11. Asosiy operatorlar

Reja:

- **Operatorlar**

> 💡 Javascriptda **operator** bu o'zgaruvchilar yoki qiymatlar ustida amal bajarish uchun ishlatiladigan maxsus belgi yoki kalit so'zdir. Operatorlar yordamida ma'lumotlarni o'zgartirish, hisob-kitob qilish, qiymatlarni solishtirish va boshqa ko'p harakatlarini amalga oshirish mumkin.

Sintaksis

```
operand1 operator operand2
```

- **Misollar**

1. **Arifmetik**:

```
const sum = 5 + 3; // '+' qo'shish operatori
console.log(sum); // 8
```

2. **Solishtirish**:

```
const isEqual = 5 === 5; // '===' solishtirish opera
tori
console.log(isEqual); // true
```

> 💡 Javascriptda ba'zi operatorlar **faqat bitta operand** bilan ishlaydi. Ular **unary operatorlar** deb ataladi.

Sintaksisi

```
operator operand
```

yoki

```
operand operator
```

▼ **Misollar**

1. **Unary Plus (`+`) -** operandni raqamga aylantiradi (agar imkon bo‘lsa).

```
const str = "5";
const num = +str; // "5" (matn)ni 5 (son)ga aylantir
adi
console.log(num); // 5
```

2. **Unary Minus** (`-`) - operandni manfiy qiladi yoki raqamga aylantiradi.

```
const num = 5;
const neg = -num; // Qiymatni manfiy qiladi
console.log(neg); // -5
```

Asosiy operatorlar turlari

1. **Arifmetik operatorlar** - matematik amallarni bajarish uchun ishlatiladi.
2. **Solishtirish (comparison) operatorlar** - qiymatlarni solishtirish uchun ishlatiladi va boolean (`true` yoki `false`) qaytaradi.
3. **Tayinlash (assignment) operatorlar** - qiymatni o‘zgaruvchiga o‘tkazish yoki qiymatni o‘zgaruvchida o‘zgartirish uchun ishlatiladi.
4. **Mantiqiy (logical) operatorlar** - mantiqiy amallarni bajaradi.
5. **Bitwise operatorlar -** ikkilik sanoq sistemasidagi sonlar bilan bit darajasida ishlash imkonini beradi.

▼ **Arifmetik (arithmetic) operatorlar**

💡 **Arifmetik operatorlar** Javascriptda matematik amallarni bajarish uchun ishlatiladi. Ular yordamida qo‘shish, ayirish, ko‘paytirish, bo‘lish va boshqa matematik amallarni bajarish mumkin.

Arifmetik operatorlarning turlari

Operator	Ma'nosi	Misol	Natija
`+`	Qo‘shish	`5 + 3`	`8`
`-`	Ayirish	`5 - 3`	`2`
`*`	Ko‘paytirish	`5 * 3`	`15`
`/`	Bo‘lish	`6 / 3`	`2`
`%`	Qoldiqni olish (modulus)	`5 % 2`	`1`
`**`	Darajaga oshirish	`2 ** 3`	`8`
`++`	Qo‘shish (1 ga oshirish) (Postfix)	`let x = 5; x++`	`x = 6`
`++`	Qo‘shish (1 ga oshirish) (Prefix)	`let x = 5; ++x`	`x = 6`
`--`	Ayirish (1 ga kamaytirish) (Postfix)	`let x = 5; x--`	`x = 4`
`--`	Ayirish (1 ga kamaytirish) (Prefix)	`let x = 5; --x`	`x = 4`

Misollar

1. **Qo‘shish**:

```
const a = 10;
const b = 5;
const result = a + b; // Qo‘shish operatori
console.log(result); // Natija: 15
```

2. **Ayirish**:

```
const a = 10;
const b = 3;
const result = a - b; // Ayirish operatori
console.log(result); // Natija: 7
```

3. **Ko‘paytirish**:

```
const a = 4;
const b = 3;
const result = a * b; // Ko‘paytirish operatori
console.log(result); // Natija: 12
```

