**arr.reverse(); обръща реда на елементите в масива**

**arr.join('')** ще вземе всички елементи на масива, ще ги превърне в стринг и ще ги долепи със сепаратор, който посочим в скобите.

**username.split('').reverse().join('');**

**arr.inludes(5)**; търси дали в масива се съдържа номера. Връща True/ False.

**replace** - функция на стринга. връща нов стринг. word.replace( '\_', symbol ); в скобите - какво искаме да заменим, нова дума.

**/** - интерпретира се като специален символ

**/t** - tab

**/n** - new line

**//** ако искаме да ползваме '/' черта трябва да сложим //

**Math.trunc**( 3.14 /число/) - взима цялото число , изрязва частта след запетаята.

**num.toString()**; превръщане на число в Стринг

**string.includes('9');** търси дали нещо се съдържа в дадения стринг

**String.fromCharCode();** връща стринг със специални символи

let letter = String.fromCharCode( 97 + num) - gets the value in decimal and transforms it to a character from the ASCII table.

**parseInt()** function parses a string argument and returns an integer / прави разбор на стринг и връща цяло число

parseInt(*string* [, *radix*])

The value to parse. If this argument is not a string, then it is converted to one using the [ToString](https://tc39.es/ecma262/" \l "sec-tostring)

parseFloat(n) - маха нулите

**toUpperCase()** method returns the calling string value converted to uppercase (the value will be converted to a string if it isn't one)

str.toUpperCase()

**split() -** превръща стринг в масив. Подрежда стринга според това, което сме подали и го превръща в масив. Създава се нова променлива let arr = str.split('i');

words = str.split(' '); разделя стринг изречение на отделни стринг думи разделение от запетая и интервал;

chars = str.split(''); разделя стринга на отделни стринг буквички, разделение с запетая

myString.split('\t') - tab

**arr.push(50);** - добавя елемент към края на масива. и връща дължината на масива ---- **arr.pop();** премахва последния елемент от масива и го връща и може да го запазим в променлива.

**arr.unshift();** - вмъква елемента в началото на масива и връща дължината на масива; ---- **arr.shift();** премахва елемент от началото на масива

**arr.toString()**; превръща масива в стринг и съединява ел. със запетая и принтира на един ред = **arr.join(' ') ;** прави същото, но трябва ние да подадем с какъв знак да съединим елемените.

**arr.includes();** дали съдържа определена стойност; работи и със стрингове.

**indexOf();** връща на кой индекс в масива се намира стойността ;

връща -1 ако не съществува. Може да търси в **String/Array**.

indexOf(2, 2); ако имаме 2 2ки в масива, посочваме че търсим 2 и от кой индекс нататък да търси. arr.indexOf(searchElement, fromIndex);

**slice();** реже и копира, не променя стария масив, създава нов масив, като използва част от друг масив. Приема selected start to end (exclusive) - 0, 1 или 2 параметъра. Старият масив остава непроменен.(2)започва от втория индекс до края.(2,5) започва от 2 индекс до 5, но без него. Ако slice(3, -2) започва от 3индекс и бори 2 числа от края.

**splice(2, 3);** start, delete-count изрязваме, изтриваме и добавяме елементи, променя оригиналния масив. (2) изрязва от 2 индекс до края. (2, 3) от 2 индекс трие 3 елемента.(2, 3, 85) трие и на тяхно място слага 85 и др. ако има.(2, 0, 95, 12) - нищо не трие, добавя 95 и 12 преди 2 индекс. Работи с отрицателни числа.

**transform elements - приемат функции**

**map()** - създава нов масив, прилагайки функция върху всеки един елемент в масив. Създаваме нов масив, копие на първоначалния **arr.map(x => x + 2); ..... myArr .map(x => x.length);**

**map(Number)** - превръща стрингови числа в числа.

**map(toString())** - превръща в стринг

**arr.map(x = > x.length);** връща дължината на елементите в масива

**filter()** - създава нов масив. само със стойности, които са true. Предикат - функция , която връща true/ false. Не променя стария масив. Нека променливата(current value) е **x =>** подаваме true/false стойност (x!== 2)

**arr.filter(x => x.length > 3);**

**arr.reverse** - обръща масива отзад напред

**sort()** - подрежда елементите в масива по сравняваща функция. не създава нов. По подразбиране елементите ще бъдат подредени по азбучен ред, нарастващо. Ако го ползваме с числа подрежда по азбучен ред.

