Pagini Web cu JavaSCRIPT

Diana Elena Diaconu

PREFAȚĂ

Limbajele de programare au apărut cu scopul de a *procesa informațiile* prin intermediul calculatorului. Teoria limbajelor formale, teoria gramaticilor şi automatelor, au făcut posibilă conceperea şi realizarea compilatoarelor ce se află la baza utilizării efective a limbajelor de programare în activitatea de procesare a informațiilor folosind calculatorul. De asemenea, multe limbaje de programare sunt utilizate prin intermediul *interpretoarelor* şi *emulatoarelor*. Spre deosebire de interpretor, un emulator nu transformă codul sursă într-un cod maşină, ci emulează execuția unui program în cadrul unei maşini virtuale. De exemplu, un program sursă **Java** (fişier text cu extensia *.java*) nu se transformă ăn cod obiect, ci în cod de octeți (*bytecodes*), salvat într-un fişier cu extensia *.class*. Codul de octeți (rezultatul compilării) generat de compilatorul Java (**javac**) este interpretat de **maşina virtuală Java** (*JVM – Java Virtual Machine*), care converteşte codul de octeți în cod maşină. În prezent, pentru majoritatea tipurilor de calculatoare există maşini virtuale Java, ce pot fi descărcate gratuit de la adresa Web www.java.sun.com. De fapt, este vorba de **tehnologia Java** ce oferă un *mediu de programare* performant destinat dezvoltării aplicațiilor Java distribuite şi independente de platforme (sisteme de operare). Prin utilizarea emulatorului (maşina virtuală Java) o aplicație Java se poate executa pe orice platformă pentru care există un emulator Java.

Sistemele de operare (numite mai recent platforme: **Windows, Unix, Linux** etc.) sunt proceduri şi programe ce reprezintă interfața între un utilizator şi *resursele* (memorie, dispozitive) unui calculator. În fapt, procesarea informațiilor se realizează prin intermediul programelor scrise într-un limbaj de programare şi care se execută prin intermediul sistemului de operare instalat pe un calculator. Complexitatea aplicațiilor de utilizare a calculatorului în diverse domenii de activitate, a determinat perfecționarea atât

a sistemelor de operare, cât şi a limbajelor de programare. Astfel, prin efortul comun al cercetătorilor (vezi serviciul **WWW**, limbajul **Prolog**), programatorilor şi inginerilor (vezi limbajele **C**, **C++**, **Java**), profesorilor (vezi **LaTex**), studenților (vezi **Linux**) etc., au fost concepute şi elaborate noi *sisteme de operare*, *noi limbaje de programare*, *noi tehnologii*. Aşa se explică rezultatele obținute în domeniul **rețelelor de calculatoare** şi în dezvoltarea **sistemului Internet**. Dezvoltarea şi răspândirea diverselor tehnologii în conceperea şi elaborarea aplicațiilor destinate procesării informațiilor, precum şi utilizarea diverselor *metode de comunicare a informațiilor*, au determinat o revoluție în domeniul calculatoarelor cunoscută sub numele de " *Tehnologia informației si comunicațiilor*".

Dacă în anii '70 inventarea şi utilizarea *microprocesorului* au însemnat o revoluție în domeniul *arhitecturii calculatoarelor*, în anii '90 a fost o adevărată revoluție atât în domeniul *rețelelor de calculatoare*, cât şi în domeniile *limbajelor de programare* (**Java** şi **JavaScript**) şi *sistemelor de operare* (**Linux**). Astfel, au apărut **tehnologiile Web**. Trebuie menționate dezvoltarea şi evoluția limbajului **C++** care în anii '80 a implementat şi dezvoltat modelul orientat spre obiecte (*modelul programării obiectuale* are rădăcini în limbajele **SmallTalk**, **Lisp** etc.) şi programarea orientată spre obiecte (*OOP-Object Oriented Programming*). La începutul anilor '90 a apărut limbajul **HTML** (Hypertext Markup Language) ce a determinat răspândirea **paginilor Web statice** şi dezvoltarea explozivă a sistemului **WWW** (*World Wide Web*). Necesitatea elaborării **paginilor Web dinamice** a determinat apariția diverselor tehnologii: **JavaScript**, **JavaServer Pages** (**JSP**), **VBScript**, **PHP**, **ASP**, **Macromedia Dreamweaver** etc., tehnologii ce unele sunt destinate pentru *aplicații server*, iar altele pentru *aplicații client*.

Profesoara *Diana Elena Diaconu* a elaborat prezenta carte ca rezultat al activității desfăşurate cu scopul perfecționării în domeniul *tehnologiilor Web*. Lucrarea prezentă este propusă ca manual opțional pentru **Informatică**. Conținutul cărții este prezentat într-o formă clară, concisă şi cu foarte multe exemple pentru asimilarea şi testarea cunoştințelor. Cartea de față îşi propune descrierea **tehnologiei JavaScript** pentru elaborarea **paginilor Web dinamice**.

Cartea este destinată tuturor celor care posedă cunoştințe de nivel mediu în domeniul limbajelor de programare. De asemenea, se adresează în special celor care doresc să se perfecționeze în utilizarea tehnologiile Web. Elevii, studenții şi cadrele didactice au prin această carte un instrument util în obținerea de competențe în utilizarea tehnologiei JavaScript privind dezvoltarea de pagini Web dinamice.

Septembrie 2006

Conf. univ. dr. Marin Vlada,
Universitatea din Bucureşti
marinvlada@gmail.com

“If I have been able to see farther than others,
it was because I stood on the shoulders of giants.”

Isaac Newton

Introducere

Obiectivul principal al acestei cărți este acela de a dezvolta la elevi deprinderea de a crea pagini Web atractive, cu conținut dinamic, având la dispoziție resurse limitate. Pentru a putea realiza exemplele din această carte, este nevoie de un editor de texte, cum ar fi Notepad şi un browser, precum Internet Explorer, Mozilla Firefox, Netscape, Opera.

Conținutul acestei cărți poate asigura suportul pentru un program şcolar de o oră pe săptămână, timp de 36 de săptămâni. La începutul fiecărui capitol / subcapitol sunt scrise principalele obiective pe care le-am gândit pentru predarea materialului respectiv iar la sfârşitul fiecărui capitol se află câteva întrebări cu care putem asigura evaluarea elevilor sau pe care le putem da ca temă.

Capitolele cărții au următorul conținut:

Capitolul 1 cuprinde un scurt istoric al Internetului, urmat de o trecere în revistă a rețelelor de calculatoare. După o clasificare a rețelelor de calculatoare în funcție de dimensiunile sale, am continuat cu analizarea celei mai extinse rețele şi anume Internetul. Aici am insistat asupra modalităților de conectare la Internet şi asupra celor mai importante grupări care influențează Internetul.

Capitolul 2 este dedicat noțiunilor de WWW (World Wide Web), URL (Uniform Resource Locator) şi HTTP (Hypertext Transfer Protocol). Tot în acest capitol, am prezentat pe scurt cum funcționează paginile Web.

Începând cu **capitolul 3**, am insistat mai mult pe aplicații scurte cu ajutorul cărora am prezentat cele mai importante elemente HTML.

Capitolul 4 conține aplicații ce au ca scop utilitatea stilurilor CSS într-o pagină Web.

În **capitolul 5** mi-am propus prezentarea pe scurt a limbajului de scriptare JavaScript printr-o scurtă introducere, după care am rezentat cu ajutorul exemplelor, operatorii şi instrucțiunile JavaScript.

Capitolul 6 prezintă modalitatea de creare şi utilizare a funcțiilor, atât a celor definite de noi, cât şi a celor predefinite.

Capitolul 7 conține evenimentele şi tratarea evenimentelor declanşate de către utilizatori şi interpretate cu ajutorul funcțiilor JavaScript.

Capitolul 8 este cel mai amplu şi conține pe lângă obiectele JavaScript şi o scurtă descriere a proprietăților, metodelor şi evenimentelor ce le aparțin.

Capitolul 1.

Internet

Obiective:
- dezvoltarea capacității de a înțelege transformările prin care a trecut Internetul.
- manifestarea unor atitudini favorabile față de cunoaştere şi de ştiință în general.

1.1. Scurt istoric

Internetul a apărut ca o necesitate a oamenilor de a comunica la distanțe mari.

Printre primele încercări ale oamenilor de a comunica cu ajutorul aparatelor ar putea fi considerat telegraful, inventat în 1840. Telegraful a fost primul instrument care a utilizat cablul pentru a transmite semnale pe distanțe mari.

În 1962 au apărut primele discuții despre conceptul de "Rețea galactică" în care se imagina un set de computere interconectate prin intermediul cărora oricine putea avea acces rapid la datele şi programele celorlalți. În esență, acest concept este foarte asemănător cu Internetul de astăzi.

Pentru a explora acest concept, în 1965 a fost creată experimental, prima rețea prin linie telefonică de viteză mică (dial-up), creând astfel prima rețea de computere, dispusă pe o suprafață întinsă.

În 1968 DARPA (Defense Advanced Research Projects Agency) împreună cu BBN (Bolt, Beranek & Newman) au dezvoltat conceptul, realizând prima rețea de computere, numită ARPAnet. ARPAnet a fost o rețea națională, construită la nivelul Statelor Unite ce se baza pe schimbul de pachete de date.

IMPs (Interface Message Processors) reprezintă interfața unui sistem independent care poate fi utilizată de orice computer din rețeaua ARPAnet. O parte din software-ul pentru rețeaua ARPAnet se află în IMPs şi o parte în host. La început, în rețea, mesajele se transmiteau de la host la host, host-urile având drepturi egale şi se utiliza o linie telefonică de 56 kbps.

Dezvoltând teoria schimbului de pachete între două host-uri, ARPA (Advanced Research Projects Agency - Agenția pentru Proiecte de Cercetare Avansată) a decis ca în 1969 U.C.L.A. (University of California Los Angeles) să devină primul nod de rețea. De acum se puteau conecta cercetătorii şi centrele de cercetare implicate, utilizând IMPs.

Al doilea nod a fost creat la Stanford Research Institute (S.R.I.) şi includea funcții cum ar fi menținerea unor tabele cu numele host-urilor pentru a realiza o mai bună adresare. Prima legătură ARPAnet a fost stabilită pe 21.11.1969 între U.C.L.A. şi S.R.I.

În 1970, au urmat încă două noduri la University of California Santa Barbara şi University of Utah. Aceste noduri au declanşat de fapt apariția Internetului.

Obiective:
- dezvoltarea capacității de a înțelege conceptul de rețea de calculatoare
- conştientizarea diferențelor dintre o rețea de calculatoare şi Internet.

1.2. Rețea de calculatoare

Definiție

*O **rețea de calculatoare** reprezintă un ansamblu de calculatoare autonome, interconectate, ce folosesc o singură tehnologie pentru a putea realiza un schimb de date şi a folosi în comun resursele.*

Într-o rețea, calculatoarele pot fi conectate prin cablu de cupru, fibră optică, radiații infraroşii, microunde sau sateliți de comunicații. Datorită distanței dintre două sau mai multe computere conectate, rețelele au fost clasificate inițial în: rețele locale şi rețele extinse.

Progresul tehnologiei şi al echipamentelor de rețea, a dus ulterior, la o clasificare mai detaliată a rețelelor în funcție de dimensiunile sale şi anume:

- **rețea de birou** – Desk Area Network – **DAN** în care fiecare componentă a computerului aflat pe birou, cum ar fi: ecranul monitorului, unitățile de CD-ROM, CD-Writer, DVD-ROM, DVD-Writer, Combo Drive, dispozitivele periferice precum: WebCam, imprimanta, scanner-ul, pot fi accesibile din rețea. O rețea de felul acesta se realizează pentru a avea acces la toate resursele ce pot fi solicitate de o aplicație de rețea.
- **rețea personală** – Home Phoneline Networking Alliance – **HPNA** utilizează o tehnologie relativ nouă ce permite construirea unei rețele private, utilizând firul de telefon existent. Accesul la Internet se realizează printr-un singur computer, ce permite conectarea la Internet şi a altor computere fără a fi necesar un router.
- **rețea locală** – Local Area Network – **LAN** se extinde pe o suprafață mai mică de 1 Kilometru şi poate fi compusă din: computere, plăci de rețea, dispozitive periferice, dispozitive de rețea, dispozitive media. Rețelele locale permit partajarea eficientă atât a resurselor software cât şi a celor hardware.
- **rețea metropolitană** – Metropolitan Area Network – **MAN** se poate întinde pe suprafața unui oraş, putându-se extinde pe o suprafață de zeci de kilometri.
- **rețea extinsă** – Wide Area Network – **WAN** se extinde pe o suprafață geografică mare (mai mare decât MAN) şi realizează interconectarea LAN-urilor. WAN-urile permit partajarea resurselor pe distanțe foarte mari, stabilirea comunicațiilor în timp real, permit servicii de poştă electronică (e-mail), comerț electronic (e-commerce), transfer de fişiere (file transfer).
- **Internet**ul nu reprezintă o singură rețea de calculatoare, ci o rețea de rețele răspândite pe tot globul.

Ştim că într-o rețea, calculatoarele folosesc o singură tehnologie pentru a comunica între ele. Daca însă avem de-a face cu două sau mai multe rețele care utilizează tehnologii diferite, de cele mai multe ori incompatibile, atunci comunicarea între calculatoarele acestor rețele se poate realiza cu ajutorul aşa numitor gateways (porți).

Un gateway (poartă) este un calculator cu ajutorul căruia se poate realiza conectarea între două sau mai multe rețele de calculatoare şi asigură conversiile necesare atât în termeni de hardware cât şi de software.

Astfel de rețele interconectate formează o inter-rețea sau internet.

O altă tehnologie realizată în scopul conectării computerelor într-o rețea de mare viteză, poartă numele de **ATM** – Asynchronous Transfer Mode – Mod de Transfer Asincron. ATM este o rețea cu comutație de pachete orientată pe conexiune ce permite o rată de transfer a datelor mai mare decât tehnologia Internet şi poate funcționa indiferent de mediul fizic. Dezavantajul acestei tehnologii este costul mai mare decât al tehnologiei Internet.

Obiective:
- dezvoltarea capacității de a înțelege conceptul de rețea de calculatoare
- conştientizarea diferențelor dintre o rețea de calculatoare şi Internet.

1.3. Ce este Internetul

Definiție

Internet-ul *este o rețea de rețele extinsă pe tot globul, alăturând multe guverne, universități, calculatoare personale şi furnizând o infrastructură pentru utilizarea documentelor de tip hipertext, serviciilor de e-mail, transfer de fişiere şi alte resurse computaționale. Această vastă rețea de rețele de computere, acționează ca o singură şi uriaşă rețea pentru a transporta date şi mesaje din orice colț al lumii, de-a lungul unor distanțe mari.*

Internetul a revoluționat lumea prin posibilitățile de comunicare şi diseminare a informațiilor, ca mediu pentru colaborare şi interacțiune între computere individuale indiferent de poziția geografică.

Caracteristici:

- Cea mai mare rețea din lume;
- Utilizează protocoalele TCP/IP şi schimbul de pachete;
- Rulează în orice substrat de comunicare.

Cum te poți conecta la Internet?

Pentru a avea acces la această vastă rețea de rețele şi la resursele de care dispune, utilizatorii trebuie să apeleze la un furnizor de servicii Internet (Internet Service Provider ISP), numit şi provider Internet. Un ISP este o organizație sau o firmă ce poate oferi utilizatorilor acces la Internet şi la alte servicii înrudite cum ar fi: Internet transit, înregistrare pentru nume de domeniu, Web hosting, conexiune prin dial-up, acces prin linie închiriată (leased line) şi colocație.

Internet transit furnizează o conexiune dedicată la Internet ce permite o rată foarte mare de transfer a datelor. Acest tip de conexiune se utilizează atunci când se crează o rețea privată ce conține mai multe computere şi toată rețeaua utilizează aceeaşi conexiune la Internet.

Nume de domeniu sau Domain Name System (DNS) permite clienților şi gazdelor Internet să se adreseze unii celorlalți utilizând nume în loc de adresă IP. Exemple de nume de domeniu: www.yahoo.com, www.einformatica.ro. Exemple de adrese IP: 193.226.18.252, 68.142.226.32.

O **adresă IP** reprezintă adresa de identificare în rețeaua Internet, a unui computer sau dispozitiv. Protocolul de bază din Internet care stabileşte modalitatea în care se realizează transferul datelor între două sau mai multe computere este IP sau Internet Protocol. În momentul de față se utilizează versiunea 4 pentru protocolul Internet, numită şi IPv4. O adresă IPv4 o putem scrie ca patru grupe de numere în baza 10 între 0 şi 255 despărțite prin puncte sau patru grupe de numere în baza 2, de la 0.0.0.0 până la 11111111.11111111.11111111.11111111. Numărul total de adrese IPv4 diferite este de 2^{32}.

Datorită dezvoltării rapide a Internetului, cererea de adrese IP unice este deosebit de mare, fapt ce poate duce la epuizarea adreselor IP unice existente. Pentru a rezolva diferența dintre cererea şi oferta de adrese IP, în prezent se dezvoltă şi se îmbunătățeşte o nouă versiune a protocolului Internet, numită IPv6. IPv6 permite o adresare pe 128 biți față de 32 de biți la IPv4 şi poate furniza un număr de cel puțin $3,4*10^{38}$ adrese IP unice.

Web hosting sau găzduire Web presupune furnizarea de către un ISP a tuturor resurselor hardware şi software de care este nevoie pentru oferirea serviciului de găzduire a unui site Web sau a unor informații în format electronic şi asigurarea conexiunii acestuia la

Internet. Web hosts pot furniza spațiu pe server-ul sau server-ele sale şi conexiune la Internet pentru servere. Odată găzduite, fişierele vor putea fi accesate prin Web.

Conexiunea prin dial-up reprezintă conexiunea la Internet printr-o linie telefonică existentă.

Modem-ul este un dispozitiv ce realizează **mo**delarea şi **dem**odelarea semnalului, adică transformarea semnalului digital utilizat de către computer în semnal analog ce se transmite prin linia telefonică şi invers. Sistemul de telefonie lucrează cu impulsuri de curent continuu care variază în frecvență şi putere, iar computerul lucrează cu date în format digital adică 0 sau 1.

Linie închiriată sau dedicată - leased line - reprezintă o linie telefonică ce asigură o conexiune permanentă între două locații. Plata pentru linia închiriată se face în rate lunare fixe, nu în funcție de apelurile efectuate, ca în cazul conexiunii prin dial-up.

Colocație reprezintă închirierea unui spațiu special amenajat, pentru funcționarea în condiții optime a propriului echipament (server Web...) care este conectat la Internet.

Cine influențează Internetul?

Multe grupuri influențează dezvoltarea Internetului, ajutând la stabilirea unor standarde şi educând oamenii să-l utilizeze cât mai corect. Printre cele mai importante grupări se numără:

- **Internet Society** (ISOC) – o societate internațională non-profit ce se ocupă cu promovarea şi dezvoltarea Internetului. Pentru mai multe informații, puteți apela la varianta electronică: http://www.isoc.org
- **Internet Engineering Task Force** (IETF) – o comunitate internațională deschisă de operatori, proiectanți de rețele, cercetători preocupați de evoluția arhitecturii şi protocoalele TCP/IP ale Internetului (http://www.ietf.org). În concluzie, sunt interesați ca Internetul să funcționeze mai bine.
- **World Wide Web Consortium** (W3C) – consorțiu internațional ce perfecționează standardele Web pentru evoluția părții Internetului care se dezvoltă cel mai rapid: World Wide Web (http://www.w3.org). W3C este interesat ca Web să funcționeze la întregul potențial.

Evaluare

1. Cum se numeşte prima rețea de calculatoare şi când a fost ea creată?
2. Care este diferența dintre o rețea de calculatoare şi Internet?
3. Cum se realizează conectarea între computerele dintr-o rețea?
4. Ce este Internetul?
5. Câte noduri avea prima rețea de calculatoare?
6. Ce este un ISP şi ce servicii poate oferi?

Răspundeți prin alegerea variantei sau a variantelor corecte:

7. O adresă IPv4 o putem scrie sub forma:
   a) 265.23.12.0; b) 127.0.0.0; c) 0.0.14.230.22; d) 192.168.12.13;
8. Cine controlează Internetul?
   a) U.S.A.; b) Uniunea Europeană; c) Internet Society; d) nimeni

Capitolul 2.

W.W.W.

Obiective:
- să înțeleagă noțiunile de WWW, URL şi HTTP
- să înțeleagă paşii care se parcurg pentru a se afişa o pagină Web.

2.1. World Wide Web

Dezvoltarea rapidă a Internetului duce la dezvoltarea celei mai utilizate laturi ale ei şi anume: World Wide Web-ul.

Definiție

WWW *(World Wide Web) reprezintă o rețea care utilizează protocolul HTTP în scopul de a stabili legături între documente aflate în diverse puncte din Internet. În acelaşi timp este utilizat ca resursă pentru servicii, informații, publicații electronice.*

Web-ul funcționează după modelul client-server. Pe server rulează o aplicație ce conține site-uri Web şi permite utilizatorilor client să le acceseze cu ajutorul unor aplicații, numite browser-e ce se află pe computerul client. Legătura dintre aplicația client şi aplicația server se poate realiza numai după ce pe ambele computere se realizează o conexiune la Internet. Server-ul ce are ca rol posibilitatea de accesare on-line a site-urilor Web este cunoscut sub numele de server Web.

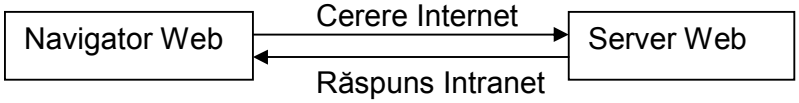


Figura 1.

În Figura 1. putem vedea care sunt paşii ce se fac pentru a fi afişată o pagină Web şi anume:

- utilizatorul scrie în bara de adrese a navigatorului (browser Web) numele site-ului, de exemplu: http://www.yahoo.com
- navigatorul Web face o cerere către Internet
- ruterele Internetului examinează cererea pentru a determina cărui server să-i fie transmisă
- ruterele transmit cererea către server-ul Web al cărui nume se află imediat în dreapta expresiei http://
- server-ul Web acceptă URL-ul şi decide dacă şi ce anume va returna ca răspuns către browser
- server-ul interpretează, execută fişierul cerut şi îl trimite către browser-ul Web, posibil modificat sau generat.
- browser-ul Web primeşte fişierul şi îl afişează pentru a putea fi văzut de către utilizator

Actualul preşedinte al World Wide Web Consortium, ***Tim Berners-Lee***, a inventat WWW în 1989, iar în 1990 a scris primul browser.

Ce este URL?

U.R.L. sau Uniform Resource Locator este o adresă Web. Ea este compusă din patru părți: numele protocolului (Ex: HTTP, scriem http://), locația site-ului (Ex: WWW), numele organizației care ține site-ul (Ex: yahoo), sufixul care identifică tipul organizației (Ex: .com). Pentru exemplul dat, introducem adresa: *http://www.yahoo.com*

Dar HTTP?

H.T.T.P. sau Hypertext Transfer Protocol (Protocol de Transfer Hipertext) este un protocol Internet care defineşte modul în care comunică browser-ele Web şi serverele şi modalitatea de transferare a documentelor HTML.

Obiective:
- să înțeleagă noțiunile de hipertext şi hipermedia
- să înțeleagă conceptul de pagină Web.

2.2. Cum funcționează paginile Web

Cele mai fascinante lucruri pe care un utilizator Internet le descoperă sunt paginile Web care conțin text, imagini, sunet, animații şi alte elemente multimedia cu care interacționează. Pentru a trece de la o pagină la alta, se utilizează legături de tip **hipertext** sau **hipermedia**.

Ca să înțelegem mai bine succesul conceptului de hipertext, ne imaginăm care este diferența dintre un text scris liniar, ca o carte şi un text cu diferite legături care ne trimit la alte pagini. Informația scrisă într-un text liniar necesită parcurgere integrală a textului pentru a ajunge acolo unde dorim, pe când în textul cu legături putem accesa legăturile care ne interesează pentru a ajunge mai rapid la informația de care avem nevoie. Această variantă este mult mai apropiată de gândirea umană, unde informația se acumulează, iar atunci când întâlnim noi informații, pentru a le reține mai uşor le asociem cu cele deja existente.

Aşadar, hipertextul nu face decât să creeze legături între mai multe pagini Web pe o porțiune de tip text. Hipermedia este un termen mai nou ce a apărut ca urmare a dezvoltării noilor tehnologii şi a posibilității de a crea legături prin fişiere ce conțin: animație, sunet, film, imagine. Hipertextul şi hipermedia oferă posibilitatea stocării unei mari cantități de informație, iar utilizatorului îi oferă posibilitatea parcurgerii documentului respectiv în ce ordine doreşte, dându-i senzația de navigare într-un spațiu virtual. De aici vine şi sintagma: “a naviga pe Internet”.

Paginile Web au fost construite inițial cu ajutorul unui limbaj de marcare a hipertextelor, numit HTML (Hypertext Markup Language). Acesta conține comenzi care-i transmit browser-ului să afişeze text, imagini, fişiere multimedia şi legături cu alte pagini Web.

Mai multe pagini Web conectate între ele prin legături de tip hipertext sau hipermedia, ce au un subiect comun, formează un site Web.

Orice site Web are o primă pagină, care se numeşte de regulă *home page* şi se salvează cu numele index.html sau index.htm. Această primă pagină este ca o copertă de carte ce trebuie să ne atragă şi să ne sugereze ce se află în interior. Un site bun este acela în care informațiile sunt uşor de găsit şi accesat, conținând legături către celelalte pagini ale site-ului printr-un meniu, iar din toate paginile există o legătură către prima pagină. Când construim design-ul unui site Web, trebuie să încercăm să creăm ceva unic, cu structura şi combinația nuanțelor de culori adaptată conținutului.

Structura unui site Web este de obicei organizată într-una din cele trei moduri:

- **Liniară.** Organizarea liniară a informațiilor presupune scrierea lor într-o anumită ordine stabilită de către autor, ca într-un roman. Parcurgerea informațiilor se face în totalitate, de la început până la sfârşit.
- **Arborescentă.** Acest mod de organizare presupune structurarea informațiilor şi organizarea lor într-o anumită ordine pe niveluri sau capitole. Pentru a putea ajunge la o anumită informație nu va fi necesară parcurgerea în întregime a conținutului, ci doar a ramurii respective a arborelui sau a capitolului. Aranjarea informațiilor se face de la modul general la detaliu, în funcție de parcurgerea arborelui.
- **Aleatoare.** Organizarea aleatoare seamănă cu gândirea umană, în care se poate ajunge de la o informație la alta aparent aleator, nestructurat, ca într-o pânză de păianjen. De la acest mod de aranjare a informațiilor vine numele Web, care în traducere liberă este *pânză de păianjen*. Organizarea aleatoare a informațiilor într-un site se aseamănă cumva cu gândirea umană asociativă pentru care orice nouă informație se leagă de o alta existentă deja prin asociere de idei. Site-ul construit după o structură aleatoare, nu conține un meniu şi nici nu are informațiile structurate în capitole, dar îl putem parcurge trecând de la o pagină la alta în momentul în care considerăm că legăturile care ni se oferă, sunt demne de interes.

Obiective:
- să înțeleagă cum ar trebui construit un site Web şi de ce;
- să poată realiza şi publica un site Web.

2.3. Construirea site-urilor Web

Pentru a construi un site Web, primul pas este realizarea paginii de început, la care "legăm" celelalte pagini ale site-ului. Legăturile dintre pagini este bine să fie realizate printr-un meniu, astfel încât din orice pagină a site-ului să putem ajunge la pagina de început. Legăturile dintre pagini ar trebui să fie realizate în aşa fel încât să nu fim nevoiți să apelăm la butoanele "Back" şi "Forward" ale browser-ului.

Un alt aspect pe care ar trebui să-l luăm în calcul este aspectul paginilor care ar trebui să fie foarte asemănător, în care să utilizăm aceleaşi fonturi pentru text, aceleaşi culori pentru fundal şi text, acelaşi meniu. Pentru formatarea eficientă a paginilor unui site, cea mai potrivită modalitate este utilizarea stilurilor, despre care o să discutăm în detaliu într-un capitol viitor.

Cum construim rapid un site?

Primul lucru care ar trebui să-l ştim este tema site-ului. După ce am luat această decizie, adunăm documentația de care avem nevoie şi imaginile pe care dorim să le prezentăm.

Înainte de introducerea imaginilor în site, acestea trebuie prelucrate cu ajutorul unor editoare de imagini. Cel mai indicat este ca imaginile pe care dorim să le prezentăm să le prelucrăm în aşa fel încât să avem o imagine de dimensiune mică, care se va încărca foarte uşor de pe Internet şi o imagine de dimensiuni mai mari, care să fie încărcată numai dacă utilizatorul doreşte acest lucru. Efectul acesta se poate realiza dacă utilizatorul dă click pe imagine, imaginea de dimensiuni mari deschizându-se într-o fereastră nouă. O altă modalitate de a introduce o imagine pe o pagină Web, fără ca timpul de încărcare a ei să supere utilizatorii este tăierea imaginii în mai multe bucăți, fiecare bucată reprezentând un fişier. În felul acesta imaginea finală se va încărca mai repede. Unele dintre cele mai utilizate programe de editare şi prelucrare de imagini sunt: Corel Photo Paint, Image Editor, Photoshop, IrfanView.

După prelucrarea imaginilor şi culegerea textului, putem utiliza un editor de pagini web, în care să aranjăm toate elementele aşa cum dorim noi. Editoare de pagini Web sunt: Macromedia Dreamweaver, FrontPage.

Odată terminat un site, acesta ar trebui să fie verificat cu mai multe browsere, dintre cele mai utilizate, cum ar fi: Internet Explorer, Mozilla Firefox, Netscape, Opera. O altă verificare pe care ar trebui să o realizăm este dacă pagina realizată de noi se vede bine la rezoluții diferite ale computerului. Cele mai utilizate rezoluții sunt: 800x600 şi 1024x768.

Să presupunem că site-ul este terminat şi verificat şi dorim să-l poată vedea oricine din Internet. Pentru aceasta avem nevoie în primul rând de spațiu pe un server Web. Se poate procura spațiu (Web Hosting) gratuit de la diverşi furnizori (providers) sau contra cost. O simplă căutare pe Internet ne poate aduce toate informațiile de care avem nevoie.

Următorul pas este transferarea fişierelor site-ului nostru de pe propriul computer, pe serverul Web al unui ISP cu ajutorul unui program FTP.

Toți aceşti paşi îi putem face pentru a realiza foarte rapid un site simplu. Dacă însă dorim ca site-ul nostru să fie foarte bine realizat, ar trebui să ştim ceea ce se află în spatele unei pagini Web şi de ce se află acolo.

Evaluare

1. Ce dimensiune ar trebui să aibă o imagine dintr-o pagină Web şi de ce?
2. Construiți un site personal şi publicați-l pe Internet.

