|  |  |  |
| --- | --- | --- |
| № | Код | Объяснение |
| 1 | Math.abs('-1'); // 1  Math.abs(-2); // 2  Math.abs(null); // 0  Math.abs('string'); // NaN  Math.abs(); // NaN | Метод вернёт [NaN](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/NaN" \o "Глобальное свойство NaN является значением, представляющим не-число (Not-A-Number).), если в него передать нечисловую строку или значение [undefined](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/undefined" \o "Значение глобального свойства undefined представляет значение undefined. Это одно из примитивных значений JavaScript.)/ничего не передавать. Метод вернёт 0, если в него передать значение [null](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/null" \o "Документация об этом ещё не написана; пожалуйста, поспособствуйте её написанию!). |
| 2 | Math.acos(-2); // NaN  Math.acos(-1); // 3.141592653589793  Math.acos(0); // 1.5707963267948966  Math.acos(0.5); // 1.0471975511965979  Math.acos(1); // 0  Math.acos(2); // NaN | Метод Math.acos() возвращает числовое значение между 0 и π радианами для значения x, лежащего в диапазоне от -1 до 1. Метод вернёт [NaN](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/NaN" \o "Глобальное свойство NaN является значением, представляющим не-число (Not-A-Number).), если значение x выйдет за этот диапазон.  Поскольку метод acos() является статическим методом объекта Math, вы всегда должны использовать его как Math.acos(), а не пытаться вызывать метод на созданном экземпляре объекта Math (поскольку объект Math не является конструктором). |
| 3 | Math.acosh(-1); // NaN  Math.acosh(0); // NaN  Math.acosh(0.5) // NaN  Math.acosh(1); // 0  Math.acosh(2); // 1.3169578969248166 | Метод Math.acosh() возвращает гиперболический арккосинус числа, то есть  ∀x≥1,Math.acosh(x)=arcosh(x)= уникальный y≥0такой, что cosh(y)=x  Для значений, меньших 1, метод Math.acosh()  возвращает [NaN](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/NaN" \o "Глобальное свойство NaN является значением, представляющим не-число (Not-A-Number).). |
| 4 | Math.asin(-2); // NaN  Math.asin(-1); // -1.5707963267948966 (-pi/2)  Math.asin(0); // 0  Math.asin(0.5); // 0.5235987755982989  Math.asin(1); // 1.570796326794897 (pi/2)  Math.asin(2); // NaN | Метод Math.asin() возвращает числовое значение между -π2 и π2 радианами для значения x, лежащего в диапазоне от -1 до 1. Метод вернёт [NaN](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/NaN" \o "Глобальное свойство NaN является значением, представляющим не-число (Not-A-Number).), если значение x выйдет за этот диапазон.  Для значений, меньших -1 или больших 1, метод Math.asin() возвращает [NaN](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/NaN" \o "Глобальное свойство NaN является значением, представляющим не-число (Not-A-Number).). |
| 5 | Math.asinh(1); // 0.881373587019543  Math.asinh(0); // 0 | Метод Math.asinh() возвращает гиперболический арксинус числа, то есть  Math.asinh(x)=arsinh(x)= уникальный y такой, что sinh(y)=x |
| 6 | Math.atan(1); // 0.7853981633974483  Math.atan(0); // 0 | Метод Math.atan() возвращает числовое значение между – π/2 и π/2 радианами. |
| 7 | Math.atanh(-2); // NaN  Math.atanh(-1); // -Infinity  Math.atanh(0); // 0  Math.atanh(0.5); // 0.5493061443340548  Math.atanh(1); // Infinity  Math.atanh(2); // NaN | Метод Math.atanh() возвращает гиперболический арктангенс числа, то есть  ∀x∊(-1,1),Math.atanh(x)=arctanh(x)= уникальный y такой, что tanh(y)=x |
| 8 | Math.atan2(90, 15); // 1.4056476493802699  Math.atan2(15, 90); // 0.16514867741462683  Math.atan2(±0, -0); // ±PI.  Math.atan2(±0, +0); // ±0.  Math.atan2(±0, -x); // ±PI для x > 0.  Math.atan2(±0, x); // ±0 дляx > 0.  Math.atan2(-y, ±0); // -PI/2 для y > 0.  Math.atan2(y, ±0); // PI/2 для y > 0.  Math.atan2(±y, -Infinity); // ±PI для конечного y > 0.  Math.atan2(±y, +Infinity); // ±0 для конечного y > 0.  Math.atan2(±Infinity, x); // ±PI/2 для конечного x.  Math.atan2(±Infinity, -Infinity); // ±3\*PI/4.  Math.atan2(±Infinity, +Infinity); // ±PI/4. | Метод Math.atan2() возвращает числовое значение от -π до π, представляющее угол тета точки (x, y). Это выраженный в радианах угол, отсчитываемый против часовой стрелки от положительного направления оси X до точки (x, y). Обратите внимание, что первым в метод передаётся координата y, а только вторым — координата x.  В метод Math.atan2() аргументы x и y передаются по отдельности, в то время, как в метод Math.atan() передаётся отношение этих двух аргументов. |