**compare** - приема два параметъра a, b.

if result < 0, a is before b;

if result > 0; a is after b;

if result = 0; a is equal to b;

**function (a, b) => a - b** подрежда числата в нарастващ ред, (b - a) в намаляващ ред

**arr.sort(function(a,b){**

**return a.length - b.length;**

**});**

**array = arr.sort((a, b) = > a - b);**

**localeCompare(**) сравнява ги лексикографски вместо ASCII таблицата /

**а.localeCompare(b)** сравнява по азбучен ред от а - z, ако искаме обратно - b.localeCompare(a)

**arr.sort((a,b) => a.localeCompare(b));**

**forEach** - като for cycle

**let result = '';**

**arr.forEach( (element, i, array) => {** /\*може да се достъпва освен елемента, индекса и масива

**element += 'a';**

**}**

**arr.reduce ((a, b) => a + b)** събира 1 и следващия елемент.след това сумата + следващия. връща сумата на масива.

let result = arrOne.reduce((sum, num) => sum += num);

**STRING METHODS**

**indexOf('Java') - посочва индекса на който се намира думата(зачита малки и големи букви)**

**lastIndexOf('Java') - ако в Стринга има еднакви думи, може да започнем отзад напред.**

**Ако искаме да изведе следващата еднаква дума, посочваме от кой индекс да започне да търси -> indexOf('Java', 5) / lastIndexOf('Java', 24)**

**substring(5, 10); започни от 5 индекс и изрежи 10 символа от началото. Посочваме начало на стринга и края. Ако не посочим краен индекс - реже до края.**

**НЕ ! substr(5, 10); започни от 5 индекс и изрежи следващите 10 символа. ! да не се ползва!**

**replace(search, replacement);** заменя само първото съвпадение.

**replace with Regex**

let regex = /dog/g;

text.replace(regex, 'cat');

**НЕ! replaceAll**

let result = myString.replace('.bg', '.com');

split(); разделя по подаден символ

let tokens = myString.split(' ').filter(c => c.length > 0) - връща без празен елемент

console.log(text.split(' ').filter(w => w == word).length);

токенизация -> от едно цяло да разделим на малки парчета

**str.repeat( count)** - даваме стринг и колко пъти да се повтаря.console.log( '\*\*\*'.repeat(6))

**trim()** маха интервалите/празни редове/ в началото и в края string.trim();

**trimStart() / trimEnd()** - маха само в началото/ само в края text = text.trimStrart();

**startsWith()**; дали даден стринг започва с определена дума. Зачита малки и големи букви.

**endsWith();** дали даден стринг завършва с определена дума.

**padStart()** допълва стринг със символи в началото (дължина, с какво допълваме) hours.padStart(2, '0');

**padEnd();** допълва стринга със символи в края на стринга.

**RegExp**

**[wk ] -** търсим една от тези букви отделно. ако я има една от тях, има един мач

**[wk ]**+ търсим поредица от тези символи

**[A-Za-z]+** всички поредици от букви.търси поредица от изброените в скоби бувки

**[A-Z][a-z]+** дума, която започва с една главна буква последвана от малки

**[\d]** - хвани една от цифрите

[\d ]+$ хвани една

**^[A-Z][a-z]+: \+[\d]+$** целия стрингот началто до края да съвпада на този регекс. [^........$]

* \w – matches any **word character** (a-z, A-Z, 0-9, \_)
* \W – matches any **non-word character** (the opposite of \w)
* \s – matches any **white-space** character
* \S – matches any **non-white-space**  character (opposite of \s)
* \d – matches any **decimal digit** (0-9)
* \D – matches any **non-decimal character** (the opposite of \d)
* **\*** – matches the previous element zero or more times
* **+** – matches the previous element one or more times
* **?** – matches the previous element zero or one time
* **{3}** – matches the previous element exactly 3 times
* **(subexpression)** – captures the matched subexpression as numbered group
* **(?:subexpression)** – defines a non-capturing group
* **(?<name>subexpression)** – defines a named capturing gro**let text = 'Today is 2015-05-11';**
* **let regex = /\d{4}-\d{2}-\d{2}/g;**
* **let containsValidDate = regex.test(text);**
* **console.log(containsValidDate); *// true***
* up