JavaScript

Acest capitol este aproape imposibil de asimilat dacă nu ați parcurs capitolele anterioare. Este important să cunoaşteți deja noțiunile de Internet, WWW şi să ştiti să creați pagini Web simple în HTML. În afară de acestea, ar fi bine să fiți familiarizați cu un limbaj de programare cum ar fi: C, C++, C#, Visual Basic sau Java, dar nu este o condiție fără de care să nu puteți trece mai departe.

Corporația Netscape Communications a creat limbajul de scriptare "LiveScript" pentru Web designers şi dezvoltatori de aplicații Web. În decembrie 1995, LiveScript a fost redenumit "JavaScript" şi lansat ca parte componentă a browser-ului Netscape Navigator 2.0 de către Netscape Corporation şi Sun Microsystems

JavaScript a fost proiectat ca fiind un limbaj de scriptare simplu, pentru programatori începători care nu cunosc Java şi pentru Web designers. Acest limbaj de scriptare este simplu deoarece la scrierea unui script nu suntem obligați să declarăm variabilele înainte de a le utiliza şi nici să utilizăm un compilator. Browser-ul este cel care interpretează scriptul şi care ne arată dacă avem greşeli. Nu este nevoie să utilizăm un compilator Java sau C, nu este necesar să instalăm diverse soft-uri pe computerul nostru pentru a putea vizualiza rezultatul. Nu avem nevoie decât de un browser. Browser-ul care a interpretat pentru prima dată scripturile JavaScript este Netscape, după care au urmat Internet Explorer, Opera, Mozilla şi altele.

JavaScript este interpretat de către browser-e şi poate funcționa indiferent de platformă. Sistemele de operare pe care poate fi interpretat un script JavaScript sunt: Windows, UNIX, Macintosh.

Cu JavaScript putem crea pagini Web dinamice, interactive, pop-up, bare de navigare, putem trimite date pentru verificare prin Internet, putem controla applet-uri Java. JavaScript este un limbaj care poate fi utilizat atât pe parte de client cât şi pe parte de server.

JavaScript este un limbaj de scriptare bazat pe obiecte, nu orientat pe obiecte cum este limbajul de programare Java. JavaScript nu este Java, chiar dacă asemănarea numelor poate duce la această confuzie. Cu JavaScript nu putem crea applet, dar îl putem interpreta, după ce îl creăm cu Java.

JavaScript este un limbaj interpretat de către browser şi nu are nevoie să fie compilat înainte de a fi rulat. Browser-ul analizează fiecare element al paginii Web pe rând, împărțindu-l în componente mai mici.

În funcție de locul unde este plasat un script, scriptul poate fi pe parte de client sau pe parte de server. Dacă scriptul rulează în browser-ul clientului, atunci îl numim script pe parte de client, aşa cum sunt majoritatea script-urilor.

Ca răspuns la provocarea lansată de Netscape, au apărut şi alte limbaje de scriptare asemănătoare:

- Microsoft a lansat propriul limbaj de scriptare cu numele VBScript, iar în iulie 1996 a lansat Internet Explorer 3.0 cu JScript inclus;
- ECMA a lansat în iunie 1997 o versiune numită ECMAScript, care este de fapt standard internețional pentru JavaScript.

Rezultatul la care trebuie sa ajungă orice firmă ce doreşte să furnizeze un limbaj de scriptare este că acel limbaj trebuie să ruleze independent de platformă pentru a "trăi" în spațiul Web. Acest lucru duce la concluzia că, atât utilizatorii cât şi dezvoltatorii de soft trebuie să lucreze împreună pentru a găsi soluții utile tuturor.

Bineînțeles că JavaScript are şi dezavantaje, cum ar fi faptul că se pot crea bucle infinite cu ajutorul instrucțiunilor repetitive sau atacuri asupra computerelor care permit rularea diverselor scripturi, dar un utilizator Internet cu puțină experiență nu ar trebui să-şi facă probleme. Orice limbaj de programare poate fi folosit atât pentru a crea lucruri bune, cât şi pentru a crea lucruri rele. JavaScript este doar un limbaj de scriptare, adică un instrument! ☺

Script-uri JavaScript

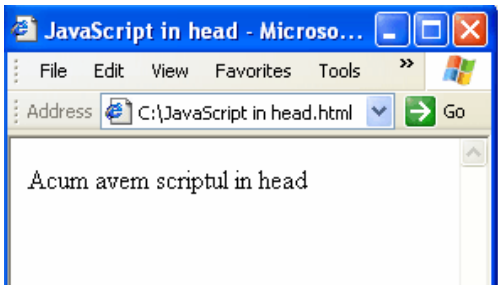
Ca şi un stil CSS, un script JavaScript poate fi scris atât în pagina Web, în antet sau în corpul documentului, cât şi în exteriorul ei, ca fişier separat. Fişierele care conțin scripturi JavaScript se salvează cu extensia .js.

