При этом создаётся массив без элементов, но с длиной. Любой элемент этого массива равен `undefined`.

```js

//+ run

let a = new Array(1,2), b = new Array(3);

alert(a[0]); // 1

alert(b[0]); // undefined

```

При арифметических операциях `undefined` становится `NaN`, поэтому и общий результат -- `NaN`.

-

content: |

Яблоко стоит 1.15, апельсин стоит 2.30.

Сколько стоят они вместе -- чему равна сумма `1.15 + 2.30` с точки зрения JavaScript?

type: single

answers:

- '`345`'

- '`3.45`'

- '`3,45`'

- Ни один из вариантов выше

correctAnswer: 3

correctAnswerComment: |

При сложении десятичных дробей проявляются ошибки округления, поэтому сумма будет другой:

```js

//+ run

alert(1.15 + 2.30);

```

-

content: |

Что выведет код ниже?

```js

let obj = {'1': 0, 1: 1, 0: 2};

alert(obj['1']);

```

type: single

answers:

- '`0`'

- '`1`'

- В коде ошибка

correctAnswer: 1

correctAnswerComment: |

При задании объекта через `{...}` -- кавычки не обязательны, они нужны лишь в случаях, когда без них нельзя, например:

```js

let obj = { "строка с пробелами" : 123 }

```

Все ключи приводятся к строке. Поэтому второй ключ равен первому и перезаписал его.

-

content: |

Что выведет код ниже?

```js

for (let key in {1:1, 0:0}) {

alert(key);

}

```

type: single

answers:

- '`1`, затем `0`'

- '`0`, затем `1`'

- В коде ошибка

- Зависит от браузера

correctAnswer: 1

correctAnswerComment: |

Все современные браузеры сортируют цифровые ключи.

-

content: |

Есть ли различия между проверками:

```js

if ( x <= 100 ) {...}

// и

if ( !(x > 100) ) {...}

```

type: single

answers:

- 'Да, существует значение `x`, для которого они работают по-разному'

- 'Нет, они полностью взаимозаменяемы.'

- Зависит от браузера.

correctAnswer: 0

correctAnswerComment: |

Различие будет при `x = NaN` (или любого другого значения, которое преобразуется к `NaN`).

-

content: |

Внимательно посмотрите на синтаксис этого кода. Что он выведет?

```js

let f = function(x) {

alert(x)

}

(function() {

f(1)

}())

```

type: single

answers:

- Никакого результата

- 'Выведет `1`'

- 'Выведет `undefined`'

- Будет ошибка

correctAnswer: 3

correctAnswerComment: |

Ошибка из-за отсутствия точки с запятой в строке `(\*)`:

```js

//+ run

let f = function(x) {

alert(x)

} // (\*)

(function() {

f(1)

}())

```

JavaScript не вставляет точку с запятой на этом месте, так как ниже идёт открывающая скобка. Поэтому этот код воспринимается так:

```js

let f = function(x) {

alert(x)

}(function() { f(1) }())

```

-

content: |

Что выведет этот код?

```js

let obj = {

"0": 1,

0: 2

};

alert( obj["0"] + obj[0] );

```

type: single

answers:

- '`2`'

- '`3`'

- '`4`'

- '`12`'

- В коде ошибка

correctAnswer: 2

correctAnswerComment: |

Дело в том, что у объектов в JavaScript ключи всегда строковые. Если в качестве ключа передано что-то ещё, то оно приводится к строке. Значения обрабатываются в порядке поступления, поэтому `0: 2` перекроет `"0": 1`.

В итоге получится объект с единственным ключом: `{"0" : 2}`.

При доступе к ключу -- `obj["0"]` и `obj[0]` эквивалентны, опять же, поскольку ключ приводится к строке.