Capitolul 3.

HTML

Obiective:
- să înțeleagă noțiunile de HTML, SGML, XHTML, XML, DTT

3.1. Introducere

Primul pas realizat de către cercetători pentru a transmite mai multe informații în rețea, către oricine doreşte să le cunoască, a fost realizarea paginilor Web, construite cu ajutorul limbajului HTML. O pagină Web scrisă cu HTML are extensia .htm sau .html. De la prima versiune HTML, care datează din 1992 până acum, adică HTML 4.01, acest limbaj a suferit multe îmbunătățiri, cum ar fi: internaționalizarea limbajului, utilizarea indiferent de platformă, printarea, permiterea scripturilor, a stilurilor precum CSS, a obiectelor şi nu în ultimul rând corectarea erorilor.

Ce este HTML?

HTML – HyperText Markup Language este un limbaj de marcare utilizat pentru crearea documentelor care conțin text, grafică, tabele, liste, imagini, sunete, secvențe video şi legături către alte documente prin World Wide Web.

Browser-ele precum: Internet Explorer, Mozilla Firefox, Netscape, Opera interpretează fişierele scrise cu HTML şi le afişează. Afişarea aceleiaşi pagini diferă puțin în funcție de varianta browser-ului. Pentru ca rezultatul să fie cât mai asemănător, indiferent de browser, trebuie să utilizăm etichetele standard HTML, adică formatul SGML. HTML este o aplicație SGML.

Ce este SGML?

SGML – Standard Generalised Mark-up Language este un sistem ce defineşte limbajele de marcare. HTML este un exemplu de limbaj de marcare.

Limbajele de marcare, ca şi cerințele utilizatorilor şi a Web-design-erilor au evoluat. Istoria computerelor ne spune că tot ceea ce nu a reuşit să se dezvolte şi să se adapteze noilor cerințe, a fost uitat şi a dispărut. În încercarea de a păstra HTML-ul, dezvoltatorii lui încearcă să-l transforme într-un limbaj de marcare bazat pe XML, modularizându-l. Toate aceste transformări se fac în ideea de a face mai uşoară combinarea lui cu alte limbaje de marcare, de a corecta problemele de accesibilitate, prelucrare a formularelor, internaționalizare care se cunosc deja. Un limbaj în plină dezvoltare, care corespunde standardelor XML este XHTML.

Ce este XHTML?

XHTML – The Extensible HyperText Markup Language este o familie de tipuri de documente şi module actuale sau viitoare care reproduc HTML. Tipurile de documente din familia XHTML se bazează pe XML.

Ce este XML?

XML – The Extensible Markup Language reprezintă un limbaj de marcare extins ce permite crearea unor documente compatibile cu SGML ce pot fi utilizate în Internet şi suportă o varietate de aplicații.

El a fost conceput pentru a avea puterea şi flexibilitatea SGML fără complexitatea lui. Un document care este strict conform cu XHTML este un document XML ce îndeplineşte criteriile unui DTD.

Ce este DTD?

DTD – Document Type Definition - reprezintă o colecție de declarații de marcare XML ce defineşte o structură de elemente şi atribute pentru a fi utilizate într-un document.

Obiective:
- să înțeleagă diferențele dintre tipurile de documente HTML;
- să poată utiliza noțiunile prezentate pentru a realiza o pagină Web

3.2. Cum se realizează o pagină Web cu HTML

Pentru crearea paginilor Web, limbajul HTML conține etichete (tag) care sunt încadrate de "<" şi ">". Etichetele permit definirea elementelor importante dintr-o pagină Web şi anume: antet, corp al paginii, paragrafe, liste, tabele, legături. Atât etichetele cât şi atributele lor sunt case-insensitive, adică pot fi scrise atât cu litere mari cât şi cu litere mici. Având în vedere faptul că noua generație de HTML, adică XHTML solicită ca etichetele să fie lower-case este bine să ne formăm deprinderea de a scrie aşa, adică cu litere mici.

Înainte de a începe să scriem codul sursă al unei pagini Web, trebuie să spunem browser-ului ce specificații utilizăm: HTML sau XHTML.

Cu HTML putem avea trei tipuri de documente: Strict, Transitional şi Frameset.

- **HTML Strict DTD** se utilizează atunci când dorim să curățăm marcajele, împreună cu CSS (Cascading Style Sheets).
  Exemplu <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  http://www.w3.org/TR/html4/strict.dtd>
- **HTML Transitional DTD** se utilizează în special pentru utilizatorii ce nu au browsere ce suportă CSS şi include atribute şi elemente prezentaționale pe care ar trebui să le scriem în stiluri separate.
  Exemplu <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  http://www.w3.org/TR/html4/loose.dtd
- **HTML Frameset DTD** se utilizează în documentele cu cadre (frame). Acesta este identic cu HTML Transitional DTD cu excepția faptului că elementul FRAMESET înlocuieşte elementul BODY.
  Exemplu <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
  http://www.w3.org/TR/html4/frameset.dtd

OBS *În această carte voi prezenta elementele importante ale limbajului HTML fără să insist asupra atributelor care sunt în contradicție cu HTML 4.01 sau nu sunt suportate de XHTML 1.0 Strict DTD. Scopul acestei cărți este de a inița cititorii în acest domeniu. Pentru mai multe detalii cu privire la recomandările W3C, vizitați: http://www.w3.org/TR/html4.*

SFAT Puteți să verificați dacă o pagină Web este validă W3C, la adresa: http://validator.w3.org/

Etichetele între care se încadrează o pagină Web sunt <html> şi </html>. Este important de reținut faptul că etichetele pe care le deschidem trebuie să le şi închidem.

Primii paşi pentru realizarea unei pagini Web sunt:

- se scrie codul sursă al paginii Web cu ajutorul unui editor de texte, cum ar fi Notepad;
- se salvează documentul cu extensia .htm sau .html;
- se deschide fişierul cu extensia .htm sau .html pentru a putea vedea cum interpretează browser-ul codul sursă HTML.

OBS *Salvați toate documentele cu care lucrați la realizarea unui site într-un singur folder. De exemplu, pentru un site aveți nevoie de mai multe pagini cu extensia .htm sau .html, imagini, fişiere multimedia.*

Structura unei pagini Web este formată din:

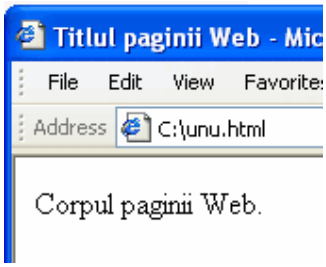
- **Antetul documentului** conține informații despre titlul documentului precum şi informații legate de conținut, cum ar fi: cuvinte cheie, numele autorului sau autorilor paginii Web. Antetul se încadrează între etichetele <head> şi </head>, şi poate conține titlul

documentului, între etichetele <title> şi </title>. Titlul documentului va apărea afişat în bara de sus a browser-ului.

- **Corpul documentului** este partea în care browser-ul afişează conținutul paginii Web. Etichetele între care se încadrează corpul documentului sunt <body> şi </body>. Dacă pagina noastră uneşte mai multe pagini în cadre (frame), atunci în loc de <body> se utilizează elementul <frameset>.

Exemplu de pagină Web scrisă cu HTML.

```
<html>
  <head>
      <title>Titlul paginii Web </title>
<!-- Autor: Diana Diaconu -->
  </head>
<body>
   Corpul paginii Web.
</body></html>
```



Comentariile se introduc între etichetele <!- -şi -- > şi pot ocupa o linie sau mai multe.
Deoarece caracterele < şi > se utilizează pentru etichete, introducerea lor într-o pagină Web necesită utilizarea unor coduri speciale, cum ar fi:

- "<" reprezintă semnul <
- ">" reprezintă semnul >
- "&" reprezintă semnul &
- """ reprezintă semnul "

Alte etichete utile ar fi:
 (break) pentru a trece la linia următoare, <b> pentru a scrie textul îngroşat,
<i> înclinat şi <u> subliniat. Etichetele se pot utiliza şi împreună, de exemplu pentru a avea un text îngroşat şi înclinat.

Exemplu de pagină Web scrisă cu coduri HTML.

```
<html>
<head><title>Coduri</title></head>
<body>
<b>Coduri pentru semnele:</b><br /><br/>
<u>&    l    t    ;</u>    <i>reprezinta
   semnul</i>&lt;<br />
<u>& g t ;</u> reprezinta semnul &gt;<br
   />
<b><u><i>&    a    m    p    ;</i></u></b>
   reprezinta semnul &amp;<br />
<i><u>&  q  u  o  t  ;</u></i>  reprezinta
   semnul &quot;
</body>
</html>
```

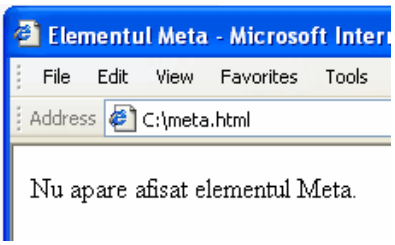
După cum vedem în exemplul de mai sus, etichetele se pot utiliza câte una sau împreună.

3.3. Elementul Meta

Limbajul HTML permite autorilor să specifice informații cu privire la conținutul documentului, autor. Atributele elementului Meta:
name= specifică proprietatea nume
content= valoarea conținutului
scheme= schema ce va fi folosită pentru a interpreta valoarea proprietății. Valoarea atributului *scheme* depinde de proprietatea nume.
http-equiv= atribut ce poate fi folosit în locul atributului name.
lang= specifică limba;

Exemplu

```
<html>
<head>
<META name="Author" lang ="ro" content="Diana
   Diaconu">
<!- -Pentru un site cu jucarii -->
<META name="keywords" lang="ro" content="joc,
   ursulet, puzzle, calut">
<!- - Pentru a specifica tipul continutului -->
<META http-equiv="Content-Type"
   content="text/html">
<!- - Pentru a specifica data crearii site-ului -
   ->
<META name="date" content="2006-01-
   01T09:34:25+00:00">
<title>Elementul Meta</title>
</head>
<body>
Nu apare afisat elementul Meta.
</body></html>
```



3.4. Culori

Paginile Web implicit au culoarea alb pentru fundal şi negru pentru text. Aceste culori se pot schimba, dacă ştim numele culorii în limba engleză sau codul culorii în hexazecimal. În continuare avem un tabel cu cele mai importante culori. Trebuie reținut faptul că putem avea foarte multe nuanțe de culori, plecând de la culorile de bază, deoarece o culoare o putem scrie ca fiind un număr în hexazecimal de la #000000 până la #FFFFFF.

Culoare	Echivalent nume	Echivalent nr. hexazecimal		Culoare	Echivalent nume	Echivalent nr. hexazecimal
Negru	Black	#000000		Verde	Green	#008000
Argintiu	Silver	#C0C0C0		Verde deschis	Lime	#00FF00
Gri	Gray	#808080		Măsliniu	Olive	#808000
Alb	White	#FFFFFF		Galben	Yellow	#FFFF00
Maron	Maroon	#800000		Albastru închis	Navy	#000080
Roşu	Red	#FF0000		Albastru	Blue	#0000FF
Violet	Purple	#800080		Albastru marin	Teal	#008080
Siclam	Fuchsia	#FF00FF		Albastru deschis	Aqua	#00FFFF

Culorile pe care le putem alege sunt de fapt combinații ale culorilor de bază: roşu (Red), verde (Green) şi albastru (Blue).

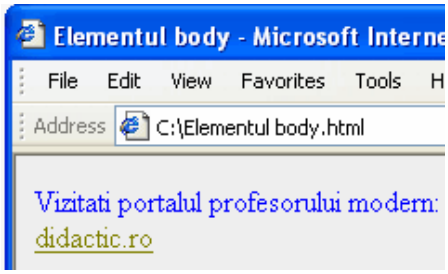
Să facem un mic experiment! Haideți să încercăm să ajungem la culoarea ciocolatei (chocolate). Cum vom face asta? Luăm mai mult roşu (D2) pentru că este o culoare puternică, adăugăm verde, dar nu prea mult (69) şi foarte puțin albastru, pentru că este o culoare rece (1E). La ce rezultat ajungem? D2691E, adică valoarea RGB #D2691E. Încercați această combinație de culori să vedeți dacă vă place (culoarea, nu ciocolata ☺).

3.5. Elementul BODY

Culorile dintr-o pagină web pot fi stabilite prin atributele elementului **body** şi anume:
backgound = atribuie o imagine de fundal pentru browser-ele vizuale;
bgcolor = atribuie o culoare fundalului paginii Web;
text = atribuie o culoare textului paginii Web;
link= atribuie o culoare legăturilor de tip hipertext nevizitate
vlink= atribuie o culoare legăturilor de tip hipertext vizitate
alink= atribuie o culoare legăturilor de tip hipertext selectate de către utilizator

Exemplu

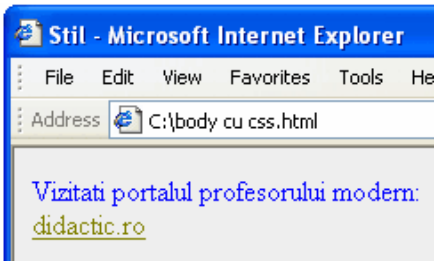
```
<html>
<head><title>Elementul body</title></head>
<body bgcolor=#EEEEEE text="blue" link=#008000
alink=#00FF00 vlink=#808000>
Vizitati portalul profesorului modern:<br /> <a
href="http://www.didactic.ro">didactic.ro</a>
</body>
</html>
```



OBS Cu toate că funcționează foarte bine, acest mod de a atribui culori elementului body, nu este de preferat deoarece în browser-ele audio se specifică inutil toate aceste atribute. Modalitatea cea mai bună este de a crea un stil propriu fiecărei formatări, aşa cum vom vedea în exemplul următor.

Exemplu

```
<html>
<head><title>Stil</title>
<style type="text/css">
body {background:#EEEEEE;color: blue}
A:link{color=#008000}
A:visited{color:#808000}
A:active{color:#00FF00}
</style></head>
<body>
Vizitati portalul profesorului modern:<br /> <a
href="http://www.didactic.ro">didactic.ro</a>
</body></html>
```



Acelaşi exemplu se poate scrie şi cu stilul salvat într-un fişier separat de documentul .html, ca în tabelul următor. Această modalitate de lucru cu stiluri este mai uşor de modificat şi are ca avantaj faptul că putem utiliza acelaşi stil pentru mai multe pagini dintr-un site Web.

ex_stil_1.css	fisier.html
`body {background:#EEEEEE;color: blue}` `A:link{color=#008000}` `A:visited{color:#808000}` `A:active{color:#00FF00}`	`<html>` `<head><title>Stil</title>` `<link rel="stylesheet" type="text/css" href="ex_stil_1.css">` `</head>` `<body>` `Vizitati portalul profesorului modern: ` `<a href="http://www.didactic.ro">didactic.ro</a>` `</body></html>`

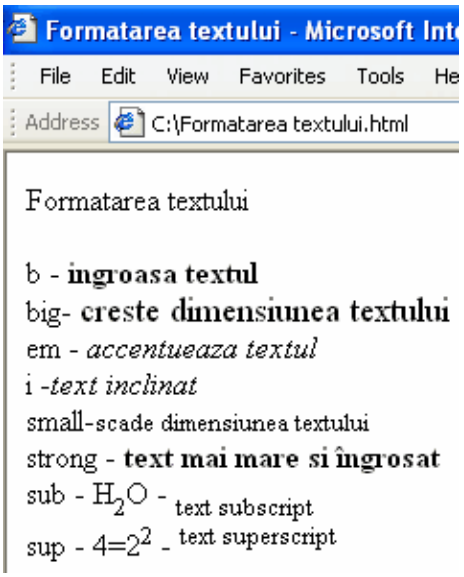
3.6. Formatarea textelor

Etichetele pentru formatarea textelor sunt:

Etichetă	Descriere
<b>	text îngroşat
<big>	creşte dimensiunea textului
	accentuează textul
<i>	text înclinat
<small>	scade dimensiunea textului
	text mai mare şi îngroşat
<sub>	text subscript
<sup>	text superscript

Exemplu În exemplul următor putem vedea interpretarea browser-ului pentru toate modalitățile de formatare a textului din tabelul de mai sus.

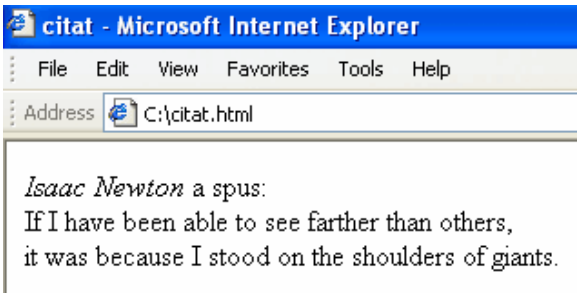
```
<html>
<head><title>Formatarea textului</title>
</head>
<body>
Formatarea textului<br><br />
b  - <b>ingroasa textul</b><br />
big- <big>creste dimensiunea textului</big><br
   />
em - <em>accentueaza textul</em><br />
i  -<i>text inclinat</i><br />
small-<small>scade                    dimensiunea
   textului</small><br />
strong    -    <strong>text    mai    mare    si
   îngrosat</strong><br />
sub     -     H<sub>2</sub>O     -     <sub>text
   subscript</sub><br />
sup     -     4=2<sup>2</sup>     -     <sup>text
   superscript</sup>
</body>
</html>
```



Alte etichete pentru formatare mai sunt: **<cite></cite>** care conțin un citat sau o referire la altă sursă, după cum se vede în exemplul următor:

Exemplu

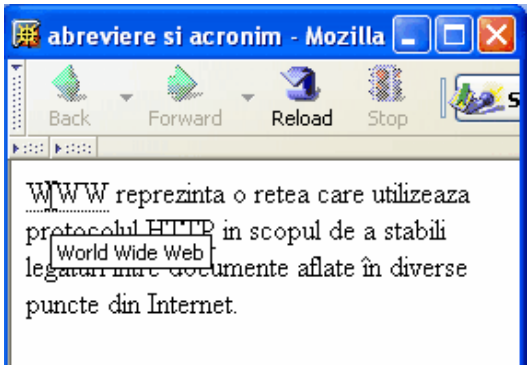
```
<html>
<head><title>citat</title></head>
<body>
<cite>Isaac Newton</cite> a spus:<br>
If I have been able to see farther than
others,<br>
  it  was  because  I  stood  on  the
shoulders of giants.<br>
</body>
</html>
```



Etichetele **<abbr>** şi **<acronym>** permit autorului să indice abrevierile şi acronimele şi se utilizează cu atributele: ***title*** care specifică ce reprezintă abrevierea respectivă şi ***lang*** care ne indică limba utilizată.

Exemplu

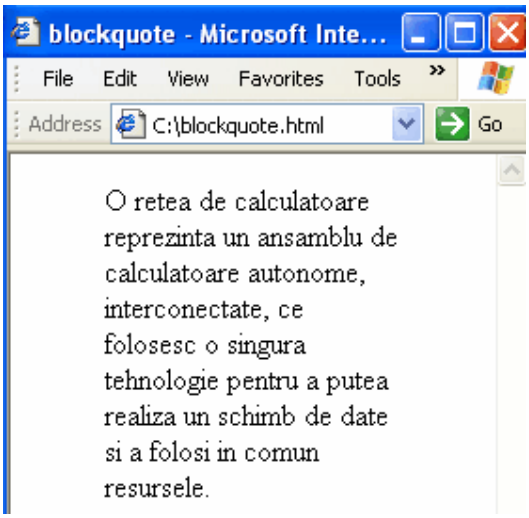
```
<html>
<head><title>abreviere                    si
acronim</title></head>
<body>
<abbr    lang="ro"    title="World    Wide
Web">WWW</abbr>  reprezinta  o  retea  care
utilizeaza protocolul
<abbr lang="ro" title="Hypertext Transfer
Protocol"> HTTP</abbr> in scopul de a
stabili legaturi intre documente aflate în
diverse puncte din Internet.<br>
</body>
</html>
```



Pentru citate se mai utilizează şi etichetele **<blockquote>** şi **<q>**, cu deosebirea că <blockquote> se utilizează pentru blocuri de text mai mari, iar <q> pentru blocuri de text mai mici.

Exemplu

```
<html>
<head><title>blockquote</title></head>
<body>
<blockquote cite="Pagini Web cu JavaScript">
<p>O retea de calculatoare reprezinta un ansamblu de calculatoare autonome, interconectate, ce folosesc o singura tehnologie pentru a putea realiza un schimb de date si a folosi in comun resursele.
</p>
</blockquote>
</body>
</html>
```



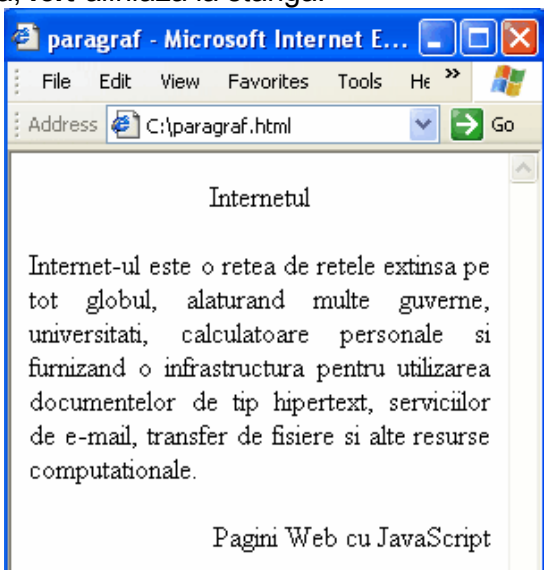
3.7. Formatarea paragrafelor

Cea mai utilizată modalitate de formatare a unei porțiuni mai mari de text este cu ajutorul paragrafelor. În HTML un paragraf se încadrează între etichetele şi şi are atributele:

id, class= identificatorii documentului;
lang= specifică limba;
title= elementul titlu;
style= stil;
align= aliniament;
onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup evenimente

Exemplu În exemplul următor am utilizat atributul **align** ce are posibilitățile: ***center***-centrează, ***justify***-aliniază stânga-dreapta, ***right***-aliniază la dreapta, ***left***-aliniază la stânga.

```
<html>
<head><title>paragraf</title>
</head>
<body>
<p title="titlul" align="center" lang="ro">
Internetul</p>
<p title="definitie" align="justify" lang="ro">
Internet-ul este o retea de retele extinsa pe tot globul, alaturand multe guverne, universitati, calculatoare personale si furnizand o infrastructura pentru utilizarea documentelor de tip hipertext, serviciilor de e-mail, transfer de fisiere si alte resurse computationale.</p>
<p title="sursa" align="right" lang="ro">
Pagini Web cu JavaScript</p></p>
</body>
</html>
```



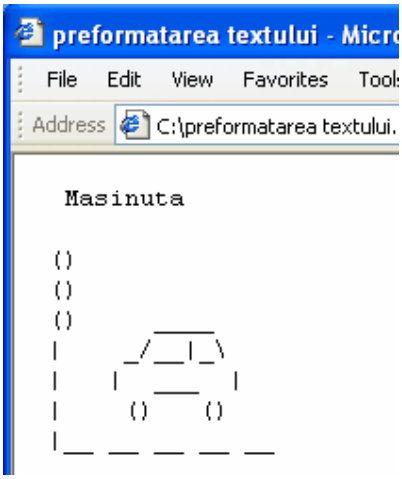
3.8. Preformatarea textului

Browser-ul ignoră formatările noastre dacă nu le încadrăm între etichetele corespunzătoare. De exemplu, dacă scriem câteva versuri frumos formatate în editorul de texte, atunci când îl salvăm, browser-ul scrie versurile unul în continuarea celuilalt, nu unul sub altul aşa cum am dori noi. Dacă dorim ca browser-ul să respecte formatarea noastră fără să utilizăm elementele: <p>,
..., atunci putem utiliza elementul <**pre**></**pre**>.

Exemplu

```
<html>
<head><title>preformatarea
textului</title></head>
<body>
<pre title="masinuta" lang="ro">
  Masinuta

  ()
  ()
  ()      ____
  |     _/__|_\
  |    |  ___  |
  |     ()   ()
  |__ __ __ __ __
</pre></body>
</html>
```



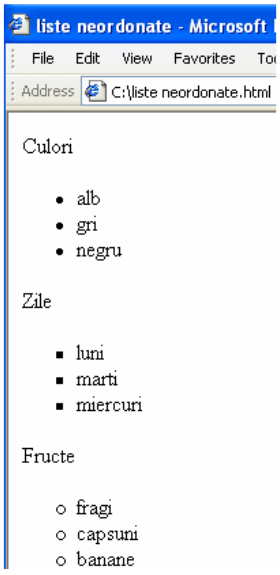
3.9. Liste

Avem trei modalități de afişare a informațiilor în: liste ordonate, liste neordonate şi definiții.

Liste neordonate - Unordered Lists (UL) - se încadrează între etichetele <**ul**></**ul**>. Fiecare element al listei începe cu eticheta <**li**>. Atributul **type** al listei neordonate poate lua valorile: "square", "circle" sau "disc", în funcție de cum dorim să fie afişat semnul din fața elemntelor listei.

Exemplu

```
<html>
<head><title>liste neordonate</title></head>
<body>
Culori
<ul>
 <li>alb</li>
 <li>gri</li>
 <li>negru</li></ul>
Zile
<ul type="square">
 <li>luni</li>
 <li>marti</li>
 <li>miercuri</li></ul>
Fructe
<ul type="circle">
 <li>fragi</li>
 <li>capsuni</li>
 <li>banane</li> </ul>
</body></html>
```

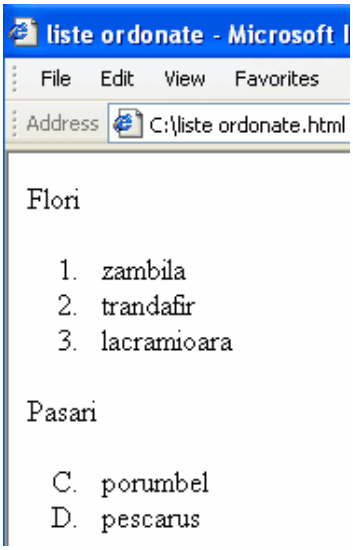


Liste ordonate – Ordered Lists (OL) - se încadrează între etichetele <**ol**></**ol**>. Fiecare element al listei începe cu eticheta <**li**>. Atributul **start** al listei ordonate specifică numărul de la care începe numărătoarea pentru itemii listei. Fără acest atribut, numărătoarea începe de la 1. Atributul **type** defineşte tipul numerotării elementelor listei în felul următor:

type=”1” pentru cifre 1, 2, 3, ...
type=”A” pentru litere mari A, B, C, ...
type=”a” pentru litere mici a, b, c, ...
type=”I” pentru cifre romane scrise cu litere mari I, II, III, IV, ...
type=”i” pentru cifre romane scrise cu litere mici i, ii, iii, iv, ...

Exemplu

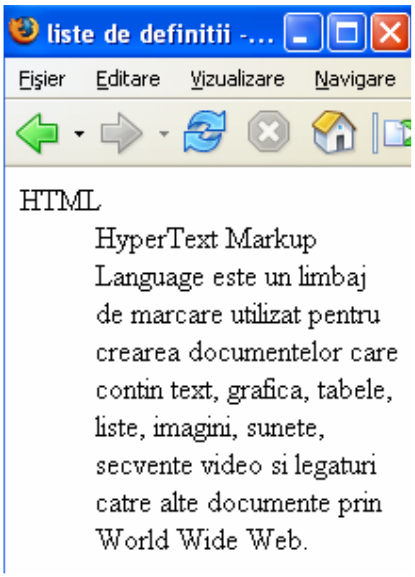
```
<html>
<head><title>liste ordonate</title></head>
<body>
Flori
<ol>
 <li>zambila</li>
 <li>trandafir</li>
 <li>lacramioara</li>
</ol>
Pasari
<ol type="A" start=3>
 <li>porumbel</li>
 <li>pescarus</li>
</ol>
</body></html>
```



Liste de definiții – Definition Lists (DL) – se încadrează între etichetele **<dl></dl>** şi sunt puțin diferite față de celelalte liste deoarece fiecare element al listei este format din două părți: termenul de definit şi definiția. Într-o definiție, termenul care trebuie definit începe cu eticheta <dt>, iar definiția termenului începe cu eticheta <dd>.

Exemplu

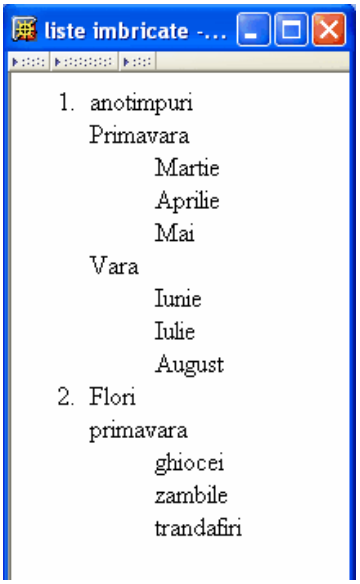
```
<html>
<head><title>liste de definitii</title></head>
<body>
<dl>
 <dt>HTML</dt>
  <dd> HyperText Markup Language este un limbaj de
marcare utilizat pentru crearea documentelor care
contin text, grafica, tabele, liste, imagini,
sunete, secvențe video si legaturi catre alte
documente prin World Wide Web.
</dd>
</dl>
</body>
</html>
```



Imbricarea listelor se realizează prin inserarea unei liste în interiorul altei liste. Se pot insera liste ordonate în interiorul celor neordonate sau invers, liste de definiții în interiorul celor ordonate, ca în cazul exemplului de mai jos şi alte modalități.

Exemplu

```
<html>
<head><title>liste imbricate</title></head>
<body>
<ol>
<li>anotimpuri</li>
<dl>
 <dt>Primavara</dt>
  <dd>Martie</dd>
  <dd>Aprilie</dd>
  <dd>Mai</dd>
<dt>Vara</dt>
  <dd>Iunie</dd>
  <dd>Iulie</dd>
  <dd>August</dd>
<li>Flori</li>
<dt>primavara</dt>
  <dd>ghiocei</dd>
  <dd>zambile</dd>
  <dd>trandafiri</dd></dl>
</ol>
</body></html>
```



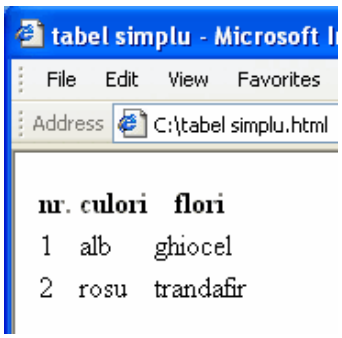
3.10. Tabele - se scriu între etichetele **<table>** şi **</table>**. Un tabel este împărţit în rânduri încadrate între etichetele **<tr>** şi **</tr>** (table row) şi fiecare rând este împărţit în celule cu ajutorul etichetelor **<td>** şi **</td>**. O celulă poate conține text, imagini, liste, paragrafe, formulare.