Pentru scripturile simple, cel mai uşor este să le introducem în interiorul paginii Web, în antet şi mai rar în corpul paginii. Scripturile complexe, care este posibil să genereze erori la interpretare sau care se încarcă mai greu este indicat să le salvăm într-un fişier separat şi să le apelăm din interiorul paginii Web. Mai este însă un motiv pentru care scripturile ar trebui scrise separat şi anume posibilitatea ca acestea să fie modificate individual, fără să modificăm toată pagina Web sau tot site-ul care utilizează acest script.

Ca şi la HTML, scriptul JavaScript se introduce între două etichete **<script>** şi **</script>** la care ar trebui să adăugăm **language="JavaScript"** pentru ca browser-ul să ştie ce fel de script urmează.

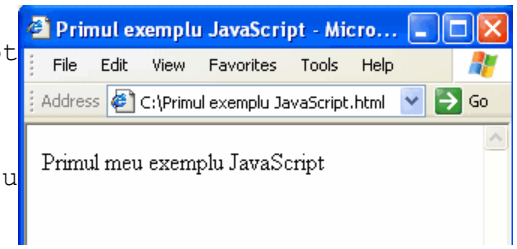
Exemplu În acest exemplu vom putea realiza cum se poate scrie un script JavaScript în *head* .

```
<html>
<head><title>JavaScript in head</title>
<script language="JavaScript">
document.write("Acum avem scriptul in head")
</script>
</head>
<body>
</body>
</html>
```



Exemplu În acest exemplu vom putea realiza cum se poate scrie un script JavaScript în *body.*

```
<html>
<head><title>Primul exemplu JavaScript
</title> </head>
<body>
<script language="JavaScript">
document.write("Primul meu exemplu
JavaScript")
</script>
</body>
</html>
```



Important! JavaScript este case-sensitive, adică este important să utilizăm numele variabilelor aşa cum le-am declarat, cu litere mari sau mici. Dacă am declarat o variabilă cu numele X1, nu o putem folosi ca x1, ci doar cu numele X1.

Comentariile din codul JavaScript se pot introduce:

- pe o singură linie cu ajutorul semnelor //
- pe mai multe linii cu ajutorul semnelor: /* */

Operatori JavaScript

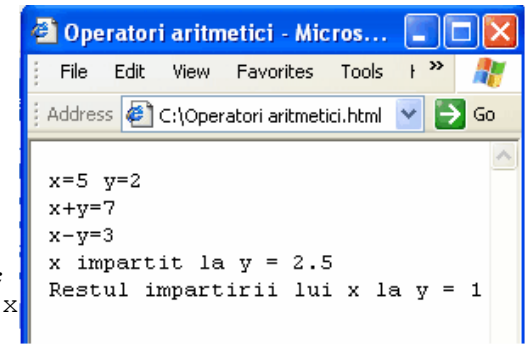
Operatori aritmetici

JavaScript conține operatorii aritmetici standard, întâlniți şi în limbajele de programare C şi C++. Din punctul de vedere al operanzilor la care se aplică, operatorii sunt de două feluri: unari şi binari. Operatorii unari se aplică unui singur operand, iar operatorii binari se aplică între doi operanzi.

Operator	Tip de operator	Descriere	Exemplu
+	unar	adunarea a două valori	+x sau a+b
-	unar	diferența a două valori	-x sau a-b
*	binar	înmulțirea a două valori	a*b
/	binar	împărțirea a două valori	a/b
%	binar	restul a două valori	a%b

Exemplu de utilizare a operatorilor aritmetici.

```
<html>
<head><title>Operatori aritmetici</title>
</head>
<body><pre>
<script  language="JavaScript">
var x=5, y=2;
document.writeln("x=5 y=2");
document.writeln("x+y=",x+y);
document.writeln("x-y=",x-y);
document.writeln("x impartit la y = ",x/y);
document.writeln("Restul  impartirii  lui  x
la y = ",x%y);
</script>
</pre>
</body></html>
```



OBS În exemplul de mai sus am declarat variabilele x şi y înainte de a le utiliza. Declararea variabilelor însă nu este obligatorie, acest limbaj ne permite să le utilizăm fără să le declarăm anterior.