-

content: |

Чему равно такое выражение?

```js

[] + false - null + true

```

type: single

answers:

- '`0`'

- '`NaN`'

- '`undefined`'

- '`1`'

correctAnswer: 1

correctAnswerComment: |

Шаги преобразования:

<ol>

<li>`[] + false = "false"`, т.к массив `[]` преобразуется к пустой строке, выходит `"" + false = "false"`.</li>

<li>Далее `"false" - null = NaN`, т.к. вычитание преобразует к числу, получается `NaN - 0 = NaN`.</li>

<li>...дальнейшие вычисления с `NaN` дают `NaN`</li>

</ol>

-

content: |

Чему равно значение выражения `4 - "5" + 0xf - "1e1"`?

type: single

answers:

- 'Цифре'

- 'Строке'

- '`NaN`'

correctAnswer: 0

correctAnswerComment: |

<ol>

<li>Первый минус превратит строку `"5"` в число `5`</li>

<li>Далее идёт сложение с числом 15, записанным в 16-ричной форме.</li>

<li>Далее строка содержит число, записанное в научной форме: `1e1 = 10`, минус преобразует эту строку в число.</li>

</ol>

Получается:

```js

4 - 5 + 15 - 10 = 4

```

-

content: |

Переменные `apple` и `Apple` (с большой буквы) -- это одна и та же или разные?

type: single

answers:

- Одна и та же

- Разные

- С большой буквы переменные называть нельзя

- Слово "apple" является зарезервированным, нельзя использовать

correctAnswer: 1

-

content: |

Есть ли разница между вызовами `i++` и `++i`?

type: single

answers:

- Разница в значении, которое возвращает такой вызов

- "Разница в значении `i` после вызова"

- Нет никакой разницы

correctAnswer: 0

-

content: |

Какие вызовы `parseInt` возвратят число?

type: multi

answers:

- '`parseInt("1px")`'

- '`parseInt("-1.2")`'

- '`parseInt("0 минут")`'

- '`parseInt("$1.2")`'

correctAnswer:

- 0

- 1

- 2

-

content: |

Какой код корректно найдёт максимальное значение в непустом массиве `arr`?

type: multi

answers:

- "`arr.reduce(function(prev, item) { return Math.max(prev, item) })`"

- "`Math.max.apply(null, arr)`"

- "`Math.max(arr)`"

- '`arr.findMax()`'

correctAnswer:

- 0

- 1

title: 'DOM, работа со страницей'

description: 'Элементы, их содержимое, координаты, обработчики событий.'

slug: 'dom'

questionsToAskCount: 20

weight: 2

questions:

-

content: |

Сколько детей (дочерних DOM-узлов) у элемента `<ul>` в примере ниже:

```html

<ul>

<li>Привет</li>

<li>Мир</li>

</ul>

```

correctAnswer: 4

answers:

- '1'

- '2'

- '3'

- '4'

- '5'

- '6'

type: 'single'

-

content: |

В каких случаях для элемента `elem` верно `elem.firstChild == elem.lastChild`?

correctAnswer:

- 0

- 1

answers:

- 'Когда у `elem` нет детей.'

- 'Когда у `elem` единственный потомок.'

- 'Когда у `elem` два или более потомков.'

type: 'multi'

-

content: |

Как получить DOM-элемент `form` в документе?

```html

<!DOCTYPE HTML>

<html>

<body>

<form name="myform">...</form>

</body>

</html>

```

correctAnswer:

- 0

- 2

- 4

answers:

- '`document.forms.myform`'

- '`document.getElementById("myform")`'

- '`document.body.children[0]`'

- '`document.body.firstChild`'

- '`document.forms[0]`'

type: 'multi'

-

content: |

Дан такой HTML:

```html

<div id="outer">

<div id="inner"></div>

</div>

```

Какие вызовы вернут `div#inner`?

correctAnswer:

- 0

- 2

answers:

- '`document.querySelector("#outer #inner")`'

- '`document.getElementById("#outer").getElementById("#inner")`'

- '`document.querySelector("#outer").children[0]`'

- '`document.getElementById("#outer").firstChild`'

type: 'multi'

-

content: |

Есть такой элемент:

```html

<input id="input" value="Привет">

```

Какие вызовы поменяют значение в нём?

correctAnswer:

- 0

- 2

answers:

- '`input.value = "Пока"`'

