# Строки

## String

В JavaScript любые текстовые данные являются строками.
Внутренний формат для строк — всегда UTF-16, вне зависимости от кодировки страницы.

# Создание строки

1) Чтобы задать строку в JavaScript, текст нужно с двух сторон закавычить. Кавычки можно использовать тремя разными способами.
   - `let single = 'single-quoted';//` Одинарные кавычки
   - `let double = "double-quoted";//` Двойные кавычки
   - `let backticks = \`backticks\`;//` братные апострофы (обратные кавычки)

*Преимущество обратных кавычек*

➔ *они могут занимать более одной строки*
➔ *можно использовать произвольные выражения ${ .. }*

2) Для работы со строками предназначен объект String, поэтому также можно использовать конструктор String:

let name = new **String**("Tom");

// typeof( `single` ) : "string"

// typeof( name) : "object"

# Спецсимволы

| Символ | Описание |
| --- | --- |
| `\n` | Перевод строки |
| `\',\"` | Кавычки |
| `\\` | Обратный слеш |
| `\t` | Знак табуляции |

\ — «символа экранирования»

```
// перевод строки добавлен с помощью символа перевода строки

let str = "Name:\n Petr";
```

# str.length – длина строки

```
let hello = "привет мир";
console.log(`В строке "${hello}"  ${hello.length} символов`);
// В строке "привет мир"  10 символов
```

# Доступ к символам

1) Получить символ, который занимает позицию `pos`, можно с помощью квадратных скобок: **`[pos]`**

```
let str = `Hello`;

// получаем первый символ

console.log(str[0]); // H
```

2) Получить символ, который занимает позицию `pos`, можно с помощью метод **str.charAt(pos)**

```
console.log(str.charAt(0));                 // H

console.log(str.charAt(str.length-1));    // o
```

*Разница только в том, что если символ с такой позицией отсутствует, тогда*

- ➔ *`[]` вернёт `undefined`,*
- ➔ *а `charAt` — пустую строку*

# Перебор символов в строке

```
for(let elem of str){}
```

```
let str = `Hello`;

for (elem of str) {
    console.log(elem)
}
```

H

e

l

l

o

# Изменение регистра

`toLowerCase()` - Преобразовывает все символы переданной строки в нижний регистр. Метод не изменяет значение самой строки.

`toUpperCase()` - Преобразовывает все символы переданной строки в верхний регистр.Метод не изменяет значение самой строки.

```
let str = `Hello`;

str.toLowerCase()  // hello

str.toUpperCase()   // HELLO
```

# Поиск подстроки

1) str.**indexOf(substr, pos)** - ищет подстроку `substr` в строке `str`, начиная с позиции `pos`, и **возвращает позицию**, на которой располагается совпадение, либо `-1` при отсутствии совпадений.

```
let str = 'Hello World';

console.log( str.indexOf('World') ); // 6

console.log( str.indexOf('Worlds') ); // -1
```

2) str.**lastIndexOf(substr, pos)** ищет подстроку `substr` в строке `str`, начиная с позиции `pos`, и **возвращает индекс** последнего вхождения подстроки

```
console.log( str.lastIndexOf('o') ); // 7
```

# Поиск подстроки

3) **str.includes(substr, pos)** возвращает `true`, если в строке `str` есть подстрока `substr`, либо `false`, если нет.

```
let str = 'Hello World';

console.log(str.includes("World")); // true

console.log(str.includes('World', 6")); // true поиск начался с 6 позиции

console.log(str.includes('World', 7)); // false поиск начался с 7 позиции
```

# Задание

Дана почта, проверить наличие @  в строке

Дана строка “Мама мыла раму”.

- На каких позициях находится ‘а’,’ма’
- Сколько раз встречается вхождение ’ам’

```
let pos = str3.indexOf('@', pos);
console.log(pos);


// найти позиции
let str3 = 'Мама мыла раму;
let pos = 0;
do {
    pos = str3.indexOf('ма', pos);
       // if (pos == -1) break;
    console.log(pos);
    pos += 1;
} while (pos > 0)
```

```
pos = -1;
while ((pos = str3.indexOf('ма', pos + 1)) != -1) {
    console.log(pos);
}
```

# Получение подстроки substr

1) str.slice(start [, end]) - позволяет получить из строки какую-то ее часть. Она принимает два параметра:
   - индекс символа в строке, начиная с которого надо проводить обрезку строки. Обязательный параметр
   - индекс, до которого надо обрезать строку, (не включая) end Необязательный параметра - если он не указан, то обрезается вся остальная часть строки

➔ ***индексы могут иметь отрицательное значение.***Отрицательный индекс указывает на индекс символа относительно конца строки

➔ ***начальный индекс должен быть меньше чем конечный***

```
let str = 'Hello World';

let world1 = str.slice(0, 5); //Hello

let world2 = str.slice(6);    //World

let world3 = str.slice(6, -3); //Wo
```

# Получение подстроки substr

2) str.substring(start [, end]) - Возвращает часть строки между start и end (не включая) end.