Operatori de incrementare / decrementare

Operatorii se mai clasifică şi în funcție de operația pe care o efectuează şi anume în:

- **operatori de incrementare** – creşte valoarea variabilei cu o unitate
- **operatori de decrementare** – scade valoarea variabilei cu o unitate;

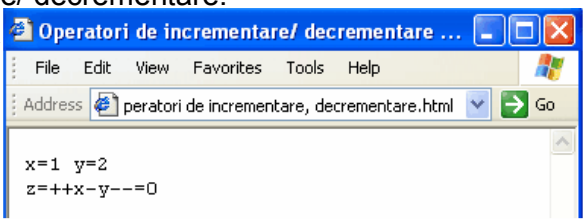
care la rândul lor sunt de două feluri:

- **prefixată** – operatorul se scrie înaintea operandului. Operația de incrementare (sau decrementare) se efectuează înainte de a se efectua operațiile expresiei în care se află.
  Exemplu dacă inițial a=1 şi x=++a; vom avea la final x=2 şi a=2, adică se incrementează variabila a cu o unitate înainte de a i se atribui variabilei x valoarea variabilei a.
- **postfixată** – operatorul se scrie după operand. Operația de incrementare (sau decrementare) se efectuează după ce se efectuează operațiile expresiei în care se află.
  Exemplu dacă inițial a=1 şi x=a++; vom avea la final x=1 şi a=2, adică se incrementează variabila a cu o unitate după ce i se va atribui variabilei x valoarea variabilei a.

Operator	Tip de operator	Descriere	Exemplu
++	incrementare	incrementare prefixată	++x sau x=x+1
		incrementare postfixată	x++ sau x=x+1
--	decrementare	decrementare prefixată	--x sau x=x-1
		decrementare postfixată	x- - sau x=x-1

Exemplu de utilizare a operatorilor de incrementare/ decrementare.

```
<html>
<head><title>Operatori de incrementare/
   decrementare </title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=1, y=2;
document.writeln("x=1 y=2");
document.writeln("z=++x-y--=",++x-y--);
</script>
</pre>
</body></html>
```



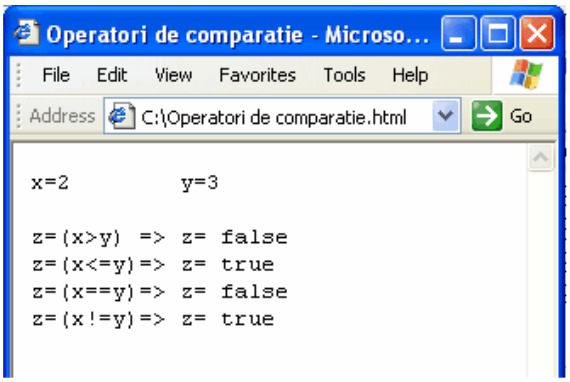
Operatori de comparație

Pentru a compara doi operanzi, utilizăm operatorii binari de comparație, descrişi în tabelul de mai jos.

Operator	Descriere	Exemplu pentru x=2 şi y=3	Rezultat
==	operator de egalitate	x==y	false
!=	operator de inegalitate	x!=y	true
>	mai mare decât...	x>y	false
<	mai mic decât...	x<y	true
>=	mai mare sau egal	x>=y	false
<=	mai mic sau egal	x<=y	true

Exemplu de utilizare a operatorilor de comparație.

```
<html>
<head>
<title>Operatori de comparatie</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=2,
y=3;
document.writeln("x=2         y=3\n");
document.writeln("z=(x>y)    =>    z=    ",
   z=(x>y));
document.writeln("z=(x<=y)=>     z=     ",
   z=(x<=y));
document.writeln("z=(x==y)=>     z=     ",
   z=(x==y));
document.writeln("z=(x!=y)=>     z=     ",
   z=(x!=y));
</script>
</pre>
</body>
</html>
```



OBS Codurile \n, \t se utilizează pentru a trece la o linie nouă, respectiv pentru a lăsa un spațiu, similar cu cel lăsat de tasta Tab. Aceste coduri se utilizează în cadrul unui script încadrat de etichetele <pre> şi </pre>.

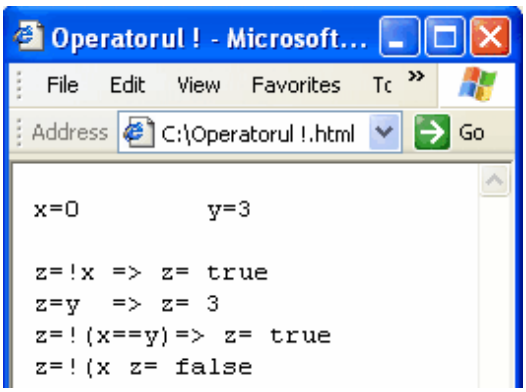
Operatori logici (booleeni)

Aceşti operatori se utilizează în cadrul unor expresii logice sau se aplică unor variabile, constante. Rezultatul unei expresii logice este tot o valoare logică. Valorile logice sunt true şi false. Zero (0) are valoarea logică false, iar orice număr diferit de zero are valoarea logică true.