- '`input.setValue("Пока")`'

- '`input.setAttribute("value", "Пока");`'

- '`input.dataset.value = "Пока";`'

- '`input["Пока"] = true;`'

- 'Все вышеперечисленные'

type: 'multi'

-

content: |

Какие варианты подключения скрипта соответствуют современному стандарту HTML?

correctAnswer:

- 0

- 1

answers:

- '`<script type="text/javascript" src="my.js"></script>`'

- '`<script src="my.js"></script>`'

- '`<script src="my.js"/>`'

- '`<хачу-javascript отсюда="my.js">`'

type: 'multi'

-

content: |

Есть кнопка `<button id="elem"></button>`, как показать в ней строку `"<hello>"`?

Результат должен быть таким:

```html

<!--+ run -->

<button id="elem">&lt;hello&gt;</button>

```

correctAnswer: 2

answers:

- '`elem.innerHTML = "<hello>"`'

- '`elem.innerText = "<hello>"`'

- '`elem.textContent = "<hello>"`'

- '`elem.setText("<hello>")`'

type: 'single'

-

content: |

Какие из этих обработчиков сработают?

```js

div.onclick = function() { alert(1) };

div.onclick = function() { alert(2) };

div.addEventListener('click', function() { alert(3) });

```

correctAnswer:

- 1

- 2

answers:

- 'Первый.'

- 'Второй.'

- 'Третий.'

type: 'multi'

-

content: |

Верно ли следующее утверждение:

Содержимое тега `script` нужно заключать в комментарии `<!-- .. -->`,

чтобы браузеры, которые не поддерживают JavaScript, работали корректно?

correctAnswer: 1

answers:

- 'Да.'

- 'Нет.'

type: 'single'

-

content: |

Можно ли скриптом перевести посетителя на другую страницу сайта?

correctAnswer: 0

answers:

- 'Да, куда угодно.'

- 'Да, но только в рамках текущего сайта.'

- 'Нет, нельзя.'

type: 'single'

-

content: |

Вызов `setTimeout(func, 0)` вызовет `func`...

correctAnswer: 3

answers:

- 'Сразу после `setTimeout`, до следующей строки кода.'

- 'Сразу после текущего скрипта, когда браузер сможет выполнить JavaScript.'

- 'Ровно через 4 миллисекунды.'

- 'Через 4 миллисекунды или больше, в зависимости от пожеланий браузера.'

type: 'single'

-

content: |

Вызов `setInterval(func, 100)` вызовет `func`...

correctAnswer: 2

answers:

- 'Ровно каждые 100 мс.'

- 'Примерно каждые 100 мс.'

- 'Обычно каждые 100 мс, но возможно, что вызовы будут происходить гораздо реже.'

type: 'single'

-

content: |

Браузер вызывает `setInterval` и `setTimeout` гораздо реже, чем обычно, если...

correctAnswer: 2

answers:

- 'Вкладка, в которой работает JavaScript, не видна и находится в фоновом режиме.'

- 'Посетитель зашёл с ноутбука, с питанием от батареи.'

- 'В обоих этих случаях.'

type: 'single'

-

content: |

Как правильно вывести приветствие через 5 секунд?

correctAnswer: 2

answers:

- '`sleep(5); alert("Привет!");`'

- '`sleep(5000); alert("Привет!");`'

- '`setTimeout(function() {alert("Привет!")}, 5000);`'

- '`setTimeout(function() {alert("Привет!")}, 5);`'

type: 'single'

-

content: |

Какое событие из этого списка не существует?

correctAnswer: 0

answers:

- '`onmousescroll`'

- '`onclick`'

- '`onmouseover`'

- '`onmousemove`'

- '`onwheel`'

- 'Все существуют.'

type: 'single'

-

content: |

Может ли скрипт во время работы страницы подключить к ней другие внешние JS-файлы?

correctAnswer: 2

answers:

- 'Да, но только один раз.'

- 'Да, но только до полной загрузки страницы.'

