JavaScript

Teljesen kezdőknek

1.

1. óra

Adattípusok Java Scriptben:

Egész vagy integer – `integer` `int`
Lebegőpontos szám – `float`
Karakter – `character` `char`
Karaktersorozat – `string`
Logikai – `boole` `boolean`

De a JavaScriptben van egy meghatározó szavunk a „var” (variable - változó).
Így nem kell külön megadni az adattípus formáját. Például:

var a = 5; -> integer
var b = 6.5; -> float
var c = A; -> char
var d = abc; -> string
var e = false; -> boole

Hogyan iratunk ki?
pl:

```
<script type="text/javascript">

document.write(’ Helló világ! ’);

</script>
```

Eredmény:

Helló világ!

Matematikai műveletek, kiíratás:

```
<script type="text/javascript">
var a = 2;
var b = 3;
document.write(a+b);
</script>
```

Az eredmény pedig: 5
Az adattípusunk integer.

String befűzése:

```
<script type="text/javascript">
var a = 2;
var b = 3;
document.write('<b>2+3=</b>' , a+b);
</script>
```

Az eredmény pedig: 2+3=5
Az adattípusunk 2+3= string és az a+b összege integer.

Újabb változó felvétele:

```
<script type="text/javascript">
var a = 2;
var b = 3;
var c = a*b
document.write(c);
</script>
```

Az eredmény pedig: 6
Az adattípusunk integer.

```
<script type="text/javascript">
var a = 2;
var b = 3;
var c = a/b
document.write(c);
</script>
```

Az eredmény pedig: 0.6666666666666666
Az adattípusunkat egyből float -nak értelmezi.

3.

A változó értekének növelése:

```
<script type="text/javascript">
var a = 2;
var b = 3;
var c = a+b
document.write(c++,’<br>’);
document.write(c);
</script>
```

Az eredmény pedig: 5
6

A ++c esetében először hozzáadja az érteket, utána kiírja.

Fordítva is működik, kivonással. c- -, vagy - -c. Ez teljesen nyilvánvaló, így erre nem írok példát.

```
<script type="text/javascript">
var a = 2;
var b = 3;
var c = a+b
document.write(c++,'<br>');
document.write(c++,'<br>');
document.write(c);
</script>
```

Az eredmény pedig: 5
6
7

Maradékos osztás:

```
<script type="text/javascript">
var a = 9;
var b = 2;
var c = a%b
document.write(c);
</script>
```

Az eredmény pedig: 1

Elágazások:

```
<script type="text/javascript">
var a = 5;
if (a>4)
{ document.write(’ Nagy szám ’);}
</script>
```

Eredmény: Nagy szám; ha az ’a’ kisebb négynél, akkor semmi.

```
<script type="text/javascript">
var a = 3;
if (a>4)
{
document.write('Nagy szam');
} else {
document.write('Kis szam');
}
</script>
```

Eredmény: Kis szám

```
<script type="text/javascript">
var a = 4;
if (a>4)
{ document.write('Nagy szam');
} else if(a==4) {
document.write('Az ’a’ pontosan 4.');
} else {
} document.write('Kis szám');
</script>
```

Az eredmeny: Az ’a’ pontosan 4.

5.

Beépített függvények:

```
<script type="text/javascript">
var a = prompt('Hany eves vagy?','');
document.write(a,'eves vagy.');
</script>
vagy
<script type="text/javascript">
var a = prompt('Hany eves vagy?','');
if (a<25)
{
document.write('Fiatal vagy.');
} else {
document.write('Felnőtt vagy');
}
</script>
```

2. óra

Elágazások:

% modulo. Osztás, amelynek az eredménye a maradék.

```
<script type="text/javascript">
var a = prompt('Adj meg egy számot!');
if(a%2==0) {
document.write('A szám páros.');
} else {
document.write('A szám páratlan.');
}
</script>
```

Viszont marad egy probléma. Ha karaktert írunk, a válasz páratlan lesz, mivel a karakter nem lesz egyenlő soha az integer nullájával, amit a program elvár.

Van egy isFinite beépített függvény, ami eldönti a változóról, hogy integer, vagy karakter, karaktersor vagy string. Például:
`isFinite(abc); // false` vagy `isFinite(3); // true.`

```
<script type="text/javascript">

var a = prompt('Adj meg egy számot!');
if(isFinite(a)) {
if(a%2==0)
{
document.write('A szám páros.');
} else {
document.write('A szám páratlan.');
}
} else {
document.write('Nem számot adtál meg!');
}
</script>
```

Van egy további problémánk. Ha nem egész számot adok meg, mindig páratlannak fogja értelmezni.
Van egy beépített függvény: a Math.floor.