Operator	Descriere	Exemplu pentru x=0 şi y=3	Rezultat
!	- operatorul logic NOT (negație logică) - operator unar - returnează o valoare logică diferită	!x !y !(x==y) !(x!=y) !(x>y) !(-25)	true false true false true false
&&	- operatorul logic AND (*şi* logic) - operator binar - returnează ***true*** dacă ambii operanzi (sau expresii) au valoarea ***true***, altfel returnează ***false.***	x&&y (!x)&&y 0&&1 (4<=2)&&(5>=4) (x==0)&&(y==3)	0 3 0 false true
||	- operatorul logic OR (*sau* logic) - operator binar - returnează ***false*** dacă ambii operanzi (sau expresii) au valoarea ***false***, altfel returnează ***true***.	x||y (!x)||y 0||1 (4<=2)||(5>=4) (x==0)||(y==3)	3 true 0 true true

Exemplu de utilizare a operatorului !

```
<html>
<head>
<title>Operatorul !</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=0, y=3;
document.writeln("x=0        y=3\n");
document.writeln("z=!x => z= ", z=!x);
document.writeln("z=y  => z= ", z=y);
document.writeln("z=!(x==y)=>     z=     ",
   z=!(x==y));
document.writeln("z=!(x<y)=>            z=
   ",z=!(x<y));
</script>
</pre>
</body>
</html>
```



&&	true	false
true	true	false
false	false	false

||	true	false
true	true	true
false	true	false

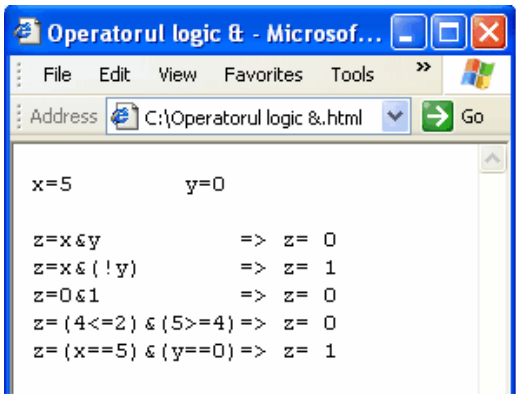
Operatori logici pe biți (bitwise)

Aceşti operatori sunt operatori binari şi acționează numai asupra operanzilor de tip întreg.

Operator	Descriere	Exemplu	Rezultat
&	- operatorul logic *şi pe biți* (bitwise and &) - returnează 1 dacă ambii operanzi sunt 1, altfel returnează 0.	0&0 1&0 0&1 1&1	0 0 0 1
|	- operatorul logic *sau pe biți* (bitwise or 1) - returnează 0 dacă ambii operanzi sunt 0, altfel returnează 1.	0|0 1|0 0|1 1|1	0 1 1 1
^	- operatorul logic *sau exclusiv pe biți* (bitwise xor ^) - returnează ^ dacă un singur operand este 1, altfel returnează 0.	0^0 1^0 0^1 1^1	0 1 1 0
<<	- operatorul logic de deplasare spre stânga a conținutului tuturor biților operandului din stânga sa, cu un număr de poziții egal cu valoarea reținută de al doilea operand. - Pozițiile rămase libere (în dreapta) vor reține valoarea 0.	***X=1010*** X<<1 X<<12 ***Y=1001*** Y<<0001 Y<<2 Y<<5	 2020 4136960 2002 4004 32032
>>	- operator logic de deplasare spre dreapta - deplasează spre dreapta conținutul tuturor biților operandului din stânga cu un număr de poziții egal cu valoarea reținută de al doilea operand.	***X=1010*** X>>1 X>>12 ***Y=1001*** Y>>0001 Y>>2 Y>>5	 505 0 500 250 31

Exemplu de utilizare a operatorului &.

```
<html>
<head>
<title>Operatorul logic &</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=5, y=0;
document.writeln("x=5         y=0\n");
document.writeln("z=x&y           => z= ",
   z=x&y);
document.writeln("z=x&(!y)        => z= ",
   z=x&(!y));
document.writeln("z=0&1           => z= ",
   z=0&1);
document.writeln("z=(4<=2)&(5>=4)=> z= ",
   z=(4<=2)&(5>=4));
document.writeln("z=(x==5)&(y==0)=> z= ",
   z=(x==5)&(y==0));
</script>
</pre></body>
</html>
```

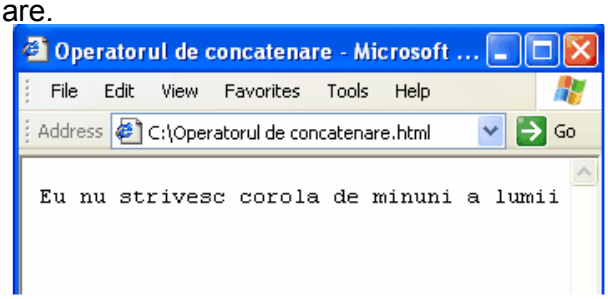


Operator pentru şiruri de caractere (string)

Operatorul de concatenare (+) este un operator binar cu ajutorul căruia se pot uni două şiruri de caractere.