| 9 | Math.cbrt(-1); // -1  Math.cbrt(0); // 0  Math.cbrt(1); // 1  Math.cbrt(2); // 1.2599210498948734 | Метод Math.cbrt() возвращает кубический корень числа, то есть Math.cbrt(x)=x3=уникальный y такой, что y3=x |
| 10 | Math.ceil(.95); // 1  Math.ceil(4); // 4  Math.ceil(7.004); // 8  Math.ceil(-0.95); // -0  Math.ceil(-4); // -4  Math.ceil(-7.004); // -7 | Метод Math.ceil() - округление в верх. Округляет аргумент до ближайшего большего целого. |
| 11 | Math.clz32(1); // 31  Math.clz32(1000); // 22  Math.clz32(); // 32  [NaN, Infinity, -Infinity, 0, -0, null, undefined, 'foo', {}, []].filter(  function(n) {  return Math.clz32(n) !== 32  }); // []  Math.clz32(true); // 31  Math.clz32(3.5); // 30 | Метод Math.clz32() возвращает количество ведущих нулевых битов в 32-битном двоичном представлении числа.  Название clz32 является сокращением для более длинного CountLeadingZeroes32 (ПосчтитатьВедущиеНули32).  Если значение x не является числом, оно будет сначала преобразовано в число, а потом в 32-битное беззнаковое целое число.  Если преобразованое 32-битное беззнаковое целое число равно 0, метод вернёт 32, поскольку все биты в числе равны 0.  Этот метод особено полезен для систем, которые компилируются в JS, например, для Emscripten. |
| 12 | Math.cos(0); // 1  Math.cos(1); // 0.5403023058681398  Math.cos(Math.PI); // -1  Math.cos(2 \* Math.PI); // 1 | Метод Math.cos() возвращает числовое значение от -1 до 1, которое представляет косинус угла. |
| 13 | Math.cosh(0); // 1  Math.cosh(1); // 1.5430806348152437  Math.cosh(-1); // 1.5430806348152437 | Метод Math.cosh() возвращает гиперболический косинус числа, который может быть выражен через число e:  Math.cosh(x)= ex+e-x2 |
| 14 | Math.exp(-1); // 0.36787944117144233  Math.exp(0); // 1  Math.exp(1); // 2.718281828459045 | Метод Math.exp() возвращает значение выражения ex, где x — аргумент метода, а e — число Эйлера, основание натурального логарифма. |
| 15 | Math.expm1(-1); // -0.6321205588285577  Math.expm1(0); // 0  Math.expm1(1); // 1.718281828459045 | Метод Math.expm1() возвращает значение выражения ex - 1, где x — аргумент метода, а e — основание натурального логарифма. |
| 16 | Math.floor( 45.95); // 45  Math.floor(-45.95); // -46 | Метод Math.floor() возвращает наибольшее целое число, которое меньше или равно данному числу. |
| 17 | Math.fround(0); // 0  Math.fround(1); // 1  Math.fround(1.337); // 1.3370000123977661  Math.fround(1.5); // 1.5  Math.fround(NaN); // NaN | Метод Math.fround() возвращает ближайшее число с плавающей запятой одинарной точности, представляющее указанное число. |
| 18 | Math.hypot(3, 4); // 5  Math.hypot(3, 4, 5); // 7.0710678118654755  Math.hypot(); // 0  Math.hypot(NaN); // NaN  Math.hypot(3, 4, 'foo'); // NaN, +'foo' => NaN  Math.hypot(3, 4, '5'); // 7.0710678118654755, +'5' => 5  Math.hypot(-3); // 3, то же самое, что и Math.abs(-3) | Метод Math.hypot() возвращает квадратный корень суммы квадратов своих аргументов, то есть  Math.hypot(v1,v2,…,vn)=  ∑i=1nvi2=v12+v22+…+vn2  \mathtt{\operatorname {Math.hypot}(v\_1, v\_2, \dots, v\_n)} = \sqrt{\sum\_{i=1}^n v\_i^2} = \sqrt{v\_1^2 + v\_2^2 + \dots + v\_n^2} |
| 19 | Math.imul(2, 4); // 8  Math.imul(-1, 8); // -8  Math.imul(-2, -2); // 4  Math.imul(0xffffffff, 5); // -5  Math.imul(0xfffffffe, 5); // -10 | Метод Math.imul() возвращает результат перемножения в стиле C двух 32-битных чисел.  Метод Math.imul() позволяет выполнять быстрое перемножение двух 32-битных чисел с семантикой C. Эта возможность полезна для таких проектов, как Emscripten. |
