Webprogramozás szakkör

Előadás 5 (2012.04.09)

Programozás alapok

Eddig amit láttunk:

- Programozás lépései
  - Feladat leírása (specifikáció)
  - Algoritmizálás, tervezés (folyamatábra, pszeudokód)
  - Programozás (elkészítjük)
  - Tesztelés (ellenőrizzük, hogy jól csináltuk-e meg)
- Hogyan írunk ki tetszőleges szöveget / számot (document.write(…))
- Kommentezés
- Változók
- Különböző típusok összefűzése (+)

Feladat 1 – írjuk ki 1-től 3-ig a számokat
Szükséges ismeretek: kiíratás

```
// Megoldás 1:
document.writeln(1);
document.writeln(2);
document.writeln(3);

// Megoldás 2:
document.write(”1, 2, 3”);    // szövegként
document.write(1, 2, 3);      // számok felsorolva
```

Operátorok

- **Műveleti**: +, -, *, /, %, ++, --
- **Relációs (összehasonlítás):** <, <=, >, >=, =, == (érték egyezés), === (típus és érték egyezés), != (nem egyenlő)
- **Logikai** (ÉS – VAGY – NEGÁLÁS (tagadás)): &&, ||, !
- **Feltételes:** változó = (feltétel) ? ha igaz : ha hamis;
- **Szövegre a + operátor szöveg összefűzését jelenti**

Feladat 2 – műveletek
Szükséges ismeretek: kiíratás, változók, operátorok

Inicializáljunk (adjunk neki értéket) néhány változót, és írjuk ki a következőket (kifejezéseket):

```
var a = 5;
var b = 8;
var c = "programozás";
var d;
var e = false;
var f = true;

document.writeln(d = a+b);
document.writeln(d);
document.writeln(b/a);
document.writeln(b*a);
document.writeln(a-b);
document.writeln(b % 3);
a++;
document.writeln(a);
a--;
document.writeln(a);

document.writeln(a == c);
document.writeln(a != c);
document.writeln(e);
document.writeln(!e);
document.writeln(e && f);
document.writeln(e && e);
document.writeln(f && f);
document.writeln(e || f);
document.writeln(e || e);
document.writeln(f || f);
document.writeln(e && e);
document.writeln(f && f);

document.writeln(a + " " + c);
document.writeln(a >= 5 ? "igggaaaz" : "hammiiiis");
document.writeln(a > 5 ? "igggaaaz" : "hammiiiis");
```

Mutassuk meg, hogy nem csak a document.write-ban végezhetjük el a műveleteket (pl. a legelső sor helyett ***d = a+b;*** , majd kiírjuk ***d***-t.

Nézzük meg, hogy mi történik az alábbi esetekben:

```
document.writeln(d);                // első sorba téve: undefined
document.writeln(c % 2);            // NaN = Not a Number
```

Az első esetben nem adtunk a változónak értéket, így definiálatlan lesz (undefined), a második esetben pedig szövegen próbálunk olyan műveletet végezni, ami számra értelmes (maradékképzés), így jelez, hogy nem számon próbálkozunk.

Következtetések:

- A műveletek sorrendjének változtatása nem ugyanazt az eredményt adja
- A program soronként fut végig, és az alkalmazott változtatások érvényben maradnak

Elágazás

If-else

Az if kötelező, az else if-ből 0..N darab lehet, az else-ből pedig 0..1 darab.

- if (feltétel) { ha teljesül }
- else if (feltétel) { ha ez teljesül }
- else { egyébként ez történik }

Feladat 3 – elágazás

Szükséges ismeretek: kiíratás, változók, operátorok, elágazás

```
var a = 5;
var b = 8;
var c = "programozás";
var d;
var e = false;
var f = true;

if (a > 3 && c == "Programozás") {
   document.writeln("IF");
}
else if (a > 3 && c == "programozás") {
   document.writeln("ELSEIF");
}
else {
   document.writeln("ELSE");
}
```

Figyeljünk a **kód tördelésére** is (space-ek, tabok)!