Etichete specifice tabelelor:

- **table** defineşte un tabel;
- **th** defineşte antetul unui tabel;
- **tr** defineşte rândul unui tabel;
- **td** defineşte celula unui tabel;
- **caption** defineşte legenda unui tabel;
- **colgroup** defineşte grupe de coloane ale unui tabel;
- **col** defineşte valoarea atributului pentru una sau mai multe coloane ale unui tabel;
- **thead** defineşte capul unui tabel;
- **tbody** defineşte corpul unui tabel;
- **tfoot** defineşte partea de jos a unui tabel.

Exemplu de tabel simplu

```
<html>
<head><title>tabel simplu</title></head>
<body>
<table>
 <tr> <th>nr.</th>
        <th>culori</th>
        <th>flori</th> </tr>
 <tr> <td>1</td>
        <td>alb</td>
        <td>ghiocel</td> </tr>
 <tr> <td>2</td>
        <td>rosu</td>
        <td>trandafir</td> </tr>
</table>
</body></html>
```

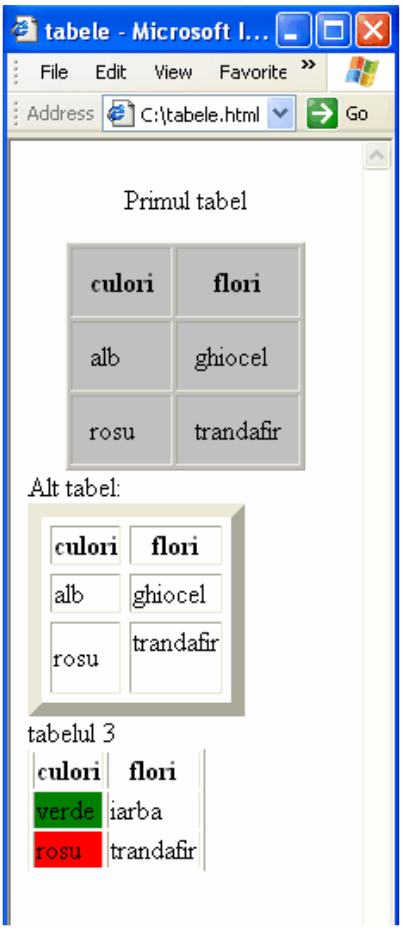


Atribute specifice tabelelor:

Atribut	Valoare	Descriere
align	left / center / right	Aliniază tabelul la stânga / centru / dreapta.
bgcolor	Numele culorii sau valoarea în hexazecimal.	Specifică culoarea fundalului tabelului.
border	pixeli	Marginea celulelor tabelului. În funcție de valoarea pe care o poate lua, va fi afişat un tabel fără margini (ca cel de mai sus border=”0”), sau cu margini de diferite dimensiuni.
cellpadding	pixeli în %	Specifică spațiul dintre conținutul şi marginile celulei.
cellspacing	pixeli în %	Specifică spațiul dintre celule.
frame	void / above / below / hsides / lhs / rhs / vsides / box / border	Specifică modalitatea de afişare a bordurii exterioare.
rules	None / groups / rows / cols / all	Specifică liniile despărțitoare orizontale / verticale.
summary	text	Specifică sumarul tabelului pentru browser-e non-vizuale
width	pixeli în %	Specifică lățimea tabelului.

Exemplu Primul tabel este centrat, are marginea 1 şi distanța dintre conținutul celulei şi margini este de 10%; al doilea tabel are marginea mai mare, de 8, spațiul dintre celule mai mare 5% iar lățimea tabelului este 50% din lățimea suprafeței de afişare a browser-ului; al treilea tabel are vizibile doar marginile verticale şi colorate fundalurile a două celule cu bgcolor.

```
<html>
<head><title>tabele</title></head>
<body>
<table border="1" bgcolor="silver" align="center"
cellpadding=10%>
<caption>Primul tabel</caption>
 <tr> <th>culori</th>
        <th>flori</th> </tr>
 <tr> <td>alb</td>
        <td>ghiocel</td> </tr>
 <tr> <td>rosu</td>
        <td>trandafir</td> </tr>
</table>
Alt tabel:
<table border="8" cellspacing=5% width=50%>
<tr> <th>culori</th>
       <th>flori</th> </tr>
<tr> <td>alb</td>
       <td>ghiocel</td> </tr>
<tr> <td>rosu</td>
       <td  background="trandafir.jpg">  trandafir
<br><br></td>  </tr>
</table>
tabelul 3
<table frame="vsides">
<tr> <th>culori</th>
       <th>flori</th> </tr>
<tr> <td bgcolor="green">verde</td>
       <td>iarba</td> </tr>
<tr> <td bgcolor="red">rosu</td>
       <td>trandafir</td>  </tr>
</table>
</body></html>
```



3.11. Legături HTML – se scriu între etichetele **<a>** şi **** (Anchor) şi pot fi de tip hipertext sau hipermedia. Cel mai important atribut al etichetei <a> este **href** care poate crea o legătură către un alt document din spațiul Web, către o imagine, un film.

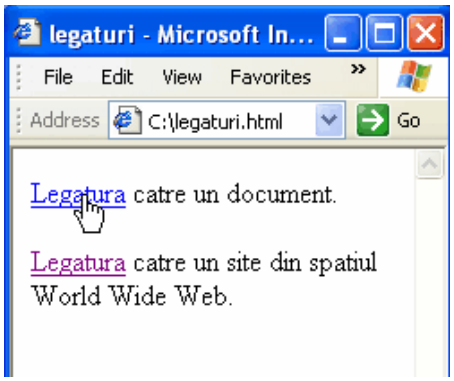
Legături relative sau absolute

Este important să ştim cum creăm o legătură către o pagină Web sau o imagine, dar trebuie să ştim şi cum să le apelăm. De exemplu, dacă avem o pagină Web pe un server la adresa: http://www.un_site_oarecare.ro/folder1/pagini/pagina1.html, atunci când o apelăm putem scrie de fiecare dată toată calea, adică să scriem calea absolută a fişierului respectiv sau putem scrie o cale relativă la fişier. Legătura relativă se referă la modalitatea de referire a paginilor. De exemplu, dacă cele două fişiere se află în acelaşi folder, putem scrie doar "pagina1.html".

Concluzia este că cea mai bună modalitate de apelare a unei pagini este prin utilizarea legăturii relative la pagina respectivă.

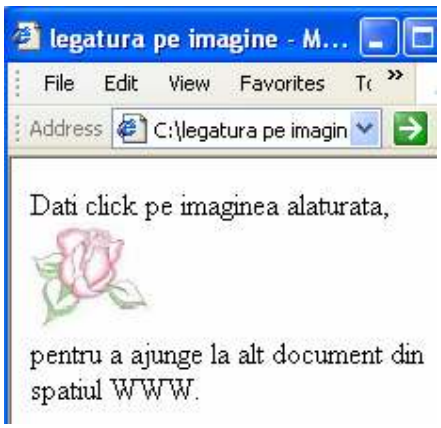
Exemplu de legătură relativă către un document care se află în acelaşi folder cu documentul apelant, respectiv un exemplu de legătură absolută către un document din spațiul Web.

```
<html>
<head><title>legaturi</title>
</head>
<body>
<a href="pagina.html">
Legatura</a> catre un document.
<p>
  <a href="http://www.einformatica.ro/">
Legatura
</a> catre un site din spatiul World Wide Web.
</p>
</body>
</html>
```



Exemplu de legătură pe o imagine.

```
<html>
<head><title>legatura pe imagine</title>
</head>
<body>
Dati click pe imaginea alaturata, <br>
<a href="fisier.html">
<img  border="0"  src="trandafir.jpg"  width="65"
height="60"><br>
</a>
pentru a ajunge la alt document din spatiul WWW.
</body>
</html>
```



Exemplu de legătură pe aceeaşi pagină. Această modalitate de legare a conținutului paginii se utilizează îndeosebi la paginile Web stufoase, cu conținut încărcat, în care se află multă informație.

```
<html>
 <head>
<title>legatura pe aceeasi pagina</title></head>
<body>
<p><a href="#au">Legatura catre luna august.
</a></p>
<h2><a name="ia">Ianuarie</a></h2>
<h2>Februarie</h2>
<h2>Martie</h2>
<h2>Aprilie</a></h2>
<h2>Mai</h2>
<h2>Iunie</h2>
<h2>Iulie</h2>
<h2><a name="au">August</a>
</h2>
<p>
<a href="#ia">Legatura catre luna ianuarie.</a></p>
<h2>Septembrie</h2>
<h2>Octombrie</h2>
<h2>Noiembrie</h2>
<h2>Decembrie</h2>
</body></html>
```

Atribute specifice elementului Anchor (<a>):

Atribut	Valoare	Descriere
charset	character_encoding	Specifică codul caracterului țintei URL.
coords	if shape="rect" then coords="*left,top,right,bottom*" if shape="circ" then coords="*centerx,centery,radius*" if shape="poly" then coords="*x1,y1,x2,y2,..,xn,yn*"	Specifică coordonatele atributului shape pentru a defini o regiune dintr-o imagine pentru maparea imaginii.
href	URL	Ținta URL a legăturii.
hreflang	language_code	Specifică limbajul țintei URL.
name	section_name	Numele ancorei. Se utilizează acest atribut pentru a crea un "semn" într-un document.
rel	Alternate / designates / stylesheet / start / next / prev / contents / index / glossary / copyright / chapter / section / subsection / appendix / help / bookmark	Specifică relațiile dintre documentul curent şi ținta URL.
rev	Alternate / designates / stylesheet / start / next / prev / contents / index / glossary / copyright / chapter / section / subsection / appendix / help / bookmark	Specifică relațiile dintre ținta URL şi documentul curent.
shape	rect / rectangle / circ / circle / poly / polygon	Defineşte tipul regiunii ce va fi definită pentru mapare în aria etichetei curente. Se utilizează cu atributul coords.
target	_blank _parent _self _top	-ținta URL se va deschide într-o nouă fereastră -ținta URL se va deschide în *parent frameset* -ținta URL se va deschide în acelaşi cadru -ținta URL se va deschide în corpul ferestrei
type	mime_type	Specifică tipul MIME al țintei URL.

3.12. Cadre HTML – HTML Frame – permit afişarea mai multor pagini Web în fereastra unui browser. Fiecare document HTML afişat în browser este numit frame (cadru) şi este independent de celelalte frame-uri. Avantajele utilizării frame-urilor sunt:

- când se îmbunătăţeşte conţinutul unui site Web se pot modifica doar cadrele respective, nu întregul site.
- site-ul se încarcă mai repede în browser.

Dezavantaje:

- este dificil de printat o pagină Web ce conţine cadre
- unele motoare de căutare nu verifică decât conţinutul cadrului principal, nu conţinutul tuturor cadrelor.

Etichetele cadrelor HTML:
<frameset> - defineşte un set de cadre;
<frame> - defineşte un cadru al ferestrei;
<noframes> - defineşte o secţiune noframe pentru browser-ele care nu pot afişa cadre;
<iframe> - defineşte un cadru într-o porţiune din interiorul ferestrei

Elementul **<frameset>** defineşte un set de cadre, unde fiecare cadru deschide separat câte o pagină Web. Pentru a realiza acest lucru, se utilizează atributele: ***cols*** – coloane şi ***rows*** – linii.

Exemplu de utilizare a cadrelor cu atributul *cols*. Paginile se salvează separat, de preferat în acelaşi folder.

Pagina principală:

```
<html>
<frameset cols = "30%, 70%">
  <frame src ="meniu.html" />
  <frame src ="pagina simpla.html" />
</frameset>
</html>
```

Pagina din stânga (meniu.html):

```
<html>
<head><title>Meniu</title></head>
<body bgcolor="silver">
<h2>Albert Einstein</h2>
"Imaginatia este mai importanta decât cunoasterea"
</body></html>
```

Pagina din dreapta (pagina simpla.html):

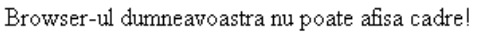
```
<html>
<head><title>Einstein</title></head>
<body>
<center>
<h2>Omul secolului</h2>
<hr><img          src="Einstein.jpg"><br>
</center>
Anul 1905 este un glorios pentru descoperirile din domeniul fizicii, an ce a provocat valuri la nivel mondial si pentru care, 2005 este considerat anul Einstein. De acum este numit <i> The man of the century</i> (Omul secolului).
</body></html>
```

Elementul **<noframes>** se utilizează pentru ca utilizatorii browser-elor ce nu pot utiliza cadre, să primească un mesaj de avertizare. Pentru ceilalți, rezultatul este ca la exemplul de mai sus.

Exemplu

Pagina principală:

```
<html>
<frameset cols = "30%, 70%">
<noframes>
  <body> Browser-ul dumneavoastră nu poate
afişa cadre!</body></noframes>
  <frame src ="meniu.html" />
  <frame src ="pagina simpla.html" />
</frameset>
</html>
```



Elementul **<iframe>** crează un frame care conține o altă pagină Web, ca în exemplul următor.

Exemplu Acest element se utilizează în cazul în care dorim să afişăm conținutul altei pagini Web pe o suprafață mică, delimitată de noi într-o anumită zonă a paginii principale.

Pagina principală:

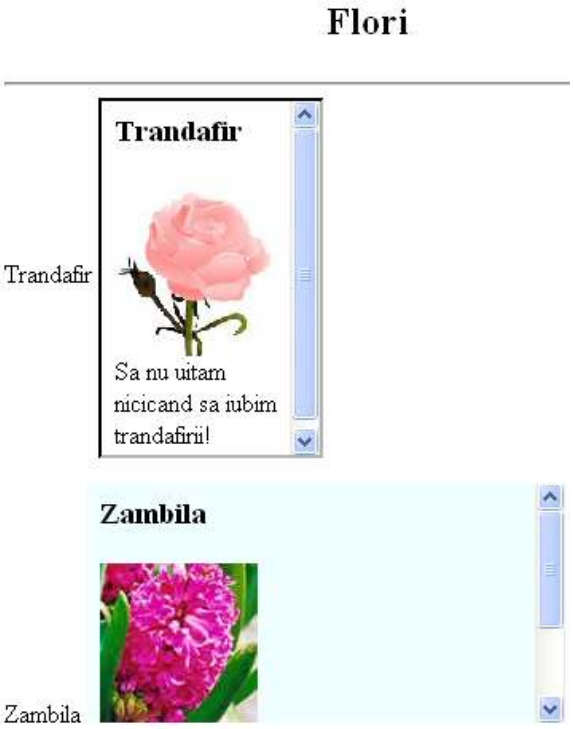
```
<html>
<head><title>Flori</title>
</head>
<body>
<h2><p align="center">Flori</p>
</h2>
<hr>Trandafir
<iframe              src="trandafir.html"
align="middle" frameborder=1 height=40%
width=30% scrolling="yes"></iframe>
<p>Zambila
<iframe                src="zambila.html"
frameborder=0></iframe></p>
</body>
</html>
```

Fişierul trandafir.html:

```
<html>
<head><title>Trandafir</title>
</head>
<body>
<h3>Trandafir</h3>
<img src="trandafir1.jpg"><br />
Sa nu uitam nicicand sa iubim
trandafirii!
</body>
</html>
```

Fişierul zambila.html:

```
<html>
<head><title>Zambila</title></head>
<body bgcolor="azure">
<h3>Zambila</h3>
<img src="zambila.jpg"><br />
Zambila este o floare de climat
subtropical-temperat, originara din
Orientul Apropiat.<br />
</body>
</html>
```



Atribute specifice elementului **iframe:**

Atribut	Valoare	Descriere
align	left / right / top / middle / bottom	Specifică modalitatea de aliniere a textului ce înconjoară elementul *iframe*.
frameborder	1 / 0	Specifică dacă iframe va afişa sau nu bordură.
height	pixeli în %	Defineşte înălțimea unui *iframe*.
longdesc	URL	Descrierea pe larg a URL-ului conținutului cadrului.
marginheight	pixeli	Defineşte marginile de sus şi de jos ale *iframe.*
marginwidth	pixeli	Defineşte marginile din stânga şi din dreapta ale *iframe.*
name	frame_name	Specifică numele unic al *iframe.*
scrolling	ys / no / auto	Defineşte bara scroll.
src	URL	URL-ul paginii Web afişată în *iframe*.
width	pixeli în %	Defineşte lățimea unui *iframe*.

Elementul <**frame**> defineşte un cadrul al ferestrei şi are următoarele atribute: *frameborder, longdesc, marginheight, marginwidth, name, noresize, scrolling, src*.

3.13. Formulare HTML – HTML Form – sunt utilizate pentru a selecta diferite tipuri de intrări ale utilizatorilor.

Etichetele formularelor HTML:
<**form**> - defineşte un formular pentru o intrare a utilizatorului;
<**input**> - defineşte intrarea unui câmp;
<**textarea**> - defineşte aria textului (text-area);
<**label**> - defineşte o etichetă de control
<**fieldset**> defineşte un fieldset;
<**legend**> - defineşte legenda unui fieldset;
<**select**> - defineşte o listă selectabilă;
<**optgroup**> - defineşte o opțiune a unui grup;
<**option**> - defineşte o opțiune;
<**button**> - defineşte un buton pe care se “apasă”

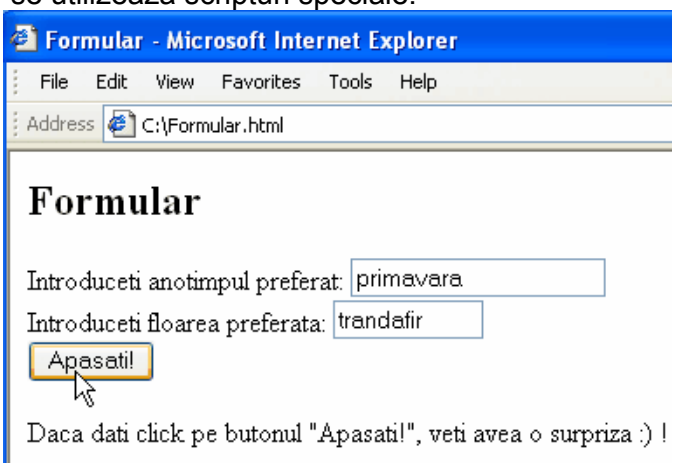
3.13.1. Elementul **form** poate conține un câmp de text, checkboxes, butoane radio. Formularele sunt utilizate pentru a trimite datele introduse de către utilizator către un URL specificat.

Atribute specifice elementului **form:**

Atribut	Valoare	Descriere
action	URL	URL-ul ce defineşte unde vor fi trimise datele când va fi “apăsat” butonul.
accept	listă cu conținut	O listă separată prin virgulă ce conține tipul de conținut pe baza căruia server-ul va procesa corect formularul.
accept-charset	charset-list	O listă separată prin virgulă ce poate conține seturi de caractere pentru formular.
enctype	mimetype	Tipul mime utilizat pentru a coda conținutul formularului.
method	get / post	method=”get” trimite conținutul formularului către URL. method=”post” trimite conținutul formularului în conținutul cererii.
name	nume formular	Defineşte numele formularului
target	_blank _self _parent _top	-ținta URL se va deschide într-o nouă fereastră -ținta URL se va deschide în *parent frameset* -ținta URL se va deschide în acelaşi cadru -ținta URL se va deschide în corpul ferestrei

Exemplu de utilizare a elementului form. În acest exemplu, utilizatorii pot introduce informații pe pagina Web şi pot acționa butonul *Apăsați!* care are ca efect deschiderea altei pagini Web. Pentru acțiuni mai complicate, se utilizează scripturi speciale.

```
<html>
<head><title>Formular</title></head>
<body>
<h2>Formular</h2>
 <form        action="surpriza.html"
method="get">
Introduceti anotimpul preferat:
<input      type="text"      name="x"
value="primavara"> <br />
Introduceti floarea preferata:
<input      type="text"      name="y"
value="trandafir" size=10 >
<br />
<input                  type="submit"
value="Apasati!">
</form>
<p>Daca   dati   click   pe   butonul
   "Apasati!", veti avea o surpriza
   :) !</p>
</body>
</html>
```



3.13.2. Elementul **input** defineşte începutul câmpului de introducere a datelor de către utilizatori.

Atribute specifice elementului **input:**

Atribut	Valoare	Descriere
accept	listă cu conținut	O listă separată prin virgulă ce conține tipul de conținut pe baza căruia server-ul va procesa corect formularul.
align	left / right / top / texttop / middle / absmiddle / baseline / bottom / absbottom	Se utilizează numai cu type=”image” şi defineşte aliniamentul textului ce urmează imaginii.
alt	text	Se utilizează numai cu type=”image” şi defineşte un text alternativ, pentru cazul în care nu se încarcă o imagine.
checked	checked	Se utilizează cu type=”checkbox”, type=”radio” şi indică faptul că elementul introdus va fi verificat la prima încărcare.
disabled	disabled	Dezactivează elementul introdus la prima încărcare.
maxlength	număr	Defineşte numărul maxim de caractere permise în câmpul de tip text.
name	nume formular	Defineşte numele formularului
readonly	readonly	Indică faptul că valoarea acestui câmp nu poate fi modificată.
size	număr de tip char	Defineşte dimensiunea elementului care va trebui introdus.
src	URL	Se utilizează numai cu type=”image” şi defineşte URL pentru imaginea care va fi afişată.
type	button / checkbox / file / hidden / image / password / radio / reset / submit / text	Defineşte tipul elementului care va fi trebui introdus.
value	value	Defineşte valori diferite în funcție de tipul butoanelor. ○ Pentru butoanele **reset** şi **submit** defineşte textul de pe buton ○ Pentru butoanele **image** defineşte rezultatul câmpului care trece prin script. ○ Pentru butoanele **checkboxes** şi **radio** defineşte rezultatul elementului introdus când se dă click pe buton. ○ Pentru câmpurile **hidden, password** şi **text** defineşte valoarea inițială a elementului.

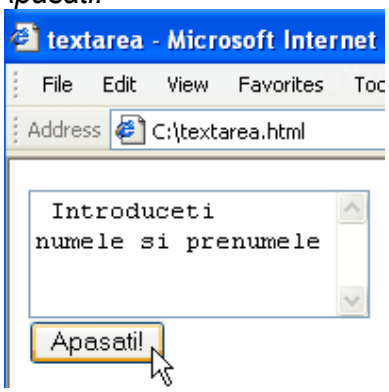
3.13.3. Elementul **textarea** defineşte aria textului în care utilizatorii pot introduce text.

Atribute specifice elementului **textarea:**

Atribut	Valoare	Descriere
cols	număr	Specifică numărul de coloane vizibile în aria de introducere a textului.
rows	număr	Specifică numărul de rânduri vizibile în aria de introducere a textului.
disabled	disabled	Dezactivează elementul introdus la prima încărcare.
name	nume	Specifică numele pentru aria de introducere a textului.
readonly	readonly	Indică faptul că valoarea acestui câmp nu poate fi modificată.

Exemplu de utilizare a elementului textarea. În acest exemplu utilizatorii pot introduce un comentariu pe o suprafață mai mare înainte de a acționa butonul *Apasati!*

```
<html>
<head><title>textarea</title></head>
<body>
<p></p>
 <form action="surpriza.html" method="get">
<textarea   name="z"   value="0"cols=20   rows=4   >
Introduceti numele si prenumele</textarea>
<br>
<input type="submit" value="Apasati!">
</form>
</body>
</html>
```



3.13.4. Elementul **label** defineşte o etichetă de control. Cu ajutorul elementului label se poate acționa un buton radio atât prin acționarea lui directă cât şi prin executarea unui click pe textul asociat butonului radio.

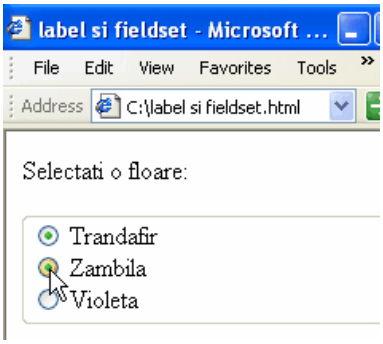
Atribut specific elementului **label:**

Atribut	Valoare	Descriere
for	numele altui câmp	Defineşte cărui element din formular îi este atribuit.

3.13.5. Elementul **fieldset** desenează un dreptunghi în jurul elementelor pe care le conține.

Exemplu de utilizare a elementelor: *label* şi *fieldset*. În acest exemplu am utilizat elementul **fieldset** pentru a încadra într-un dreptunghi butoanele radio. Fiecare buton radio are câte o etichetă (label) ce permite selectarea butonului dacă acționăm asupra textului ce aparține butonului respectiv.

```
<html>
<head><title>label si fieldset</title></head>
<body>
<p>Selectati o floare:</p>
<form name="input" action="">
<fieldset>
<input type="radio" name="x" id="trandafir">
<label for="trandafir">Trandafir</label><br>
<input type="radio" name="zambila" id="zambila">
<label for="zambila">Zambila</label><br>
<input type="radio" name="violeta" id="violeta">
<label for="violeta">Violeta</label>
</fieldset></form>
</body></html>
```



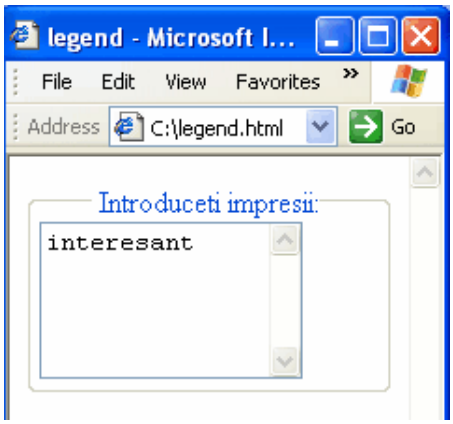
3.13.6. Elementul **legend** defineşte legenda elementului pentru fieldset.

Atribut specific elementului **legend:**

Atribut	Valoare	Descriere
align	top / bottom / left / right	Defineşte modalitatea de aliniere a conținutului din fieldset. Dacă nu este definit nimic, alinierea este *top*.

Exemplu de utilizare a elementului: *legend*.

```
<html>
<head>
<title>legend</title>
</head>
<body>
<fieldset>
<legend align="center">
Introduceti impresii:</legend>
<textarea cols=15 rows=5>interesant
</textarea>
</fieldset>
</form>
</body>
</html>
```



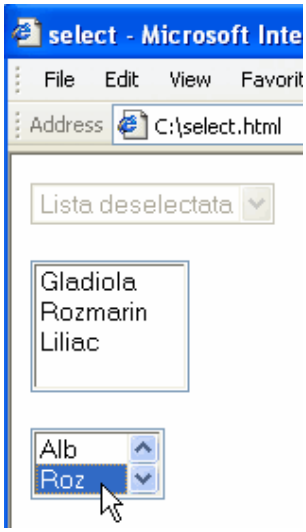
3.13.7. Elementul **select** permite selectarea unui element dintr-o listă.

Atribute specifice elementului **select:**

Atribut	Valoare	Descriere
disabled	disabled	Dezactivează lista.
multiple	multiple	Permite selectarea mai multor itemi în acelaşi timp.
name	nume	Permite definirea unui nume.
size	număr	Defineşte un număr vizibil de itemi din listă.

Exemplu de utilizare a elementului: *select*.

```
<html>
<head><title>select</title>
</head>
<body>
<select disabled>
  <option value ="zero">
     Lista deselectata</option>
</select>
<p>
<select multiple>
  <option value ="gladiola">Gladiola</option>
  <option value ="rozmarin">Rozmarin</option>
  <option value ="liliac">Liliac</option>
</select></p>
<select size=2>
  <option value ="alb">Alb</option>
  <option value ="roz">Roz</option>
  <option value ="rosu">Rosu</option>
  <option value ="galben" >Galben</option>
</select>
</body>
</html>
```



3.13.8. Elementul **optgroup** defineşte opțiunea de grup şi permite gruparea mai multor opțiuni.

Atribute specifice elementului **select:**

Atribut	Valoare	Descriere
disabled	disabled	Dezactivează opțiunea de grup la prima încărcare.
label	text label	Defineşte label pentru un grup de opțiuni.

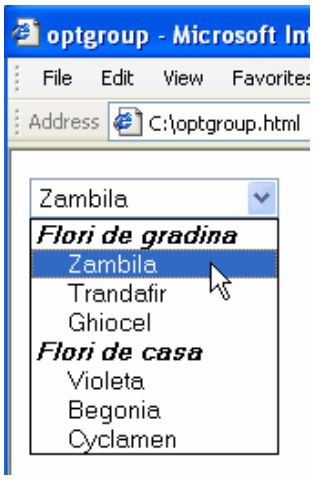
3.13.9. Elementul **option** defineşte opțiunea dintr-o listă.

Atribute specifice elementului **option:**

Atribut	Valoare	Descriere
disabled	disabled	Specifică faptul că opțiunea ar trebui dezactivată la prima încărcare.
label	text label	Defineşte *label* pentru un grup de opțiuni.
selected	selected	Specifică faptul că opțiunea ar trebui să apară selectată.
value	text	Defineşte valoarea opțiunii pentru a fi trimisă către server.

Exemplu de utilizare a elementelor: *optgroup* şi *option*.

```
<html>
<head><title>optgroup</title></head>
<body>
<select>
<optgroup label="Flori de gradina">
<option value ="zambila">Zambila</option>
<option value ="trandafir">Trandafir</option>
<option value ="ghiocel">Ghiocel</option>
</optgroup>
<optgroup label="Flori de casa">
<option value ="violeta">Violeta</option>
<option value ="begonia">Begonia</option>
<option value ="cyclamen">Cyclamen</option>
</optgroup></select>
</body></html>
```



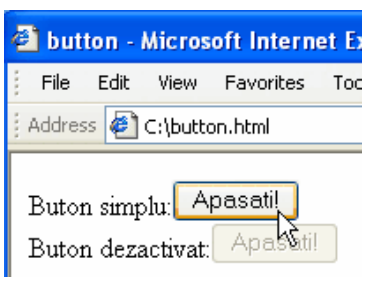
3.13.10. Elementul **button** defineşte un buton pe care puteți "apăsa".

Atribute specifice elementului **option:**

Atribut	Valoare	Descriere
disabled	disabled	Dezactivează butonul.
name	Numele butonului	Specifică numele butonului.
type	button / reset / submit	Defineşte tipul butonului.
value	o valoare	Defineşte valoarea inițială a butonului care va putea fi schimbată pe parcurs.

Exemplu de utilizare a elementului: *button*.

```
<html>
<head><title>button</title>
</head>
<body>
Buton simplu:<button>Apasati!</button><br />
Buton                          dezactivat:<button
disabled>Apasati!</button>
</body></html>
```



3.14. Imagini HTML – HTML Images – sunt utilizate pentru a afişa imagini într-un document cu ajutorul etichetei <**img**>. Atributul cel mai important este **src** (**s**ou**rc**e), care permite definirea sursei imaginii care va fi afişată. Pentru a ține minte mai uşor etichetele, trebuie să reținem că *img* vine de la image (imagine) şi *src* de la source (sursă).