4. **Bo‘lish**:

```
const a = 20;
const b = 4;
const result = a / b; // Bo‘lish operatori
console.log(result); // Natija: 5
```

5. **Qoldiqni olish**:

```
const a = 10;
const b = 3;
const result = a % b; // Qoldiqni olish operatori
console.log(result); // Natija: 1
```

6. **Darajaga oshirish**:

```
const base = 2;
const exponent = 3;
const result = base ** exponent; // Darajaga oshirish o
peratori
console.log(result); // Natija: 8
```

7. **Postfix Increment (`x++`)**

```
let x = 5;
let y = x++;
console.log(x); // 6 (x avval y ga qiymatini beradi, ke
yin 1 ga oshadi)
console.log(y); // 5 (x ning dastlabki qiymati y ga ber
ilgan)
```

8. **Prefix Increment (`++x`)**

```
let x = 5;
let y = ++x;
console.log(x); // 6 (x avval 1 ga oshadi)
console.log(y); // 6 (x ning yangi qiymati y ga berilga
n)
```

9. **Postfix Decrement (`x--`)**

```
let x = 5;
let y = x--;
console.log(x); // 4 (x avval y ga qiymatini beradi, ke
yin 1 ga kamayadi)
console.log(y); // 5 (x ning dastlabki qiymati y ga ber
ilgan)
```

10. **Prefix Decrement (`--x`)**

```
let x = 5;
let y = --x;
console.log(x); // 4 (x avval 1 ga kamayadi)
console.log(y); // 4 (x ning yangi qiymati y ga berilga
n)
```

▼ **Solishtirish (comparison) operatorlar**

💡 **Solishtirish operatorlari** Javascriptda qiymatlarni solishtirish va natijani `true` yoki `false` ko‘rinishida qaytarish uchun ishlatiladi.

Solishtirish Operatorlari Turlari

Operator	Ma'nosi	Misol	Natija
`==`	Teng	`5 == '5'`	`true`
`===`	Qattiq teng (turi bilan birga)	`5 === '5'`	`false`
`!=`	Teng emas	`5 != 3`	`true`
`!==`	Qattiq teng emas	`5 !== '5'`	`true`
`>`	Katta	`5 > 3`	`true`
`<`	Kichik	`5 < 3`	`false`
`>=`	Katta yoki teng	`5 >= 5`	`true`
`<=`	Kichik yoki teng	`5 <= 4`	`false`

Misollar

1. **Tenglik (`==`)**:

```
const a = 5;
const b = "5";
console.log(a == b); // true, qiymat teng bo'lgani uchu
n
```

2. **Qattiq tenglik (`===`)**:

```
const a = 5;
const b = "5";
console.log(a === b); // false, chunki turi har xil
```

3. **Teng emas (`!=`)**:

```
const a = 5;
const b = 3;
console.log(a != b); // true, qiymatlar teng emas
```

4. **Katta (`>`)**:

```
const a = 7;
const b = 5;
console.log(a > b); // true, 7 katta 5 dan
```

5. **Kichik yoki teng (`<=`)**:

```
const a = 10;
const b = 10;
console.log(a <= b); // true, 10 kichik yoki teng 10 ga
```

`==` va `===` orasidagi farq

- `==`: Faqat qiymatni tekshiradi. Agar qiymatlar teng bo‘lsa, `true` qaytaradi (turini inobatga olmaydi).

```
5 == '5'; // true
```

- `===`: Qiymat va turini birga tekshiradi.

```
5 === '5'; // false
```

▼ **Tayinlash (assignment) operatorlar**

> 💡 **Tayinlash operatorlari** Javascriptda qiymatlarni o‘zgaruvchilarga belgilash yoki mavjud qiymatni o‘zgartirish uchun ishlatiladi. Tayinlash operatorlari yordamida oddiy qiymat o‘zlashtirishdan tortib, matematik hisob-kitoblarni qisqartirilgan shaklda bajarish mumkin.