- 'Да, сколько угодно файлов когда угодно.'

type: 'single'

-

content: |

Какое событие не может быть вызвано кликом мыши?

correctAnswer:

- 2

answers:

- '`onfocus`'

- '`onclick`'

- '`onkeydown`'

- '`onmousedown`'

type: 'multi'

-

content: |

Можно ли инициировать DOM-событие из JavaScript?

Например, сэмулировать клик мышкой на элементе, чтобы JavaScript-код кликнул за пользователя, и сработали соответствующие обработчики.

correctAnswer: 0

answers:

- 'Да, можно.'

- 'Нет, нельзя.'

type: 'single'

-

content: |

Где в документе может располагаться тег `script` по стандарту HTML?

correctAnswer: 2

answers:

- 'Только в `<head>`.'

- 'Только в `<body>`.'

- 'В `<head>` или в `<body>`.'

- 'Где угодно, главное, чтобы был.'

type: 'single'

-

content: |

Чему равно значение свойства `checked` в коде ниже?

```js

var input = document.createElement('input');

input.setAttribute('checked', 'checked');

// input.checked = ?

```

correctAnswer: 2

answers:

- '`"checked"`'

- '`"true"`'

- '`true`'

type: 'single'

-

content: |

Посмотрим на HTML-документ:

```html

<table>

<tr>

<td id="hi">Привет!</td>

</tr>

</table>

```

Каков будет результат `document.getElementById('hi').parentNode.parentNode.tagName`?

correctAnswer: 3

answers:

- '`TR`'

- '`TD`'

- '`TABLE`'

- 'Другой.'

type: 'single'

-

content: |

Чем отличаются стандарты DOM Level 2 и DOM Level 3?

type: 'single'

answers:

- 'DOM Level 2 -- это свойства второго уровня, а DOM Level 3 -- третьего.'

- 'DOM Level 3 появился позже чем DOM Level 2, он доопределяет DOM-свойства и события.'

- 'DOM Level 3 работает в 3 раза быстрее, чем DOM Level 1, а DOM Level 2 -- только в 2 раза.'

- 'DOM Level 3 описывает новые свойства, которых нет в DOM Level 2.'

- 'DOM Level 3 описывает только события, а DOM Level 2 -- только свойства элементов.'

- 'Хорош выдумывать, какой ещё DOM Level? Есть только "просто DOM".'

correctAnswer: 1

-

content: |

Как получить HTML-содержимое DOM-элемента `elem`?

correctAnswer: 1

answers:

- '`elem.html`'

- '`elem.innerHTML`'

- '`elem.content`'

type: 'single'

-

content: |

Можно ли из JavaScript получить содержимое комментария?

correctAnswer: 0

answers:

- 'Да, комментарий -- DOM-узел, который можно получить.'

- 'Нет, комментарии есть в HTML, но отсутствуют в DOM.'

- 'Можно получить, но только до окончания загрузки страницы.'

type: 'single'

-

content: |

Можно ли из JavaScript получить содержимое комментария?

correctAnswer: 0

answers:

- 'Да, комментарий -- DOM-узел, можно получить всегда.'

- 'Нет, комментарии есть в HTML, но отсутствуют в DOM.'

- 'Можно получить, но только до окончания загрузки страницы.'

type: 'single'

-

content: |

Что выведет скрипт на этой странице?

```html

<p id="elem">Привет, мир!</p>

<script>

elem.outerHTML = "Прощай, жестокий мир!"

alert(elem.outerHTML);

</script>

```

correctAnswer: 4

correctAnswerComment: |

На странице теперь нет элемента с id="elem", он заменён на текст.

answers:

- 'Выведет `undefined`, теперь `elem` -- текстовый узел.'

- 'Прощай, жестокий мир!'

- '`<p id="elem">Привет, мир!</p>`'

- '`<p id="elem">Прощай, жестокий мир!</p>`'

- 'Будет ошибка.'

type: 'single'

-

content: |

Что выведет этот код?

```html

<p id="elem">Привет!</p>

<script>

elem.textContent = elem.outerHTML

alert(elem.innerHTML);

</script>

```

correctAnswer: 2

answers:

- 'Привет!'

