Cuprins

Introducere –definițiile

Ce înseamnă WWW (World Wide Web)?

- foarte des World Wide Web (WWW) este denumit prescurtat "**Web-ul"**.
- este o rețea de calculatoare mondială.
- se foloseşte protocolul de comunicație **HTTP (HyperText Transfer Protocol)**.

Cum funcționează WWW?

- informația este salvata în documente numite pagini web.
- paginile web sunt fişiere text salvate pe un calculator numit **server de web**.
- calculatoarele care accesează paginile web sunt denumite **clienți web**.
- pentru a vizualiza paginile web un client web are nevoie de un program denumit
- **browser**.
- browsere cunoscute: **Internet Explorer, Netscape Navigator, Opera, Mozila .....**

Cum aduce browserul paginile de web?

- browserul face o cerere către server (**request**).
- cerere standard către un server conține o adresa.
- exemplu: **http://www.csac.ulbsibiu.ro/orar.htm**

Cum afişează browser-ul paginile?

- toate paginile web conțin **instrucțiuni (pentru afişaj) –** spre norocul nostru acestea nu trebuie compilate ele sunt doar interpretate iar în caz de eşec browserul nu afişează nimic
- instrucțiunile sunt denumite **tag-uri HTML** (marcatori).

Exemplu:

```
<p>Acesta este un paragraf si este un tag
foarte important în standardul HTML 4</p>.
```

Cine face standardele web?

- un grup nonguvernamental denumit **W3C**.
- W3C înseamnă **World Wide Web Consortium**.
- W3C stabilesc specificațiile pentru **standardele web**.
- cele mai importante standarde web sunt **HTML, CSS şi XML**.

Ce este un fişier HTML?

- HTML este prescurtarea pentru **H**yper **T**ext **M**arkup **L**anguage
- un fişier HTML este un fişier text care conține **tag-uri**
- un fişier HTML trebuie să aibă extensia **htm** sau **html**
- un fişier HTML poate fi creat cu ajutorul unui **editor simplu de text** (ex: NOTEPAD)

1 – Structura standard a unui document html (HTML 4.01)

Primul cod html

Putem porni fără teama la scrierea fişierelor html pentru că niciodată nu veți vedea fereastra urât colorata a compilatorului care te anunță că ai ceva erori şi chiar mai multe „warning-uri" umbrită de privirea dojenitoare a profesorului care îți da de înțeles că iarăşi ai uitat „un punct şi virgulă".
Pornim un editor simplu - Notepad. Tastati urmatoarea secvență:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<title>Prima</title>
</head>
<body>
Buna ziua Web. <b>textul e scris
ingrosat</b>
</body>
</html>
```

Observație: Nu lăsați un spațiu după „<" pentru că browserul nu mai recunoaşte instrucțiunea.
Salvam fişierul cu numele "`prima.htm`".
Pornim browserul şi deschidem fişierul prima.htm
Ar trebui să se afişeze:

Explicarea exemplului:
Înainte de a începe observam că avem mai multe **perechi** de tag-uri. Primul marchează începutul unui bloc de text iar tag-ul care începe cu „/" şi are acelaşi nume cu tag-ul de deschidere marchează terminarea blocului de text.
Primul tag dintr-un document HTML este un comentariu.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

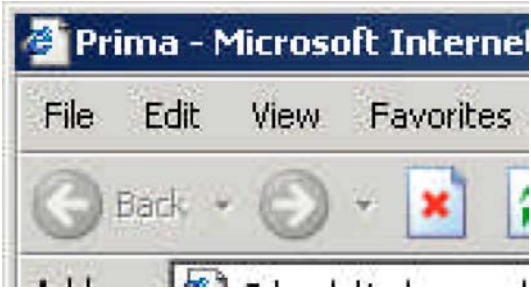
Acest comentariu este important pentru standardizarea documentelor html. Pentru web-ul semantic care va urma este deosebit de importantă respectarea standardelor. Motoarele de căutare inteligente vor „şti" însemnătatea tag-urilor iar astfel rezultatele căutărilor vor fi mult mai precise. Comentariul din exemplul de mai sus specifică faptul că documentul html care urmează este scris conform standardului `HTML 4.01 Transitional` iar acest standard se găseşte la URN-ul

`http://www.w3.org/TR/html4/loose.dtd`

`Tag-ul <html>`. Acesta are rolul de a anunța browserul că urmează un document HTML. La sfârşitul documentului avem `</html>` acesta atenționând browserul ca a ajuns la sfârşitul documentului HTML.
În continuare avem doua delimitări importante ale documentului

partea de **head** şi cea de **body**
Blocul de text dintre `<head>` şi `</head>` este informația din header.
Foarte important:
informația din cadrul header-ului nu se afişează în fereastra browserului. Ea este destinată descrierii documentului.

Tag-urile <title> marchează titlul documentului si acesta va fi afişat pe fereastra browserului.



Tag-urile de `<body>`delimitează zona de text care va fi afişată în cadrul ferestrei browser-ului. Textul delimitat de si va fi afişat îngroşat.

Ce extensie folosim pentru fişierele documentelor web: HTM sau HTML?

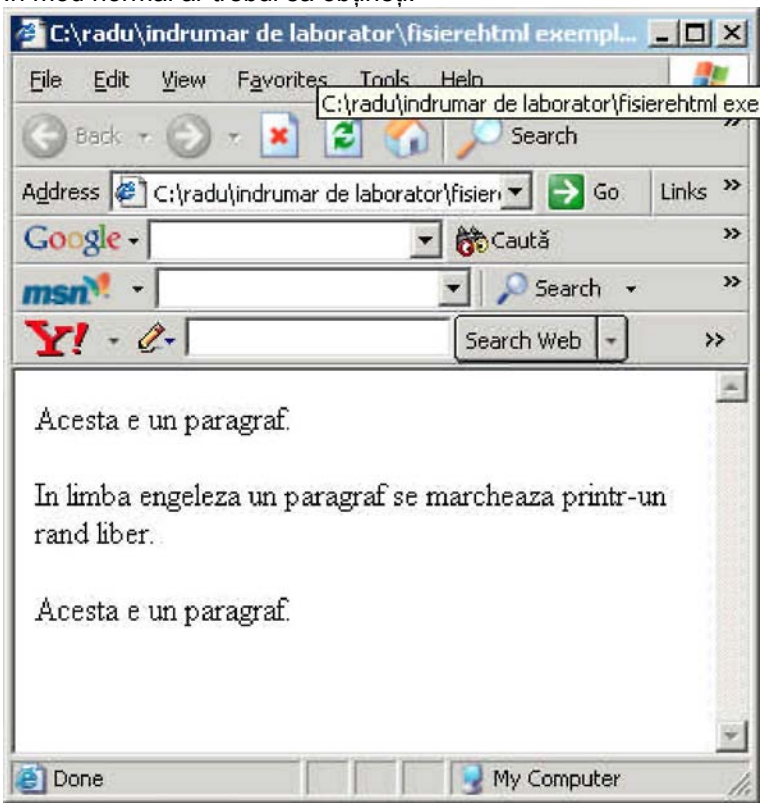
În principiu nu contează pentru că ambele sunt interpretate la fel. Extensia „HTM" a devenit mai populară datorită faptului că majoritatea extensiilor fişierelor sunt formate din 3 litere. Oricum consecvența în utilizarea unei singure extensii este foarte importantă.

Exemplu:
Tastați următorul cod:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
```

```
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<body>
<p>Acesta e un paragraf.</p>
<p>In limba engeleza un paragraf se marcheaza
printr-un rand liber.</p>
<p>Acesta e un paragraf.</p>
</body>
</html>
```

