**MINISTERUL EDUCAŢIEI AL REPUBLICII MOLDOVA**

**UNIVERSITATEA TEHNICĂ A MOLDOVEI**

**FACULTATEA CALCULATOARE, INFORMATICǍ**

**ȘI MICROELECTRONICǍ**

**CATEDRA AUTOMATICĂ ȘI TEHNOLOGII INFORMAŢIONALE**

**Raport**

**la disciplina Sisteme de gestiune a cunoștințelor și data mining**

**Tema: JavaScript questions**

**Laboratorul 2**

**A efectuat: st. gr. TI-171M Cebotari Daria**

**A verificat: lect. sup. Poddukin Vladimir**

**Chișinău 2018**

**Sarcina 1**

"Hello JavaScript" Creati o pagina HTML care la incarcare arata un alert pop-up window cu text 'Pagina se incarca'. Dupa ce utilizatorul apasa OK, arata al doilea alert 'pagina a fost incvarcata cu succes'.

Rezolvați in 2 moduri:

1. Cu JavaScript intern

2. Cu JavaScript pastrat intr-un sau 2 fisiere externe

**Rezolvare:**

1 JavaScript intern

<!DOCTYPE html>

<html>

<head>

<title>Exercitiu 1</title>

</head>

<body>

<h1>Exercitiul 1</h1>

<script type="text/javascript">

alert("Pagina se incarca.");

window.onload = function(){

alert("Pagina a fost incarcata cu succes");

};

</script>

</body>

</html>

2 JavaScript pastrat in fisier

//index.html

<!DOCTYPE html>

<html>

<head>

<title>Exercitiu 1.2</title>

</head>

<body>

<h1>Exercitiul 1.2</h1>

<script type="text/javascript" src="./externalScript.js"></script>

</body>

</html>

//externalScript.js

alert("Pagina se incarca.");

window.onload = function(){

alert("Pagina a fost incarcata cu succes");

};

**Sarcina 2**

Ordine de executare a scripturilor

In exemplu de mai jos sunt conectate intr-o pagina 2 scripturi small.js и big.js.

Daca presupunem ca small.js se executa cu mult mai rapuid decat big.js, care va fi executat primul ?

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

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

Dar asa?

<script async src="big.js"></script>

<script async src="small.js"></script>

Dar daca asa?

<script defer src="big.js"></script>

<script defer src="small.js"></script>

Scripturile (adica referintele la ele sunt plasate inauntru tagului <body> ! Ceea ce noi punem in tag <head> nu se executa la incarcara paginii, asa scripturi trebuie pornite undeva din <body>

**Rezolvare:**

**small.js**

alert("Small JavaScript");

**big.js**

for (var i = 0; i < 90000; i++){

console.log(i);

}

alert("Big JavaScript");

**index.html**

<!DOCTYPE html>

<html>

<head>

<title>Exercitiu 2</title>

</head>

<body>

<h1>Exercitiul 2</h1>

<!--

<script type="text/javascript" src="./big.js"></script>

<script type="text/javascript" src="./small.js"></script>

<script async type="text/javascript" src="./big.js"></script>

<script async type="text/javascript" src="./small.js"></script>

<script defer type="text/javascript" src="./big.js"></script>

<script defer type="text/javascript" src="./small.js"></script>

-->

</body>

</html>

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

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

In cazul dat, se incarcarca dentii big.js apoi small.js

<script async src="big.js"></script>

<script async src="small.js"></script>

In cazul dat, se incarcarca dentii big.js apoi small.js

<script defer src="big.js"></script>

<script defer src="small.js"></script>

In cazul dat, se incarcarca dentii big.js apoi small.js

**Sarcina 3**

Clasificati numele variabelor in trei categorii:

-bune,

-nerecomandate,

-inadmisibile:

var adminName;

var imia\_Admina;

var numele\_Adminului;

var admin\_Name;

var AdminName;

var ADMINNAME;

var имяАдмина;

var \_;

var 1stAdmin;

var primaryAdmin;

var $;

var i;

var j;

var getCount;

var count;

var getCount();

var int;

var admin-Name;

Problema dezvolta cultura scrierii codului profesionist.