Switch-case (NEM VOLT ÓRÁN)

Ugyanazt valósítja meg, mint az IF-ELSE szerkezet, csak annyi különbséggel, hogy akkor tudjuk ezt használni, ha ugyanaz a kifejezés szerepelne minden feltételben, csak az a kérdés, hogy mi az értéke.

IF-ELSE példa:

```
if (n == 1) {
        document.writeln(„ha 1-el egyenlő az „n””);
}
else if (n == 2) {
        document.writeln(„ha 2-vel egyenlő az „n””);
}
else {
        document.writeln(„ha nem 1 és nem 2”);
}
```

UGYANEZ switch-case-el:

```
switch(n) {
case 1:
  document.writeln(„ha 1-el egyenlő az „n””);
  break;
case 2:
  document.writeln(„ha 2-vel egyenlő az „n””);
  break;
default:
  document.writeln(„ha nem 1 és nem 2”);
}
```

A ***break;*** utasítások azért kellenek, hogy kilépjünk a szerkezetből, és a switch lezáró kapcsos zárójele után folytatódjon a program futása. Ha nem tennénk ki, de belefutnánk például a ***case 1:*** ágba, akkor utána megvizsgálja az összes többi ágat is, így olyan lesz, mintha nem a fenti IF-ELSE IF-ELSE szerkezet lenne, hanem az alábbi (két különálló egymás utáni IF):

```
if (n == 1) {
          ha 1-el egyenlő az „n”
}
if (n == 2) {
          ha 2-vel egyenlő az „n”
}
ha nem 1 és nem 2
```

Ciklusok

For ciklus

Adott kezdőértéktől, adott végértékig egy változót növelünk, és ellenőrizzük, hogy mikor teljesül a végérték feltétele.

```
for (ciklusváltozó_neve = kezdőérték; végérték feltétel; ciklusváltozó változtatása) {
   <minden ciklusban végrehajtandó utasítások>
}
```

A ciklusváltozót ha a for()-ban deklaráljuk, akkor csak azon belül lesz érvényes.

Feladat 4 – írjuk ki 1-től 10-ig a számokat (fejlesszük tovább az 1. feladatot)
Szükséges ismeretek: kiíratás, változók, operátorok, for ciklus

```
for (var i = 0; i < 11; i++) {
   document.writeln(i);
}
```

Feladat 5 – írjuk ki 1-től 10-ig a páros számokat
Szükséges ismeretek: kiíratás, változók, operátorok, for ciklus

```
for (var i = 2; i < 11; i = i+2) {
   document.writeln(i);
}
```

Feladat 6 – ÖNÁLLÓ FELADAT – írjuk ki 1-től 10-ig a páratlan, nem 3-al osztható számokat
Szükséges ismeretek: kiíratás, változók, operátorok, elágazás, for ciklus

```
for (var i = 1; i < 11; i = i + 2) {
        if (i % 3 != 0) {
                document.writeln(i);
        }
}
```

Az előző feladat is megoldható cikluson belüli elágazással vizsgálva, hogy 2-vel osztható-e.

While ciklus

- Egy adott feltétel teljesüléséig megy a ciklus
- 2 típus
    - Előltesztelő: 0..N darab futás
        - Megvizsgáljuk a feltételt, és ha nem teljesük, akkor be se lépünk a ciklusba
        - Ha teljesül, akkor addig ismétlünk, amíg a feltétel teljesül

- Hátultesztelő: 1..N darab futás
  - Mindenképpen egyszer belépünk, futtatjuk a ciklust, utána megvizsgáljuk a feltételt, és ha IGAZ, akkor folytatjuk a ciklust, ha HAMIS, akkor kilépünk belőle.
  - Ha folytatjuk, akkor addig megy, amíg IGAZ a feltétel

