Introduction to Web Programming – Lab 8

Mitrea Dan Alexandru
danalexmitrea@gmail.com

JavaScript

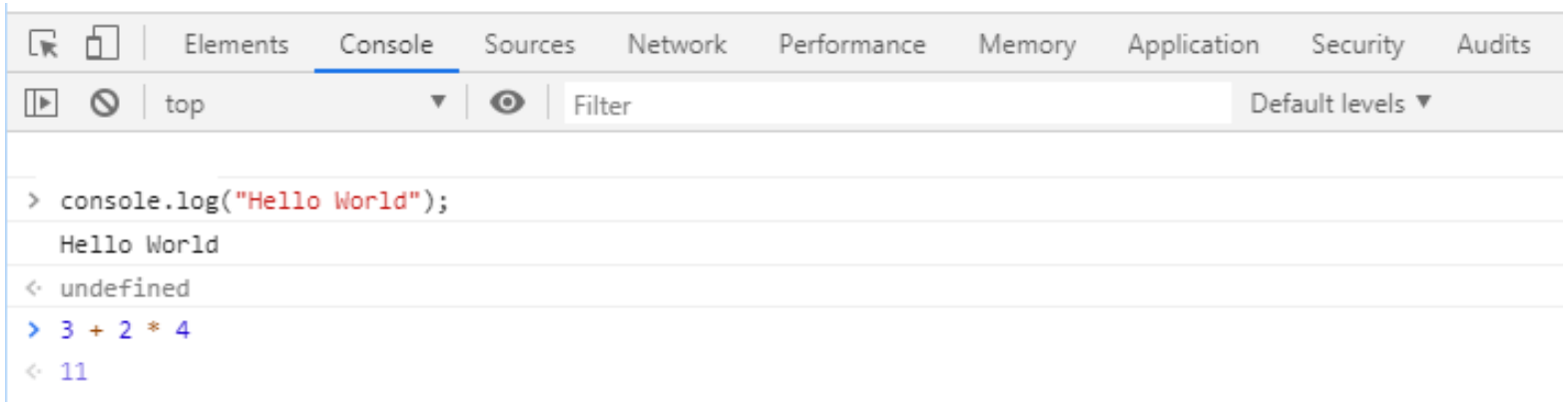
JavaScript is one of the **3 languages** all web developers **must** learn:

1. **HTML** - to markup the content of web pages

2. **CSS** - to describe the look and formatting of web pages

3. **JavaScript** - to program the behavior of web pages

Execution


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
top
Filter
Default levels
> console.log("Hello World");
Hello World
undefined
> 3 + 2 * 4
11

Include JavaScript into HTML

In HTML, JavaScript code must be inserted between <script> and </script> tags

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <script type="text/javascript">
        // script code
    </script>
</head>
<body>


    <script type="text/javascript">
        // script code
    </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <script src="script.js"></script>
</head>
<body>


    <script src="script.js"></script>
</body>
</html>
```

Scripts can also be placed in external files. JavaScript files have the **file extension .js**.

Variables

Variable	Exemple
String	**let** name = “John”;
Number	**let** grade = 100;
Boolean	**let** isPass= true;
undefined	**let** name = undefined;
null	**let** myVariable = null;
Array	**let** myArr = [“John”, “Jane”, 100];
Object	**let** objVariable = {nom: “John”, age: 18};
Function	**function** area(length, width) { ... }

Variables

STATIC
statically-typed

String title= “Java”;

DYNAMIC
dynamically-typed

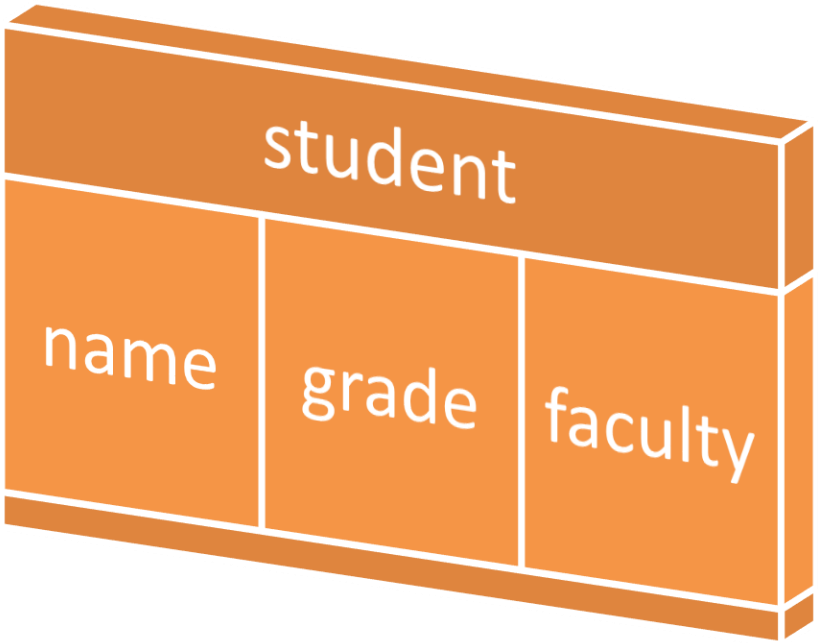
let title = “JavaScript”;

Use **typeof** to check the type of a variable

Objet