Formatul unei imagini

Imaginile au diferite extensii, în funcție de programul cu care au fost prelucrate sau create. În spațiul Web, formatele care sunt recunoscute de către browser-e sunt: .gif, .jpg şi .jpeg, unde gif este prescurtarea de la Graphics Interchange Format, iar jpg sau jpeg sunt prescurtări de la Joint Photographic Experts Group. Un alt format care este recunoscut doar de către Internet Explorer este .bmp (bitmap), dar pentru faptul că nu este recunoscut de către alte bowser-e, nu vă recomand să-l folosiți într-o pagină Web.

Etichetele imaginilor HTML:
<**img**> - defineşte o imagine;
<**map**> - defineşte maparea unei imagini pe parte de client;
<**area**> - defineşte o suprafață pe care putem da click din interiorul unei imagini mapate.

3.14.1. Elementul **img** defineşte elementul **img** al unei imagini.

Atribute specifice elementului **img:**

Atribut	Valoare	Descriere
alt	text	Defineşte o descriere a imaginii.
src	URL	URL al imaginii care va fi afişată.
align	top / bottom / middle / left / right	Specifică modalitatea de aliniere a imaginii în concordanță cu textul înconjurător.
border	pixeli	Defineşte bordura din jurul imaginii.
height	Pixeli în %	Defineşte înălțimea unei imagini.
hspace	pixeli	Defineşte spațiul alb din stânga şi din dreapta imaginii
ismap	URL	Defineşte imaginea pe parte se server a imaginii mapate.
longdesc	URL	URL al unui document care conține o descriere mai lungă a imaginii.
usemap	URL	Defineşte imaginea pe parte de client a imaginii mapate.
vspace	pixeli	Defineşte spațiul alb din partea de sus şi din partea de jos a imaginii.
width	Pixeli în %	Setează lățimea unei imagini.

Exemplu de utilizare a elementului: *img*. Dacă imaginea se află în acelaşi folder în care avem şi pagina Web, atunci o apelăm prin numele ei, alfel suntem nevoiți să scriem calea unde se află imaginea, pentru ca browser-ul să ştie unde să o caute. Este de preferat să nu dăm toată calea în care se află imaginea, adică: “c:/fisiere/site/imagini/t2.jpg” sau alta, ci doar “imagini/t2.jpg”. Diferența constă în faptul că dacă dăm toată calea, în momentul în care mutăm pagina Web pe alt computer, nu mai vedem imaginile, pentru că au altă cale.

```
<html>
<head><title>imagini</title>
</head>
<body>
<p>
Imagine din acelasi folder:
<img  src="t1.jpg"  alt="Trandafir"  width="70"
height="70" align="bottom"></p>
Imagine din alt folder:
<img     src="imagini/t2.jpg"     alt="Zambila"
width="70" height="70" align="top"></p>
Imagine din spatiul WWW:
<img
src="http://www.einformatica.ro/imagini/copert
a5.jpg" alt="Revista"

width="75" height="100" align="middle"></p>
</body>
</html>
```

Imagine din acelasi folder: 

Imagine din alt folder: 

Imagine din spatiul WWW: 

3.14.2. Elementul **map** defineşte maparea imaginii pe parte de client, asta presupune că imaginea conține zone în care poți executa click pentru a realiza o acțiune.

Atribute specifice elementului **map:**

Atribut	Valoare	Descriere
id	nume	Defineşte un nume unic pentru eticheta *map*.
name	nume	Defineşte un nume unic pentru eticheta *map*.

3.14.3. Elementul **area** defineşte o regiune dintr-o imagine mapată.

Atribute specifice elementului **area:**

Atribut	Valoare	Descriere
alt	text	Specifică textul alternativ pentru suprafața mapată
coords	dacă shape=”rect”, coords= ”stânga, sus, dreapta, jos” dacă shape=”circ”, coords= “centruX, centruY, raza” dacă shape=”poly”, coords=”x1, y1, x2, y2,…,xn,yn”	Specifică coordonatele pentru suprafața pe care se poate executa click.
href	URL	Specifică ținta URL a suprafeței mapate.
nohref	true / false	Exclude o suprafață din maparea imaginii.
shape	rect / rectangle / circ / circle / poly / polygon	Defineşte forma unei suprafețe.
target	_blank/ _parent/ _self/ _top	Defineşte locul unde se va deschide ținta URL. Astfel, pentru: _blank – ținta URL va fi într-o nouă fereastră; _self – ținta URL se va deschide în acelaşi cadru în care s-a dat click; _parent – ținta URL se va deschide parent frameset; _top – ținta URL se va deschide în corpul ferestrei.

Exemplu de utilizare a elementelor: *map* şi *area*. În imaginea alăturată, pentru a mapa imaginea am plecat de la coordonatele x=0 şi y=0 (colțul stânga sus) până la coordonatele x=200 şi y=150 (dreapta jos). Pentru laptop am utilizat o suprafață dreptunghiulară, iar pentru neuron (litera O din cuvântul Informatica) am utilizat un cerc. Pentru cercul neuronului am luat coordonatele: x=50, y=70 şi raza cercului=15.

```
<html>
<head><title>maparea imaginii</title>
</head>
<body>
<p>Dati click pe laptop:</p>
<img src ="coperta5.jpg" border=0 width ="150"
height ="200" alt="Revista Informatica"
usemap="#rev">
<map id ="rev" name="rev">
<area shape ="rect" coords ="7,10,40,30" href
="2005.htm" target ="_blank" alt="2005">
<area shape ="circle" coords ="50,70,15"
  href ="neuron.htm" target ="_blank"
  alt="Neuron">
<area shape ="rect" coords ="50,150,80,180" href
="Laptop.htm" target ="_blank" alt="Laptop">
</map>
</body>
</html>
```

3.15. Fundal HTML – HTML Background – este utilizat pentru a îmbunătăți aspectul unui site Web. Pentru a realiza acest lucru, trebuie găsită o gamă de culori şi nuanțe de culori pentru fundal şi text astfel încât să se potrivească şi să rezulte un aspect plăcut. Un lucru important în alegerea fundalului este realizarea unui contrast destul de puternic pentru ca utilizatorii să poată parcurge conținutul site-ului cu uşurință.

Ca fundal se pot alege nuanțe de culori sau imagini. În alegerea unei imagini pentru fundal trebuie să ținem seama de impactul pe care o să-l aibă asupra utilizatorilor. Dacă imaginea are un contrast puternic, ar fi indicată o prelucrare cu ajutorul unui editor de imagini, cum ar fi: Adobe Photoshop, Corel Drow, Paint Shop Pro şi exemplele pot continua.

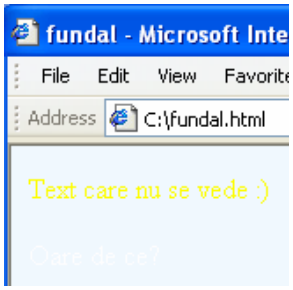
Un alt aspect important ar fi faptul că o imagine de dimensiuni mici se multiplică pe suprafața fundalului şi poate dăuna aspectului final. Pentru a rezolva acest lucru, am putea crea noi o imagine de fundal simetrică, în aşa fel încât dacă aşezăm mai multe imagini una lângă alta, rezultatul să fie plăcut.

Din ce în ce mai puține site-uri profesioniste renunță la imaginile pentru fundal în favoarea unei singure culori, de regulă alb sau gri deschis. Cea mai potrivită culoare pentru text, în cazul acesta este negru.

În continuare voi prezenta exemple pozitive de combinații de culori şi exemple la care imaginea de fundal nu se potriveşte cu nuanța aleasă pentru text.

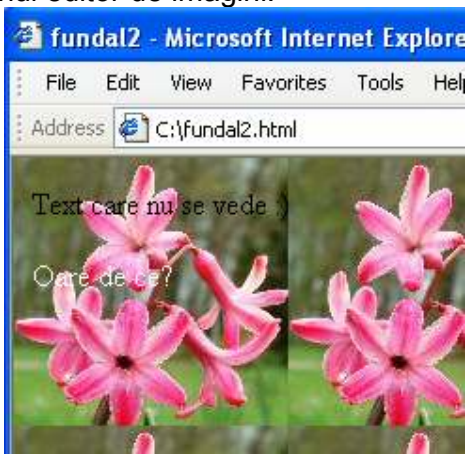
Exemplu de utilizare a elementului: *bgcolor*. Alegerea nuanțelor de culori este total greşită datorită contrastului foarte slab dintre ele. Culorile alb şi galben pentru text sunt dificil de asociat cu fundalul, poate cu excepția culorii negre.

```
<html>
<head><title>fundal</title>
</head>
<body bgcolor="aliceblue" text="yellow">
<p>Text care nu se vede :)</p>
<font color="white">
Oare de ce?</font>
</body></html>
```



Exemplu de utilizare a elementului: *background.* Aici, alegerea ca fundal a unei flori nu este potrivită datorită faptului că imaginea are un contrast foarte puternic. Totusi, vom putea utiliza floarea după o prelucrare cu ajutorul unui editor de imagini.

```
<html>
<head><title>fundal2</title>
</head>
<body background="zambila roz.jpg">
<p>Text care nu se vede :)</p>
<font color="white">
Oare de ce?</font>
</body>
</html>
```



Exemplu Se vede vreo diferență între cele două exemple? Bineînțeles. Acest lucru se realizează după o prelucrare atentă a imaginii. Primul lucru pe care trebuie să-l facem este să avem o imagine într-adevăr bună, adică să aibă o rezoluție mare, după care căutăm un element care ne place pentru a-l prelucra. După ce vă alegeți obiectul, îl izolați, scăpați de fundal, îi aplicați diverse efecte (blur, illumination, ...) în funcție de preferințe şi îi adăugați un fundal potrivit, de preferat o singură culoare. Dacă aveți o imagine ca aceasta, care conține o floare, orice alt element decorativ încarcă pagina. Nu trebuie să uităm scopul final şi anume faptul că un site trebuie să transmită în primul rând informație.

```
<html>
<head><title>fundal cu zambile
roz</title>
</head>
<body background=”zambila6.jpg”
text=”hotpink”>
<h2>
<p align=”center”>Nu-i asa ca va
plac
zambilele?</p></h2>
</body>
</html>
```

OBS Înainte de a alege o imagine pentru fundal trebuie să ne punem următoarele întrebări:

- Imaginea aleasă de noi va creşte cu mult timpul de încărcare a paginii?
- Se potrivesc culorile predominante ale imaginii alese de noi cu restul paginii ?
- Imaginea transmite aceeaşi idee vizitatorului, ca restul site-ului?
- Imaginea de fundal distrage atenția vizitatorului site-ului?

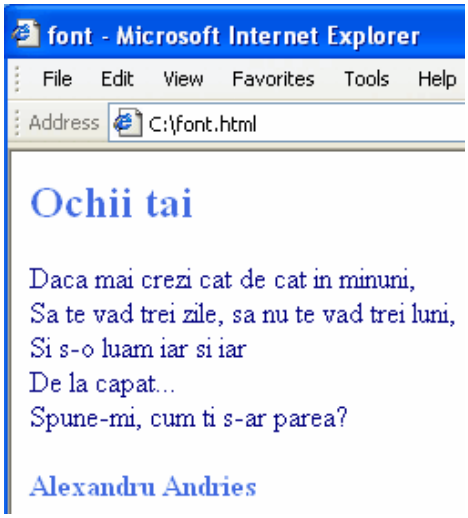
3.16. Fonturi HTML – HTML Fonts realizează formatarea unui text. Formatarea este de preferat să o realizați cu ajutorul stilurilor, ca în exemplul de mai jos.

Atribute specifice elementului **font:**

Atribut	Valoare	Descriere
color	nume culoare / valoare culoare	Defineşte culoarea fontului cu ajutorul unui nume predefinit scris în limba engleză sau cu ajutorul unui număr în hexazecimal..
face	nume font	Defineşte numele fontului, cum ar fi: face=”Arial”
size	valoare număr + valoare număr - valoare număr	Defineşte dimensiunea fontului. Creşte dimensiunea fontului. Scade dimensiunea fontului.

Exemplu În acest exemplu am utilizat câte un stil pentru a formata elementele <h2>, <p> şi <h4> dintr-o pagină web.

```
<html>
<head><title>font</title>
</head>
<body>
<h2                    style="color:royalblue"
face="Verdana">Ochii tai</h2>
<p style="color:navy" face="Monotype Corsiva">
Daca mai crezi cat de cat in minuni,<br/>
Sa te vad trei zile, sa nu te vad trei
luni,<br/>
Si s-o luam iar si iar<br/>
De la capat...<br/>
Spune-mi, cum ti s-ar parea?</p>
<h4 style="color:royalblue" face="Verdana">
Alexandru Andries</h4>
</body>
</html>
```



3.17. Trucuri cu HTML – Până acum am formatat paginile integral, pentru a ajunge la un aspect uniform. Dar dacă vrem ca pagina noastră să arate ca o revistă, cu textul scris în coloane, imaginile aranjate în aceeaşi poziție relativă față de text, indiferent de browser şi de rezoluție, atunci apelăm la coloanele tabelelor.

Exemplu

```
<html>
<head><title>coloane</title></head>
<body>
<center><h2>Clasificarea retelelor in functie de dimensiunile sale</h2></center>
<table>
<tr><td bgcolor="DDDDDD"><b>Retea de birou</b></td>
<td><b>Retea personala</b></td></tr>
<tr><td bgcolor="EEEEEE"> Desk Area Network - DAN în care fiecare componentă a computerului aflat pe birou, cum ar fi: ecranul monitorului, unitățile de CD-ROM, CD-Writer, DVD-ROM, DVD-Writer, Combo Drive, dispozitivele periferice precum: WebCam, imprimanta, scanner-ul, pot fi accesibile din rețea</td>
<td> Home Phoneline Networking Alliance - HPNA utilizează o tehnologie relativ nouă ce permite construirea unei rețele private, utilizând firul de telefon existent. Accesul la Internet se realizează printr-un singur computer, ce permite conectarea la Internet şi a altor computere fără a fi necesar un router.</td> </tr>
</table>
</body>
</html>
```

Evaluare

1. Care este diferența dintre HTML şi SGML?
2. Cum ar trebui să fie fundalul unei pagini Web şi de ce?
3. Creați o pagină Web în care să prezentați date despre dumneavoastră.
4. Care este diferența dintre o legătură relativă şi o legătură absolută?

Răspundeți prin alegerea variantei sau variantelor corecte:

5. Formatul imaginilor care sunt recunoscute de către majoritatea browser-elor web sunt:
   a) .jpg, .gif, .bmp;
   b) .jpeg, .jpg, .bmp;
   c) .jpg, .gif, .jpeg;
   d) .jpg, .jpeg, .gif, .bmp;

Capitolul 4.

C.S.S.

Obiective:
- să înțeleagă noțiunea de stil HTML
- să înțeleagă paşii care se parcurg pentru a se introduce un stil într-o pagină Web.

4.1. Cascading Style Sheet

Până acum am învățat cum să creăm pagini Web frumos formatate, cu text înclinat sau subliniat, de dimensiuni sau culori diferite. Acum vă imaginați că aveți de construit un site care să conțină text formatat în acelaşi fel şi pentru acest lucru trebuie să scrieți cod pentru fiecare porțiune de text separat şi pentru fiecare pagină Web. În felul acesta o pagină Web devine din ce în ce mai încărcată. Putem schimba acest lucru dacă în loc de formatarea separată a porțiunilor de text, am utiliza stiluri scrise separat într-un fişier pe care l-am putea apela din interiorul tuturor paginilor site-ului nostru. Cu alte cuvinte, am putea scrie un singur fişier ce conține CSS, cu ajutorul căruia să formatăm toate paginile Web dintr-un site.

O pagină Web scrisă cu CSS poate fi interpretată mai uşor şi corect de către diferite browser-e, cum ar fi cele destinate persoanelor cu dizabilități. Aceste browser-e parcurg paginile Web şi le transformă. Principalele acțiuni sunt: mărirea sau micşorarea dimensiunii textului afişat, transformarea conținutului paginii Web în fişiere audio sau fişiere interpretabile de către un dispozitiv Braille.

Pentru a interpreta cât mai corect conținutul unei pagini Web, acesta trebuie să poată fi "văzut" simplu, neformatat, toate formatările să fie făcute cu ajutorul CSS-urilor.

În afara acestor avantaje, CSS este folosit pentru a putea crea pagini cu stil, care să încânte privirea utilizatorilor.

Avantajele utilizării CSS:
- paginile se încarcă mai repede;
- se poate modifica foarte uşor aspectul unui întreg site;
- avem mult mai puțin cod de scris pentru o pagină Web;
- site-ul are mai multe şanse să fie interpretat corect de către cât mai multe browser-e;

Etichetele stilurilor HTML:
<**style**> - defineşte stilul unui document HTML;
<**link**> - defineşte o resursă ca referință;
<**div**> - defineşte o secțiune dintr-un document (element de nivel bloc generic);
<**span**> - defineşte ataşarea CSS la o mică porțiune a unui rând dintr-un document (element inline generic);
<**font**> - defineşte dimensiunea, culoarea, tipul textului;
<**basefont**> - defineşte fontul de bază dintr-un document;
<**center**> - centrează textul dintr-un document.
OBS. În limbajul HTML există două tipuri de elemente: *elemente de nivel bloc* (care sunt afişate cu începere de pe un rând nou; *elemente inline* (care sunt afişate pe acelaşi rând). Elementele ** ** şi **** pot fi utilizate într-o gamă largă de situații şi acceptă atribute comune, cu excepția că elementul nu acceptă atributul **align**, deoarece este element de interior de rând. Ambele acceptă atributul **title** ce impune browserului afişarea unui text pe ecran atunci când mouse-ul rămâne timp de câteva secunde deasupra conținutului definit de ** **, respectiv ****.

4.1.1. Elementul **style** se poate afla în antetul documentului, în interiorul unui element dintr-o pagină Web sau în exterior.

Atribute specifice elementului **style:**

Atribut	Valoare	Descriere
type	text/css	Defineşte conținutul unui type.
media	screen / tty / tv / projection / handheld / print / braille / aural / all	Defineşte destinația mediului ales pentru stil.

Un stil este compus din două părți: selectorul şi declarația. Selectorul este un element HTML ce poate realiza formatarea paginii Web (cum ar fi: body, H1, p) şi acționează ca o legătură între pagina Web şi stil. Declarația conține elementele ce vor duce la formatarea propriu-zisă a documentului.

Selector	Declarația	
	Proprietatea	Valoarea
Body	{ background-color:	beige}
H2	{color:	red}
P	{font-family:	"Verdana"}

În exemplul de mai sus, stilul H2 declarat în antetul paginii Web, va schimba culoarea tuturor textelor din pagină, ce utilizează H2.

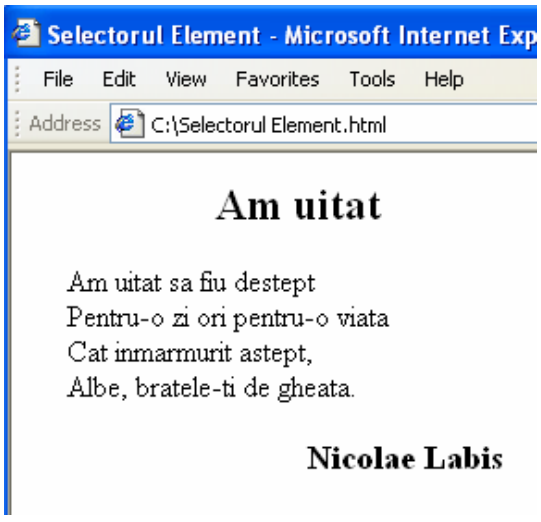
Un selector direcționează un stil către un anumit loc dintr-un document HTML. El poate fi construit în multe feluri, prin combinații între blocurile:

- **Element**
- **Class**
- **Id**

Selectorul ***Element*** poate fi definit cu ajutorul etichetelor de genul: <H2>, <p> care formatează o porțiune de text, într-un anumit fel. Odată definit stilul cu ajutorul acestor etichete, atunci când le utilizăm în cadrul paginii Web, ele vor avea acelaşi efect în toată pagina. De exemplu, dacă pentru H2 definim în cadrul stilului şi culoarea maron, atunci, <H2> va avea această culoare oriunde va fi utilizat în cadrul paginii Web. În concluzie, acest tip de selector are dezavantajul că odată definită eticheta, nu mai poate fi utilizată cu efectul ei implicit, ci doar cu cel dat de stilul definit.

Exemplu de utilizare a selectorului ***Element***.

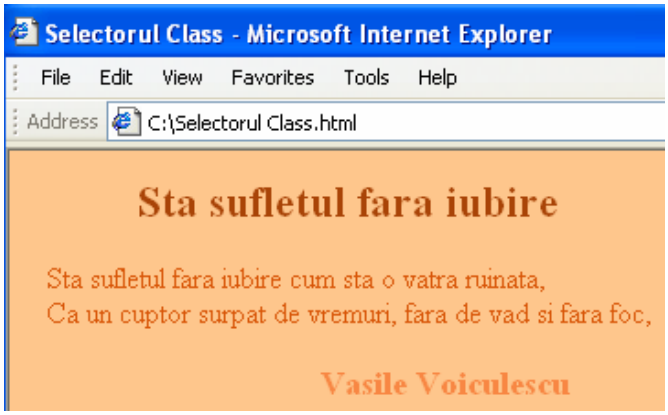
```
<html>
 <head><title>Selectorul Element</title>
<style type="text/css">
H2 { margin-left: 100px; }
H3 { margin-left: 150px;}
p { margin-left: 20px;}
</style>
</head>
<body>
<H2>Am uitat </H2>
<p> Am uitat sa fiu destept<br />
Pentru-o zi ori pentru-o viata<br />
Cat inmarmurit astept,<br />
Albe, bratele-ti de gheata.<br />
</p>
<H3>Nicolae Labis </H3>
</body>
</html>
```



Selectorul ***Class*** poate fi definit cu ajutorul unui nume specific, diferit de cuvintele utilizate în cadrul etichetelor HTML. Acest lucru poate fi interpretat ca un avantaj prin faptul că nu se schimbă efectele implicite ale etichetelor HTML, ci se creează stiluri cu nume distincte, cum ar fi: stil1, stil2, titlul_poeziei. Pentru a defini un stil, scriem .numele_class urmat de formatările specifice stilului, cum ar fi: { font-size: 14px; }. Pentru a putea fi utilizat în cadrul corpului paginii, apelăm stilul din cadrul oricărei etichete dorim, cum ar fi: sau <body class="numele_class">.

Exemplu de utilizare a selectorului ***Class***.

```
<html>
 <head><title>Selectorul
Class</title>
<style type="text/css">
.fundal {background: #FFC68C;}
.titlul { margin-left: 60px;
          color:#A94407;  }
.versuri { margin-left: 10px;
           color: #D95709;}
.autor { margin-left: 160px;
         color: #F88743;}
</style>
</head>
<body class="fundal">
<H2 class="titlul">Sta sufletul fara
iubire </h2>
<p class="versuri">Sta sufletul fara
iubire cum sta o vatra ruinata,<br />
Ca un cuptor surpat de vremuri, fara
de vad si fara foc,<br /></p>
<H3  class="autor">Vasile  Voiculescu
</H3>
</body></html>
```



Selectorul ***ID*** este unic într-un document HTML şi se utilizează cu semnul # înaintea numelui. De exemplu, se poate defini un selector: #xyz { color: red; } care se apelează <p IP=xyz>. Acest selector este identic cu selectorul ***class*** cu excepția faptului că putem avea un singur ***ID*** într-un document HTML.

Exemplu de utilizare a selectorului ***ID***.

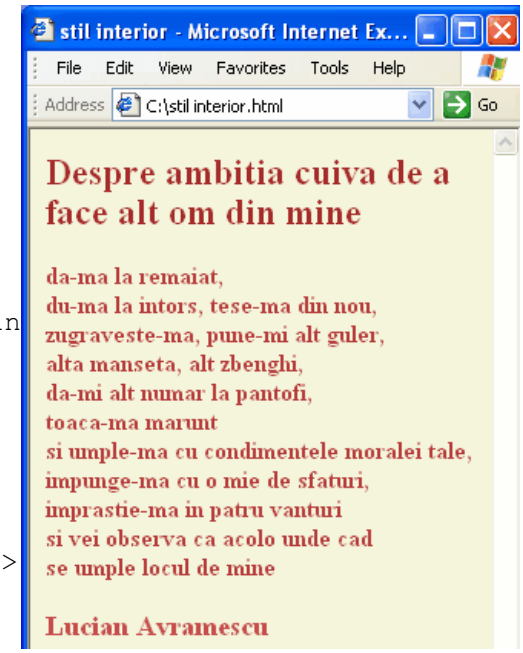
```
<html>
 <head><title>Selectorul ID</title>
<style type="text/css">
#fundal {background: #E5E5E5;
         color: #330099;}
#titlul { margin-left: 60px; }
#versuri { margin-left: 10px; }
#autor { margin-left: 160px; }
</style>
</head>
<body ID=fundal>
<H2 ID=titlul>Te port in mine </h2>
<p ID=versuri>
Te port in suflet, ca pe-un vas de pret,<br />
Ca pe-o comoara-nchisa cu peceti<br />
<H3 ID="autor">Zorica Latcu   </H3>
</body></html>
```

CSS reprezintă un mecanism de adăugare a stilurilor, cum ar fi: *fonts, colors*, într-o pagină Web. Sunt trei modalități de introducere a unui stil:

- **Stil extern** - este un fişier separat de pagina web, care se salvează cu extensia .css şi este util în cazurile când acelaşi stil apare în mai multe pagini Web. În felul acesta, prin schimbarea unui stil, toate paginile care-l folosesc se vor modifica.
- **Stil intern** – se utilizează de către o singură pagină Web şi se află în codul sursă al paginii, în antet.
- **Stil inline** – se utilizează doar ca stil unic aplicabil unui singur element, nu întregi pagini Web.

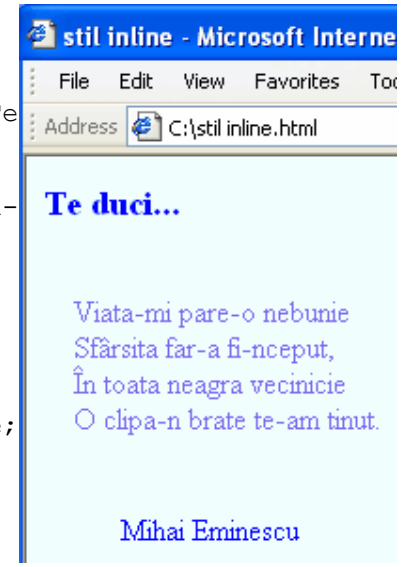
Exemplu de utilizare a stilului ***interior***.

```
<html>
<head><title>stil interior</title>
<style type="text/css">
body {background-color:beige }
p {margin-left:20px }
h2 {color:brown}
h4 {color:#B63B3B}
h3 {color:#C74C4C}
</style></head>
<body>
<h2>Despre ambitia cuiva de a face alt om din
mine </h2>
<h4>da-ma la remaiat,<br />
du-ma la intors, tese-ma din nou,<br />
zugraveste-ma, pune-mi alt guler,<br />
alta manseta, alt zbenghi,<br />
da-mi alt numar la pantofi,<br />
toaca-ma marunt<br />
si umple-ma cu condimentele moralei tale,<br />
impunge-ma cu o mie de sfaturi,<br />
imprastie-ma in patru vanturi<br />
si vei observa ca acolo unde cad<br />
se umple locul de mine<br /><br /></h4>
<h3>Lucian Avramescu</h3>
</body></html>
```



Exemplu de utilizare a stilului ***inline***.

```
<html>
<head><title>stil inline</title></head>
<body style="background:azure">
<h3  style="font-weight:  bold;  color:blue">Te
duci... </h3>
<br />
<p    style="color:mediumslateblue;    margin-
left:15px">
Viata-mi pare-o nebunie<br />
Sfârsita far-a fi-nceput,<br />
În toata neagra vecinicie<br />
O clipa-n brate te-am tinut.
</p>
<p  style="font-weight:  normal;  color:blue;
margin left:40px">Mihai Eminescu</p>
</body>
</html>
```



Exemplu de utilizare a stilului ***extern***.
Programul principal:

```
<html>
<head><title>stil extern</title>
<link      rel="stylesheet"      type="text/css"
href="stil1.css" >
</head>
<body class="stil1">
<h3>Peste varfuri</h3>
<p>
De ce taci, cand fermecata<br />
Inima-mi spre tine-ntorn?<br />
Mai suna-vei, dulce corn,<br />
Pentru mine vre odata?<br />
</p>
<p>Mihai Eminescu</p>
</body></html>
```

Fişierul "stil1.css":

```
.stil1
{  font-family: "Courier New", Courier, mono;
   font-size: 14px;
   font-style: normal;
   color: #0000FF;
   background-color: #33CCFF;
   text-align: center;
   border-bottom-width: medium}
```

4.1.2. Elementul **link** realizează o legătură între două documente.

Atribute specifice elementului **link:**

Atribut	Valoare	Descriere
charset	charset	Defineşte codul caracterului țintei URL. Valoarea inițială este "ISO-8859-1"
href	URL	Ținta URL a resursei.
hreflang	codul limbajului	Defineşte limbajul de bază a țintei URL.
media	all / braille / print / projection / screen / speech	Defineşte tipul de dispozitiv pe care va fi afişat documentul.
rel	alternate / appendix / bookmark / chapter / contents / copyright / glossary / help / home / index / next / prev / section / start / stylesheet / subsection	Defineşte relația dintre documentul curent şi ținta documentului.
rev	alternate / appendix / bookmark / chapter / contents / copyright / glossary / help / home / index / next / prev / section / start / stylesheet / subsection	Defineşte relația dintre ținta documentului şi documentul curent.
target	_blank _self _top _parent	-ținta URL se va deschide într-o nouă fereastră -ținta URL se va deschide în acelaşi cadru în care s-a executat click. -ținta URL se va deschide în *parent frameset* -ținta URL se va deschide în corpul ferestrei
type	MIME_type: text/css sau text/javascript sau image/gif	Specifică tipul MIME al țintei URL.

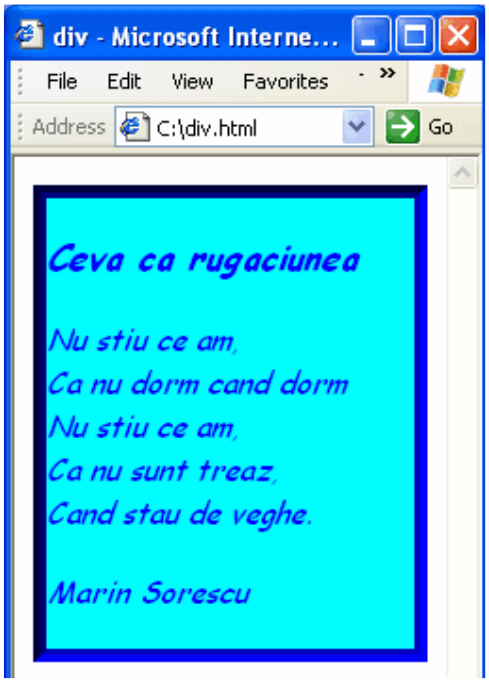
4.1.3. Elementul **div** defineşte o secțiune dintr-un document.