Exemplu de utilizare a operatorului de concatenare.

```
<html>
<head><title>Operatorul de
concatenare</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x="Eu nu strivesc ";
    y="corola de minuni";
    z="a lumii";
document.writeln(x+y+" "+z);
</script>
</pre></body>
</html>
```



Operatori de atribuire

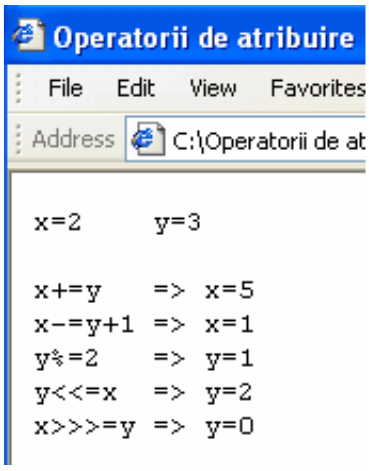
Operatorii de atribuire se utilizează între un operand pe care îl scriem în partea stângă şi o constantă, o variabilă sau o expresie în partea dreaptă a operatorului.

Nu se scriu expresii în partea stângă a unui operator de atribuire.

Operator	Exemplu prescurtat	Exemplu
=	x=3 z=2*x+y*x	x=3 z=2*x+y*x
+=	x+=1	x=x+1
-=	x-=1	x=x-1
*=	x*=3	x=x*3
/=	x/=3	x=x/3
%=	x%=3	x=x%3
<<=	x<<=3	x=x<<3
>>=	x>>=2	x=x>>2
>>>=	x>>>=5	x=x>>>5
&=	x&=6	x=x&6
^=	x^=2	x=x^2
|=	x|=y	x=x|y

Exemplu de utilizare a operatorilor de atribuire.

```
<html>
<head>
 <title>Operatorii de atribuire</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=2;
    y=3;
document.writeln("x=2    y=3\n");
document.writeln("x+=y   => x=",x+=y);
document.writeln("x-=y+1 => x=",x-=y+1);
document.writeln("y%=2   => y=",y%=2);
document.writeln("y<<=x  => y=",y<<=x);
document.writeln("x>>>=y => y=",x>>>=y);
</script></pre>
</body></html>
```



Operatorul condițional

Operatorul condițional se utilizează în cadrul expresiilor condiționale.

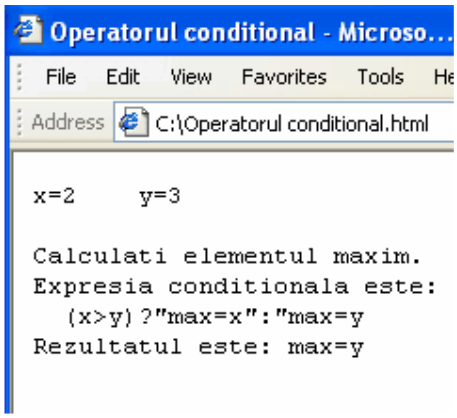
Forma generală:

(cond.)?expr.1:expr.2

Se evaluează condiția ***cond.***, iar dacă este adevărată sau diferită de 0, atunci se execută expresia ***expr.1***, altfel se execută expresia ***expr.2***.

Exemplu de utilizare a operatorului condițional.

```
<html>
<head><title>Operatorul conditional</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=5;    y=7;
document.writeln("x=2    y=3\n");
document.writeln("Calculati elementul maxim.");
document.writeln("Expresia conditionala este:");
document.writeln("  (x>y)?\"max=x\":\"max=y");
document.writeln("Rezultatul este:
",(x>y)?"max=x":"max=y");
</script></pre>
</body></html>
```



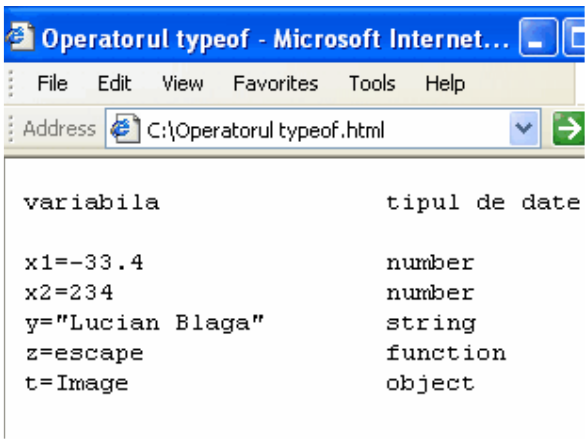
Operatorul typeof

Operatorul *typeof* returnează tipul de date conținut de operandul său.
Tipurile de date pe care le poate returna sunt:

- ***string*** – pentru şiruri de caractere
- ***number*** – pentru numere
- ***function*** – pentru funcțiile JavaScript
- ***object*** – pentru obiectele JavaScript