```
let name = "Jean";
let grade = 10;
let faculty = "FILS";
```

```
let student = {
    name: "Jean",
    grade: 10,
    faculty:  "FILS"
};
```



The operators

Operator	Symbol(s)	Example
Sum / Concatenation	+	2 + 3; “Hello” + “ ” + “World”;
Subtraction, multiplication, division	- * /	2 * 3;
Assignment Operator	=	let myVariable = 100;
Identity operator	===	myVariable === 10;
Negative operator and inequality operator	! !==	myVariable !== 10;

Comments

```
/*
        Here is a comment that can be on several lines
 */

// Here is a comment on a single line
```

JavaScript debugging

Press **F12** and use **debugging** in any browser

Why use **debugging**:

- Allows debugging of scripts
- Step by step execution
- Adding breakpoints
- Watch expressions
- Visualize the DOM tree

Function Syntax

A JavaScript function is a block of code designed to perform a particular task.
A JavaScript function is executed when "something" invokes it (calls it).
A JavaScript function is defined with the **function** keyword, followed by a **name**, followed by parentheses**()**.

Syntax

```
function nameFunction (parameter-list) {
        instructions
}
```

Function Syntax - Example

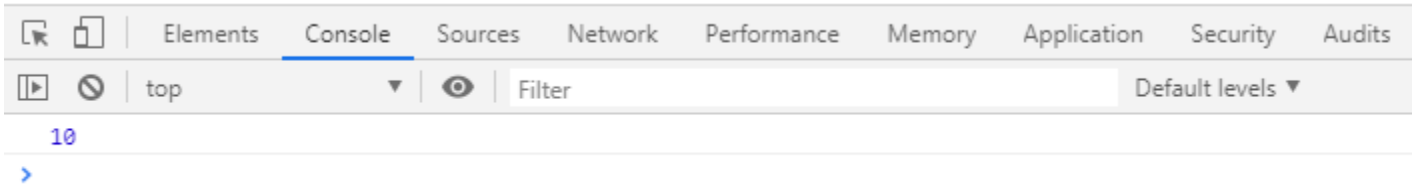
file example.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>FILS</title>
</head>
<body>


    <script src="js/basic.js"></script>

    <script>
        let value = area(5, 2);
        console.log(value);
    </script>

</body>
</html>
```

file basic.js

```js
function area(length, width) {
    let result = length * width;
    return result;
}
```

Console



Conditional structures

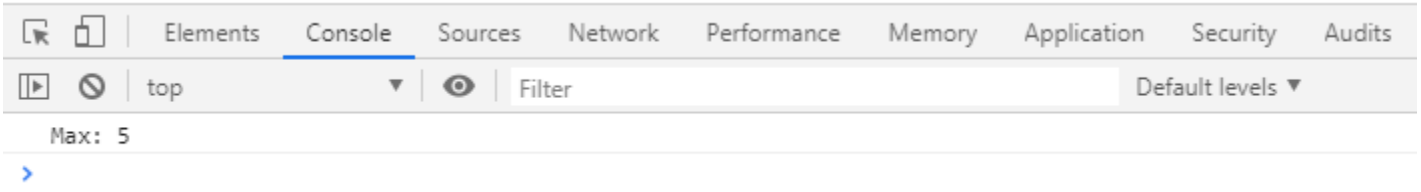
file example.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>FILS</title>
</head>
<body>