| 20 | Math.log(-1); // NaN, выход за пределы диапазона  Math.log(0); // -Infinity  Math.log(1); // 0  Math.log(10); // 2.302585092994046 | Метод Math.log() возвращает натуральный (по основанию e) логарифм числа, то есть  ∀x>0,Math.log(x)=ln(x)=уникальныйyтакой, чтоey=x  Если значение x меньше 0, возвращаемое значение всегда будет равным [NaN](https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/NaN" \o "Глобальное свойство NaN является значением, представляющим не-число (Not-A-Number).). |
| 21 | Math.log1p(1); // 0.6931471805599453  Math.log1p(0); // 0  Math.log1p(-1); // -Infinity  Math.log1p(-2); // NaN | Метод Math.log1p() возвращает натуральный (по основанию e) логарифм 1 + числа, то есть  ∀x>-1,Math.log1p(x)=ln(1+x)  Если значение x меньше -1, возвращаемое значение всегда будет равным NaN. |
| 22 | Math.log10(2); // 0.3010299956639812  Math.log10(1); // 0  Math.log10(0); // -Infinity  Math.log10(-2); // NaN  Math.log10(100000); // 5 | Метод Math.log10() возвращает десятичный (по основанию 10) логарифм числа, то есть  ∀x>0,Math.log10(x)=log10(x)=уникальныйyтакой, что10y=x  Если значение x меньше 0, возвращаемое значение всегда будет равным NaN. |
| 23 | Math.log2(3); // 1.584962500721156  Math.log2(2); // 1  Math.log2(1); // 0  Math.log2(0); // -Infinity  Math.log2(-2); // NaN  Math.log2(1024); // 10 | Метод Math.log2() возвращает двоичный (по основанию 2) логарифм числа, то есть  ∀x>0,Math.log2(x)=log2(x)=уникальныйyтакой, что2y=x  Если значение x меньше 0, возвращаемое значение всегда будет равным NaN. |
| 24 | Math.max(10, 20); // 20  Math.max(-10, -20); // -10  Math.max(-10, 20); // 20  function getMaxOfArray(numArray) {  return Math.max.apply(null, numArray);  } | Метод Math.max() возвращает наибольшее из нуля или более чисел.  При вызове без аргументов результатом вызова будет значение -Infinity.  Если хотя бы один из аргументов не может быть преобразован в число, результатом будет NaN.  Следующая функция использует метод Function.prototype.apply() для нахождения максимального элемента в числовом массиве. Вызов getMaxOfArray([1, 2, 3]) эквивалентен вызову Math.max(1, 2, 3), однако вы можете использовать функцию getMaxOfArray() вместе с программно сконструированными массивами любого размера. |
| 25 | var x = 10, y = -20;  var z = Math.min(x, y);  var x = f(foo);  if (x > boundary) {  x = boundary;  }  var x = Math.min(f(foo), boundary); | Метод Math.min() возвращает наименьшее из нуля или более чисел.  В этом примере ищется минимум из x и y и присваивается переменной z:  Метод Math.min() часто используется для обрезания значения таким образом, чтобы оно всегда не превосходило некоторую границу. Например, такое условие  может быть переписано в виде  Подобным образом может использоваться и метод Math.max() для обрезания значения по нижней границе. |
| 26 | Math.pow(7, 2); // 49 | Метод Math.pow() возвращает основание, возведённое в степень показатель, то есть, значение выражения основание показатель.  Math.pow(base, exponent)  Base - Основание степени.  Exponent - Показатель степени, в которую возводится основание base. |
| 27 | // Возвращает случайное число между 0 (включительно) и 1 (не включая 1)  function getRandom() {  return Math.random();  }  // Возвращает случайное число между min (включительно) и max (не включая max)  function getRandomArbitrary(min, max) {  return Math.random() \* (max - min) + min;  }  // Возвращает случайное целое число между min (включительно) и max (не включая max)  // Использование метода Math.round() даст вам неравномерное распределение!  function getRandomInt(min, max) {  return Math.floor(Math.random() \* (max - min)) + min;  } | Метод Math.random() возвращает псевдослучайное число с плавающей запятой из диапазона [0, 1), то есть, от 0 (включительно) до 1 (но не включая 1), которое затем можно отмасштабировать до нужного диапазона. Реализация сама выбирает начальное зерно для алгоритма генерации случайных чисел; оно не может быть выбрано или сброшено пользователем.  Примечание: метод Math.random() не предоставляет криптографически стойкие случайные числа. Не используйте его ни для чего, связанного с безопасностью. Вместо него используйте Web Crypto API (API криптографии в вебе) и более точный метод window.crypto.getRandomValues(). |