A valós szám egész alakját veszi. Ha a valós szám egész alakja egyenlő a valós számmal, akkor egész számot adtam meg.

```
<script type="text/javascript">

var a = prompt('Adj meg egy számot!');
if(isFinite(a)){
if(Math.floor(a)==a) {
if(a%2==0)
{
document.write('A szám páros.');
}
else
{
document.write('A szám páratlan.');
}
}
else
{
document.write('A valós számok nem lehetnek párosak, páratlanok.');
}
}
else
{
document.write('Nem számot adtál meg!');
}

</script>
```

Logikai operatorok:

Ezek a ’ vagy ’ és az ’ és ’. A vagy ’OR’ jele a: ||, az és ’AND’ jele a kettő &&.

```
<script type="text/javascript">
var a = 5;
var b = 3;
if(a==5 && b==2)
 {
document.write('Rendben');
} else {
document.write(' Várok  még. ');
}
</script>
```

Az eredmény a ’ Várok még. ’ lesz, mindaddig, míg át nem írom a b értekét 2-re.

Ha csak az egyiknek kell igaznak lennie, akkor ’vagy’ operatort használunk.

```
<script type="text/javascript">
var a = 5;
var b = 3;
if(a==5 || b==2)
 {
document.write('Rendben');
} else {
document.write(' Várok  még. ');
}
</script>
```

Csinosítsuk a scriptet!

```
<script type="text/javascript">
var a = prompt('Adj meg egy egész számot! ');
var b = 3;
if(a==5 || b==2)
 {
document.write('Rendben');
} else {
document.write(' Várok  még. ');
}
</script>
```

Több ’ vagy’ és ’ és’ kapcsolatot is használhatok egyszerre. ’ a ||b && c ’

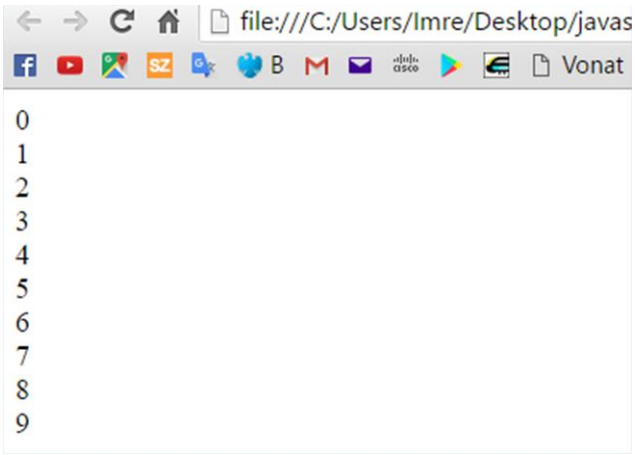
For ciklus:

for(var a = 0; a<10; ++a)

- Első futáskor az "a" változó 0 (**a=0;** első paraméter)
- Minden egyes ciklus ismétléskor "a" változó értéke egyel növekszik (**++a;** harmadik paraméter)
- A ciklus addig fut, amíg a kisebb 10 (**a<10;** második paraméter)
- Ha kettesével akarok lépni akkor **a+=2** a **++a** helyett.
- Visszafele számolunk: **for (a=20; a>=0; --a)**

```
<script type="text/javascript">
for(var a = 0; a<10; ++a) {
document.write(a,'<br>')}
</script>
```

Eredmeny:



Második feladat, for ciklusra:

```
<script type="text/javascript">
for(var a = 0; a<181; ++a) {
        var b = a*Math.PI/180; // radiánra átváltás
        document.write(a, "° = sin ", Math.sin(b),'<br>');
}
</script>
```

Eredmény:

sin0° = 0
sin1° = 0.01745240643728351
sin2° = 0.03489949670250097
sin3° = 0.05233595624294383
sin4° = 0.0697564737441253
sin5° = 0.08715574274765817
sin6° = 0.10452846326765346
sin7° = 0.12186934340514748
sin8° = 0.13917310096006544
sin9° = 0.15643446504023087
sin10° = 0.17364817766693033
sin11° = 0.1908089953765448