Atribute specifice elementului **div:**

Atribut	Valoare	Descriere
align	left / right / center / justify	Defineşte modalitatea de aliniere din elementul *div*.

Exemplu de utilizare a elementului **div**. În acest exemplu am utilizat stilul div pentru a scrie textul cu culoarea albastră.

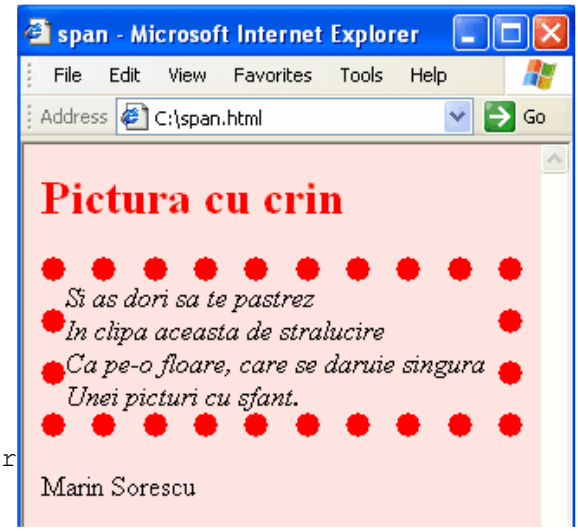
```
<html>
<head><title>div</title>
<style type="text/css">
div
 {color:blue; background:cyan;
font-family:cursive;
font-style:italic;
font-size:12pt;
border-style: inset;
border-width: 0.07in;
border-color: blue;}
</style>
</head>
<body>
<div>
<h3>Ceva ca rugaciunea</h3>
Nu stiu ce am,<br />
Ca nu dorm cand dorm<br />
Nu stiu ce am,<br />
Ca nu sunt treaz,<br />
Cand stau de veghe.<br />
<p>Marin Sorescu</p>
</div>
</body></html>
```



4.1.4. Elementul **span** se utilizează pentru a grupa mai multe elemente inline într-un document.

Exemplu de utilizare a elementului **span**.

```
<html>
<head><title>span</title>
</head>
<body style="background: mistyrose">
<span type="text/css" style="color:red;
font-size:20pt;
font-weight:bold;">
Pictura cu crin</span>
<p type="text/css" style="border:dotted;
border-width:0.15in;
border-color:red;
font-style:italic;">
Si as dori să te pastrez<br />
In clipa aceasta de stralucire<br />
Ca pe-o floare, care se daruie singura<br
/>
Unei picturi cu sfant.</p>
<p>Marin Sorescu</p>
</span>
</body></html>
```



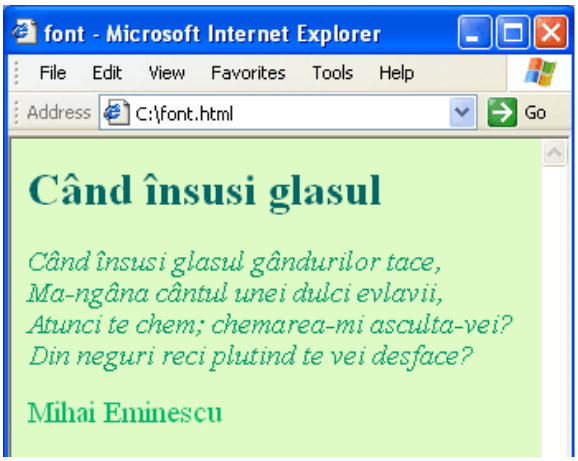
4.1.5. Elementul **font** specifică dimensiunea, culoarea, tipul textului (*ex.*font=”Times New Roman”).

Atribute specifice elementului **font:**

Atribut	Valoare	Descriere
color	nume culoare / valoare culoare	Defineşte culoarea textului.
face	listă nume font	Defineşte fontul textului din elementul *font*.
size	un număr între 1 şi 7	Defineşte dimensiunea textului din elementul *font*.

Exemplu de utilizare a elementului **font**.

```
<html>
<head><title>font</title>
</head>
<body style="background: #DEFAC5">
<font type="text/css"
style="color:#006666;
font-size:20pt;
font-weight:bold;">
Când însuşi glasul </font><br /><br />
<font type="text/css"
style="color:#009966;
font-size:13pt;
font-style:italic;">
Când însuşi glasul gândurilor tace,<br />
Mă-ngână cântul unei dulci evlavii,<br />
Atunci te chem; chemarea-mi asculta-
vei?<br />
Din neguri reci plutind te vei desface?
</font><br />
<font type="text/css"
style="color:#00BB66;
font-size:14pt;
line-height:50px;">
Mihai Eminescu</font>
</body>
</html>
```



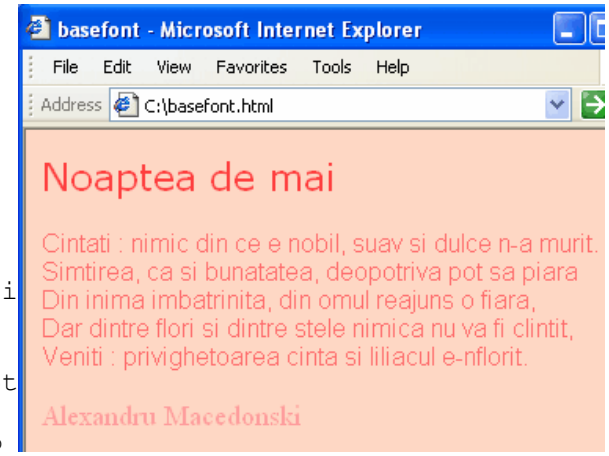
4.1.6. Elementul **basefont** defineşte fontul de bază dintr-un document. Dacă nu este setat, fontul de bază este inițializat cu 3.

Atribute specifice elementului **basefont:**

Atribut	Valoare	Descriere
color	nume culoare / valoare culoare	Defineşte culoarea textului.
face	listă nume font	Defineşte fontul textului din elementul *font*.
size	un număr între 1 şi 7	Defineşte dimensiunea textului din elementul *font*.

Exemplu de utilizare a elementului **basefont**. Acest exemplu, chiar dacă este reuşit, nu utilizează stiluri. În concluzie nu este acceptat de HTML 4.01 şi nici nu este suportat în XHTML 1.0 Strict DTD.

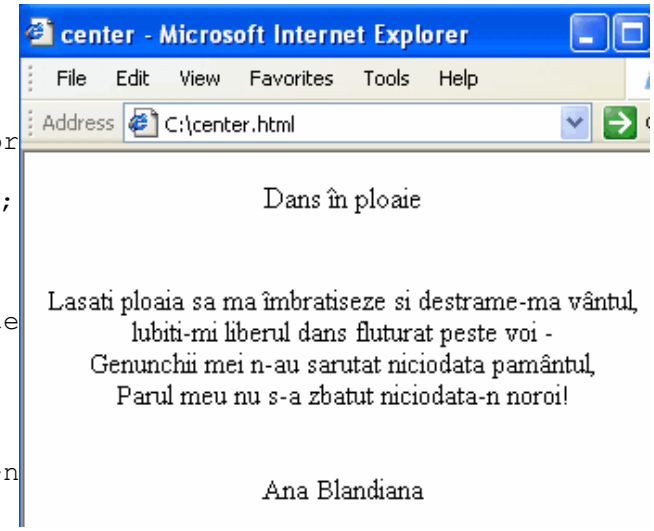
```
<html>
<head><title>basefont</title>
</head>
<body style="background: #FFD7C4">
<basefont size=5 color="#FF3E3E"
face="Verdana">
Noaptea de mai </basefont><br /><br />
<basefont size=3 color="#FF6C6C"
face="Arial">
Cintati : nimic din ce e nobil, suav si
dulce
n-a murit.<br />
Simtirea, ca si bunatatea, deopotriva pot
sa piara<br />
Din inima imbatrinita, din omul reajuns o
fiara,<br />
Dar dintre flori si dintre stele nimica
nu va
fi clintit,<br />
Veniti : privighetoarea cinta si liliacul
e-nflorit.
</basefont><br /><br />
<basefont size=4 color="#FF9D9D"
face="Times">
Alexandru Macedonski</basefont>
</body></html>
```



4.1.7. Elementul **center** centrează textul dintr-un document şi se scrie între etichetele ** ** şi ** **.

Exemplu de utilizare a elementului **center**. În acest exemplu, titlul este scris centrat cu ajutorul etichetei (nu este stil), iar pentru restul textului, acelaşi efect este produs cu ajutorul stilului CSS (varianta acceptată de HTML 4.01 şi XHTML 1.0 Strict DTD).

```
<html>
<head><title>center</title>
</head>
<body>
<center>Dans în ploaie </center><br
/><br />
<p type="text/css" style="font-size=15;
text-align:center;">
Lăsaţi ploaia să mă îmbrăţişeze şi
destrame-mă vântul,<br />
lubiţi-mi liberul dans fluturat peste
voi
-<br />
Genunchii mei n-au sărutat niciodată
pământul,<br />
Părul meu nu s-a zbătut niciodată-n
noroi!<br />
<br /><br />
Ana Blandiana</p>
</body>
</html>
```



4.2. Atribute CSS

Pentru fundal se utilizează atributele:

- **background** – introduce imagini sau culori de fundal. Imaginile se introduc cu ajutorul URL-ului, iar culorile cu ajutorul numelui predefinit în limba engleză (*Ex.* white, green) sau a valorii RGB (*Ex.* #FFFFFF pentru alb).
- **color** – introduce culoarea textului paginii Web cu ajutorul numelui sau al valorii RGB.

Fonturile se introduc cu ajutorul următoarelor atribute:

- **font-family** – introduce tipul fontului cu ajutorul numelui sau al familiei de fonturi (*Ex* Arial)
- **font-size** – introduce dimensiunea fontului, care poate fi dată în pixeli (px), inch (in), puncte (pt) sau centimetri (cm).
- **font-style** – introduce un text cursiv înclinat (*Ex* italic) sau normal
- **font-weight** – introduce grosimea fontului care poate fi dată cu ajutorul dimensiunilor predefinite de la subțire la gros: extra-light, demi-light, medium, demi-bold, bold, extra-bold.

Aranjarea textului este destul de dificilă dacă nu avem la îndemână instrumentele necesare. Pentru a rezolva această problemă, stabilim marginile cu ajutorul atributelor:

- **margin-left** – stabileşte distanța dintre textul documentului şi marginea din stânga a paginii;
- **margin-right** - stabileşte distanța dintre textul documentului şi marginea din dreapta a paginii;
- **margin-top** – stabileşte distanța dintre textul documentului şi marginea de sus a paginii;

Aranjarea secvențelor de text se realizează şi cu ajutorul atributelor:

- **text-align** – realizează alinierea textului şi poate avea valorile:
  - **left** (stânga), de exemplu: text-align=”left”;
  - **center** (centru);
  - **right** (dreapta);
  - **justify** (stânga şi dreapta);
- **line-height** – stabileşte distanța dintre rândurile textului;
- **text-decoration** – stabileşte anumite decorații pentru diferite porțiuni de text, cu ajutorul atributelor:
  - **none** (nimic);
  - **underline** (subliniat);
  - **italic** (înclinat);
  - **line-through** (tăiat)
- **text-indent** – stabileşte distanța primului rând dintr-un paragraf, față de marginea din stânga;
- **border-style** – stabileşte tipul chenarului în care va fi încadrat textul şi are atributele:
  - **none** (nimic);
  - **groove;**
  - **dotted** (chenar cu buline);
  - **dashed** (chenar cu linie întreruptă);
  - **solid** (chenar plin)**;**
  - **double** (chenar cu două linii)**;**
  - **ridge** (chenar cu două nuanțe de culori)**;**
  - **inset;**
  - **outset**
- **border-width** – stabileşte grosimea chenarului;
- **border color** – stabileşte culoarea chenarului cu ajutorul numelui sau al valorii în hexazecimal

În continuare voi prezenta un exemplu de utilizare a elementelor CSS, în care nu folosim elementele predefinite, ci definim altele noi. Pentru fiecare porțiune de text voi utiliza un

stil definit în antetul documentului. La utilizarea stilului respectiv, atributul class va primi numele definit în antet.

```
<html>
<head>
<title>class</title>
<style type="text/css">
.titlul_poeziei {   font-family: Georgia, "Times
    New Roman", Times, serif; font-size: 24px;
    font-style: normal; line-height: 10mm; font-
    weight: bolder; font-variant: normal; color:
    #0000FF; letter-spacing: 2mm; word-spacing:
    10mm;     text-align: center; vertical-align:
    top;  border-color:  black  black  #0000FF;
    border-bottom-width: medium}
.textul_poeziei {   font-family: Georgia, "Times
    New Roman", Times, serif; font-size: 14px;
    font-style:  oblique;  font-weight:  lighter;
    color: #0033FF}
.autorul_poeziei {  font-family: Arial, Helvetica,
    sans-serif;   font-size:   16px;   font-style:
    italic; line-height: 20mm; font-weight: bold;
    color: #0000FF; text-indent: 50pt}
</style>
</head>
<body bgcolor="#FFFFFF" text="#000000">
<p  class="titlul_poeziei">Fior </p>
<p   class="textul_poeziei">Cine   esti,   ori   ce
esti,<br>
  Abur ori duh cobor&acirc;t din povesti,<br>
  Unda prelinsa sa ma &icirc;nvenine,<br>
  Stea fulgerata &icirc;n mine? </p>
<p class="autorul_poeziei">Nicolae Labis</p>
</body></html>
```

OBS

Un element HTML poate avea doar un singur atribut class. Este greşit să scriem

```
<p class="titlul_poeziei" class="textul_poeziei">
      Fior
</p>
```

În exemplul de mai sus, stilurile: .titlul_poeziei, .textul_poeziei, .autorul_poeziei pot fi utilizate de către orice elemente HTML, nu numai de către elementul <p>.

Dimensiunea textului

După cum am spus mai sus, textul poate fi redimensionat cu ajutorul următoarelor elemente:

Point sau pt, Dimensiunea standard a textului afişat într-un editor de texte este de 10pt. Când utilizați pt, cel mai indicat este să folosiți dimensiuni mai mari sau egale cu 10 pt. Acest mod de dimensionare a textului este utilizat în special de către cei care doresc să tipărească documentul şi doresc o bună încadrare a textului în pagină.

Exemplu: p1 {font-size: 20 pt;}

Ems este o unitate de măsură relativă la orice element părinte a textului care-l conține. De exemplu, dacă definim elementul părinte

p1 {font-size: 20 pt;}, după care definim

p2{font-size: 2em;}

dimensiunea celui de-al doilea text este de două ori mai mare decât a primului text.

Pixel sau **px** este o modalitate de a dimensiona textul cu ajutorul dimensiunii unui pixel. Acest mod este util pentru afişarea unui text, dar nu şi pentru printarea lui.

P3 {font-size: 33px; }

Procent sau **%** este o dimensiune relativă la orice element părinte, ca şi ems.

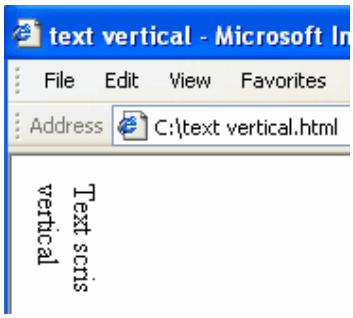
P4 {font-size: 150%; }

Alte unități de măsură care pot fi utilizate sunt: in (inch), ex (x-height), cm (centimetri), mm (milimetri), pc (picas). Aceste elemente pot da o dimensiune clară a textului, spre deosebire de următoarele care sunt parte a specificațiilor CSS şi definesc o dimensiune aproximativă: xx-small, x-small, small, medium, large, x-large, xx-large. Am spus aproximativă deoarece browser-ele interpretează puțin diferit aceste elemente.

4.3. Aplicații interesante cu CSS

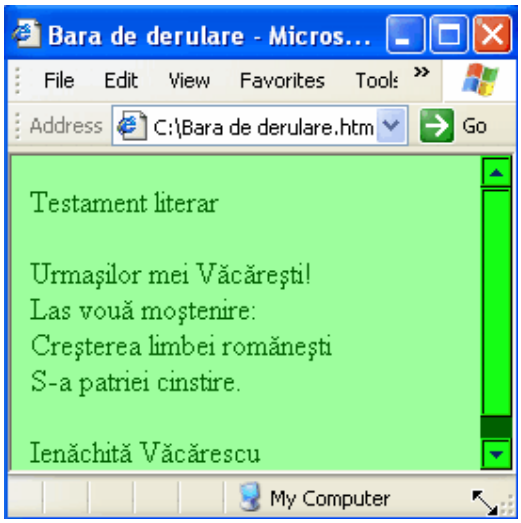
Text scris vertical În acest exemplu vom utiliza un stil ce face ca textul să apară afişat pe verticală.

```
<html>
<head>
<title>text vertical</title>
<style>
.stil
 {  writing-mode: tb-rl;
   filter:   flipH() flipV();  }
</style>
</head>
<body>
<span class="stil">
Text scris vertical
</span>
</body></html>
```

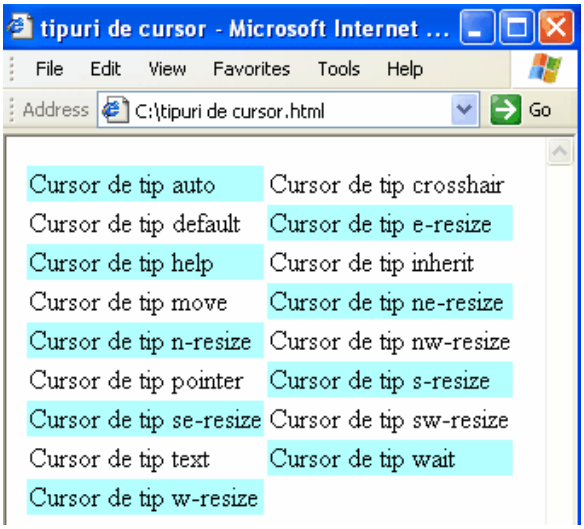


Scroll colorat Stilurile se pot aplica şi asupra barei de derulare, după cum putem vedea în exemplul acesta .

```
<html>
<head>
<title>Bara de derulare</title>
<style>
body
 {  background-color: #9DFF9D;
    scrollbar-3dlight-color:#15FF15;
    scrollbar-arrow-color:#001155;
    scrollbar-base-color:#033000;
    scrollbar-darkshadow-color:#000d00;
    scrollbar-face-color:15FF15;
    scrollbar-highlight-color:#0d0000;
    scrollbar-track-color:#006000;
    scrollbar-shadow-color:#22d000  }
p  {color: #006000;}
</style></head>
<body>
<p>Testament literar <br /><br />
Urma&#351;ilor                              mei
   V&#259;c&#259;re&#351;ti!<br />
Las vou&#259; mo&#351;tenire:<br />
Cre&#351;terea  limbei  rom&#259;ne&#351;ti<br
   />
S-a patriei cinstire.<br /><br />
Ien&#259;chit&#259; V&#259;c&#259;rescu</p>
</body></html>
```



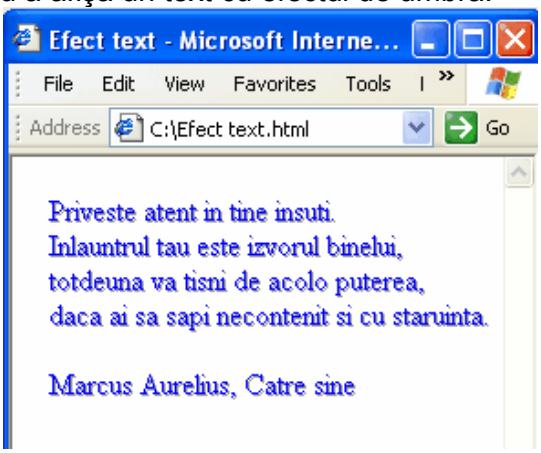
Tipuri de cursor Am utilizat în acest exemplu toate tipurile de cursor pe care le putem utiliza. După cum veți putea vedea mai jos, în dreptul cursorului de tip help apare un semn de întrebare, cursorul de tip crosshair apare ca o cruciulița, cel de tip pointer ca o mânuță. Toate aceste tipuri de cursor le veți putea utiliza în diverse exemple, în funcție de situație.




```
<html>
<head><title>tipuri de cursor</title>
<style type=text/css>
.s1 {background-color:#B3FFFF;
      cursor:auto;}
.s2 {cursor:crosshair;}
.s3 {cursor:default;}
.s4 {background-color:#B3FFFF;
      cursor:e-resize;}
.s5 {background-color:#B3FFFF;
      cursor:help;}
.s6 {cursor:inherit;}
.s7 {cursor:move;}
.s8 {background-color:#B3FFFF;
      cursor:ne-resize ;}
.s9 {background-color:#B3FFFF;
      cursor:n-resize;}
.s10 {cursor:nw-resize;}
.s11 {cursor:pointer;}
.s12 {background-color:#B3FFFF;
       cursor:s-resize;}
.s13 {background-color:#B3FFFF;
       cursor:se-resize;}
.s14 {cursor:sw-resize;}
.s15 {cursor:text;}
.s16 {background-color:#B3FFFF;
       cursor:wait;}
.s17 {background-color:#B3FFFF;
       cursor:w-resize;}
</style>
</head>
<body>
<table>
<tr><td class=s1>Cursor de tip auto</td>
    <td     class=s2>Cursor     de     tip
crosshair</td></tr>
<tr><td     class=s3>Cursor     de     tip
default</td>
    <td   class=s4>Cursor   de   tip   e-
resize</td></tr>
<tr><td class=s5>Cursor de tip help</td>
    <td     class=s6>Cursor     de     tip
inherit</td></tr>
<tr><td class=s7>Cursor de tip move</td>
    <td   class=s8>Cursor   de   tip   ne-
resize</td></tr>
<tr><td   class=s9>Cursor   de   tip   n-
resize</td>
    <td   class=s10>Cursor   de   tip   nw-
resize</td></tr>
<tr><td     class=s11>Cursor     de     tip
pointer</td>
```

```
    <td   class=s12>Cursor   de   tip   s-
resize</td></tr>
<tr><td   class=s13>Cursor   de   tip   se-
resize</td>
    <td   class=s14>Cursor   de   tip   sw-
resize</td></tr>
<tr><td class=s15>Cursor de tip text</td>
    <td     class=s16>Cursor     de     tip
wait</td></tr>
<tr><td   class=s17>Cursor   de   tip   w-
resize</td></tr>
</table>
</body>
</html>
```

Text cu umbră În acest exemplu am utilizat stiluri pentru a afişa un text cu efectul de umbră.




```
<html>
<head>
<title>Efect text</title>
<STYLE type=text/css>
DIV
  {  position: absolute;
     font-size: 16px;
     left: 20px;
     top: 20px;  }
</STYLE>
</head>
<body style="margin: 0px">
<STYLE type=text/css>
.umbra
  {  left: 1px;
     top: 1px;
     color: #999999;   }
.original
  {  left: 0px;
     top: 0px;
     color: #0000FF;  }
</STYLE>
<DIV class=citat>
  <DIV  class=umbra>Priveste  atent  in  tine
insuti. <br />
  Inlauntrul  tau  este  izvorul  binelui,  <br
/>
  totdeuna va tisni de acolo puterea, <br />
  daca   ai   sa   sapi   necontenit   si   cu
staruinta.<br /><br />
  Marcus Aurelius, Catre sine</DIV>
  <DIV class=original>Priveste atent in tine
insuti. <br />
  Inlauntrul  tau  este  izvorul  binelui,  <br
/>
  totdeuna va tisni de acolo puterea, <br />
  daca   ai   sa   sapi   necontenit   si   cu
staruinta.<br /><br />
  Marcus Aurelius, Catre sine</DIV>
</DIV>
</body>
</html>
```

Evaluare

6. Care sunt modalitățile de introducere a unui stil?
7. Cum putem modifica rapid aspectul unui întreg site Web?
8. Creați o pagină Web ce prezintă formația de muzică preferată de dumneavoastră, în care să utilizați numai stiluri CSS.
9. Creați trei pagini Web ce conțin stilurile: extern, intern şi inline. Faceți o analiză comparativă între cele trei modalități de introducere a stilurilor, insistând pe avantajele şi deosebirile dintre ele.
10. Browser-ele ignoră “spațiile albe”?
    a) da;
    b) nu

Capitolul 5.

JavaScript

Obiective:
- să înțeleagă noțiunea de JavaScript
- să înțeleagă avantajele şi dezavantajele utilizării scripturilor JavaScript.

5.1. Introducere

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. 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! ☺

5.2. Unde scriem script-urile JavaScript

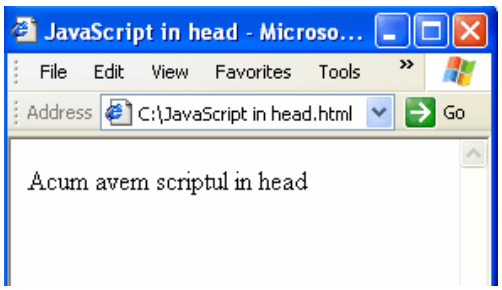
Ca şi un stil CSS, un script JavaScript poate fi scris atât în pagina Web, în antet sau în corpul documentului, cât şi în exteriorul ei, ca fişier separat. Fişierele care conțin scripturi JavaScript se salvează cu extensia .js.

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.

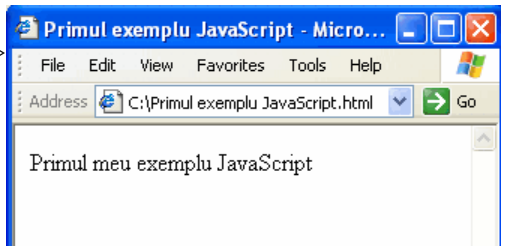
Exemplu Cum scriem un script JavaScript în *head.*

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



Exemplu Cum scriem 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.

5.3. Operatori JavaScript

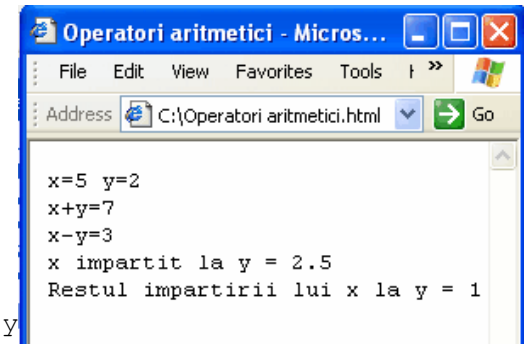
5.3.1. Operatori aritmetici

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.

5.3.2. 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;

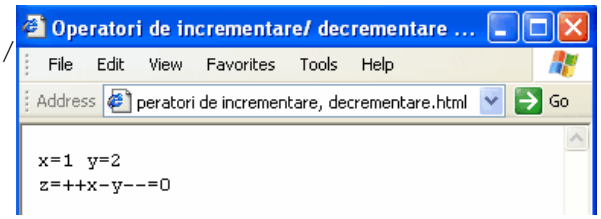
care la rândul lor sunt de două feluri:

- **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>
```



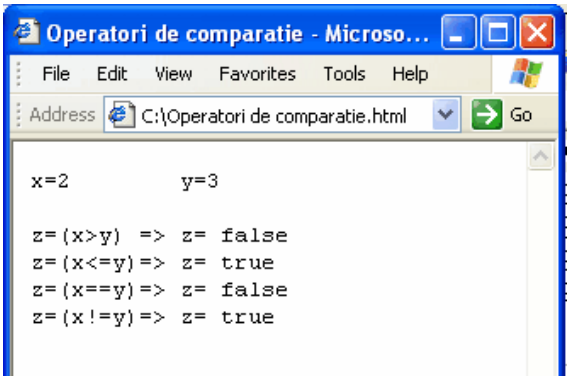
5.3.3. Operatori de comparație

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>.

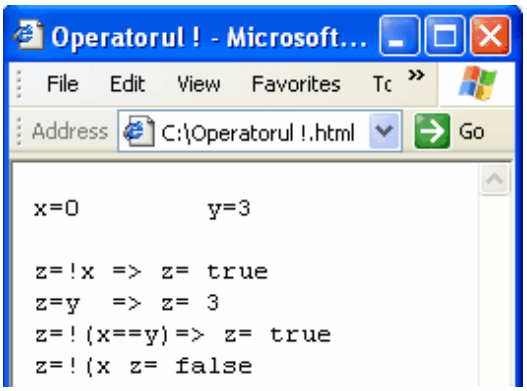
5.3.4. Operatori logici (booleeni)

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>
```

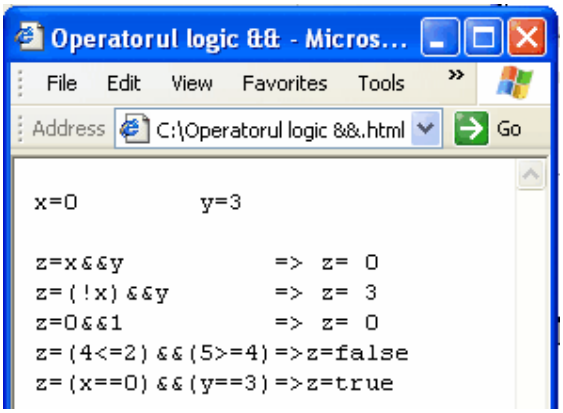


&&	true	false
true	true	false
false	false	false

||	true	false
true	true	true
false	true	false

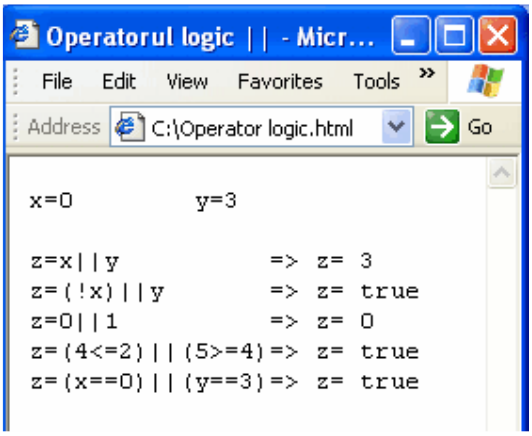
Exemplu de utilizare a operatorului &&.

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



Exemplu de utilizare a operatorului ||.