Exemplu de utilizare a operatorului *typeof*.

```
<html>
<head><title>Operatorul typeof</title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("variabila\t\ttipul de
   date\n");
var x1=-33.4;
document.writeln("x1=-33.4\t\t",typeof
   x1);
x2=234;
document.writeln("x2=234\t\t\t",typeof
   x2);
y="Lucian Blaga";
document.writeln("y=\"Lucian
   Blaga\"\t",typeof y);
z=escape;
document.writeln("z=escape\t\t",typeof z);
t=Image;
document.writeln("t=Image\t\t\t",typeof
   t);
</script>
</pre>
</body></html>
```



Instrucțiuni

În JavaScript instrucțiunile se clasifică în: instrucțiuni primitive, instrucțiuni condiționate (de decizie) şi instrucțiuni repetitive. Instrucțiunile reprezintă acțiunile ce trebuiesc executate pentru a putea obține anumite rezultate.

Instrucțiuni primitive

Instrucțiunile primitive se clasifică în:

- ***Instrucțiunea vidă*** **;**

Atunci când nu este necesară nici o prelucrare a datelor, putem utiliza instrucțiunea; Această instrucțiune nu returnează nici un rezultat, dar este necesară utilizarea ei.

- ***Instrucțiunea compusă***

Pentru cazul în care trebuiesc executate mai multe instrucțiuni împreună se utilizează instrucțiunea compusă:

```
{
  instrucțiune 1;
  instrucțiune 2;
  .....
  instrucțiune n;
}
```

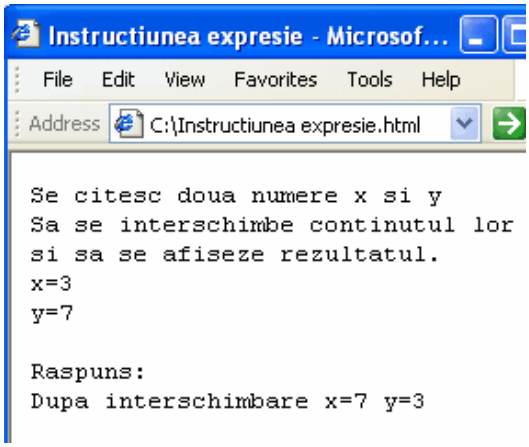
Instrucțiunea compusă se utilizează atunci când dorim să grupăm mai multe instrucțiuni, pentru a putea fi executate în ordine.

- ***Instrucțiunea expresie***

O instrucțiune expresie poate fi: o expresie, o atribuire sau apelul unei funcții.

Exemplu de utilizare a instrucțiunii expresie, în cadrul interschimbării conținutului a două variabile.

```
<html>
<head><title>Instructiunea expresie
      </title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citesc doua numere
   x si y\nSa se interschimbe continutul
   lor\nsi sa se afiseze rezultatul.");
x=eval(prompt("Dati x:"));
y=eval(prompt("Dati y:"));
document.writeln("x=",x,"\ny=",y);
document.writeln("\nRaspuns:");
z=x;  x=y;  y=z;
document.writeln("Dupa interschimbare
   x=",x," y=",y);
</script></pre>
</body>
</html>
```



OBS În acest exemplu am utilizat funcția ***eval***, ce ne permite evaluarea unor date citite de la tastatură prin intermediul unei căsuțe de dialog. Vom studia acestă funcție mai târziu.

Instrucțiuni de decizie

Instrucțiunile condiționate (de decizie) se clasifică în:

- ***Instrucțiunea If***

Se utilizează pentru a lua o decizie în funcție de o condiție dată.

Sintaxa pentru instrucțiunea de decizie simplă:

if (condiție) Principiul de execuție este următorul:

{ instrucțiune_1; } - se evaluează condiția;
else - dacă se îndeplineşte condiția (condiție), se execută instrucțiunea instrucțiune_1, altfel se execută instrucțiunea instrucțiune_2.
{ instrucțiune_2; }

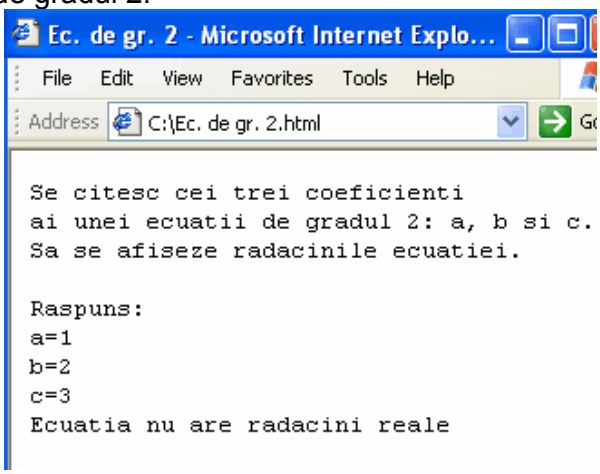
Sintaxa pentru instrucțiunea de decizie compusă:

if (condiție)
{ instrucțiune_01;
instrucțiune_02;
.....
instrucțiune_0n; }
else
{ instrucțiune_1;
instrucțiune_2;
.....
instrucțiune_n; }