    <script src="js/basic.js"></script>

    <script>
        let result = max(4, 5);
        console.log("Max: " + result);
    </script>

</body>
</html>
```

file basic.js

```
function max(a, b) {
    if (a > b) {
        return a;
    } else {
        return b;
    }
}
```

Console



Repetition structure

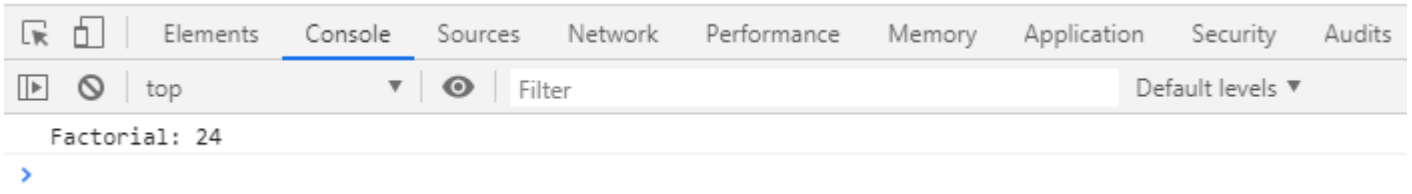
file example.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>FILS</title>
</head>
<body>


    <script src="js/basic.js"></script>

    <script>
        console.log("Factorial: "
                    + factorialLoopFor(4));
    </script>

</body>
</html>
```

file basic.js

```js
function factorialLoopFor(n) {
    let resultat = 1;

    for (let i = 1; i <= n; i++) {
        resultat = resultat * i;
    }

    return resultat;
}
```

Console



Repetition structure

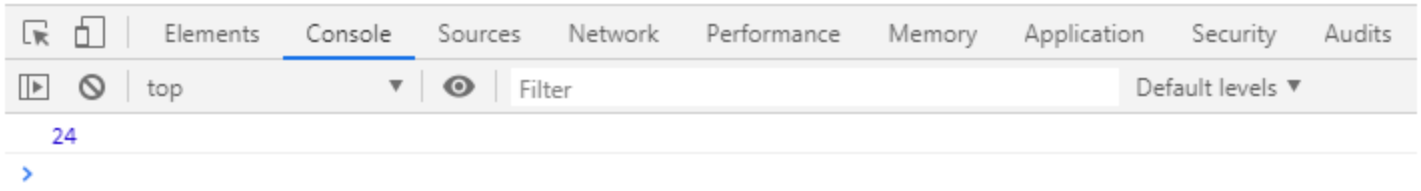
file example.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>FILS</title>
</head>
<body>


    <script src="js/basic.js"></script>

    <script>
        console.log(factorialLoopWhile(4));

    </script>

</body>
</html>
```

file basic.js

```js
function factorialLoopWhile(n) {
    let resultat = 1;
    let i = 1;