- '`<p id="elem">Привет!</p>`'

- '`&lt;p id="elem"&gt;Привет!&lt;/p&gt;`'

- 'Будет ошибка.'

type: 'single'

-

content: |

Как получить атрибут `test` у элемента `<div id="elem" test="5">`?

correctAnswer:

- 1

- 2

- 3

answers:

- '`elem.test`'

- '`elem.getAttribute("test")`'

- '`elem.attributes.test.value`'

- '`elem.attributes[1].value`'

- '`elem.dataset.test`'

type: 'multi'

-

content: |

Сколько потомков будет у `<div>` после кода ниже?

```html

<div id="div"></div>

<p id="p">Привет</p>

<script>

div.appendChild(p);

p.appendChild(div);

</script>

```

correctAnswer: 3

answers:

- '`0`'

- '`1`'

- '`2`'

- 'В коде ошибка.'

type: 'single'

-

content: |

Сколько потомков будет у `<div>` после кода ниже?

```html

<div id="div"></div>

<p id="p">Привет</p>

<script>

div.appendChild(p);

div.appendChild(p);

</script>

```

correctAnswer: 1

answers:

- '`0`'

- '`1`'

- '`2`'

- 'В коде ошибка.'

type: 'single'

-

content: |

Сколько детей будет у `<div>` после кода ниже?

```html

<div id="div">текст</div>

<script>

div.insertAdjacentHTML('beforeBegin', '<p>Привет</p>');

div.insertAdjacentHTML('afterEnd', '<p>Пока</p>');

</script>

```

correctAnswer: 1

answers:

- '`0`'

- '`1`'

- '`2`'

- '`3`'

- 'В коде ошибка.'

type: 'single'

-

content: |

Сколько детей будет у `<div>` после кода ниже?

```html

<div id="div">текст</div>

<script>

div.insertAdjacentHTML('afterBegin', '<p>Привет</p>');

div.insertAdjacentHTML('beforeEnd', '<p>Пока</p>');

</script>

```

correctAnswer: 3

answers:

- '`0`'

- '`1`'

- '`2`'

- '`3`'

- 'В коде ошибка.'

type: 'single'

-

content: |

Что будет, если вызвать `document.write(str)` после загрузки страницы?

correctAnswer: 1

answers:

- 'Строка `str` допишется в конец документа.'

- 'Содержимое документа будет полностью заменено на строку `str`.'

- 'Будет ошибка.'

type: 'single'

-

content: |

Как получить размер отступа `margin-top` у `<div>`?

```html

<style>

div { margin: 10px; }

</style>

<div id="div"></div>

```

correctAnswer: 3

answers:

- '`div.style.marginTop`'

- '`div.style["margin-top"]`'

- '`getComputedStyle(div, "margin").top`'

- '`getComputedStyle(div).marginTop`'

- '`getComputedStyle(div).margin.top`'

type: 'single'

-

content: |

Как получить ширину \*произвольного\* DOM-элемента, если у него нет `border` и `padding`?

correctAnswer:

- 2

correctAnswerComment: 'Например, для обычного `<span>` первые два метода вернут `NaN`, так как у строчного элемента нет CSS-ширины. А `clientWidth` будет равен нулю.

'

answers:

- '`parseInt(elem.style.width)`'

- '`parseInt(getComputedStyle(elem).width)`'

- '`elem.offsetWidth`'

- '`elem.clientWidth`'

type: 'multi'

-

content: |

Как получить полную ширину \*блочного\* DOM-элемента, если у него есть `border` и `padding`?

correctAnswer:

- 1

- 2

answers:

- '`parseInt(elem.style.width)`'

- '`parseInt(getComputedStyle(elem).width)`'

- '`elem.offsetWidth`'

- '`elem.clientWidth`'

type: 'multi'

-

content: |

При каком условии `elem.scrollHeight == elem.clientHeight`?

correctAnswer: 3

answers:

- 'Содержимое элемента полностью прокручено вниз.'