Principiul de execuție este similar cu cel al instrucțiunii de decizie simplă.
- se evaluează condiția (cond.);
- dacă se îndeplineşte condiția (condiție), se execută instrucțiunile de la instrucțiune_01 până la instrucțiune_0n;
- dacă nu se îndeplineşte condiția (condondiție), se execută instrucțiunile de la instructiune_1 până la instructiune_n.

Exemplu de utilizare a instrucțiunii if, în ecuația de gradul 2.

```
<html>
<head><title>
      Ec. de gr. 2</title></head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citesc cei trei
   coeficienti\nai unei ecuatii de
   gradul 2: a, b si c.\nSa se afiseze
   radacinile ecuatiei.");
a=eval(prompt("Dati a:"));
b=eval(prompt("Dati b:"));
c=eval(prompt("Dati c:"));
document.writeln("\nRaspuns:");
document.writeln("a=",a,"\nb=",b,"\nc="
   ,c);
if (a==0)
   {x=-c/b;
    document.writeln("Ecuatia este de
   gradul I\nSolutia ecuatiei este
   x=",x);}
else
 {d=b*b-4*a*c;
   if (d<0)
    {document.writeln("Ecuatia nu are
   radacini reale");}
   else {x1=(-b+Math.sqrt(d))/(2*a);
         x2=(-b-Math.sqrt(d))/(2*a);
         document.writeln("x1=",x1);
         document.writeln("x2=",x1);}}
</script></pre>
</body></html>
```



OBS În acest exemplu am utilizat metoda sqrt() ce returnează rădăcina pătrată a unei valori numerice. Aceasta este o metodă a obiectului *Math*.

- ***Instrucțiunea switch..case***

Instrucțiunea de decizie multiplă se utilizează atunci când, în urma evaluării unei expresii, trebuie să optăm între mai multe cazuri date. Acelaşi lucru îl putem realiza şi cu ajutorul instrucțiunii if. Diferența constă în faptul că dacă avem de optat între mai mult de

două cazuri, folosirea instrucțiunii ***if*** devine greoaie, caz în care putem apela la instrucțiunea ***switch***.

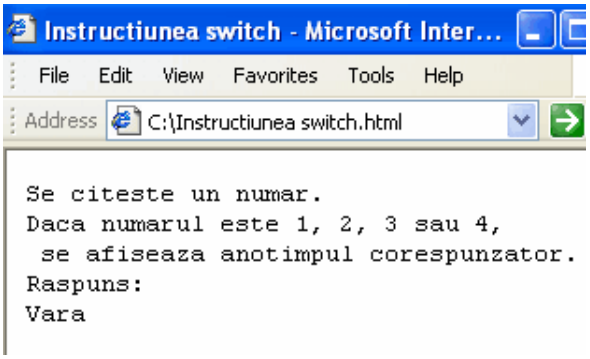
Sintaxa pentru instrucțiunea de decizie multiplă simplă:

switch (expresie)
{
case valoare_1;
instrucțiune_1; break;
case valoare_2;
instrucțiune_2; break;
……..
case valoare_n;
instrucțiune_n; break;
[default: instrucțiune_n+1];
}

Principiul de execuție este următorul:
- se evaluează expresia;
- dacă valoarea expresiei este egală cu *valoare_1*, se execută *instrucțiune_1*
- dacă valoarea expresiei nu este egală cu *valoare_1*, se verifică dacă este egală cu *valoare_2*, pentru a se executa *instrucțiune_2*,
- dacă valoarea expresiei nu este egală cu *valoare_2* se repetă paşii până la *valoare_n*
- dacă valoarea expresie nu este egală cu nici o valoare dintre: *valoare_1...valoare_n*, se execută *instrucțiune_n+1*

Exemplu de utilizare a instrucțiunii ***switch***.

```
<html>
<head>
 <title>   Instructiunea switch
   </title>
</head>
<body>
 <pre>
<script  language="JavaScript">
document.writeln("Se citeste un
   numar.\nDaca numarul este 1, 2, 3
   sau 4,\n se afiseaza anotimpul
   corespunzator.");
x=eval(prompt("Dati numarul corespunzator anotimpului:"));
document.writeln("Raspuns:");
switch(x)
 {
  case 1: document.writeln("Primavara"); break;
  case 2: document.writeln("Vara");
    break;
  case 3: document.writeln("Toamna"); break;
  case 4: document.writeln("Iarna"); break;
  default: document.writeln("Dati un numar intre 1 si 4!");
 }
</script>
</pre>
</body>
</html>
```