    while (i <= n) {
        resultat = resultat * i;
        i++;
    }

    return resultat;
}
```

Console



String functions

charAt(index)
Returns the character (exactly one UTF-16 code unit) at the specified index.

charCodeAt(index)
Returns a number that is the UTF-16 code unit value at the given index.

concat(string2[, string3, ..., stringN])
Combines the text of two strings and returns a new string.

includes(searchString[, position])
Determines whether one string may be found within another string.

endsWith(searchString[, length])
Determines whether a string ends with the characters of another string.

indexOf(searchValue[, fromIndex])
Returns the index within the calling String object of the first occurrence of the specified value, or -1 if not found.

lastIndexOf(searchValue[, fromIndex])
Returns the index within the calling String object of the last occurrence of the specified value, or -1 if not found.

localeCompare(compareString[, locales[, options]])
Returns a number indicating whether a reference string comes before or after or is the same as the given string in sort order.

repeat(count)
Returns a string consisting of the elements of the object repeated the given times.

replace(substr, function)
Used to find a match between a regular expression and a string, and to replace the matched substring with a new substring.

slice(beginIndex[, endIndex])
Extracts a section of a string and returns a new string.

split([separator[, limit]])
Splits a String object into an array of strings by separating the string into substrings.

startsWith(searchString[, position])
Determines whether a string begins with the characters of another string.

substr(start[, length])
Returns the characters in a string beginning at the specified location through the specified number of characters.

substring(indexStart[, indexEnd])
Returns the characters in a string between two indexes into the string.

toLowerCase()
Returns the calling string value converted to lower case.

toUpperCase()
Returns the calling string value converted to uppercase.

trim()
Trims whitespace from the beginning and end of the string.

Array functions

copyWithin(target[, start[, end]])
Copies a sequence of array elements within the array.

fill(value[, start[, end]])
Fills all the elements of an array from a start index to an end index with a static value.

pop()
Removes the last element from an array and returns that element.

push(element1[, ...[, elementN]])
Adds one or more elements to the end of an array and returns the new length of the array.

reverse()
Reverses the order of the elements of an array in place — the first becomes the last, and the last becomes the first.

shift()
Removes the first element from an array and returns that element.

sort([compareFunction])
Sorts the elements of an array in place and returns the array.

splice(start[, deleteCount[, item1[, item2[, ...]]]])
Adds and/or removes elements from an array.

unshift(element1[, ...[, elementN]])
Adds one or more elements to the front of an array and returns the new length of the array.

concat(value1[, value2[, ...[, valueN]]])
Returns a new array comprised of this array joined with other array(s) and/or value(s).

includes(searchElement[, fromIndex])
Determines whether an array contains a certain element, returning true or false as appropriate.

indexOf(searchElement[, fromIndex])
Returns the first (least) index of an element within the array equal to the specified value, or -1 if none is found.

join([separator])
Joins all elements of an array into a string.

lastIndexOf(searchElement[, fromIndex])
Returns the last (greatest) index of an element within the array equal to the specified value, or -1 if none is found.

slice([begin[, end]])
Extracts a section of an array and returns a new array.

IMPORTANT!!! Use the functions which you receive from the lab slides.

Create a file for each exercise and include the JavaScript script inside the file.

Example 1

Calculate the sum of the digits of a given number.

```
function sumDigits(n) {
        .....
}

console.log(sumDigits(123));
console.log(sumDigits(12301));
console.log(sumDigits(1010));
```

Output:

6
7
2

Example 1

Calculate the sum of the digits of a given number.

```
<script>
   function sumDigits(n) {
        let sum = 0;

        while(n > 0) {
            sum = sum + n % 10;
            n = parseInt(n / 10);
        }

        return sum;
    }

    console.log(sumDigits(123));
    console.log(sumDigits(12301));
    console.log(sumDigits(1010));
</script>
```

Output:

6

7

2

Example 2

Delete an element from an array.

```
function deleteElement(arr, element) {
        .....
}

let arrTest = [5, 7, 1, 4, 8];

console.log(deleteElement(arrTest, 4));
console.log(deleteElement(arrTest, 9));
console.log(deleteElement(arrTest, 7));
```

Output:

```
[5, 7, 1, 8]
[5, 7, 1, 8]
[ 5, 1, 8]
```

Example 2

Delete an element from an array.

```
<script>
    let arrTest = [5, 7, 1, 4, 8];

    function deleteElement(arr, element) {
        let index = arr.indexOf(element);

        if (index !== -1) {
            arr.splice(index, 1);
        }

        return arr;
    }

    console.log(deleteElement(arrTest, 4));
    console.log(deleteElement(arrTest, 9));
    console.log(deleteElement(arrTest, 7));
</script>
```

Output:

[5, 7, 1, 8]
[5, 7, 1, 8]
[5, 1, 8]

Example 3

Write a function that simulates a monopoly dice. The function must return the sum, but if the numbers are equal, generate the dice again and add the values to the previous sum.

```
function monopolyDice() {
        .....
}

console.log(monopolyDice());
```

Output:

```
5  2        1  1
7           3  3
           5  5
           0
```

Example 3

Write a function that simulates a monopoly dice. The function must return the sum, but if the numbers are equal, generate the dice again and add the values to the previous sum.

