Урок 3.3. Логические операторы. Операторы сравнения. Ветвление

Домашнее задание

Строки

Объявление строки

- Существует три типа кавычек для задания строк:

```
const str1 = 'Одинарные кавычк';
const str2 = "Двойные кавычки";
const str3 = `Наклонные кавычки (или косые)`
```

- Одинарные используются в большинстве случаев при объявлении обычных строк
- Двойные иногда используются, чтобы внутрь строки можно было вставить апостроф `'` и не сломать объявление строки
- Я не рекомендую использовать двойные кавычки. Для написания апострофа можно воспользоваться экранированием:

```
// апостроф неэкранирован, объявление строки сломалось
const str = 'I'm a programmer;

// апостроф экранирован и будет внутри строки
const str = 'I\'m a programmer';
```

- Косые кавычки используются в основном, чтобы вставить внутрь строки какую-то переменную:

```
const num = 6;
const str = `Мой опыт работы - ${num} лет`;
```

- Также строку с косыми кавычками можно переносить на новую строку в коде:

```
const str = `Эту строчкуя перенёспри объявлении`;
```

Методы строк

- `toUpperCase`, `toLowerCase` - приведение строки к верхнему и нижнему регистру соответственно

```
const str = 'строка'.toUpperCase(); // СТРОКА
const str2 = 'СтРока'.toLowerCase(); // строка
```

- `includes` - проверяет, есть ли подстрока в данной строке

```
console.log('Строка'.includes('Стр')); // true
console.log('Строка'.includes('отр')); // false
```

- `startsWith`, `endsWith` - проверяют, начинается (заканчивается) ли строка с данной подстроки

```
console.log('Строка'.startsWith('Стр')); // true
console.log('Строка'.endsWith('ка')); // true
```

- `replace`, `replaceAll` - поменять в данной строке подстроку на другую подстроку

```
// Заменяет только первое вхождение подстроки (первую найденную подстроку)

// 'а роза упала на лапу Азора'// Заменяет все вхождения подстроки
// (все найденные подстроки)
const strReplace = 'А роза упала на лапу Азора'.replace('А', 'а');

// 'а роза упала на лапу азора'
const strReplaceAll = 'А роза упала на лапу Азора'.replaceAll('А', 'а');
```

- `trim` - убирает пробелы слева и справа от строки

```
const str = '     строка с пробелами по краям       '.trim();
// 'строка с пробелами по краям'
```

- `str[i]` - получает i-ый символ строки

```
const str = 'Строка';
console.log(str[2]); // 'p' (нумерация с 0)
```

Ветвление

Логические операторы

- `||` - или
- `&&` - и
- `!` - не

Использование в выражениях

```
const a = null;
const b = 'Строка';
const c = 0;
// resultOr присвоится первое не-falsey значение.
const resultOr = a || b || c; // result: 'Строка'

// resultAnd присвоится значение первой falsey-переменной (слева-направо)
// Если все оказались true, значит присвоится значение последней
const resultAnd = a && b && c; // result: null
```

Операторы сравнения

- `==` - сравнение по значению, тип не учитывается (`1 == '1'` - вернёт `true`)
- `===` - сравнение по значению и по типу (`1 === '1'` - вернёт `false`)

- `!=` - сравнение с отрицанием (НЕ равно), тип не учитывается
- `!==` - сравнение с отрицанием (НЕ равно) с учётом типа операндов
- `>` - больше
- `<` - меньше
- `>=` - больше или равно
- `<=` - меньше или равно

Использование в выражениях

```
// result присвоится результат сравнения.
const result = '1' == 1; // result: true
const result2 = '1' === 1; // result: false
```

`if`

Синтаксис конструкции ветвления (ЕСЛИ..ТО)

```
// код перед условием, выполнится в любом случае

if (УСЛОВИЕ) {
  // Код, который выполнится, если УСЛОВИЕ вычислится в true
}

// код после условия: выполнится в любом случае
```

Синтаксис конструкции ветвления (ЕСЛИ..ТО..ИНАЧЕ)

```
// код перед условием, выполнится в любом случае

if (УСЛОВИЕ) {
  // Код, который выполнится, если УСЛОВИЕ вычислится в true
} else {
  // Код, который выполнится, если УСЛОВИЕ вычислится в false
}

// код после условия: выполнится в любом случае
```

Пример

```
const age = 26;
const name = 'Andrew';
const lastName = 'Gulin';

// можно комбинировать несколько условий с
// помощью логических операторов.
if (age > 18 && age < 30) {
    console.log('Ваш возраст от 18 до 30');
}

// у логического И приоритет больше, поэтому
// если нет скобок, то И выполнится первым, и затем ИЛИ
// Если нужен другой приоритет, можно воспользоваться скобками
if ((name === 'Andrew' || name === 'Андрей') && age > 18) {
    console.log('Условие выполнилось')
}
```

`switch`

Пример

```
const someVariable = 'ЗНАЧЕНИЕ';

switch (someVariable) {
    case 'Значение 1': console.log('1'); break;
    case 'Значение 2': {
      // можно выполнять несколько операций, для этого
      // надо поставить фигурные скобки
    } break;
    default: {
      // Если someVariable не совпало ни с одним значением case,
      // выполняется default.
    }
}
```