| 28 | // Вернёт значение 20  x = Math.round(20.49);  // Вернёт значение 21  x = Math.round(20.5);  // Вернёт значение -20  x = Math.round(-20.5);  // Вернёт значение -21  x = Math.round(-20.51);  // Вернёт значение 1 (!)  // Обратите внимание на ошибку округления из-за неточности арифметики с плавающей запятой  // Сравните этот результат с результатом Math.round10(1.005, -2) из следующего примера  x = Math.round(1.005\*100)/100; | Метод Math.round() возвращает число, округлённое к ближайшему целому.  Если дробная часть числа больше, либо равна 0,5, аргумент будет округлён до ближайшего большего целого. Если дробная часть числа меньше 0,5, аргумент будет округлён до ближайшего меньшего целого. |
| 29 | Math.sign(3); // 1  Math.sign(-3); // -1  Math.sign('-3'); // -1  Math.sign(0); // 0  Math.sign(-0); // -0  Math.sign(NaN); // NaN  Math.sign('foo'); // NaN  Math.sign(); // NaN | Метод Math.sign() возвращает знак числа, указывающий на то, является ли число отрицательным, положительным или нулём.  Метод может вернуть одно из пяти значений: 1, -1, 0, -0 и NaN, которые представляют, соответственно «положительное число», «отрицательное число», «положительный ноль», «отрицательный ноль» и NaN.  Обратите внимание, что аргумент, переданный в этот метод, будет неявно преобразован в число. |
| 30 | Math.sin(0); // 0  Math.sin(1); // 0.8414709848078965  Math.sin(Math.PI / 2); // 1 | Метод Math.sin() возвращает синус числа.  Метод Math.sin() возвращает числовое значение от -1 до 1, которое представляет синус переданного (в радианах) угла. |
| 31 | Math.sinh(0); // 0  Math.sinh(1); // 1.1752011936438014 | Метод Math.sinh() возвращает гиперболический синус числа, который может быть выражен через число e: Math.sinh(x)= ex-e-x2 |
| 32 | Math.sqrt(9); // 3  Math.sqrt(2); // 1.414213562373095  Math.sqrt(1); // 1  Math.sqrt(0); // 0  Math.sqrt(-1); // NaN  Math.sqrt(-0); // -0 | Метод Math.sqrt() возвращает квадратный корень числа, то есть ∀x≥0,Math.sqrt(x)=  x=уникальный y≥0 такой, что xy2=x  Квадратный корень заданного числа. Если число отрицательное, то вернется NaN. |
| 33 | Math.tan(1); // 1.5574077246549023  function getTanDeg(deg) {  var rad = deg \* Math.PI/180;  return Math.tan(rad);  } | Метод Math.tan() возвращает тангенс числа.  Метод Math.tan() возвращает числовое значение, представляющее тангенс угла.  Поскольку метод Math.tan() принимает радианы, а чаще проще работать с градусами, следующая функция принимает значение в градусах, преобразует их в радианы и возвращает тангенс. |
| 34 | Math.tanh(0); // 0  Math.tanh(Infinity); // 1  Math.tanh(1); // 0.7615941559557649 | Метод Math.tanh() возвращает гиперболический тангенс числа, то есть  tanhx= sinhxcoshx = ex-e-xex+e-x = e2x-1e2x+1 |
| 35 | Math.trunc(13.37); // 13  Math.trunc(42.84); // 42  Math.trunc(0.123); // 0  Math.trunc(-0.123); // -0  Math.trunc('-1.123'); // -1  Math.trunc(NaN); // NaN  Math.trunc('foo'); // NaN  Math.trunc(); // NaN | Метод Math.trunc() возвращает целую часть числа путём удаления всех дробных знаков.  В отличие от других трёх методов объекта Math — Math.floor(), Math.ceil() и Math.round() — метод Math.trunc() работает очень просто и прямолинейно: отбрасывает десятичную запятую и все цифры после неё, не обращая внимания на знак аргумента.  Таким образом, если аргумент является положительным числом, метод Math.trunc() работает аналогично методу Math.floor(), а если отрицательным — то аналогично методу Math.ceil().  Обратите внимание, что аргумент, переданный в этот метод, будет неявно преобразован в число. |
| 36 | **Math.toSource()** | Возвращает строку "Math" |

https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global\_Objects/Math

http://javascript.ru/math