- VIGYÁZZUNK!
  - A feltételnek valamikor hamisnak kell lennie, különben végtelen ciklusba kerülünk, vagyis soha nem fog a program leállni, hanem ismétlődik folyamatosan a megadott parancs

```
// ELÖLTESZTELŐ
while (feltétel) {
        <utasítások>
}
// HÁTULTESZTELŐ
do {
        <utasítások>
} while (feltétel);
```

Feladat 7 – írjuk ki while ciklussal 1-től 10-ig a számokat
Szükséges ismeretek: kiíratás, változók, operátorok, while ciklus

```
do {
        document.writeln(i);
        i++;
} while (i <= 10);
```

Próbáljuk ki a kétféle ciklust, milyen feltételre fut az egyik 0-szor, a másik 1-szer? (Pl: i < 0)

Változók elnevezése

Ciklusváltozók: egybetűsek, tipikusan: **i,j,k**
Segédváltozó: ideiglenes használat (pl. két változó értékének megcserélése) – **tmp**
Egyéb változó: funkciója szerint szoktuk elnevezni

Összefoglalás

- **Kiíratás:** *document.writeln("szöveg")* → hogy legyen valami kimenetünk
- **Változók:**
  - egyszerű (pl: *var x = 5;*)
  - összetett (objektum) – tömbök (később lesz róla szó)
- **Operátorok**
  - műveletek, összehasonlítás
  - egy változóra használt – például: *!a*
  - két változóra használt – például: *a+b*
- **A program hogyan futhat le?**
  - **Szekvenciálisan**: egymás után írt utasítások, azok megadásának sorrendjében futnak le (lásd pl. Feladat 2 egymás utáni kiíratásai)
  - **Elágazások**: feltételektől függően döntünk arról, hogy mit hajtson végre a program
    - IF - ELSE IF – ELSE
    - SWITCH - CASE
  - **Ismétlések**: ciklusokkal egymás után többször hajtunk végre hasonló kódrészleteket (esetleg különböző paraméterekkel)
    - FOR ciklus
    - WHILE ciklus

Órai feladatok

1. Írjuk ki a számokat 10-től 5-ig, tehát visszafelé.
2. Írjunk ki egy 5x5-ös szorzótáblát táblázatos formában.
    a. Szükséges ismeretek: kiíratás, változók, operátorok, ciklusok, HTML

```
<script type="text/javascript">
document.write("<table>");
for (var i = 1; i <= 5; i++) {
        document.write("<tr>");
        for (var j = 1; j <= 5; j++) {
                document.write("<td>" + i*j + "</td>");
        }
        document.write("</tr>");
}
document.write("</table>");
</script>
```

Otthoni feladatok

Ajánlott példák

1. Írj **programot**, mely megcseréli két változó értékét.
2. Írj **programot**, mely kiírja a Fibonacci-sorozatot 1-től 100-ig.
    a. Fibonacci sorozat:
        i. az első két eleme 1
        ii. mindegyik elem az előtte lévő kettő összege, így a harmadik az első kettőé, vagyis 1 + 1 = 2
        iii. a negyedik: 1 + 2 = 3
        iv. az ötödik: 2 + 3 = 5
        v. és így tovább…
    b. Helyes megoldás: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89

További gyakorló példák

1. Írj **programot**, mely egy tetszőleges számot négyzetre emel.
3. Írj **programot**, ami meghatározza egy tetszőleges szám faktoriálisát!
    c. Faktoriális (!): 10! = 1*2*3*4*…*9*10 (1-től az adott számig a számok szorzata)
    d. Helyes megoldás 5-re: 120 (= 1*2*3*4*5)
2. Írj **programot**, mely a következőképpen viselkedik
    e. Kiírja a számokat 1-től 50-ig sorban, de
        i. ha 3-al osztható, akkor a szám helyett azt írja, hogy: ”X”
        ii. ha 5-el osztható, akkor a szám helyett azt írja, hogy: ”Y”
        iii. ha 3-al és 5-el is, akkor: ”JOKER”
3. Írj **programot**, mely kiírja a legkisebb 10 prímszámot.