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



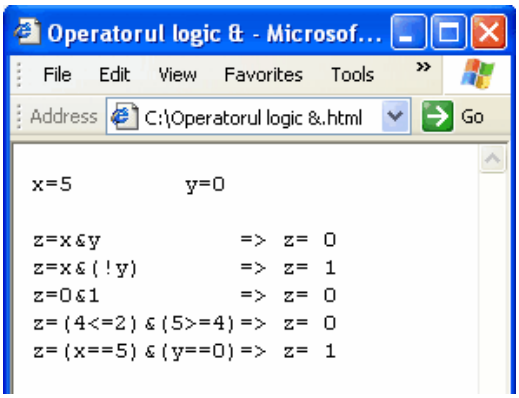
5.3.5. Operatori logici pe biți (bitwise)

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

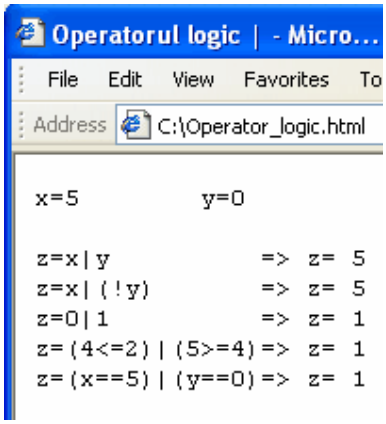
Exemplu de utilizare a operatorului &.

```
<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>
```



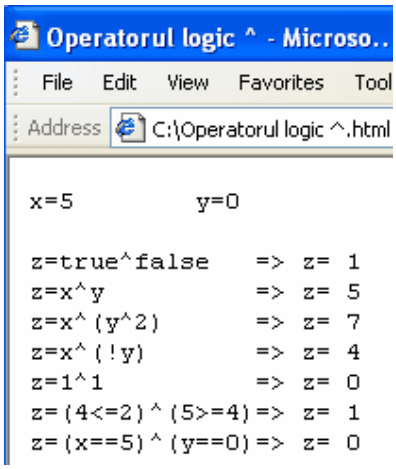
Exemplu de utilizare a operatorului |.

```
<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>
```



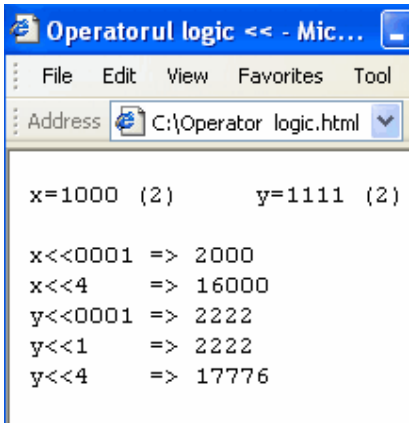
Exemplu de utilizare a operatorului ^.

```
<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=true^false   => z= ",
   z=true^false);
document.writeln("z=x^y          => z= ", z=x^y);
document.writeln("z=x^(y^2)      => z= ", z=x^(y^2));
document.writeln("z=x^(!y)       => z= ", z=x^(!y));
document.writeln("z=1^1          => z= ", z=1^1);
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>
```



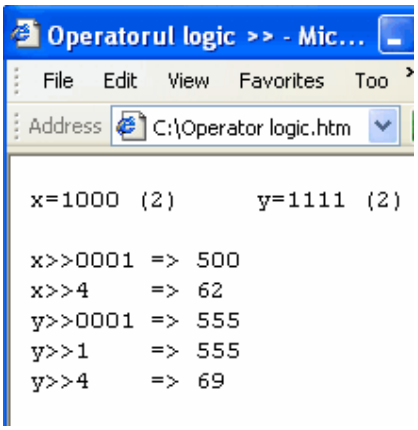
Exemplu de utilizare a operatorului <<.

```
<html>
<head><title>Operatorul logic <<</title></head>
<body>
<pre>
<script  language="JavaScript">
var x=1000 , y=1111;
document.writeln("x=1000 (2)      y=1111 (2)\n");
document.writeln("x<<0001 => ", x<<0001);
document.writeln("x<<4    => ", x<<4);
document.writeln("y<<0001 => ", y<<0001);
document.writeln("y<<1    => ", y<<1);
document.writeln("y<<4    => ", y<<4);
</script>
</pre>
</body>
</html>
```



Exemplu de utilizare a operatorului >>.

```
<html>
<head><title>Operatorul logic >></title>
</head>
<body>
<pre>
<script  language="JavaScript">
var x=1000 , y=1111;
document.writeln("x=1000 (2)      y=1111
(2)\n");
document.writeln("x>>0001 => ", x>>0001);
document.writeln("x>>4    => ", x>>4);
document.writeln("y>>0001 => ", y>>0001);
document.writeln("y>>1    => ", y>>1);
document.writeln("y>>4    => ", y>>4);
</script>
</pre>
</body>
</html>
```

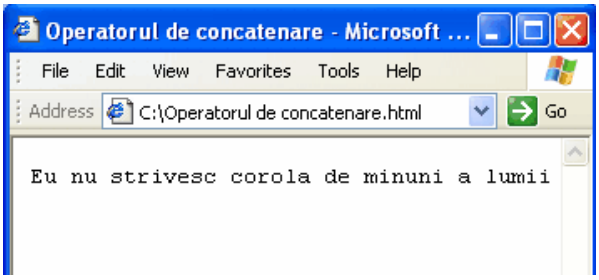


5.3.6. Operator pentru şiruri de caractere (string)

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>
```



5.3.7. Operatori de atribuire

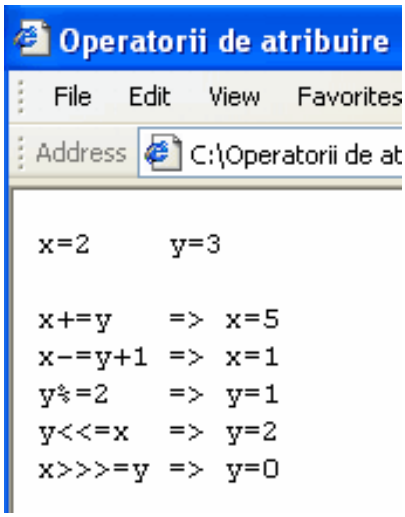
Operatorii de atribuire se utilizează între un operand pe care îl scriem în partea stângă şi o constantă, o variabilă sau o expresie în partea dreaptă a operatorului.

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>
```



5.3.8. Operatorul condițional

Operatorul condițional se utilizează în cadrul expresiilor condiționale.

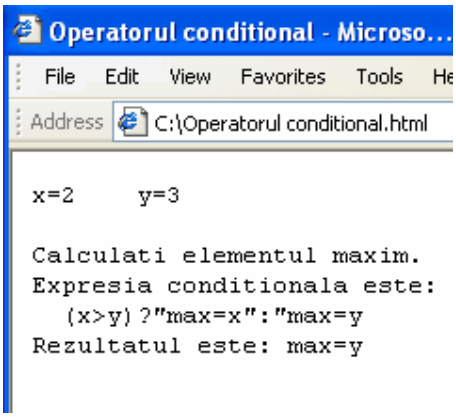
Forma generală:

(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>
```



5.3.9. Operatorul typeof

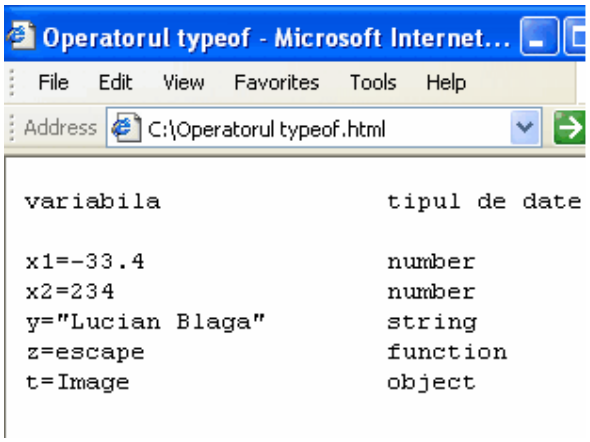
Operatorul *typeof* returnează tipul de date conținut de operandul său.

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>
```



Evaluare

11. Faceți o analiză comparativă între JavaScript şi Java, prezentând avantajele şi dezavantajele utilizării lor.
12. Creați un script JavaScript ce afişează dacă variabilele x=3.14 şi y=44 sunt diferite sau nu.

5.4. 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.

5.4.1. 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;
}
```

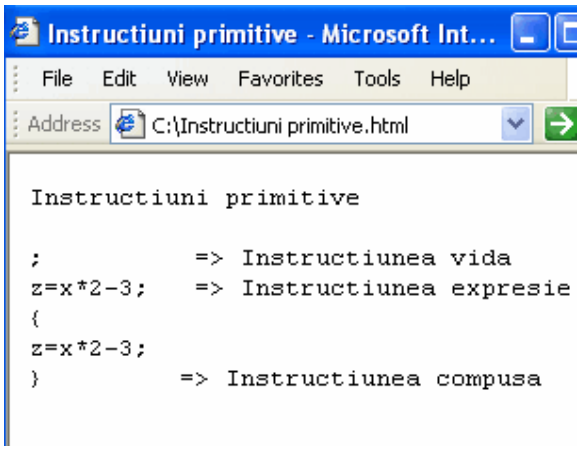
Instrucțiunea compusă se utilizează atunci când dorim să grupăm mai multe instrucțiuni, pentru a putea fi executate în ordine.

- ***Instrucțiunea expresie***

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

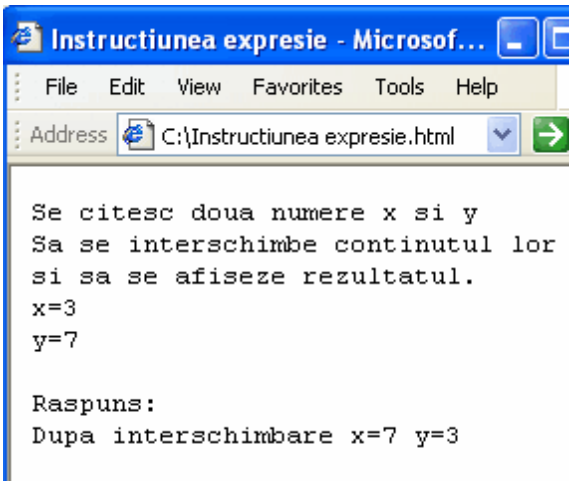
Exemplu de utilizare a instrucțiunilor primitive.

```
<html>
<head><title>Instructiuni
primitive</title></head>
<body>
<pre><script  language="JavaScript">
document.writeln("Instructiuni
   primitive\n");
{var x=2;
z=x*2-3;
;}
document.writeln(";          =>
   Instructiunea vida");
document.writeln("z=x*2-3;    =>
   Instructiunea expresie");
document.writeln("{\nz=x*2-3;\n}\t  =>
   Instructiunea compusa");
</script>
</pre>
</body></html>
```



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

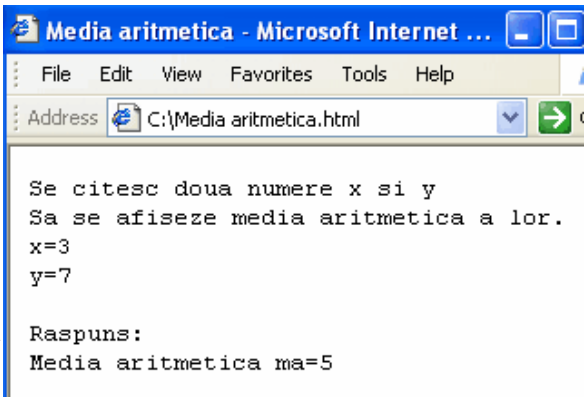
```
<html>
<head>
 <title>
   Instructiunea expresie
 </title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citesc doua numere x
   si y\nSa se interschimbe continutul
   lor\nsi sa se afiseze rezultatul.");
x=eval(prompt("Dati x:"));
y=eval(prompt("Dati y:"));
document.writeln("x=",x,"\ny=",y);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

Exemplu de utilizare a instrucțiunii expresie, pentru calcularea mediei aritmetice a două numere citite de la tastatură.

```
<html>
<head>
 <title>
   Media aritmetica
 </title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citesc doua numere x
    si y\nSa se afiseze media aritmetica a
    lor.");
x=eval(prompt("Dati x:"));
y=eval(prompt("Dati y:"));
document.writeln("x=",x,"\ny=",y);
document.writeln("\nRaspuns:");
ma=(x+y)/2;
document.writeln("Media aritmetica ma=",
    ma);
</script>
</pre>
</body>
</html>
```



5.4.2. 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 (cond.)
   { instr. 1; }
else
   { instr. 2; }
```

Principiul de execuție este următorul:
- se evaluează condiția;
- dacă se îndeplineşte condiția (cond.), se execută instrucțiunea instr.1, altfel se execută instrucțiunea instr.2.

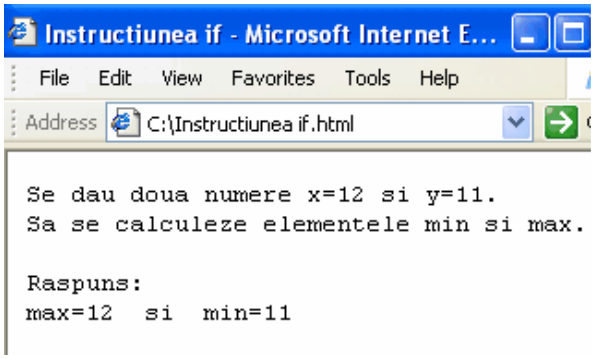
Sintaxa pentru instrucțiunea de decizie compusă:

```
if (cond.)
   { instr. 01;
     instr. 02;
     ........
     instr. 0n; }
else
   { instr. 1;
     instr. 2;
     ........
     instr. n; }
```

Principiul de execuție este similar cu cel al instrucțiunii de decizie simplă.
- se evaluează condiția (cond.);
- dacă se îndeplineşte condiția (cond.), se execută instrucțiunile de la instr.01 până la instr.0n;
- dacă nu se îndeplineşte condiția (cond.), se execută instrucțiunile de la instr.1 până la instr.n.

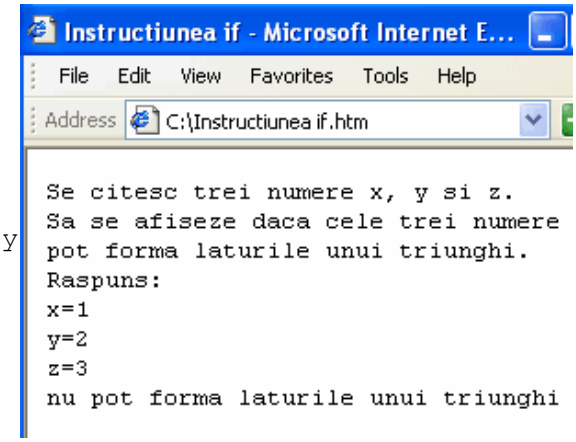
Exemplu de utilizare a instrucțiunii if.

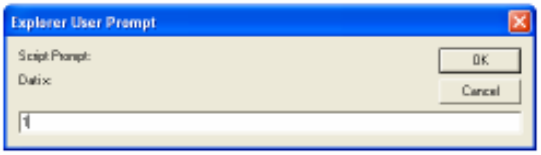
```
<html>
<head>
 <title>
   Instructiunea if
 </title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se dau doua numere x=12
   si y=11.\nSa se calculeze elementele
   min si max.\n");
document.writeln("Raspuns:");
var x=12; y=11;
if (x>y)
   {
     document.writeln("max=",x,"\tsi
      min=",y);
   }
else
   {
     document.write("max=",y);
     document.writeln("\tsi  min=",x);
   }
</script>
</pre>
</body>
</html>
```



Exemplu de utilizare a instrucțiunii if.

```
<html>
<head>
 <title>Instructiunea if</title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citesc trei numere x, y
   si z.\nSa se afiseze daca cele trei
   numere\npot forma laturile unui
   triunghi.");
document.writeln("Raspuns:");
x=eval(prompt("Dati x:"));
y=eval(prompt("Dati y:"));
z=eval(prompt("Dati z:"));
if
   ((x>0)&&(y>0)&&(z>0)&&(x+y>z)&&(x+z>y)&&(
   y+z>x))
{document.writeln("x=",x,"\ny=",y,"\nz=",z,"
   \npot forma laturile unui triunghi");}
else
{document.writeln("x=",x,"\ny=",y,"\nz=",z,"
   \nnu pot forma laturile unui triunghi");}
</script>
</pre>
</body>
</html>
```

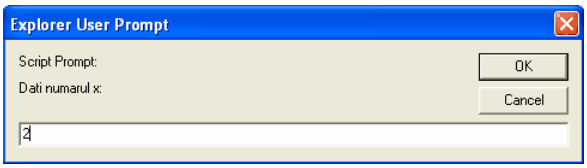


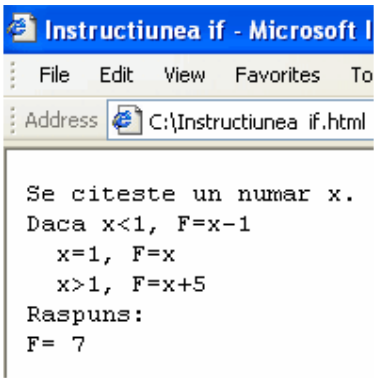


Exemplu de utilizare a instrucțiunii ***if***.

$$F=\begin{cases} x-1 & , x<1 \\ x & , x=1 \\ x+5 & , x>1 \end{cases}$$

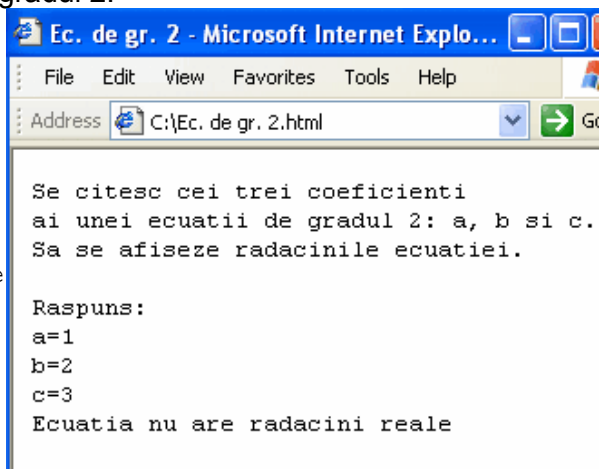
```
<html>
<head>
 <title>Instructiunea if</title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citeste un numar
   x.\nDaca x<1, F=x-1\n  x=1, F=x\n  x>1,
   F=x+5");
x=eval(prompt("Dati numarul x:"));
document.writeln("Raspuns:");
if (x<1)
    document.writeln("F= ",x-1);
else
    {
     if (x==1)
      document.writeln("F= ",x);
    else
      document.writeln("F= ",x+5);
    }
</script>
</pre>
</body></html>
```





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

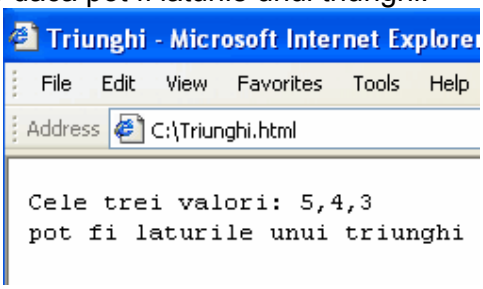
```
<html>
<head><title>Ec. de gr. 2</title>
</head>
<body>
<pre>
<script  language="JavaScript">
document.writeln("Se citesc cei trei
   coeficienti\nai unei ecuatii de gradul
   2: a, b si c.\nSa se afiseze radacinile
   ecuatiei.");
a=eval(prompt("Dati a:"));
b=eval(prompt("Dati b:"));
c=eval(prompt("Dati 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*.

Exemplu Se citesc trei numere a, b şi c. Să se precizeze dacă pot fi laturile unui triunghi.

```
<html>
<head><title>Triunghi</title></head>
<body>
<pre>
<script  language="JavaScript">
a=eval(prompt("Dati a="));
b=eval(prompt("Dati b="));
c=eval(prompt("Dati c="));
if((a>0)&&(b>0)&&(c>0)&&(a+b>c)&&(a+c>b)&&(b+
   c>a))
  document.writeln("Cele trei valori:
   ",a,",",b,",",c,"\npot fi laturile unui
   triunghi");
else
 document.writeln("Cele trei valori:
   ",a,",",b,",",c,"\nNU pot fi laturile unui
   triunghi");
</script></pre>
</body></html>
```



➢ ***Instrucțiunea switch..case***

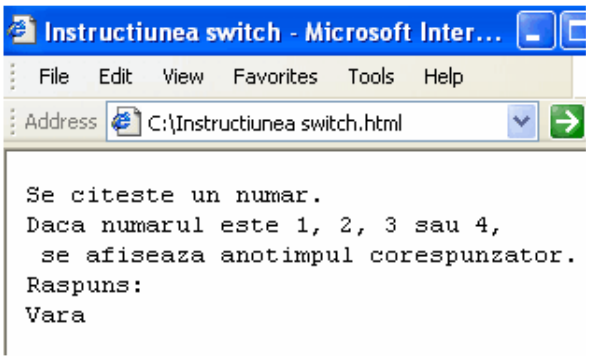
Instrucțiunea de decizie multiplă se utilizează atunci când, în urma evaluării unei expresii, trebuie să optăm între mai multe cazuri date. Acelaşi lucru îl putem realiza şi cu ajutorul instrucțiunii if. Diferența constă în faptul că dacă avem de optat între mai mult de două cazuri, folosirea instrucțiunii ***if*** devine greoaie, caz în care putem apela la instrucțiunea ***switch***.

Sintaxa pentru instrucțiunea de decizie multiplă simplă:

switch (expresie)	Principiul de execuție este următorul:
{	- se evaluează expresia;
case valoare_1;	- dacă valoarea expresiei este egală cu *valoare_1*, se
instr_1; break;	execută *instr_1*
case valoare_2;	- dacă valoarea expresiei nu este egală cu *valoare_1*, se
instr_2; break;	verifică dacă este egală cu *valoare_2*, pentru a se
……..	executa *instr_2*,
case valoare_n;	- dacă valoarea expresiei nu este egală cu *valoare_2* se
instr_n; break;	repetă paşii până la *valoare_n*
[default: instr_n+1];	- dacă valoarea expresie nu este egală cu nici o valoare
}	dintre: *valoare_1...valoare_n*, se execută *instr_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>
```



5.4.3. 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***

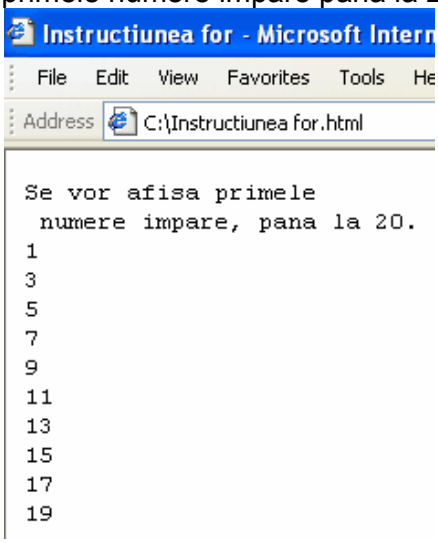
Sintaxa pentru instrucțiunea repetitivă cu număr cunoscut de paşi ***for***:

for (val1; cond.;pas)
instrucțiune

Principiul de execuție este următorul:
- se evaluează condiția ***cond.***;
- dacă se îndeplineşte condiția ***cond.*** , se incrementează valoarea inițială ***val1.*** cu valoarea pasului ***pas.*** şi se repetă execuția instrucțiunii până când nu se mai îndeplineşte condiția.

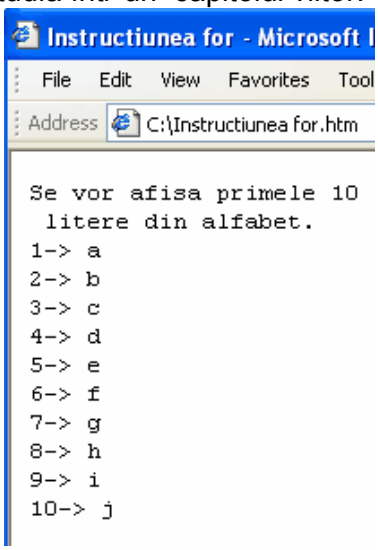
Exemplu de utilizare a instrucțiunii ***for***. Se vor afişa primele numere impare până la 20.

```
<html>
<head>
 <title>Instructiunea for</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var i=1;
document.writeln("Se vor afisa primele\n numere
    impare, pana la 20.");
for(i=1; i<=20; i=i+2)
 document.writeln(i);
</script>
</pre>
</body></html>
```



Exemplu În acest exemplu, se vor afişa literele de la a la j. Pentru aceasta, am inițializat variabila x cu valoarea Unicode a caracterului "a", de la care am dorit să începem afişarea. Condiția este să afişeze până la valoarea Unicode a caracterului "j", cu pasul 1, toate caracterele corespunzătoare acestor valori Unicode. Metodele *charCodeAt()* şi *fromCharCode()* le vom studia într-un capitolul viitor.

```
<html>
<head>
 <title>Instructiunea for</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var y=1;
document.writeln("Se vor afisa primele
   10\n litere din alfabet.");
for(x="a".charCodeAt(0);
   x<="j".charCodeAt(0); x++)
{document.writeln(y,"->
   ",String.fromCharCode(x));
 y++;}
</script>
</pre>
</body>
</html>
```



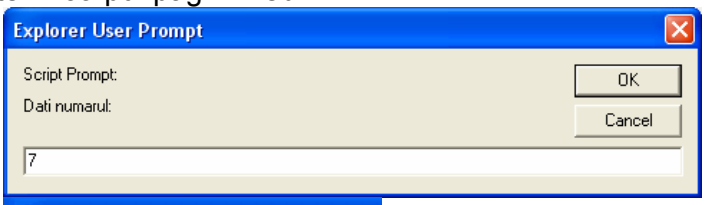
Exemplu Vom afişa în ordine descrescătoare numerele cuprinse între 15 şi 7.

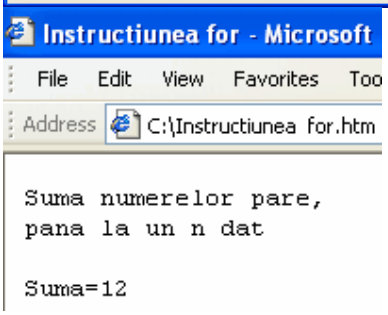
```
<html>
<head>
 <title>Instructiunea for</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var i=15;
document.writeln("Se vor afisa in ordine
descrescatoare,\nnumerele de la 15 la 7.");
for(i=15;i>=7;i--)
document.writeln(i);
</script></pre>
</body></html>
```

```
Se vor afisa in ordine descrescatoare,
numerele de la 15 la 7.
15
14
13
12
11
10
9
8
7
```

Exemplu Vom afişa suma numerelor pare până la un n citit de la tastatură. Pentru a realiza acest lucru, am utilizat metoda *prompt()* ce deschide o fereastră în care se cere numărul n şi funcția *eval()* ce evaluează valoarea introdusă de către utilizator. Datele de început şi de sfârşit sunt afişate în corpul pagini web.