Instrucțiuni repetitive

Instrucțiunile repetitive se clasifică în:
- instrucțiuni cu număr cunoscut de paşi: ***for***
- instrucțiuni cu număr necunoscut de paşi: ***while şi do..while***

➢ ***Instrucțiunea for***

Sintaxa pentru instrucțiunea repetitivă cu număr cunoscut de paşi ***for***:

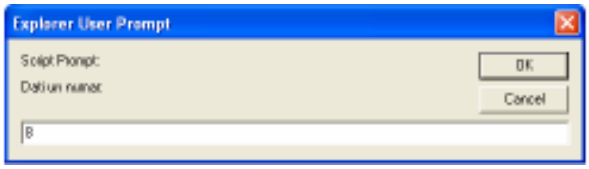
for (valoare1; condiție;pas) Principiul de execuție este următorul:

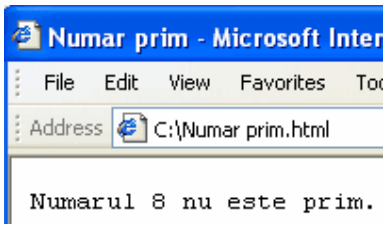
instrucțiune

- se evaluează condiția ***condiție***;
- dacă se îndeplineşte condiția ***condiție*** , se incrementează valoarea inițială ***valoare1*** cu valoarea pasului ***pas.*** şi se repetă execuția instrucțiunii până când nu se mai îndeplineşte condiția.

Exemplu Să se afişeze dacă un număr citit de la tastatură este prim sau nu. Am utilizat funcția *sqrt()* ce returnează radicalul unui număr.

```
<html>
<head>
 <title>Numar prim</title></head>
<body>
<pre>
<script  language="JavaScript">
n=eval(prompt("Dati un numar:"));
prim=1;
for(i=2;i<=Math.sqrt(n);i++)
 if (n%i==0)
   prim=0;
if (prim==1)
  document.writeln("Numarul ",n," este
    prim.");
else
  document.writeln("Numarul ",n," nu
    este prim.");
</script></pre>
</body></html>
```





➢ ***Instrucțiunea while***

Sintaxa pentru instrucțiunea repetitivă cu test inițial şi cu număr necunoscut de paşi ***while***:

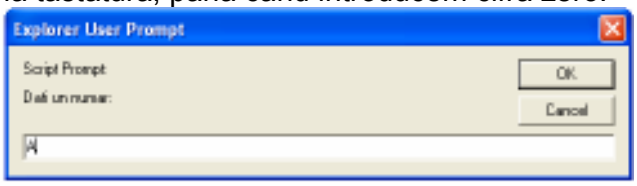
while (condiție)
instrucțiune

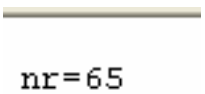
Principiul de execuție este următorul:

- se evaluează condiția ***condiție***;
- cât timp se îndeplineşte condiția ***condiție***, se execută instrucțiunea ***instrucțiune***;
- când valoarea condiției devine 0 (zero), adică fals, se trece la următoarea instrucțiune.

Exemplu de utilizare a instrucțiunii ***while***. Să se afişeze valoarea corespunzătoare *Unicode* pentru toate numerele introduse de la tastatură, până când introducem cifra zero.

```
<html>
<head>
 <title>Instructiunea while </title>
</head>
<body>
<pre>
<script  language="JavaScript">
n=prompt("Dati un numar:","");
while (n!=0)
 {
  document.writeln("nr=",
    n.charCodeAt(0));
  n=prompt("Dati un numar:","");
 }
</script>
</pre>
```





```
</body>
</html>
```

- ***Instrucțiunea do...while***

Sintaxa pentru instrucțiunea repetitivă cu test final şi cu număr necunoscut de paşi ***do...while***:

```
do
 {
   instrucțiune 1;
   instrucțiune 2;
   ……..
   instrucțiune n;
 }
while  ( condiție)
```

Principiul de execuție este următorul:
- se execută instrucțiunile;
- se evaluează condiția ***condiție***;
- dacă valoarea rezultată este diferită de zero (adevărat), se repetă execuția instrucțiunii până când aceasta devine zero (fals);
- se trece la instrucțiunea următoare

Exemplu Se citeşte un număr natural pozitiv. Să se descompună în factori primi şi să se afişeze rezultatul.

```
<html>
<head>
 <title>Factori primi</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var i=2;
n=eval(prompt("Dati un numar:"));
do
  {
    putere=0;
    while (n%i==0)
     {
       putere=putere+1;
       n=Math.floor(n/i);
     }
  if (putere!=0)
    document.writeln(i," la puterea
",putere);
    i++;
  }
while (n!=1)
</script></pre>
</body>
</html>
```