- 'Страница прокручена так, что элемент полностью видим и находится в границах окна.'

- 'Высота элемента равна высоте полосы прокрутки.'

- 'Содержимое элемента полностью видимо, в нём нет прокрутки.'

- 'У элемента нет `padding`.'

type: 'single'

-

content: |

Какое значение содержит "внешнюю" высоту элемента, то есть вертикальный размер, который он занимает во внешнем контейнере?

correctAnswer: 0

answers:

- '`offsetHeight`'

- '`outerHeight`'

- '`clientHeight`'

- '`scrollHeight`'

- 'Ни одно из вышеперечисленных.'

type: 'single'

-

content: |

Когда у элемента гарантированно совпадают `offsetHeight` и `clientHeight`?

correctAnswer: 1

answers:

- 'Когда у элемента нет `border`.'

- 'Когда у элемента нет `border` и `padding`.'

- 'Когда у элемента нет `border`, `padding` и `margin`.'

- 'Когда у элемента нет прокрутки.'

type: 'single'

-

content: |

Результат `elem.getBoundingClientRect()` содержит отрицательное значение `top: -10`.

Что это означает?

correctAnswer: 1

answers:

- 'Что верх элемента вылезает за верхнюю границу документа.'

- 'Что верх элемента вылезает за верхнюю границу окна.'

- 'Что содержимое элемента прокручено на `10px`.'

- 'Ничего не означает, браузер чудит.'

type: 'single'

-

content: |

Результат `elem.getBoundingClientRect()` содержит отрицательное значение `bottom: -10`.

Что это означает?

correctAnswer: 3

answers:

- 'Что низ элемента вылезает за нижнюю границу документа.'

- 'Что низ элемента вылезает за нижнюю границу окна.'

- 'Что низ элемента вылезает за верхнюю границу документа.'

- 'Что низ элемента вылезает за верхнюю границу окна.'

- 'Что содержимое элемента прокручено на `10px`.'

- 'Такого не может быть.'

type: 'single'

-

content: |

Какие варианты корректно перебирают все формы на странице?

correctAnswer:

- 1

- 3

answers:

-

title: 'Вариант 1.'

description: |

```js

var elems = document.querySelectorAll("form");

for (var i in elems) console.log(elems[i]);

```

-

description: |

```js

var elems = document.querySelectorAll("form");

for (var i = 0; i < elems.length; i++) console.log(elems[i]);

```

title: 'Вариант 2.'

-

description: |

```js

var elems = document.getElementsByTagName("form");

for (var i in elems) console.log(elems[i]);

```

title: 'Вариант 3.'

-

description: |

```js

var elems = document.forms;

for (var i = 0; i < elems.length; i++) console.log(elems[i]);

```

title: 'Вариант 4.'

type: 'multi'

-

content: |

Есть два вложенных элемента:

```html

<a id="outer" href="http://site.com">

<em id="inner">Текст ссылки</em>

</a>

```

На внутреннем стоит обработчик:

```js

inner.onclick = function(e) { e.preventDefault(); }

```

Произойдёт ли при клике на `#inner` переход по ссылке?

correctAnswer: 1

answers:

- 'Да.'

- 'Нет.'

- 'Будет ошибка.'

type: 'single'

-

content: |

Есть ссылка:

```html

<a href="http://site.com">Текст ссылки</a>

```

На документе стоит обработчик:

```js

document.onclick = function(e) { e.preventDefault(); }

```

Произойдёт ли при клике на ссылку переход по ней?

correctAnswer: 1

answers:

- 'Да.'

- 'Нет.'

- 'Зависит от положения звёзд в момент клика.'

type: 'single'

-

content: |

Есть элемент <input type="button" id="elem"/>. На нём два обработчика:

```js

elem.addEventListener("click", handler1, false);

elem.addEventListener("click", handler2, true);

```

Какой из них сработает раньше при клике на `elem`?

correctAnswer: 0

correctAnswerComment: |

Оба обработчика сработают не на стадии всплытия и погружения, а на самом элементе, то есть на стадии AT\_TARGET.