Az első komolyabb feladat. Írassuk ki egy képzeletbeli Otto motor dugattyújának mozgását, a felső holtponttól, az alsó holtpontig. A főtengely sugara 10 cm, a hajtókar hossza a.) 30 cm b.) 15 cm c.) 10cm. Az eredmény a felső holtponttól való elmozdulás legyen mm-ben, a főtengely elfordulásával fokonként.

```
<script type="text/javascript">

var haj = prompt('hajtokar hossza')

for(var a = 0; a<180; ++a) {


        var alfa = a*Math.PI/180; // radiánra átváltás //

        var x2 = 10*Math.cos(alfa); // A fotengely haromszoge

        var x1 = Math.sqrt((haj*haj)-((10*Math.sin(alfa)*(10*Math.sin(alfa))))); //
Hajtokar haromszoge

        var c = (x1+x2)*10;

        var cc = Math.floor(c); // Csak az egesz ertekeket vizsgaljuk

        document.write( a,'° ---------',100-cc+(haj*10), 'mm ','<br>'); // Ha 300 mm a
hajtokar + 100 mm a sugar, osszesen 400 mm. Ebbol vesszuk el az atfogo meretet. A
kor sugara 100 mm, igy az eredmenynek 0-rol kell indulni, es 200-ra kell vegzodni.

        }

</script>
```

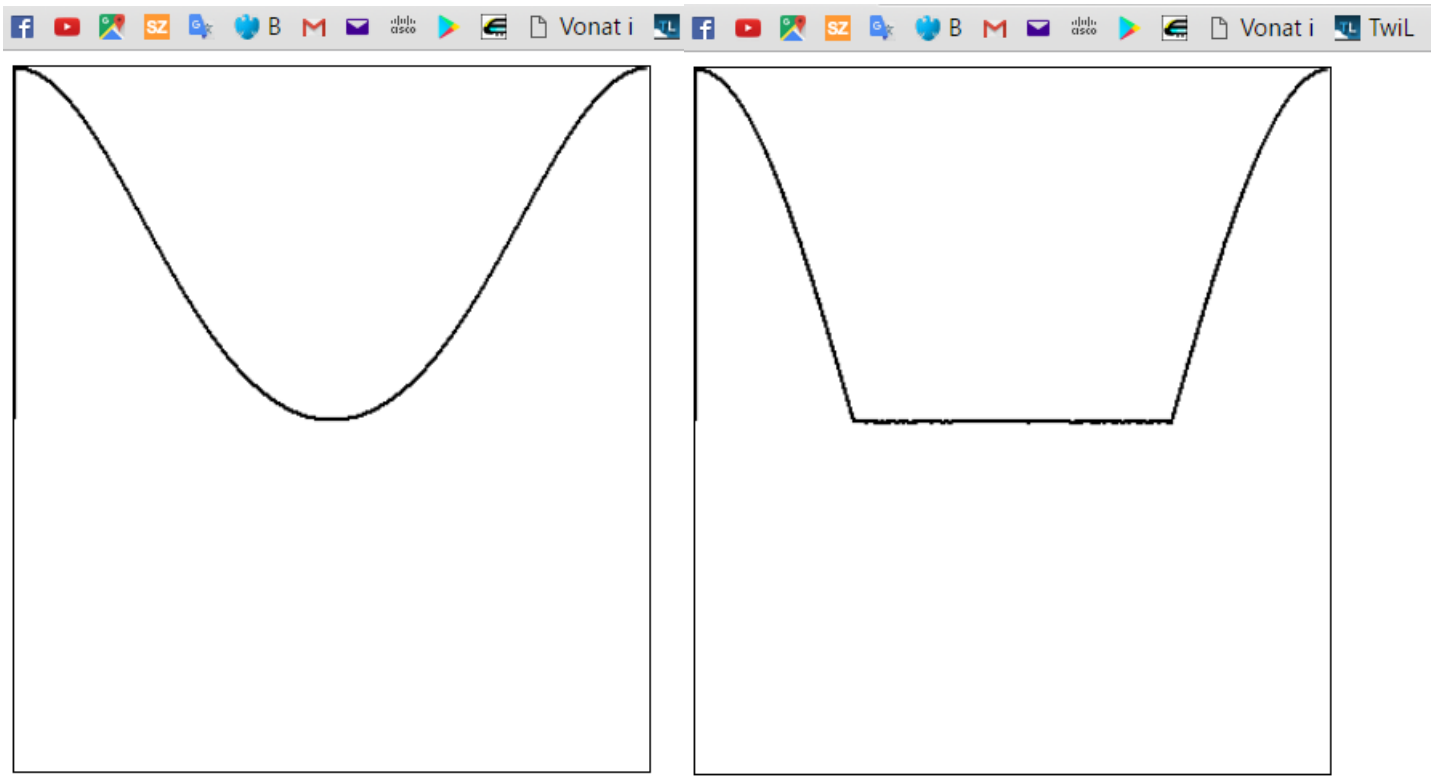
Működik. Be kell illeszteni egy üres HTML dokumentumba, es megvizsgálni, vajon szabályos szinusz hullámot ír le a dugattyú mozgása!