```
<script>
    function monopolyDice() {
        let dice1 = 0;
        let dice2 = 0;
        let sum = 0;
        let counter = 0;

        while (dice1 === dice2) {
            dice1 = parseInt((Math.random() * 6 + 1));
            dice2 = parseInt((Math.random() * 6 + 1));

            sum += dice1 + dice2;
            counter++;
            console.log(dice1 + " " + dice2);

            if (counter === 3 && dice1 === dice2) {
                sum = 0;
                break;
            }
        }

        return sum;
    }

    console.log(monopolyDice());
</script>
```

Output:

```
5 2
7
```

```
1 1
3 3
5 5
0
```

Exercise 1

Calculate the sum of even numbers smaller than n.

```
function sum(n) {
        .....
}

console.log(sum(20));
```

Output:

90

Exercise 2

Calculate the sum of the odd numbers between 20 and 40.

```
function sum(x, y) {
        .....
}

console.log(sum(20, 40));
```

Output:

300

Exercise 3

Calculate the following sum:

$$1^1 + 2^2 + 3^3 + \ ... + n^n$$

```
function sum(n) {
        .....
}

console.log(sum(1));
console.log(sum(2));
console.log(sum(3));
```

Hint: Math.pow(base, exponent)

Output:

1
5
32

Exercise 4

Calculate the following sum:

$$1 \times 2 \times 3 + 2 \times 3 \times 4 + 3 \times 4 \times 5 + \cdots$$

```
function sum(n) {
        ..........
}

console.log(sum(1));
console.log(sum(2));
console.log(sum(3));
```

Output:

6
30
90

Exercise 5

Calculate the following sum:

$$1! + 2! + 3! + \cdots + n!$$

```
function sum(n) {
            .....
}

console.log(sum(1));
console.log(sum(2));
console.log(sum(3));
```

Output:

1
3
9

Exercise 6

Calculate the following sum:

$$1 + (1 + 2) + (1 + 2 + 3) + \ldots + (1 + 2 + 3 + \ldots + n)$$

```
function sum(n) {
        ..........
}

console.log(sum(1));
console.log(sum(2));
console.log(sum(3));
```

Output:

1
4
10

Exercise 7

Calculate the value of the polynomial.

$$3x^3 - x^2 + 2x + 1$$

```
function polynom(x) {
        .....
}

console.log(polynom(1));
console.log(polynom(2));
```

Output:

5

25

Exercise 8

Calculate the value of the polynomial.

$$x^4 + 4x^3 - 3x - 2$$

```
function polynom(x) {
        .....
}

console.log(polynom(1));
console.log(polynom(2));
```

Output:

0

40

Exercise 9

Write a function that takes the following array. If the element of the array is odd, change the value of the array with 1, if the element of the array is even, change the value of the array with 0. Use the function fill to change the value of the element.

```
let arr = [3, 5, 1, 4, 2];
```

Output:

[1, 1, 1, 0, 0]

Exercise 10

Write a function that takes the following array. Sort the array and swap the first element with the last element.

```
let arr = [3, 5, 1, 4, 2];
```

Output:

[5, 2, 3, 4, 1]

Exercise 11

Write a function that takes the following array. Sort the array and double the middle element(i.e. write it two times).

```
let arr = [3, 5, 1, 4, 2];
```

Output:

[1, 2, 3, 3, 4, 5]

Exercise 12

Calculate the arithmetic mean..

```
function mean(tab) {
        .....
}

let arr = [3, 5, 1, 4, 2];

console.log(mean(arr));
```

Output:

3

Exercise 13

Find the maximum value from an array.

```
function maxValue(tab) {
        .....
}

let arr = [1, 5, 8, 4, 2, 7];

console.log(maxValue(arr));
```

Output:

8

Exercise 14

Calculate the scalar product of two vectors.

$$a \times b = a_1 \times b_1 + a_2 \times b_2 + \ldots + a_n \times b_n$$

```
function scalar(x, y) {
            .....
}

let a = [3, -1, 6];
let b = [5, 5, 3];

console.log(scalar(a, b));
```

Output:

28

Exercise 15

Delete the maximum value from an array.

```
function deleteValue(tab) {
        .....
}

let arr = [1, 5, 8, 4, 2, 7];

console.log(deleteValue(arr));
```

Output:

[1, 5, 4, 2, 7];

Exercise 16

Move at the end the maximum value from an array.

