ОПЕРАТОРЫ

Арифметические операторы

+	сложение
-	вычитание
*	умножение
/	деление
%	взятие остатка от деления
**	возведение в степень

Оператор сложения + используется для:

1. сложения чисел
2. конкатенации строк
3. для преобразования строки с числом в число

Остальные операторы преобразуют операнды к числу (Nan, если преобразование не является возможным)

Операторы присваивания

=	let a = 12	a присвоили значение 12
+=	a += 12	краткая форма от a = a + 12
-=	a -= 12	краткая форма от a = a - 12
*=	a *= 12	краткая форма от a = a * 12
/=	a /= 12	краткая форма от a = a / 12
%=	a %= 12	краткая форма от a = a % 12

Инкремент и декремент

i++	инкремент (постфиксная форма) сначала возвращает значение, потом увеличивает
++i	инкремент (префиксная форма) сначала увеличивает, а потом возвращает значение
i--	декремент (постфиксная форма) сначала возвращает значение, потом уменьшает
--i	декремент (префиксная форма) сначала уменьшает, а потом возвращает значение

В операциях сравнения сравниваются два операнда, и возвращается значение типа boolean: true, если выражение верно, false, если выражение неверно.

Операторы сравнения

>	больше
<	меньше
==	равно
>=	больше или равно
<=	меньше или равно
!=	не равно
===	строгое равентсво
!==	строгое неравентсво

Операторы сравнения

При сравнении значений разных типов происходит приведение к числу, за исключением строгого равенства === и строгого неравенства !==

При сравнении с использованием строгого равенства приведение типов не происходит, значения сравниваются на полное совпадение

```
1 console.log('3' == 3);     // true
2 console.log('3' === 3);    // false
3 console.log('3' === '3'); // true
4 console.log(3 === 3);      // true
```

Возвращают значение типа boolean

Логические операторы

&&	a && b	вернет true, если a и b - true (если a - false, b не вычисляется)
||	a || b	вернет true, если a или b - true (если a - true, b не вычисляется)
!	!a	меняет значение a на противоположное

Тернарный оператор используется для оценки выражений типа boolean

Цель тернарного оператора заключается в том, чтобы решить, какое значение должно быть присвоено переменной.

```
переменная x = выражение ? выражение1 : выражение2;
```

Если выражение равно true, то вычисляется выражение1 и его результат становится результатом выполнения всего оператора.
В противном случае вычисляется выражение2.
Выражение1 и выражение2 должны возвращать значение одинакового (или совместимого) типа.

Примеры с тернарным оператором

```
let isActive = true;
let code = isActive ? 1 : 0; /* 1 */

let n = 9;
let res = n % 2 == 0 ? n / 2 : n * 2; /* 18 */
```