Jelenítsük meg a görbét a képernyőn!

```
<body>
<canvas id="myCanvas" width="400" height="400" style="border:1px solid #000000;"></canvas>
	<script type="text/javascript">
	var csd = document.getElementById("myCanvas");
	var ctx = csd.getContext("2d");
	ctx.fillStyle = "#FF0000";
	var haj = prompt('Hajtokar hossza 10 es 30 kozott?');
	var a = 0;
	ctx.moveTo(a,200);
	var trigger = setInterval(function(){
			var alfa = a*Math.PI/180; // radiánra átváltás //
		var x2 = 10*Math.cos(alfa); // A fotengely haromszoge
		var x1 = Math.sqrt((haj*haj)-((10*Math.sin(alfa)*(10*Math.sin(alfa))))); // Hajtokar haromszoge
		var c = (x1+x2)*10;
		var c1 = Math.floor(c); // Csak az egesz ertekeket vizsgaljuk
		var cc = (100-c1+(haj*10));
		ctx.lineTo(a,cc);
		ctx.stroke();
		a++;
		if (a==360){
			clearInterval(trigger);
		}
	}, 30);
	</script>
</body>
```

13.

Az eredmény:



1. eset

2. eset

A hajtókar méretét első esetben 20 cm-nek vettük. A dugattyú mozgása torzított szinusz hullám.
A hajtókar méretét második esetben 10 cm-nek vettük. Mivel a főtengely sugara is 10 cm, 90 fokos elfordulásnál a hajtókar a dugattyú mozgásával már, szintén 90 fokos szöget fog bezárni, majd 90-tol 270 fokig a dugattyú az alsó holtpontban fog tartózkodni. A háromszögünk átfogója 0 lesz 90 – tol 270 – fokig.

Ha 9-cm hosszura vennénk a hajtókart, mivel a főtengely sugara 10 cm, 1 cm hiány lesz, es a hajtókarnak el kellene szakadni.

Mindez csak arra jó, hogy a leg szélsőségesebb esetben bemutassa, mennyire torzítja a dugattyú szinuszos mozgását a hajtókar mérete.

Végtelen hosszú hajtókar eseten viszont nem befolyásolná.

3. óra

While és DoWhile ciklus: Be kérünk egy számot. Ha nem számot írunk be újra kéri. Ha szám, kiíratja. De mindaddig, amíg nem számot írunk, be újra kéri a számot.

```
<script type="text/javascript">
		var alfa = prompt('Adj meg egy szöget!');
		while (!isFinite(alfa)) {
			alfa = prompt('Adj meg egy szöget!');
			}
		document.write('A beirt szög:',alfa,' °');
</script>
```

A második téma a DoWhile ciklus:

Ugyan úgy tudjuk használni, de a program rövidebb, gyakorlatilag egy sorral kevesebb, mivel ha karaktereket írunk be ugyan oda tér vissza. es újra végrehajtja az első sort.

```
<script type="text/javascript">
		do {var alfa = prompt('Adj meg egy szoget!');
			}
			while (!isFinite(alfa));
		document.write('A beirt szog:',alfa,' °');
</script>
```

Ki tehetjük vágólapra, es beilleszthetjük egy HTML dokumentumba.

Véletlen szám generálás a Math.random segítségével:

Egy játékot készítünk. Generalunk egy számot 0 es 1 között. Megszorozzuk 100-al, es hozzáadunk 1-et. Így kapunk egy számsor lehetőség et 1 es 100 között.

A ciklus mindaddig fut körbe-körbe, míg a tipp nem lesz egyenlő a-val. Ha egyenlő megszakad, es kiírja: Gratulálok!

A két elágazással pedig csinálunk egy kis segítséget.

A következő függvényeket használtuk:

Math.floor – egész értek

Math.random – random szám generálás 0 es 1 között.