```
<html>
<head>
 <title>Instructiunea for </title>
</head>
<body>
<pre>
<script language="JavaScript">
document.writeln("Suma numerelor
pare,\npana la un n dat\n");
n=eval(prompt("Dati numarul:"));
var suma=0;
for (i=0; i<=n; i=i+2)
  suma=suma+i;
document.writeln("Suma=",suma);
</script>
</pre>
</body>
</html>
```

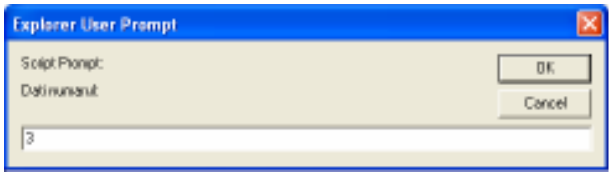


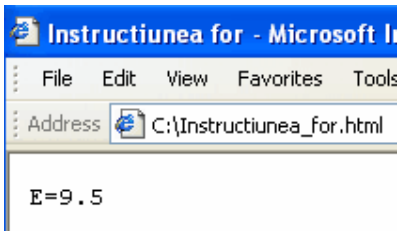


Exemplu Să se calculeze şi să se afişeze următoarea expresie:

$$E = \frac{1*3}{1} + \frac{2*4}{1*2} + \frac{3*5}{1*2*3} + \ldots + \frac{n*(n+2)}{1*2*\ldots*n}$$

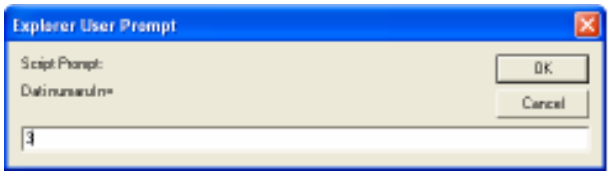
```
<html>
<head>
 <title>Instructiunea for</title>
</head>
<body><pre>
<script language="JavaScript">
 n=eval(prompt("Dati numarul:"));
var e=0;p=1;
for (i=1; i<=n; i++)
  {p=p*i;
   e=e+(i*(i+2))/p;}
document.writeln("E=",e);
</script></pre>
```

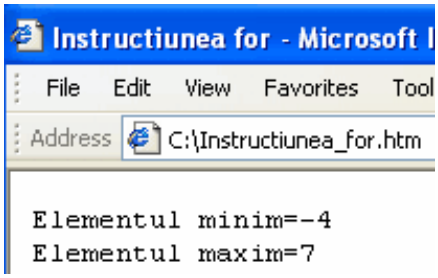




```
</body></html>
```

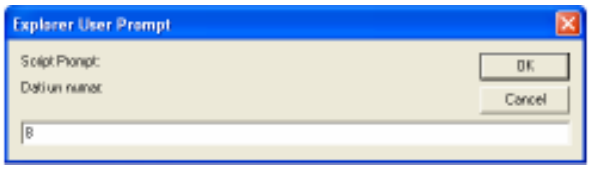
Exemplu Să se afişeze minimul şi maximul din n numere citite de la tastatură.

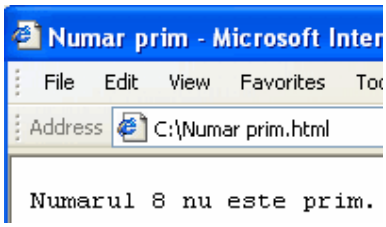





```
<html>
<head>
 <title>
   Instructiunea for
 </title>
</head>
<body>
 <pre>
<script language="JavaScript">
 n=eval(prompt("Dati numarul n="));
 x=eval(prompt("Dati numarul 1"));
   min=x;
   max=x;
 for (i=2; i<=n; i++)
  {
   x=eval(prompt("Dati numarul ",i));
   if (x<min)
     min=x;
   if (x>max)
     max=x;
  }
document.writeln("Elementul
   minim=",min);
document.writeln("Elementul
   maxim=",max);
</script>
</pre>
</body></html>
```

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***

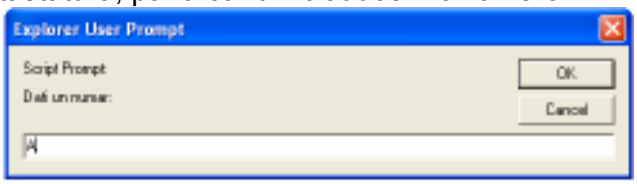
Sintaxa pentru instrucțiunea repetitivă cu test inițial şi cu număr necunoscut de paşi ***while***:

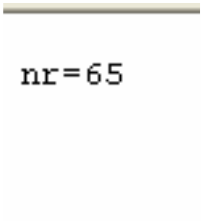
while (cond.)
instrucțiune

Principiul de execuție este următorul:
- se evaluează condiția ***cond.***;
- cât timp se îndeplineşte condiția ***cond.***, 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.

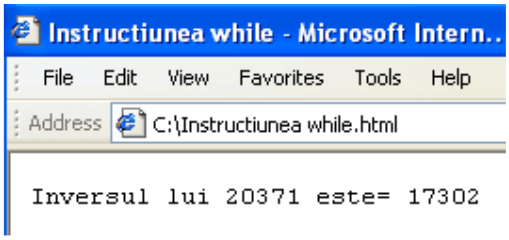
```
<html>
<head>
 <title>Instructiunea while </title>
</head>
<body>
<pre>
<script  language="JavaScript">
n=prompt("Dati un numar:","");
while (n!=0)
 {
  document.writeln("nr=",
    n.charCodeAt(0));
  n=prompt("Dati un numar:","");
 }
</script>
</pre>
</body></html>
```





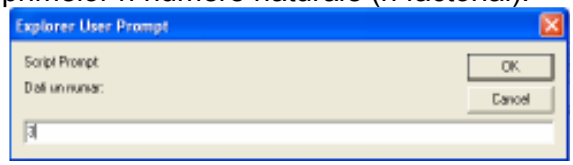
Exemplu Să se afişeze inversul unui număr citit de la tastatură.

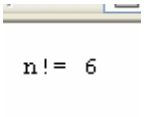
```
<html>
<head>
 <title>Instructiunea while</title>
</head>
<body>
<pre>
<script  language="JavaScript">
n=prompt("Dati un numar:");
x=n;
inv=0;
while (n!=0)
 {inv=inv*10+n%10;
  n=Math.floor(n/10);}
document.writeln("Inversul lui ",x," este=
",inv);
</script>
</pre>
</body></html>
```



Exemplu Să se calculeze şi să se afişeze produsul primelor n numere naturale (n factorial).

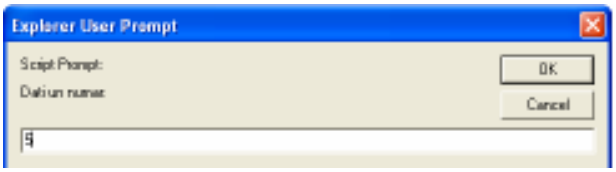
```
<html>
<head>
 <title>n!</title>
</head>
<body>
<pre>
<script  language="JavaScript">
n=eval(prompt("Dati un numar:"));
i=1;p=1;
while(i<=n)
 {
  p=p*i;
  i=i+1;
 }
document.writeln("n!= ",p);
</script></pre>
</body></html>
```

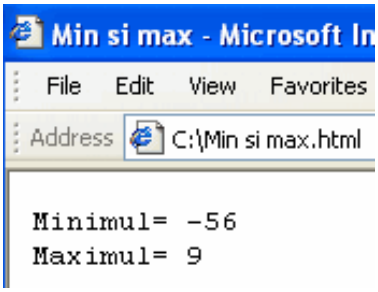




Exemplu Se citesc numere până la întâlnirea cifrei zero (0). Să se afişeze cel mai mic şi cel mai mare număr citit.

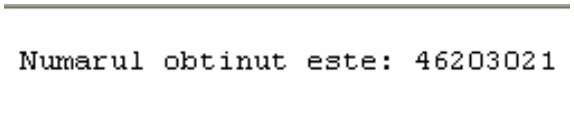
```
<html>
<head>
 <title>
   Min si max
 </title>
</head>
<body>
<pre>
<script  language="JavaScript">
n=eval(prompt("Dati un numar:"));
min=n;
max=n;
 while(n!=0)
 {
  n=eval(prompt("Dati un numar:"));
  if (n<min)
     min=n;
  if (n>max)
     max=n;
 }
  document.writeln("Minimul= ",min);
  document.writeln("Maximul= ",max);
</script>
</pre>
</body>
</html>
```





Exemplu Se citesc pe rând cifrele unui număr. Să se afişeze numărul obținut prin aranjarea cifrelor în ordinea citirii lor.

```
<html>
<head>
 <title>
   Cifre
 </title>
</head>
<body>
<pre>
<script  language="JavaScript">
var n=0;
x=eval(prompt("Dati prima cifra:"));
while ((x>=0)&&(x<=9))
 {
 n=n*10+x;
 x=eval(prompt("Dati urmatoarea cifra:"));
 }
document.writeln("Numarul obtinut este: ",
    n);
</script>
</pre>
</body>
</html>
```



Exemplu Se citeşte un număr în baza 2. Să se afişeze acel număr în baza 10. De exemplu, dacă se citeşte 111_2, se va afişa 7_{10}.

```
<html>
<head>                                          Numarul 111 in baza 10 este: 7
 <title>Baza zece</title>
</head>
<body>
<pre>
<script language="JavaScript">
var putere=1;
var bzece=0;
var k=0;
bdoi=eval(prompt("Dati numarul in baza
    2:"));
x=bdoi;
while (x!=0)
 {
  r=x%10;
  if ((r!=0)&&(r!=1))
    k=1;
  bzece=bzece+r*putere;
  putere=putere*2;
  x=Math.floor(x/10);
 }
if (k==0)
  document.writeln("Numarul ",bdoi," in baza
    10 este: ",bzece);
else
  document.writeln("Numarul nu este in baza
    2");
</script>
</pre>
</body>
</html>
```

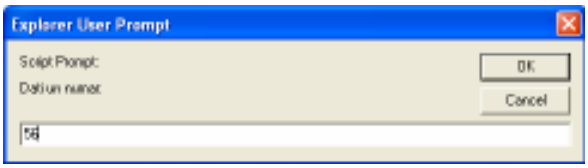
- ***Instrucțiunea do...while***

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

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

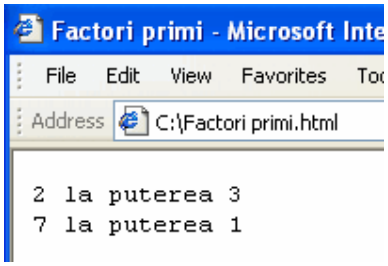
Principiul de execuție este următorul:
- se execută instrucțiunile;
- se evaluează condiția ***cond.***;
- dacă valoarea rezultată este diferită de zero (adevărat), se repetă execuția instrucțiunii până când aceasta devine zero (fals);
- se trece la instrucțiunea următoare

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



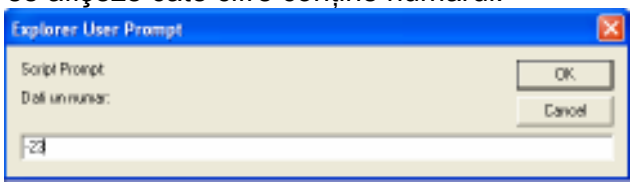

```
<html>
<head>
 <title>Factori primi</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var i=2;
```

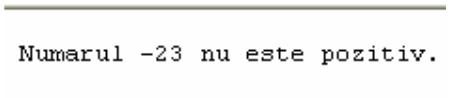
```
n=eval(prompt("Dati un numar:"));
do
  {
    putere=0;
    while (n%i==0)
     {
       putere=putere+1;
       n=Math.floor(n/i);
     }
  if (putere!=0)
    document.writeln(i," la puterea
",putere);
    i++;
  }
while (n!=1)
</script></pre>
</body></html>
```



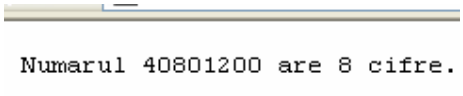
Exemplu Se citeşte un număr natural pozitiv. Să se afişeze câte cifre conține numărul.

```
<html>
<head>
 <title>Nr. cifre</title>
</head>
<body>
<pre>
<script  language="JavaScript">
var s=0;
n=eval(prompt("Dati un numar:"));
x=n;
if (n>=0)
{
  do
    {  n=Math.floor(n/10);
       s++;  }
  while (n!=0)
    document.writeln("Numarul ",x," are
",s," cifre.");
}
else
    document.writeln("Numarul ",x," nu
este pozitiv.");
</script>
</pre>
</body>
</html>
```





sau



Evaluare

1. Care sunt operatorii aritmetici?
2. Unde se pot scrie script-urile JavaScript?
3. Ce fel de operator este + (plus)?
4. Care sunt instrucțiunile repetitive?
5. Realizați un script ce permite calcularea suma numerelor pare până la un număr n citit de la tastatură.
6. În interiorul căror etichete se încadrează codul sursă JavaScript?
   a) <js> şi </js>
   b) <script> şi </script>

c) <JavaScript> şi <JavaScript>
d) <java> şi <java>

7. Unde este corect să poziționăm codul JavaScript?
   a) În secțiunile: < head > sau < body >
   b) În secțiunea < title >
   c) În secțiunea < head >
   d) În secțiunea < body >
8. Operatorii unari sunt:
   a) + / %
   b) + -
   c) * /
   d) + - %
9. Cum afişăm pe ecran: "Salutari!"?
   a) Writeln ("Salutari!")
   b) document.writeln (Salutari!)
   c) alert ("Salutari!")
   d) document.writeln ("Salutari!")
10. Cum afişăm într-un buton alert: "Salutari!"?
   a) writeln . alert ("Salutari!")
   b) msgBox ("Salutari!")
   c) alert ("Salutari!")
   d) alert . writeln ("Salutari!")
11. Cum scriem: "Dacă x este mai mic decât y, atunci minim=x, altfel minim=y"?
   a) if (x < y) then minim=x else minim=y
   b) (x < y)?minim=x:minim=y
   c) minim=(x < y)?x:y
   d) if x< y minim=x else minim=y

Capitolul 6.

Funcții JavaScript

Obiective:
- să înțeleagă noțiunea de funcție JavaScript
- să poată utiliza o funcție JavaScript acolo unde este necesar.
- să înțeleagă noțiunile de apelare şi parametrii ai funcțiilor
- să poată crea un script ce utilizează funcții predefinite JavaScript

6.1. Introducere

Atunci când dorim ca pagina Web să îndeplinească o anumită cerință, creăm o funcție JavaScript ce va avea ca scop executarea acelei cerințe. Funcțiile JavaScript le putem crea noi sau putem utiliza funcții predefinite ale limbajului, cum ar fi: *escape(), eval(),* pe care o să le studiem în acest capitol. Apelarea unei funcții se face prin numele ei, atunci când este necesar. O funcție poate fi poziționată în partea de sus a paginii, adică în antet sau în exterior, într-un alt fişier.

6.2. Definirea funcțiilor

Definirea unei funcții constă în cuvântul cheie **function**, urmat de: numele funcției, o listă cu parametri separați prin virgulă şi instrucțiunile scrise între acolade. O funcție poate fi apelată de către o altă funcție sau de către ea însăşi (se autoapelează).

Sintaxa unei funcții ce conține parametri:

Partea de declarare a funcției.	**function Nume (variabila1, variabila2,...., variabilan)** **{** **instrucțiuni ;** **}**
Partea de apelare a funcției.	**Nume (variabila1, variabila2,...., variabilan)**

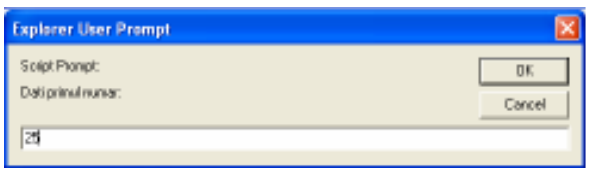
Funcțiile pot fi declarate şi apelate chiar dacă nu conțin parametri.

Partea de declarare a funcției.	**function Nume ()** **{** **instrucțiuni ;** **}**
Partea de apelare a funcției.	**Nume ()**

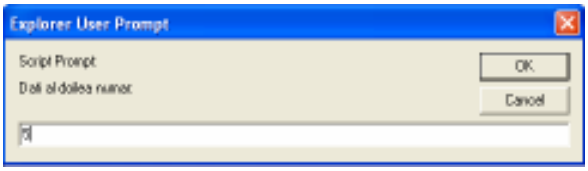
Pentru a rezolva problema din exemplul de mai jos, am apelat funcția cmmdc, care, pentru a returna cel mai mare divizor comun dintre două valori x şi y primite ca parametri, se autoapelează până când se îndeplineşte condiția y=0. Autoapelarea unei funcții poartă numele de recursivitate.

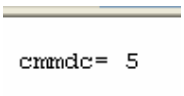
Exemplu de utilizare a unei funcții JavaScript în *head.* Se citesc două numere naturale de la tastatură. Să se afişeze cel mai mare divizor comun al celor două numere, utilizând algoritmul lui Euclid.

```
<html>
<head>
 <title>
   Functie JavaScript in head
 </title>
<script  language="JavaScript">
```



```
function cmmdc(x,y)
    {   if (y==0)
            return x;
        else
            return cmmdc(y,x%y);    }
</script>
</head>
<body><pre>
<script  language="JavaScript">
x=eval(prompt("Dati primul numar:"));
y=eval(prompt("Dati al doilea numar:"));
  document.writeln("cmmdc= ",cmmdc(x,y));
</script>
</pre>
</body></html>
```





OBS În exemplul de mai sus am utilizat **return** pentru a întoarce un rezultat prin numele funcției.

6.3. Apelarea funcțiilor

Definirea unei funcții nu înseamnă neapărat că acea funcție se va apela. Putem defini funcții care nu se vor executa niciodată, dar şi funcții care pot fi apelate de una sau mai multe funcții. Apelarea unei funcții presupune executarea unei acțiuni specifice, pentru anumiți parametri.

În exemplul de mai sus, se citesc doi parametri x şi y, după care se efectuează apelul funcției pentru parametri citiți.

Mai jos, putem vedea o funcție fără parametri, care va fi apelată doar dacă executăm click pe un buton. În acest exemplu, nu au fost necesari parametri, de aceea nu există.

Exemplu de utilizare a unei funcții JavaScript în *head* fără parametri. Se apasă pe butonul pe care scrie “Apasati!” şi va apare o fereastră în care scrie “Salut!”.

```
<html>
<head>
 <title>Functie fara parametri </title>
<script type="text/JavaScript">
function f()
 {
   alert("Salut!")
 }
</script>
</head>
<body>
<input     type="button"     onclick="f()"
   value="Apasati!">
</body>
</html>
```

Exemplu de utilizare a unei funcții JavaScript *exterioară.* Pentru aceasta, se crează o funcție simplă, ca cea de mai jos, care se salvează cu extensia **.js**. În cazul nostru, am salvat fişierul cu numele **ff.js**.

```
function f()
{
alert('Salutari       dintr-un       script
exterior!');
}
```

Odată salvat, fişierul poate fi apelat din interiorul unui script, ca în exemplul de mai jos. După ce este apelat fişierul ff.js, putem utiliza funcția sau funcțiile pe care le conține fişierul respectiv. În cazul nostru, apelăm funcția **f()**.

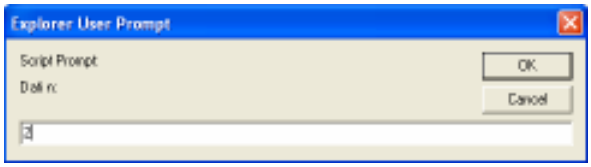
```
<html>
<head>
<title>Functie externa</title>
<SCRIPT language="JavaScript" SRC="ff.js">
</SCRIPT>
</head>
<body>
<A HREF="javascript:f()">Dati click pentru
un mesaj!</A>
</body></html>
```

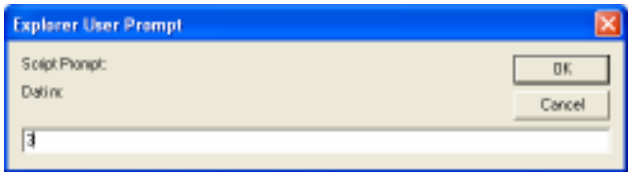
Dati click pentru un mesaj!

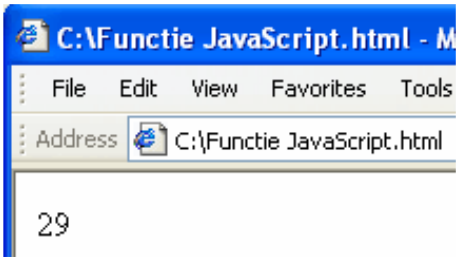
OBS Apelarea fişierului se face din antet (head).

Exemplu Se citesc două numere naturale m şi n. Scriptul următor va calcula funcția lui Ackermann. La încărcarea paginii, se apelează o funcție f() care citeşte două valori n şi m, după care apelează funcția recursivă Ackermann, de parametrii n şi m citiți. Rezultatul se va afişa pe ecran.

```
<html>
<head>
<title>Functie JavaScript</title>
</head>
<body onLoad="f()">
<script language="JavaScript">
function f()
{ n=eval(prompt("Dati n: "));
  m=eval(prompt("Dati m: "));
  document.writeln(Ackermann(m,n));  }
function Ackermann(m,n)
{
 if (m==0)
   return n+1;
 else
   if (n==0)
     return Ackermann(m-1,1);
   else
     return              Ackermann(m-1,
Ackermann(m,n-1))  }
</script>
</body></html>
```

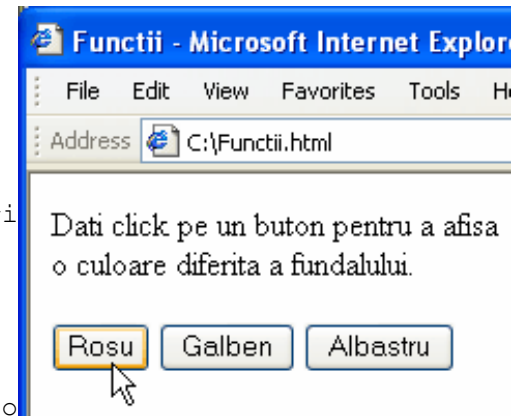






Exemplu În acest exemplu avem funcția **culori**, care va putea fi apelată de mai multe ori, pentru valori diferite. Dacă dăm click pe butonul pe care scrie **Rosu**, se va transmite funcției prin variabila x, valoarea **Red**. Această valoare va defini culoarea de fundal a unei pagini Web.

```
<html>
<head>
 <title>Functii</title>
<script type="text/JavaScript">
function culori(x)
 {
 document.writeln("<body bgcolor="+x+">");
 document.writeln("Pagina cu fundal de culori
diferite.");
 }
</script>
</head>
<body>
<form>
Dati click pe un buton pentru a afisa <br />o
```



```
   culoare diferita a fundalului.<br /><br />
<input  type="button"  onclick="culori('Red')"
   value="Rosu">
<input          type="button"          onclick=
   "culori('Yellow')" value="Galben">
<input  type="button"  onclick="culori('Blue')"
   value="Albastru">
</form>
</body>
</html>
```

Pagina cu fundal de culori diferite.

Exemplu Un exemplu similar cu cel de mai sus, dar în care avem doi parametri transmişi la apelul realizat de către evenimentul *onClick*. Cei doi parametri vor modifica valorile implicite ale culorii fundalului şi textului.

```
<html>
<head>
 <title>Functii</title>
<script type="text/JavaScript">
function culori(x,y)
 {
   document.writeln("<body  bgcolor="+x+"  text  =
   "+y+">");
   document.writeln("<b> Pagina cu fundal si text
   <br /> de culori diferite.");
 }
</script>
</head>
<body>
<form>
Dati  click  pe  un  buton  pentru  a  afisa  <br /> o
   culoare  diferita  a  fundalului  si  a<br  />
   textului<br /><br />
<input           type="button"           onclick=
   "culori('Red','blue')" value="Rosu">
<input           type="button"           onclick=
   "culori('black','Yellow')" value="Galben">
<input  type="button"  onclick="culori('green')"
   value="Albastru">
</form>
</body>
</html>
```

Pagina cu fundal si text de culori diferite.

6.4. Parametri funcțiilor

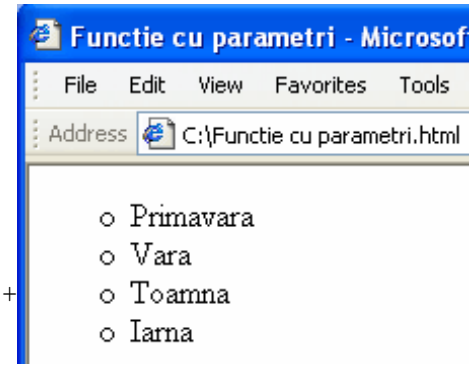
Parametri pe care-i utilizăm atunci când apelăm o funcție poartă numele de parametri efectivi sau actuali. Ei specifică valorile care vor fi prelucrate în cadrul funcției.

Parametri aflați în partea de declarare a funcției poartă numele de parametri formali. Atât parametri formali cât şi parametri efectivi pot lipsi atunci când utilizăm o funcție. Numărul parametrilor formali trebuie să fie egal cu numărul parametrilor efectivi.

OBS Excepție de la această regulă face proprietatea *arguments* ce poate fi utilizată cu un număr diferit de parametri la apelul funcției. Pentru a determina numărul de parametri din apel, utilizăm în cadrul funcției, *arguments.length. arguments* este un masiv (vector unidimensional) ce are ca primă valoare arguments[0].

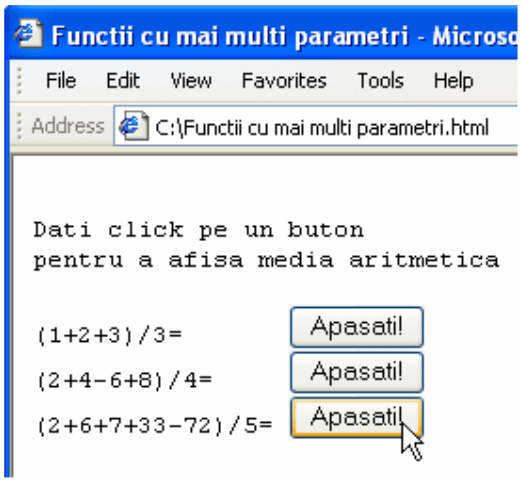
Exemplu În acest exemplu, se apelează funcția ***lista*** din interiorul unei instrucțiuni de afişare, prin lista(“U”, “Primavara”, "Vara", "Toamna", "Iarna"). La execuția funcției, se parcurg parametri cu ajutorul arguments.length. Parametrul “U” de la apel se introduce în funcție pentru a afişa o listă neordonată (Unordered Lists) UL. Pentru a afişa o listă ordonată, trebuie să înlocuim U cu O, de la Ordered lists.

```
<html>
<head>
 <title>
   Functie cu parametri
 </title>
<script type="text/JavaScript">
function lista(x)
 {
  document.write("<" + x + "L  type=circle>")
  for (var i=1; i<lista.arguments.length; i++)
    document.write("<LI>"
       lista.arguments[i]+"</LI>")
    document.write("</" + x + "L>")
 }
</script>
</head>
<body>
<script type="text/JavaScript">
write(lista("U",  "Primavara",  "Vara",  "Toamna",
    "Iarna"));
</script>
</body>
</html>
```



Exemplu Un exemplu asemănător cu cel de mai sus, dar în care aceeaşi funcție va putea fi apelată de mai multe ori, pentru a realiza calcularea mediei cu mai mulți sau mai puțini parametri.

```
<html>
<head>
 <title>
   Functii cu mai multi parametri
 </title>
<script type="text/JavaScript">
function medie()
 {
   var s = 0
   for(var i=0; i<arguments.length; i++)
      s=s+arguments[i] ;
   var ma=s/arguments.length
alert(ma);
 }
</script>
</head>
<body>
<pre>
<form>
Dati click pe un buton pentru a afisa media
   aritmetica
<br /><br />
(1+2+3)/3=   <input   type="button"   onclick=
   "medie(1,2,3)" value="Apasati!">
(2+4-6+8)/4=   <input   type="button"   onclick=
   "medie(2,4,-6,8)" value="Apasati!">
(2+6+7+33-72)/5= <input type="button" onclick=
   "medie(2,6,7,33,-72)"
value="Apasati!">
</form>
</pre>
</body>
</html>
```

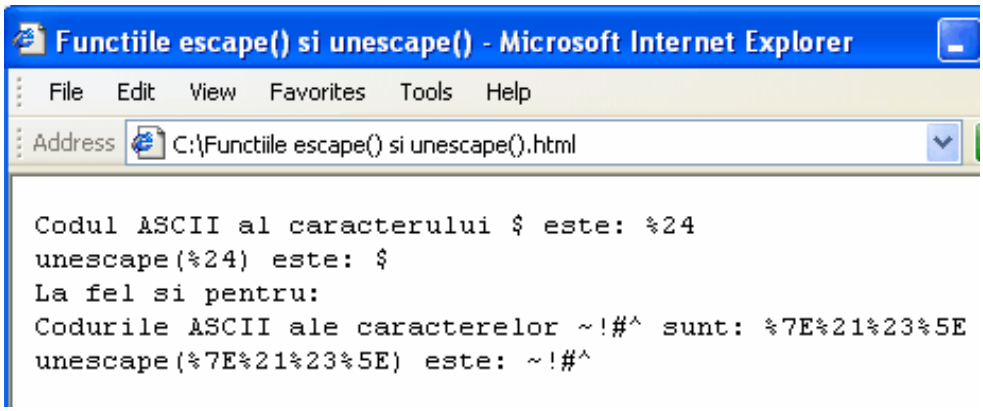


6.5. Funcții predefinite

În acest capitol vom studia cele mai importante funcții predefinite JavaScript. Printre cele mai utilizate sunt: eval, isFinite, isNaN, parseInt, parseFloat, Number, String, encodeURI, decodeURI, encodeURIComponent, decodeURIComponent

Funcțiile escape() şi **unescape()** transformă seturi de caractere admise de ISO Latin-1 în coduri ASCII (scrise în hexazecimal) şi invers.
Pentru început vom face un exemplu simplu de transformare a şirurilor de caractere în codurile ASCII corespunzătoare şi invers.

Exemplu Se transformă mai întâi caracterul $ în codul ASCII corespunzător şi înapoi din codul ASCII în caracterul inițial. După aceasta, se transformă un şir de caractere în codul ASCII corespunzător şi invers.




```
<html>
<head>
 <title>Functiile escape() si unescape() </title>
</head>
<body>
<pre>
<script language="JavaScript">
   x=escape("$");
      document.writeln("Codul ASCII al caracterului $ este: "+x);
   y=unescape(x);
      document.writeln("unescape("+x+") este: "+y);
      document.writeln("La fel si pentru: ");
   x=escape("~!#^");
      document.writeln("Codurile ASCII ale caracterelor ~!#^ sunt: "+x);
   y=unescape(x);
      document.writeln("unescape("+x+") este: "+y);
</script>
</pre>
</body>
</html>
```

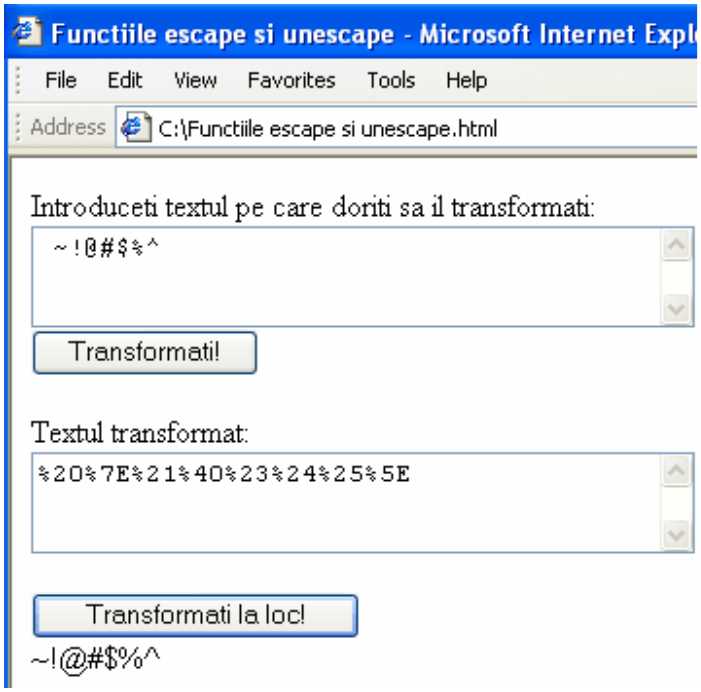
Chiar dacă încă nu am parcurs toate elementele necesare pentru exemplul următor, vă puteți face o impresie despre modul de transformare interactiv pe care-l puteți realiza cu ajutorul funcțiilor JavaScript.

Exemplu În corpul paginii introducem un formular care apelează două funcții din interiorul unui script. Scriptul este apelat la începutul paginii, în antet (head), prin src="FTR.js" .

```
<html>
<head>
 <title>
   Functiile escape si unescape
 </title>
<script type="text/javascript" src="FTR.js"> </script>
</head>
<body>
 <form name="x" action="">
   Introduceti textul pe care doriti sa il transformati:<br />
   <textarea name="tr" cols="40" rows="3">
   </textarea><br />
   <input       type="button"       name="action"       value="Transformati!"
        onClick="Transform()"/>
<br/><br />
Textul transformat:<br />
   <textarea name="re_tr" cols="40" rows="3">
   </textarea><br /><br />
    <input   type="button"   name="action"   value="Transformati   la   loc!"
        onClick="ReTransform()" /> <br />
    <div id="RTr">
    </div>
 </form>
</body>
</html>
```

Scriptul “FTR.js” este cel de mai jos. În acest script avem două funcții: Transform(), care transformă caracterele în coduri ASCII şi ReTransform() care converteşte codurile ASCII în caracterele introduse inițial.