```
function moveMax(tab) {
        .....
}

let arr = [1, 5, 8, 4, 2, 7];

console.log(moveMax(arr));
```

Output:

[1, 5, 4, 2, 7, 8];

Exercise 17

Fill an array with random numbers from 0 to 100.

```
function fillArrayRandom(n) {
        .....
}

console.log(fillArrayRandom(5));
```

Output:

[37, 89, 64, 57, 53];

Exercise 18

Calculate the "median" of these numbers (The median = the value in the middle of the array, after its sorting. If the number of elements is even, the median is the arithmetic mean of the numbers in the middle)

```
function median(tab) {
        .....
}

let arr1 = [3, 5, 1, 4, 2];
let arr2 = [3, 5, 1, 4, 2, 7];

console.log(median(arr1));
console.log(median(arr2));
```

Output:

3

3.5

Exercise 19

Find the longest name in a given array.

```
function longestName(tab) {
        .....
}

let arr = ["Albert", "Christophe", "John", "Bernard"];

console.log(longestName(arr));
```

Output:

Christophe

Exercise 20

Create a function that will concatenate two arrays.

```
function concatenate(x, y) {
        .....
}

let arr1 = [3, 2, 4];
let arr2 = [6, 5, 1, 7];

console.log(concatenate(arr1, arr2));
```

Output:

[3, 2, 4, 6, 5, 1, 7]

Exercise 21

Write a function that receives the following array.

```
let arr = ['remove opt 1', 'buy', 'remove opt 3', 'sell'];
```

The function removes from the array the options which start with the word remove.

```
let newArr = removeOption(arr);
for (let i = 0; i < newArr.length; i++) {
        console.log(newArr[i]);
}
```

Output:

buy
sell

Exercise 22

Count the times lowercase and uppercase letters appear in a string.

```
function countChar(phrase, character) {
        .....
}

let text = "A JavaScript function is a block of code designed to
perform a particular task";

console.log(countChar(text, 'a'));
console.log(countChar(text, 'b'));
console.log(countChar(text, 's'));
console.log(countChar(text, 'j'));
```

Output:

8 1 4 1

Exercise 23

Reverse the letters of a word.

```
function reverseWord(word) {
        .....
}

let text = "JavaScript";

console.log(reverseWord(text));
```

Output:

tpircSavaJ

Exercise 24

Convert a name like in the following example.

```
function convertName(name) {
        .....
}

let text = "John Doe";

console.log(convertName(text));
```

Output:

J. Doe

Exercise 25

Convert a name like in the following example.

```
function convertName(name) {
        .....
}

let text = "John Doe";

console.log(convertName(text));
```

Output:

J. D.

Exercise 26

Write a function to hide email addresses. The first part must be divided in two equal parts and hide it like in the example.

```
function protect(mail) {
        .....
}

console.log(protect("test@ing.pub.ro"));
console.log(protect("danalex@ing.pub.ro"));
console.log(protect("mailupb2018@fing.pub.ro"));
```

Output:

te...@ing.pub.ro
dan...@ing.pub.ro
mailu...@fing.pub.ro

Exercise 27

Write a function to hide an email addresse. Display the first and last letter and substitute with a dot the other letters.

```
function protect(mail) {
        .....
}

console.log(protect("test@ing.pub.ro"));
console.log(protect("danalex@ing.pub.ro"));
console.log(protect("mailupb2018@fing.pub.ro"));
```

Output:

t..t@ing.pub.ro
d.....x@ing.pub.ro
m.....8@fing.pub.ro

Exercise 28

Reverse the letters of all the words inside a phrase.

```
function reversePhrase(phrase)  {
            .....
}

let text = "A JavaScript function is a block of code designed to
perform a particular task";

console.log(reversePhrase(text));
```

Output:

A tpircSavaJ noitcnuf si a kcolb fo edoc dengised ot mrofrep a ralucitrap ksat

Exercise 29

Write a function that remove vowels from string.

```
function removeVowels(str)  {
        .....
}

console.log(removeVowels('remove vowels'));
console.log(removeVowels('hello fils’));
```

Output:

rmv vwls
hll fls