A prompt-tal pedig egy számot kertünk be.

```
<script type="text/javascript">

        var a = Math.floor(Math.random()*100)+1;
        document.write('Gondoltam egy szamra 1 es 100 kozott.','<br>');
        do {
            var tip = prompt('Adj meg egy szamot!');
            if (tip>a) {
                document.write('Kisebbet!','<br>');
            }else if (tip<a){
                document.write('Nagyobbat!','<br>');
            }
        } while(tip!=a);
        document.write('Gratulálok, a gondolt szám az,',a,'<br>');
</script>
```

Stringek, stringes függvények:

Stringet megadhatunk str = ’valami’ alakban, vagy new String(’valami’) alakban, amely kezelése később könnyebb lesz. A str.length kiírja, hány karakterből áll a string.

```
<script type="text/javascript">
        //var str = 'Bogar';
        var str = new String(prompt('Add meg a neved'));
        document.write('Szia  ',str,'!  ',str.length,' karakter a neved.');
</script>
```

Eredmény, ha Annamari-t írok be: Szia Annamari! 8 karakter a neved.

További stringes függvények: Majd lesznek, bocs. véletlenül tovább mentem.

Függvények:

Miért is lesz ez hasznos nekünk? Ha van egy hosszú folyamat, es azt sokszor fel kell használnunk, akkor ezt elnevezzük, es csak a paraméterezett függvény nevet írjuk be.

Például:

```
<script type="text/javascript">
	function kiir() {
		document.write('Ezt akarom kiiratni.');
	}
	kiir();
</script>
```

Számoljuk ki egy háromszög területét:

```
<script type="text/javascript">
	function terulet_triangular() {
		return (a*m)/2
	}
		var a = prompt('Add meg a haromszog alapjat!');
		var m = prompt('Add meg a haromszog magassagat!');
	document.write('A haromszog terulete:',terulet_triangular(a,m));

</script>
```

Ha a bemenet 10 es 20, az eredményünk 100 lesz, tehát helyes. Viszont, ha a return után az a+b – t akarjuk kiszámítani, 1020 lesz az eredmény. Miért?

Itt a JS nem tudja, milyen adattípussal van dolga, és stringeket ad össze. Ezt sokféleképpen ki küszöbölhetjük, nem biztos, hogy elegáns, most én megszoroznám mindkét érteket 1-el. Stringeket össze lehet adni, viszont nem lehet szorozni. Meg egy megjegyzés. A return ’a’ es ’m’ – je, nem biztos, hogy egyenlő a document.write ’a’ es ’m’ – évél. Csupán a szabályt adja meg.

Tehát amit tudnunk kell: function valami() {Ezt fogja megcsinálni}. Beírom valami, es a kapcsos zárójelben levő dolgot megcsinálja.

Mennyi van még hátra? Úgy számolom 50 oldal lesz a nagyon alapok. Mire lesz ez elég? Arra, mint a regényíráshoz megtanulni írni, olvasni. Nélküle nem megy, de ez semmire nem elég.

onClick:

```
<!DOCTYPE html>
<html>
<head>
	<title>JavaScript</title>
<script type="text/javascript">
	function kerulet()
	{
		var a = prompt('Add meg a teglalap a oldalat!');
		var b = prompt('Add meg a teglalap b oldalat!');
		var k =  (2*(1*a+1*b));
		alert('A teglalap kerulete: ' + k);
	}
</script>

</head>
<body>
<button onClick="kerulet()">kerulet</button>
</body>
</html>
```

Fontos, az ’alert’ – nél, a kiíratáskor, nem vessző kell a változó elé, hanem egy + jel!

4. óra

Megkeressük, hogy a szám prímszám, vagy van osztója 1 – en es önmagán kívül:

```
<script type="text/javascript">
		function prime() {
			var x = prompt('Adj meg egy 1-nél nagyobb egész számot!');
			if(x>1 && Math.floor(x)==x)  {    // kizárjuk a kettőnél kisebb számokat, es nem egész számokat! Ekkor a string kizárására mar nincs szükség. Ha csak 1-nel nagyobb számot fogadok el, az csak szám lehet.
				if (eldont(x)) { // 'eldont function true' értéke prím számot jelent.
				alert('A szam Prim.');
			}else{
				alert('A szam nem Prim'); // Nyilván az ellentéte nem prím.
			}
		}else{
			alert('Egesz szamot adj meg!');
			}
		}
	function eldont(x) {
		var i = 2; // Lesz egy hiba a kódban. A kettő nem prím, es az 1 prím számként szerepel. Ez a házifeladat. Meg kell oldani!
		while(i<=x/2 && x%i!=0 ) // Nem osztunk végig, csak az értek felével, mert nincs értelme, ha az  osztásnál van maradék, akkor szintén megszakad a ciklus.
			{
				++i;
			}
	return x==2 || x/i>Math.floor(x/i); // Ha van maradék, vagy az érték 2  a return igaz. A szám prím.
		}
	</script>
</head>
<body>
<input type="button" value="Prim" onClick="prime()" />
</body>
</html>
```