- ➔ ***Если `start` больше `end`, то метод `substring` сработает так, как если бы аргументы были поменяны местами.***
- ➔ ***Отрицательные значения `substring`,не поддерживает, они интерпретируются как `0`.***

```
str  = 'Hello World';

world1 = str.substring(0, 5); //Hello

world2 = str.substring(5,0); //Hello
```

# Получение подстроки substr

- str.substr(start [, length]) - Возвращает часть строки от `start` длины `length`. Позволяет указать длину вместо конечной позиции.

➔ ***начение первого аргумента может быть отрицательным, тогда позиция определяется с конца***

```
str  = 'Hello World';

world1 = str.substr(0, 5); //Hello - с 0 позиции всего пять символов

world2 = str.substr(3, 2); //lo -
```

# Объединение строк

concat() - Объединяет текст из двух или более строк и возвращает новую строку.

```
let hello = "Привет ";
let world = "мир";
hello = hello.concat(world);
console.log(hello); // Привет мир

let fr = "дружба";
let str  = hello.concat(world, ' ', fr); Привет мир дружба
let str_1 = hello+ ' ' +world +' 'fr
```

# Замена подстроки

str.replace(‘первое подстрока’,’вторая подстрока’) - заменяет первое вхождение одной подстроки на другую. Возвращает новую строку

```
let hello = "Добрый день";
hello = hello.replace("день", "вечер");
console.log(hello); // Добрый вечер
```

# Удаление пробелов

- `str.trim()` — убирает пробелы в начале и конце строки.
- `str.`**trimStart()**: удаляет пробел с начала строки
- `str.`**trimEnd()**: удаляет пробел с конца строки
- `str.`**trimLeft()**: удаляет пробел с левой части строки
- `str.`**trimRight()**: удаляет пробел с правой части строки

# Разделение строки в массив

str.split([разделитель,размер массива]) разбивает строку на массив подстрок по определенному разделителю. В качестве разделителя используется строка, которая передается в метод

Разделитель указывается первым необязательным параметром. Если он не задан - вернется вся строка. Если он задан как пустые кавычки '' - то каждый символ строки попадет в отдельный элемент массива.

Вторым необязательным параметром можно указать максимальное количество элементов в получившемся массиве

```
message = "   Сегодня была прекрасная погода ";
message = message.trim();
messageParts = message.split(" ");
messageParts2 = message.split('');
console.log(messageParts);  // ["Сегодня", "была", "прекрасная", "погода"]
```

# объединение элементов массива в строку

arr.join([разделитель]) - Метод join объединяет элементы массива в строку с указанным разделителем (он будет вставлен между элементами массива).По умолчанию разделителем станет запятая:

```
let arr = [1, 2, 3];

let str = arr.join('   ');

console.log(str); // 1 2 3
```

# Задание

**//дана строка чисел, преобразовать в массив чисел**

```
let a = "1 2 3 4 5 6 ";
a = a.trim();
let arr = a.split(" ").map(function(elem) {
    elem = parseInt(elem);
    return elem;
});
console.log(arr[1] + arr[3]); //6
```

Перевернем символы строки в обратном порядке

```
let str = '1 2 3 4 5 6 7 8 9';

let arr1 = str.split(' '); //из строки в массив

let arr2 = arr1.reverse(); // обратно

let result = arr2.join('; '); //из массива в строку


//let result = str.split(' ').reverse().join ' ;');
```

```
let str10 = 'My big string lol, big big';
let www = str10.replace(' big', ''); // My string lol, big big
let www = str10.split(' big').join(''); // My string lol,
let www = str10.split(' ').join(''); //  удалить пробелы из строки
—-------------------------
const test1 = (str) => {
    str = str.replace(' big', '');
    console.log(str); // only first 'big' removed

const test2 = (str) => {
    str = str.split(' big').join('');
    console.log(str); // all 'big';
```