Это полноценная отдельная стадия обработки события, не всплытие или погружение.

Обработчики на ней срабатывают в порядке назначения.

answers:

- 'Первый.'

- 'Второй.'

- 'Стандарт не указывает точную последовательность.'

type: 'single'

-

content: |

Есть элемент <div id="elem"><input/></div>. На нём два обработчика:

```js

elem.addEventListener("click", handler1, false);

elem.addEventListener("click", handler2, true);

```

Какой из них сработает раньше при клике на `<input>`?

correctAnswer: 1

correctAnswerComment: |

Сначала происходит стадия погружения, на которой срабатывают обработчики от корня до `elem`, в данном случае `handler2`.

Затем -- стадия "на элементе" и стадия всплытия, на которой срабатывают обработчики выше `elem`, в данном случае `handler1`.

answers:

- 'Первый.'

- 'Второй.'

- 'Ни один.'

- 'Стандарт не указывает точную последовательность.'

type: 'single'

-

content: |

Есть обработчик события на элементе:

```js

elem.onclick = function(event) {

// ...

}

```

Какие значения внутри этого обработчика будут всегда одинаковы?

correctAnswer: 2

answers:

- '`event.target` и `event.currentTarget`.'

- '`event.target` и `this`.'

- '`event.currentTarget` и `this`.'

- 'Ни одна пара из указанных выше.'

type: 'single'

-

content: |

Если функция-обработчик события `onclick` выполняется долго,

и посетитель во время её работы нажал на кнопку клавиатуры -- что будет с возникшим событием `keydown`?

correctAnswer: 2

answers:

- 'Будет проигнорировано, так как браузер занят.'

- 'Обработчик `onclick` приостановится, браузер обработает `keydown`, а потом возобновится обработка `onclick`.'

- 'Событие `keydown` будет обработано после `onclick`.'

- 'Если событие на том же элементе, что и `onclick`, то будет проигнорировано, иначе -- обработано после `onclick`.'

type: 'single'

-

content: |

Какие из этих свойств элемента могут указывать на текстовый узел?

correctAnswer:

- 1

- 4

answers:

- '`firstElementChild`'

- '`firstChild`'

- '`parentNode`'

- '`parentElement`'

- '`nextSibling`'

type: 'multi'

-

content: |

На ссылке `<a href="..." id="link">` стоит обработчик события:

```js

link.addEventListener('click', function(event) {

// ...

});

```

Какой код нужно записать в обработчик события, чтобы переход по ссылке не состоялся?

Если подходят несколько вариантов из перечисленных -- выберите несколько.

correctAnswer:

- 2

answers:

- '`event.stopPropagation()`'

- '`event.stopImmediatePropagation()`'

- '`event.preventDefault()`'

- '`return false`'

- '`event.cancel()`'

type: 'multi'

-

content: |

Этот код ничего не выведет. Почему?

```js

document.onclick = function(event) {

alert(event.type);

};

document.body.dispatchEvent(new CustomEvent("click"));

```

correctAnswer: 3

answers:

- 'Из JavaScript нельзя генерировать встроенные события.'

- 'Обработчик через `on-свойство` не срабатывает для событий, сгенерированных из JavaScript.'

- 'Встроенное событие нужно генерировать иначе, конструктор `CustomEvent` надо заменить на другой.'

- 'При генерации события не указано, что оно должно всплывать.'

- 'Боже, что это?'

type: 'single'

-

content: |

Какие события из этого списка возникнут при нажатии и отпускании клавиши "Shift"? Другие клавиши не нажимаются.

correctAnswer:

- 0

- 1

answers:

- '`keydown`'

- '`keyup`'

- '`keypress`'

- 'Никаких не возникнет'

type: 'multi'

-

content: |

Какие события из этого списка возникнут при нажатии и отпускании клавиши "пробел"? Другие клавиши не нажимаются.

correctAnswer:

- 0

- 1

- 2

answers:

- '`keydown`'

- '`keyup`'

- '`keypress`'

- 'Никаких не возникнет'

type: 'multi'