Tayinlash operatorlarining turlari

Operator	Ma'nosi	Misol	Natija
`=`	Oddiy tayinlash	`x = 5`	`x = 5`
`+=`	Qo‘shib tayinlash	`x += 3`	`x = x + 3`
`-=`	Ayirib tayinlash	`x -= 2`	`x = x - 2`
`*=`	Ko‘paytirib tayinlash	`x *= 4`	`x = x * 4`
`/=`	Bo‘lib tayinlash	`x /= 2`	`x = x / 2`
`%=`	Qoldiqni tayinlash	`x %= 3`	`x = x % 3`
`**=`	Darajaga oshirib tayinlash	`x **= 2`	`x = x ** 2`

Misollar

1. **Oddiy tayinlash (`=`):**

```
const a = 5; // a ga 5 qiymati tayinlandi
console.log(a); // 5
```

2. **Qo‘shib tayinlash (`+=`):**

```
let a = 5;
a += 3; // a = a + 3
console.log(a); // 8
```

3. **Ayirib tayinlash (`-=`):**

```
let a = 10;
a -= 2; // a = a - 2
console.log(a); // 8
```

4. **Ko‘paytirib tayinlash (`*=`):**

```
let a = 4;
a *= 3; // a = a * 3
```

```
console.log(a); // 12
```

5. **Bo‘lib tayinlash (`/=`):**

```
let a = 20;
a /= 4; // a = a / 4
console.log(a); // 5
```

6. **Darajaga oshirib tayinlash (`**=`):**

```
let a = 2;
a **= 3; // a = a ** 3
console.log(a); // 8
```

7. **Qoldiqni tayinlash (`%=`):**

```
let a = 10;
a %= 3; // a = a % 3
console.log(a); // 1
```

▼ **Suhbat savollari 💼**

▼ **Asosiy operator turlari qaysi?**

1. **Aritmetik operatorlar** - matematik amallarni bajarish uchun ishlatiladi.
2. **Solishtirish (comparison) operatorlar** - qiymatlarni solishtirish uchun ishlatiladi va boolean (`true` yoki `false`) qaytaradi.
3. **Tayinlash (assignment) operatorlar** - qiymatni o‘zgaruvchiga o‘tkazish yoki qiymatni o‘zgaruvchida o‘zgartirish uchun ishlatiladi.
4. **Mantiqiy (logical) operatorlar** - mantiqiy amallarni bajaradi.
5. **Bitwise operatorlar -** sonlar bilan bit darajasida ishlash imkonini beradi.

▼ **`%` operatorining vazifasi nima?**

`%` operatori ikkita sonni bo‘lgandan keyin qoldiqni qaytaradi. Masalan: `10 % 3` ning natijasi `1`.

▼ **Javascriptda `=` va `==` o‘rtasidagi farq nima?**

- `=` operatori qiymatni o‘zgaruvchiga tayinlash uchun ishlatiladi. Bu operator o‘ng tarafdagi qiymatni chap tarafdagi o‘zgaruvchiga yozadi.

```
const x = 10; // 10 qiymati x o'zgaruvchisiga tayinl
andi
const y = 5;  // 5 qiymati y o'zgaruvchisiga tayinla
ndi
```

- `==` operatori ikkita qiymatni taqqoslaydi va ularning **qiymatini bir xil** deb hisoblaydi, agar kerak bo‘lsa, turini avtomatik o‘zgartiradi (type coercion).

```
console.log(5 == "5"); // true
```

▼ **Javascriptda `==` va `===` o‘rtasidagi farq nima?**

- `==` qiymatlarni taqqoslaydi va kerak bo‘lsa turini avtomatik o‘zgartiradi (type coercion).
- `===` esa qiymat va turini bir vaqtda taqqoslaydi.

```
5 == "5";  // true
5 === "5"; // false
```

▼ **`+=` va `-=` operatorlari nima qiladi?**

- `+=` qiymatni qo‘shadi va natijani o‘zgaruvchiga qayta tayinlaydi.
- `-=` qiymatni ayiradi va natijani o‘zgaruvchiga qayta tayinlaydi.

```
let x = 5;
x += 3; // x hozir 8
x -= 2; // x hozir 6
```