```
function Transform()
 {
   cod = escape(x.tr.value);
   cod = cod.replace(/\//g,"%2F");
   cod = cod.replace(/\?/g,"%3F");
   cod = cod.replace(/=/g,"%3D");
   cod = cod.replace(/&/g,"%26");
   cod = cod.replace(/@/g,"%40");
   x.re_tr.value = cod;
 }
function ReTransform()
{
   RTr.innerHTML =
      unescape(x.re_tr.value);
}
```



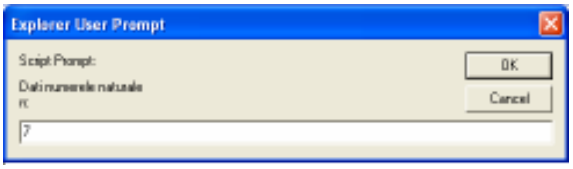
Funcția eval() evaluează un şir de caractere, o expresie, o variabilă sau anumite proprietăți ale unor obiecte existente. Sintaxa funcției este: ***eval(string)***
Dacă ***string*** reprezintă o expresie, atunci **eval** va evalua acea expresie şi va returna rezultatul.

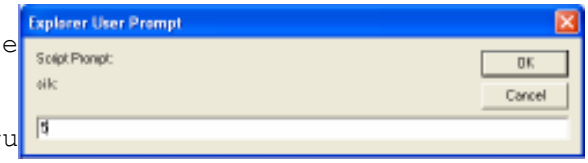
Exemplu de utilizare a funcției **eval()**. Se citesc numerele naturale n şi k. Să se calculeze C_n^k, utilizând relația de recurență:

$$C_n^k = \begin{cases} 1 & pentru \quad k = 0 \quad sau \quad k = n \\ C_{n-1}^{k-1} + C_{n-1}^{k} & pentru \quad k = 1, 2, \ldots n-1 \end{cases}$$

În variabilele n şi k se depune rezultatul evaluării funcției **eval()**. Cele două valori sunt transmise funcției *combinări* ce va calcula recursiv numărul de combinări cerute.

```
<html>
<head>
 <title>Functia eval()</title>
</head>
<body>
<pre>
<script type="text/javascript">
var n=eval(prompt("Dati numerele naturale
\nn:"));
var k=eval(prompt("si k:"));
document.writeln("Combinarile       pentru
numerele date: ",combinari(n,k));
function combinari(n,k)
 {
  if ((k==0)||(k==n))
    return 1;
   else
    return combinari(n-1,k-1)+combinari(n-
1,k);
 }
</script>
</pre>
</body></html>
```

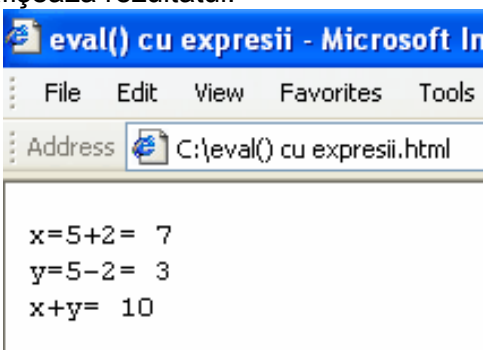





```
Combinarile pentru numerele date: 21
```

Exemplu de utilizare a expresiilor aritmetice. Se evaluează pe rând două expresii aritmetice, după care se combină rezultatele şi se afişează rezultatul.

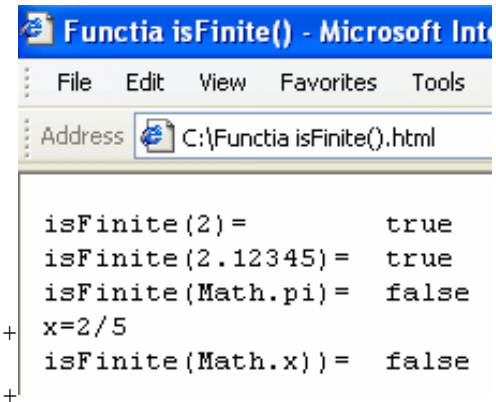
```
<html>
<head>
 <title>eval() cu expresii</title>
</head>
<body>
<pre>
<script type="text/javascript">
    var x=eval("5+2");
    var y=eval("5-2");
    document.writeln("x=5+2= ",x);
    document.writeln("y=5-2= ",y);
    document.writeln("x+y= ", eval(x+y));
</script>
</pre>
</body></html>
```



Funcția isFinite() evaluează un parametru pentru a determina dacă este număr finit sau nu. Dacă parametrul primit tinde către +∞ sau către -∞, dacă este *NaN* sau nu are limite finite, funcția va returna false, altfel va returna true.

Exemplu Am utilizat pentru verificarea funcției isFinite(), atât numere ce au zecimale finite, cât şi numere ale căror zecimale nu sunt într-un număr finit.

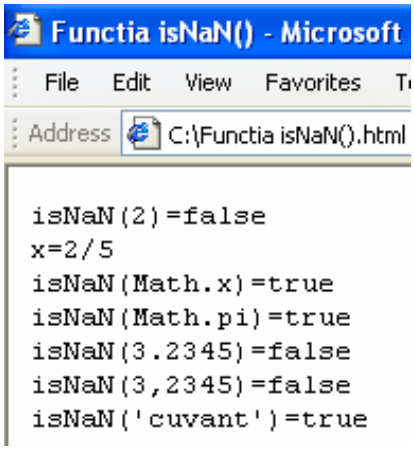
```
<html>
<head>
 <title>
   Functia isFinite()
 </title>
</head>
<body>
<pre>
<script type="text/JavaScript">
document.writeln("isFinite(2)="+isFinite(2));
document.writeln("isFinite(2.12345)=        "+
   isFinite(2.12345));
document.writeln("isFinite(Math.pi)=        "+
   isFinite(Math.pi));
var x=2/5;
document.writeln("x=2/5 <br />
   isFinite(Math.x))=  "+isFinite(Math.x));
</script>
</pre>
</body>
</html>
```



Funcția isNaN() evaluează argumentul pentru a determina dacă este număr sau nu. În cazul în care argumentul este număr, returnează ***false***, altfel returnează ***true***.

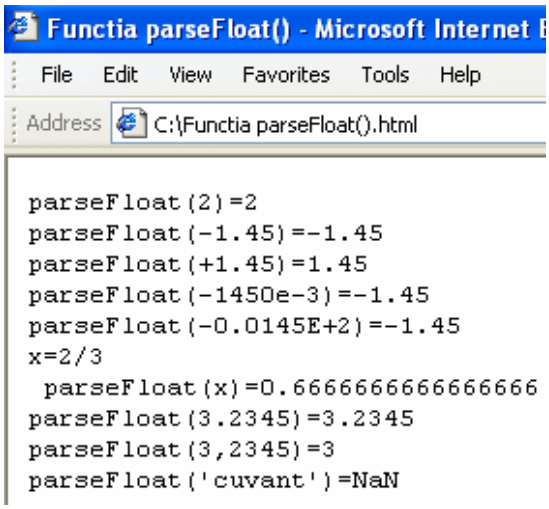
Exemplu de utilizare a funcției isNaN().

```
<html>
<head>
 <title>
   Functia isNaN()
 </title>
</head>
<body>
<pre>
<script type="text/JavaScript">
 document.writeln("isNaN(2)="+isNaN(2));
 document.writeln("x=2/5   <br />
    isNaN(Math.x)="+isNaN(Math.x));
 document.writeln("isNaN(Math.pi)="+isNaN(Math.pi));
 document.writeln("isNaN(3.2345)="+isNaN(3.2345));
 document.writeln("isNaN(3,2345)="+isNaN(3,2345));
 document.writeln("isNaN('cuvant')="+isNaN('cuvant')
    );
</script>
</pre>
</body>
</html>
```



Funcția parseFloat() transformă un şir de caractere într-un număr de tip Float. Dacă primul caracter din şir nu este număr, funcția va returna *NaN* (adică Not a Number).
Sintaxa: ***parseFloat*** *(şir de caractere)*

Exemplu În exemplul următor am utilizat ca şir de caractere, numere pozitive şi negative, numere cu zecimale şi cu exponent. La cel de-al patrulea exemplu, am utilizat -1450e-3, unde *e* reprezintă 10 la puterea dată de numărul care urmează, respectiv -3. Deci -1450e-3 este $-1450*10^{-3}$ adică -1.45.

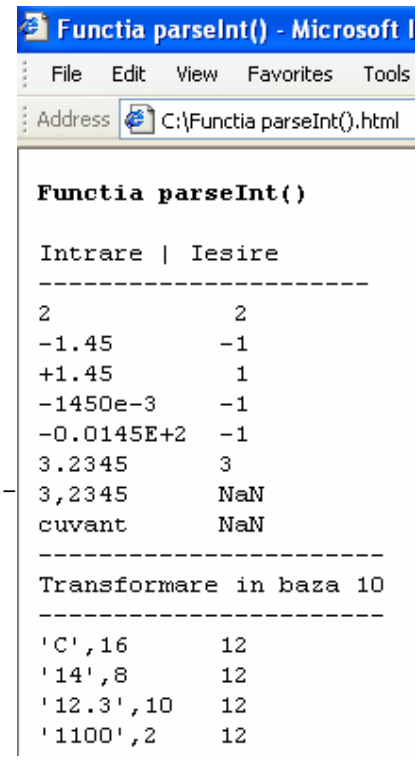



```
<html>
<head>
 <title>Functia parseFloat()</title>
</head>
<body>
<pre>
<script type="text/JavaScript">
 document.writeln("parseFloat(2)="+parseFloat(2));
 document.writeln("parseFloat(-1.45)="+parseFloat(-1.45));
 document.writeln("parseFloat(+1.45)="+parseFloat(+1.45));
 document.writeln("parseFloat(-1450e-3)="+parseFloat(-1450e-3));
 document.writeln("parseFloat(-0.0145E+2)="+parseFloat(-0.0145E+2));
 var x=2/3;
 document.writeln("x=2/3   <br /> parseFloat(x)="+parseFloat(x));
 document.writeln("parseFloat(3.2345)="+parseFloat(3.2345));
 document.writeln("parseFloat(3,2345)="+parseFloat(3,2345));
 document.writeln("parseFloat('cuvant')="+parseFloat('cuvant'));
</script>
</pre>
</body>
</html>
```

Funcția parseInt() transformă un şir de caractere într-un număr de tip Int. Lângă şirul de caractere, poate fi specificată baza de numerație din care va fi transformat în baza 10. Cu ajutorul funcției *parseInt(),* putem transforma un număr din bazele de numerație 2, 8, 10, 16, în baza 10, dacă va fi specificată baza din care va fi transformat numărul. Dacă primul caracter din şir nu este număr, sau dacă nu va fi specificată corect baza de numerație, funcția va returna NaN (adică Not a Number).
Sintaxa: ***parseInt*** *(şir de caractere[,baza de numerație])*

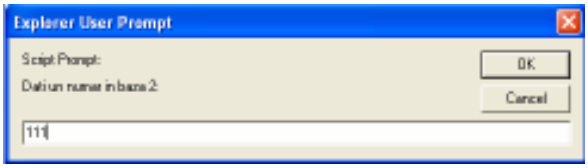
Exemplu În prima parte a exemplului de mai jos, am transformat diferite numere în numere întregi, iar în a doua parte am trasformat numere scrise în diverse baze de numerație, în baza 10.

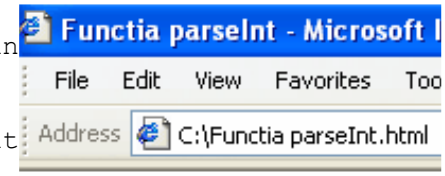
```
<html>
<head>
 <title>Functia parseInt()</title>
</head>
<body>
<pre>
<script type="text/JavaScript">
document.writeln("<b>Functia parseInt()
</b><br />");
document.writeln("Intrare | Iesire");
document.writeln("----------------------");
document.writeln("2              "+parseInt(2));
document.writeln("-1.45        "+parseInt(-1.45));
document.writeln("+1.45         "+parseInt(+1.45));
document.writeln("-1450e-3   "+parseInt(-1450e-3));
document.writeln("-0.0145E+2              "+parseInt(-
   0.0145E+2));
document.writeln("3.2345       "+parseInt(3.2345));
document.writeln("3,2345       "+parseInt(3,2345));
document.writeln("cuvant    "+parseInt('cuvant'));
document.writeln("-----------------------");
document.writeln("Transformare in baza 10");
document.writeln("-----------------------");
document.writeln("'C',16       "+parseInt("C",16));
document.writeln("'14',8       "+parseInt("14",8));
document.writeln("'12.3',10
   "+parseInt("12.3",10));
document.writeln("'1100',2 "+parseInt("1100",2));
</script></pre>
</body></html>
```



Exemplu În următorul exemplu transformăm un număr citit de la tastatură, din baza 2 în baza 10. Pentru aceasta, am transformat numărul cu ***parseInt*** şi am verificat cu ***isNaN***, dacă ceea ce a tastat utilizatorul este sau nu număr.

```
<html>
<head>
 <title>Functia parseInt</title>
<script  language="JavaScript">
function t(x)
 { y=parseInt(x,2);
     if (isNaN(y))
       document.writeln("Nu  ati  tastat  un
      numar in baza 2!");
     else
   document.writeln("Numarul    transformat
      \nin baza 10 este:<b> ",y,"</b>");
 }
</script></head>
<body><pre>
<script  language="JavaScript">
x=eval(prompt("Dati   un   numar   in   baza
2:"));
t(x);
</script></pre>
</body></html>
```

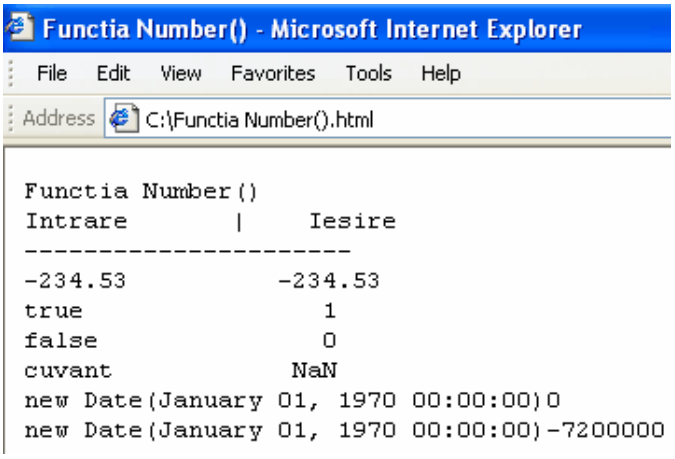




Funcția Number() transformă parametrul unui obiect specificat într-un număr, reprezentând valoarea obiectului respectiv. Dacă valoarea respectivă nu este un număr, va fi returnat *NaN*. Dacă obiectul specificat este *Date,* funcția va returna valoarea în milisecunde, măsurată din 1 ianuarie 1979 UTC (GMT), pozitivă după această dată şi negativă înainte.
Sintaxa: ***Number*** *(obiect)*

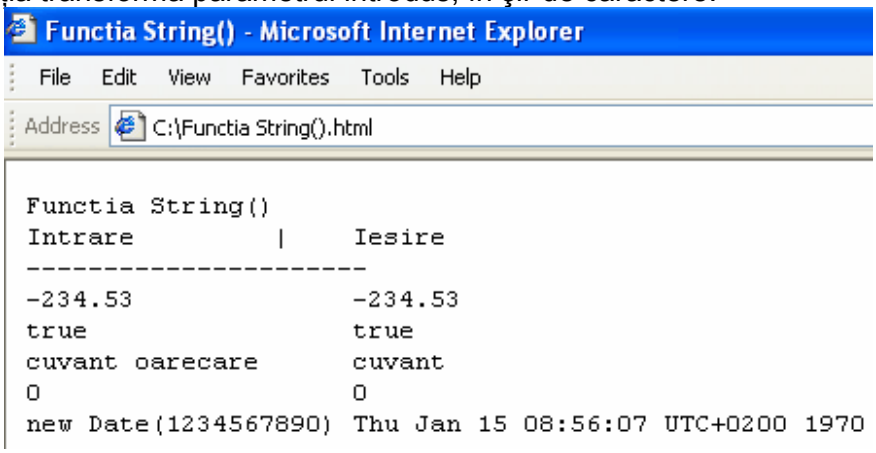
Exemplu Funcția returnează zero dacă data specificată ca parametru, va fi cea din enunț, +2, pentru Bucureşti. Dacă scriem ora 00:00:00, ne va returna o dată negativă, reprezentând cele două ore transformate în milisecunde, deoarece pentru Bucureşti avem GMT+2.

```
<html>
<head><title>Functia Number()</title>
</head>
<body><pre>
<script  language="JavaScript">
document.writeln("Functia Number()");
document.writeln("Intrare            |
   Iesire");
document.writeln("-------------------
   ---");
document.writeln("-234.53
   "+Number(-234.53));
document.writeln("true
   "+Number(true));
document.writeln("false
   "+Number(false));
document.writeln("cuvant
   "+Number("cuvant"));
document.writeln("new     Date(January
   01,    1970    00:00:00)"+Number(new
   Date("January 01,
   1970 02:00:00")));
document.writeln("new     Date(January
   01,    1970    00:00:00)"+Number(new
   Date("January 01,
   1970 00:00:00")));
</script></pre>
</body></html>
```



Funcția String() transformă un obiect specificat într-un String (şir de caractere). Dacă obiectul este Date, funcția va returna un şir de caractere ce reprezintă data calendaristică, după cum putem vedea în exemplul de mai jos.
Sintaxa: ***String*** *(obiect)*

Exemplu Funcția transformă parametrul introdus, în şir de caractere.



```
<html>
<head>
 <title>Functia String()</title>
</head>
<body>
<pre>
<script  language="JavaScript">
 document.writeln("Functia String()");
 document.writeln("Intrare         |    Iesire");
 document.writeln("----------------------");
 document.writeln("-234.53              "+String(-234.53));
 document.writeln("true                 "+String(true));
 document.writeln("cuvant oarecare      "+String("cuvant"));
 document.writeln("0                    "+String(0));
 document.writeln("new Date(1234567890) "+String(new Date(1234567890)));
</script>
</pre>
</body>
</html>
```

6.6. Funcții şi stiluri

Nu este suficient să ştim să utilizăm funcții şi stiluri. Important este ca tot ceea ce ştim să putem utiliza pentru a realiza diverse aplicații mai complexe. Tocmai acest lucru voi încerca să realizez în exemplul de mai jos.

Stiluri pentru câmpul textarea

În acest exemplu am introdus un text într-un câmp de tip textarea pe care îl pot afişa în trei moduri, utilizând stiluri diferite. Pentru aceasta, am definit cele trei stiluri s1, s2 şi s3, în antetul paginii Web. Stilurile sunt apelate prin intermediul funcțiilor: f2, f3 şi f4, la declanşarea evenimentului onClick prin apăsarea butoanelor: Stil 1, Stil 2 şi Stil 3. Funcțiile f2, f3 şi f4 utilizează masivul a, care pentru prima valoare (t=0), va utiliza stilul s1, pentru a doua valoare (t=1), va utiliza stilul s2 şi pentru a treia valoare (t=2), va utiliza stilul s3.

În concluzie, scriem un text în câmpul textarea, apăsăm butonul pe care scrie **Apasati!**, alegem unul din stiluri prin apăsarea butoanelor: Stil 1, Stil 2 sau Stil 3, după care apăsăm butonul pe care scrie **click**. Implicit, textul afişat este cel scris în body şi anume o strofă dintr-o poezie scrisă de Ienăchiță Văcărescu.

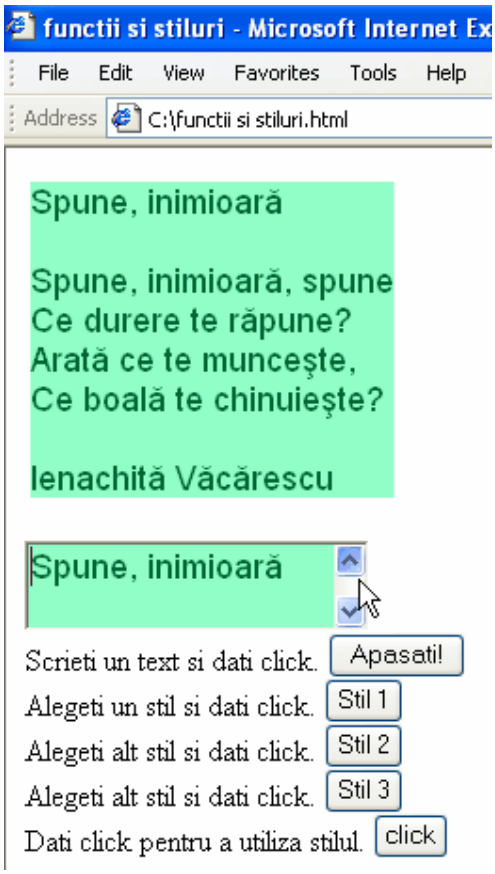
```
<html>
<head>
 <title>functii si stiluri</title>
<style>
.s1
 {  font-family: Arial, Helvetica, sans-serif;
    font-size: 18px;
    color: #006633;
    background-color: #91FFC8;
 }
.s2
 {  font-family: Georgia, Times New Roman, Times, serif;
    font-size: 16px;
    color: #990000;
    background-color: #FF8282;
 }
.s3
 {  font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 18px;
    color: #3333FF;
```

```
    background-color: #9B9BFF;
 }
</style>
<script>
function f1()
 {
   var reg=/\n/g;
   var getElementById=document.formular.ta.value;
   y=getElementById.replace(reg,"<BR>");
   document.getElementById("target").innerHTML=y;
  }
var t=0;
var a=new Array("s1","s2","s3");
function f2()
 {
   t=0;
   document.getElementById("abc").className=a[t];
 }
function f3()
 {
   t=1;
   document.getElementById("abc").className=a[t];
 }
function f4()
 {
   t=2;
   document.getElementById("abc").className=a[t];
 }
function f()
 {
   document.getElementById("target").className=document.
            getElementById("abc").className;
 }
</script>
</head>
<body>
   <table>
   <tr>
     <td>
         <div id=target class=s2>Scrieti un text si dati click!
         </div>
     </td>
   </tr>
   </table>
<form name=formular>
   <textarea name=ta class=s1 id=abc>
      Spune, inimioar&#259;

      Spune, inimioar&#259;, spune
      Ce durere te r&#259;pune?
      Arat&#259; ce te munce&#351;te,
      Ce boal&#259; te chinuie&#351;te?

      Ienachit&#259; V&#259;c&#259;rescu
   </textarea>
   <br />
   Scrieti un text si dati click.
   <input type=button value="Apasati!" onclick="f1()"><br />
   Alegeti un stil si dati click.
```

```
   <input type=button value="Stil 1" onclick="f2()"><br />
   Alegeti alt stil si dati click.
   <input type=button value="Stil 2" onclick="f3()"><br />
   Alegeti alt stil si dati click.
   <input type=button value="Stil 3" onclick="f4()"><br />
   Dati click pentru a utiliza stilul.
   <input type=button value="click" onclick="f()"><br />
</form>
</body>
</html>
```



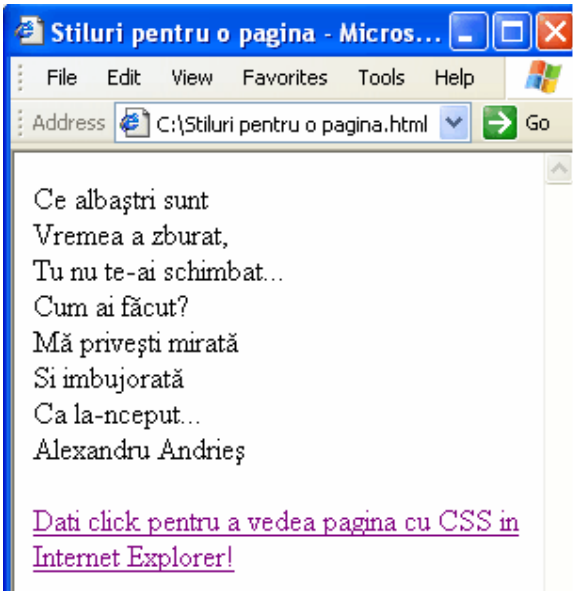
Stil diferit pentru o pagină

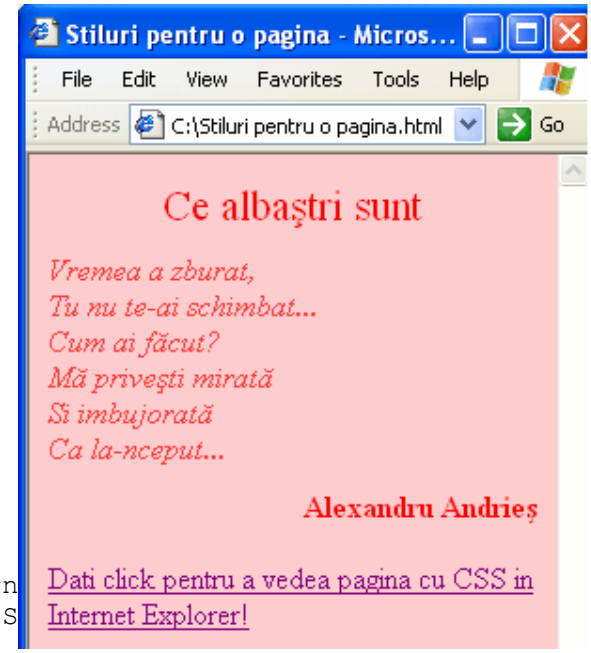
Atunci când vizităm o pagină, de cele mai multe ori avem impresia că are importanță numai conținutul, nu şi aspectul paginii. Mai jos, am să încerc să demonstrez faptul că puțină atenție acordată aspectului, nu strică niciodată. Deci, efectuând click pe legătura din partea de jos a paginii, se va apela funcția f() care va da atribute diferite stilurilor utilizate. Tot funcția f() ne va atenționa în cazul în care utilizăm alt browser, în afară de Internet Explorer, în care se poate folosi acest exemplu.

```
<html>
<head>
 <title>Stiluri pentru o pagina</title>
<script type=text/javascript>
function f()
 {
  if(!document.all)
   { alert("Nu folositi Internet Explorer!")
     return
   }
 document.body.style.backgroundColor="#FFCCCC"
 document.all.a1.style.color="red"
 document.all.a1.style.fontSize="22px"
 document.all.a1.style.textAlign="center"

 document.all.a2.style.color="#FF3C3C"
 document.all.a2.style.fontStyle="oblique"
 document.all.a2.style.textAlign="left"
 document.all.a2.style.marginTop="12px"

 document.all.a3.style.color="red"
 document.all.a3.style.fontWeight="bold"
 document.all.a3.style.fontStyle="normal"
 document.all.a3.style.textAlign="right"
 document.all.a3.style.marginTop="12px"
}
</script>
</head>
<body>
<div id=a1>Ce alba&#351;tri sunt</div>
<div id=a2>
Vremea a zburat,<br />
Tu nu te-ai schimbat...<br />
Cum ai f&#259;cut?<br />
M&#259;  prive&#351;ti mirat&#259;<br />
Si imbujorat&#259;<br />
Ca la-nceput...<br /></div>
<div id=a3>Alexandru Andrie&#351;</div>
<br><a       href="#"       onClick="f();return
false">Dati click pentru a vedea pagina cu CSS
in Internet Explorer!</a>
</body>
</html>
```





Butonul alert decorat

Cu ajutorul funcțiilor şi a stilurilor, putem decora butonul alert, după cum dorim noi. Am utilizat stiluri pentru toate elementele butonului alert: fundal, margini, imagine pentru fundal, poziția butonului de inchidere, distanța dintre litere, distanță pentru margini.

```
<html>
<head>
 <title>Butonul alert</title>
<style type="text/css">
#dreptunghi
 { position:absolute;
   width:auto;
   height:auto;
   top:0px;
   left:0px;
   z-index:10000;  }
```

```
#drA
 { position:relative;
   width:300px;
   min-height:100px;
   margin-top:50px;
   border:2px solid #FFFF00;
   background-repeat:no-repeat;
   background-position:20px 30px;
   visiblity:hidden;
   background-image: url(1trandafiri_albi.jpg);
   background-color: #FFFFCC;  }
#dreptunghi > #drA
 { position:fixed;  }
#drA h1
 { margin: 0;
   font:bold 0.9em verdana,arial;
   background-color:#FFFF6C;
   color:#000000;
   padding:2px 0 2px 5px;
   text-align: center;
   border-top-width: thin;
   border-right-width: thin;
   border-bottom-width: thin;
   border-left-width: thin;
   border-top-style: none;
   border-right-style: none;
   border-bottom-style: none;
   border-left-style: none;  }
#drA p
 { font:0.7em verdana,arial;
   height:50px;
   padding-left:5px;
   margin-left:55px;  }
#drA #Inchide_Buton
 { display:block;
   position:relative;
   padding:3px;
   border:2px solid #FFFF6c;
   width:100px;
   text-transform:none;
   text-align:center;
   color:#000000;
   background-color:#FFFF6c;
   font-family: Geneva, Arial, Helvetica, sans-serif;
   font-size: 14px;
   margin-top: 5px;
   margin-right: auto;
   margin-bottom: 5px;
   margin-left: auto;
   text-decoration: none;
   font-weight: bold;
   letter-spacing: 0.3em;
   left: 100px;
   bottom: 20px;  }
h1
 { margin:0;
   padding:4px;
   font:bold 1.5em verdana;
   border-bottom:1px solid #000;  }
.important
 { background-color:#F5FCC8;
```

```
   padding:2px;  }
</style>
<script type="text/javascript">
var titlul_alert = "Mesajul transmis";
var text_buton = "Inchideti!!";
if(document.getElementById)
 {
   window.alert = function(x)
    { f(x);  }
 }
function f(x)
 { d = document;
   if(d.getElementById("dreptunghi"))
      return;
   obiect         =         d.getElementsByTagName("body")[0].appendChild
   (d.createElement("div"));
   obiect.id = "dreptunghi";
   obiect.style.height = d.documentElement.scrollHeight + "px";
   obiectA = obiect.appendChild(d.createElement("div"));
   obiectA.id = "drA";
   if(d.all       &&       !window.opera)       obiectA.style.top       =
   document.documentElement.scrollTop + "px";
   obiectA.style.left       =       (d.documentElement.scrollWidth       -
   obiectA.offsetWidth)/2 + "px";
   obiectA.style.visiblity="visible";
   h1 = obiectA.appendChild(d.createElement("h1"));
   h1.appendChild(d.createTextNode(titlul_alert));
   msg = obiectA.appendChild(d.createElement("p"));
   msg.appendChild(d.createTextNode(x));
   Buton = obiectA.appendChild(d.createElement("a"));
   Buton.id = "Inchide_Buton";
   Buton.appendChild(d.createTextNode(text_buton));
   Buton.href = "#";
   Buton.onclick = function()
                          { ra();
                            return false;
                           }
 }
function ra()
 {document.getElementsByTagName("body")[0].removeChild(document.
   getElementById ("dreptunghi"));
 }
</script>
</head>
<body>
<p align="center">
<input  type="button"  value  ="Dati  click  pentru  a  vedea  mesajul"
onClick="alert('Mesajul pe care l-ati dorit :)');" /></p>
</body>
</html>
```

Evaluare

10. Cum se defineşte o funcție?
11. Cum se apelează o funcție?
12. Ce funcție transformă un şir de caractere într-un număr de tip întreg?
13. Scrieți un script ce permite citirea conținutului a două variabile şi o funcție ce calculează elementele minim şi maxim dintre cele două numere citite.
14. Cum apelăm funcția cu numele f, de parametrii x şi y?
    a) f(x,y)
    b) f parametri x, y
    c) function f(x,y)
    d) f(x and y)
15. Cum creăm o funcție cu numele f de parametri x şi y?
    a) function:x,y
    b) function=f(x, y)
    c) function f(var x, var y)
    d) function f(x, y)