**Rezolvare:**

- bune:

var adminName;

var primaryAdmin;

var i;

var j;

var getCount;

var count;

- nerecomandate:

var imia\_Admina;

var numele\_Adminului;

var admin\_Name;

var AdminName;

var ADMINNAME;

var 1stAdmin;

var $;

var int;

var имяАдмина;

var \_;

- inadmisibile:

var getCount();

var admin-Name;

**Sarcina 4**

Cu ce va fi egala valoarea lui x in exemplu de mai jos (explicati)?

var a = 2;

var x = 3 + (a \*= 3);

**Rezolvare:**

a \*= 3 => 6

x = 3 + 6 = 9

**Sarcina 5**

Cum va comporta codul urmator. Explicati:

alert(alert(1) && alert(2));

**Rezolvare:**

Chemarea metodei alert() nu returneaza nici o valoare, alt fel spus returneaza undefined. Deci al doilea alert() nici nu se va chema.

**Sarcina 6**

Calculati valoarea expresiilor daca aceasta este posibil. Explicati rezultatele primite

"" + 1 + 0 //totul se transforma in string si se concatineaza, in rezultat obtinem 10

"" - 1 + 0 //conversia numerica a stringului, il face in 0, apoi se face calculul algebric si

obtinem -1

true + false //conversia numerica 1 + 0, obtinem 1

6 / "3" //conversia numerica, stringul „3” se transforma in numar, in rezultat obtinem 2

"2" \* "3" //conversie numerica, Number("2") \* Number("3") = 6

4 + 5 + "px" //se face adunare 4 + 5, apoi concatenare de stringuri, obtinem "9px"

"$" + 4 + 5 //totul se transforma in string, se concateneaza in rezultat obtinem "$45"

"4" – 2 //conversie numerica a stringului „4”, apoi se face calculul algebric si obinem 2

"4px" – 2 //rezultat NaN, deoarece Number("4px") = NaN

7 / 0 //impartirea la 0 ne da infinit

" -9\n" + 5 //se concatineaza stringul. Rezultat „-9\n5”, 5 va fi din linia noua

" -9\n" – 5 //Number(" -9\n") - 5 = -9 - 5 = -14, rezultat -14

5 && 2 //incearca sa transforme 5 in false, si daca nu poate, returneaza expresia 2

2 && 5 //incearca sa transforme 2 in false, si daca nu poate, returneaza expresia 5

5 || 0 //incearca sa transforme 5 in true, si daca nu poate sal faca true, returneaza 0, da

daca poate, returneaza 5

0 || 5 //incearca sa transforme 0 in true, si daca nu poate sal faca true, returneaza 0, da

daca poate, returneaza 5

null + 1 //Number(null) + 1 = 0 + 1 = 1

undefined + 1 //Number(undefined) + 1 = NaN + 1 = NaN

null == "\n0\n" //Boolean(null) == Boolean("\n0\n") = false

Number(null) == Number("\n0\n") = true

+null == +"\n0\n" //Boolean(+null) == Boolean(+"\n0\n") = true

//Number(+null) == Number(+"\n0\n") = true

**Sarcina 7**

Scrieți functie min(a,b) care returnează valoarea minimă între a și b:

Exemple de apelare:

min(2, 5) == 2

min(3, -1) == -1

min(1, 1) == 1

Nota: prezentați trei variante de declarație a funcției - Function Declaration, Function Expression și new Function

**Rezolvare:**

<!DOCTYPE html>

<html>

<head>

<title>Exercitiu 7</title>

</head>

<body>

<h1>Exercitiul 7</h1>

<script type="text/javascript">

//Function declaratoin

function min(a, b) {

return a < b ? a : b;

}

//Function expression

var min = function(a,b) {

return a < b ? a : b;

}

//Function new Function

var min = new Function("a", "b", "return a < b ? a : b");

</script>

</body>

</html>

**Sarcina 8**

Analizați codul de mai jos din punct de vedere a formatării și convențiilor de cod recomandate pentru JavaScript.

function pow(x,n)

{

var result=1;

for(var i=0;i<n;i++) {result\*=x;}

return result;

}

x=prompt("x?",'')

n=prompt("n?",'')

if (n<0)

{

alert('Puterea'+n+'nu este procesată în cadrul exemplului dat, introduceți o putere >0');

}

else

{

alert(pow(x,n))

}

**Rezolvare:**

<!DOCTYPE html>

<html>

<head>

<title>Exercitiu 7</title>

</head>

<body>

<h1>Exercitiul 7</h1>

<script type="text/javascript">

function pow(x, n) {

var result = 1;

for (var i = 0; i < n; i++) {

result \*= x;

}

return result;

}

x = prompt("x?", '')

n = prompt("n?", '')

if (n < 0) {

alert('Puterea' + n + 'nu este procesată în cadrul exemplului dat, introduceți o putere >0');

} else {

alert(pow(x, n))

}

</script>

</body>

</html>