0101

file:///C:/Users/ansku/OneDrive%20-%20H%C3%A4meen%20ammattikorkeakoulu/Tiedostot/Fullstack/0101.html

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<p id = "tulos"></p>

<script>

var v1 = "Haloo";

var v2 = "Maailma!";

var tulos = "<h1>" + v1 + " " + v2 + "</h1>";

console.log(tulos);

var para = document.getElementById("tulos")

para.innerHTML = tulos;

</script>

</body>

</html>

0102

Kuva, joka sisältää kohteen teksti, ohjelmisto, Multimediaohjelmisto, Tietokonekuvake

Kuvaus luotu automaattisestiKuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, Verkkosivusto

Kuvaus luotu automaattisesti

0103

file:///C:/Users/ansku/OneDrive%20-%20H%C3%A4meen%20ammattikorkeakoulu/Tiedostot/Fullstack/0103.html

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<script>

let table = document.createElement("table");

let row = table.insertRow();

let cell1 = row.insertCell();

let cell2 = row.insertCell();

let cell3 = row.insertCell();

let cell4 = row.insertCell();

cell1.innerHTML = "JavaScriptin";

cell2.innerHTML = "opiskelu";

cell3.innerHTML = "on";

cell4.innerHTML = "ihanaa!";

document.body.appendChild(table);

</script>

</body>

</html>

0104

file:///C:/Users/ansku/OneDrive%20-%20H%C3%A4meen%20ammattikorkeakoulu/Tiedostot/Fullstack/0104.html

<!DOCTYPE html>

<html>

<head>

</head>

<body>

<script>

Array.prototype.min = function(){

    var min = parseInt(this[this.length-1]), el;

    for(var i=this.length-2; i>=0; i--){

        el = parseInt(this[i]);

        if(el<min){

            min = el;

        }

    }

    return min;

};

Array.prototype.max = function(){

    var max = parseInt(this[this.length-1]), el;

    for(var i=this.length-2; i>=0; i--){

        el = parseInt(this[i]);

        if(el>max){

            max = el;

        }

    }

    return max;

};

var array = [3,1,5,];

document.write(array);

document.write('</br>');

document.write(array.max());

</script>

</body>

</html>

0201

<file:///C:/Users/ansku/OneDrive%20-%20H%C3%A4meen%20ammattikorkeakoulu/Tiedostot/Fullstack/0201.html>

Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, näyttö

Kuvaus luotu automaattisesti

0202

0203

const print = (str, taskTime) => {

setTimeout(function() {

console.log(`${str} finished!`)

}, taskTime)

}

const execute = (str, taskTime, callback) => {

callback(str, taskTime)

}

console.log("Task1 to execution...")

execute('Task1', 2000, print)

console.log("Task2 to execution...")

execute('Task2', 2000, print)

console.log("Task3 to execution...")

execute('Task3', 500, print)

console.log("Last code line executed!")

1. print nuolifunktio pyytää parametreja str ja taskTime.

Hypätään riville 7.

2. execute nuolifunktio pyytää parametreja str, taskTime ja callback.

Hypätään riville 11.

3. Kirjoitetaan konsoliin "Task1 to execution..."

Seuraavalle riville.

4. Annetaan execute nuolifunktiolle parametreihin arvot 'Task1', 2000 sekä 'print' callbackiksi.

Hypätään riville 8.

5. Callbackinä nyt print, jolle parametreinä arvot 'Task1' ja 2000

Hypätään riville 2.

6. printin suoritteissa setTimeout funktio, joka kirjoittaa annetun 'Task1' + "finished!" konsoliin, kun on kulunut 2000ms (2s)

Hypätään riville 14.

7. Kirjoitetaan konsoliin "Task2 to execution..."

Seuraavalle riville.

8. executelle annetaan parametrien arvoiksi 'Task2', 2000 sekä print callbackiksi.

Hypätään riville 8.

9. Callback print saa arvot 'Task2' ja 2000.

Hypätään riville 2.

10. setTimeout kirjailee konsoliin "Task2 finished!" kun on kulunut 2000ms (2s)

Hypätään riville 17.

11. Kirjoitetaan konsoliin "Task3 to execution..."

Seuraavalle riville.

12. executelle 'Task3', 500 ja print callbackiksi.

Hypätään riville 8.

13. callback print saa arvot 'Task3' ja 500.

Hypätään riville 2.

14. setTimeout saa arvot 'Task3' ja 500. Kirjoittaa konsoliin "Task3 finished!" 500ms (0.5s) kuluttua.

Hypätään riville 20.

15. Kirjoitetaan konsoliin "Last code line executed!"

16. Noin puolen sekunnin kuluttua konsoliin ilmestyy "Task 3 finished!" kuten luvattiin.

17. Noin kahden sekunnin kuluttua konsoliin ilmestyy "Task1 finished!" ja sen jälkeen vasta "Task2 finished!". Vaikka niillä on sama aika Task1 pitäisi olla valmis ennen Task2, koska se suoritettiin ensin.

0204