In mod normal ar trebui sa obțineți:



Observație:

În exemplele de cod html ce vor urma nu se va reproduce în totalitate structura standard a unui document html. Această cerință rămâne valabilă în realizarea exemplelor de către studenți.

2 – Tagurile HTML

Fişierele **HTML** sunt fişiere text care conțin elemente **HTML.** Elementele HTML sunt definite cu ajutorul tag-urilor.

Tag-urile HTML

- Tag-urile HTML se folosesc pentru a delimita **elemente HTML**; forma unui tag este **<nume_tag>**
- De obicei tag-urile sunt perechi: **<nume_tag>** bla, bla **</nume_tag>**
- Textul delimitat de cele doua tag-uri se numeşte **conținutul elementului**

Elementele HTML

Revenim la exemplul dat:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<head>
<title>prima</title> </head>
<body>
Buna ziua Web <b>bla bla</b>
</body>
</html>
```

Din acest fişier **un element HTML** este: `<b>bla bla</b>`
Tag-ul <b> are ca scop definirea unui element HTML care sa fie afişat bold. Un alt exemplu de element HTML:

```
<body>
Buna ziua Web <b>bla bla</b>
```

```
</body>
```

Tag-ul <body> defineşte un element HTML care conține corpul documentului HTML.

Recomandare:
W3C recomanda utilizarea literelor mici în scrierea tag-urilor, aceasta fiind recomandarea pentru standardul HTML 4. Pentru cei ce se pregătesc să scrie tag-uri conform standardului XHTML 1.0 utilizarea literelor mici este obligatorie.

Atributele tag-urilor

Tag-urile pot avea atribute

<nume_tag *atr*> bla, bla **</nume_tag>**

In exemplul de mai sus ***atr*** este atributul tagului. Aceste atribute pot aduce informații suplimentare despre elementele HTML.

Exemplu:

```
<body bgcolor="red">.
```

În exemplul de mai sus tag-ul <body> a primit atributul “bgcolor” adică se specifică browserului că fundalul paginii este roşu. Întotdeauna atributele sunt însoțite de o valoare care trebuie scrisă între “ “
La noi valoarea atributului **bgcolor** este **red.**

Observații:
1. Întotdeauna atributul pentru tag se adaugă doar tag-ului de deschidere. Adică

```
<body bgcolor="red">.
Bla.. Bla
</body>.
```

2. Valorile atributelor pentru atribute trebuie să fie scrise întotdeauna între ghilimele. Se pot utiliza si ‘ ‘. În cazurile rare când valoarea atributului necesita ghilimele atunci se poate proceda conform

exemplului:

```
name='Ion zis "Macelarul" '
```

Tag-urile HTML de bază

Cele mai importante tag-uri in HTML sunt tag-urile care definesc

- antelele (headings)
- paragrafe (paragraphs)
- trecerea forțată la rând nou (line breaks).

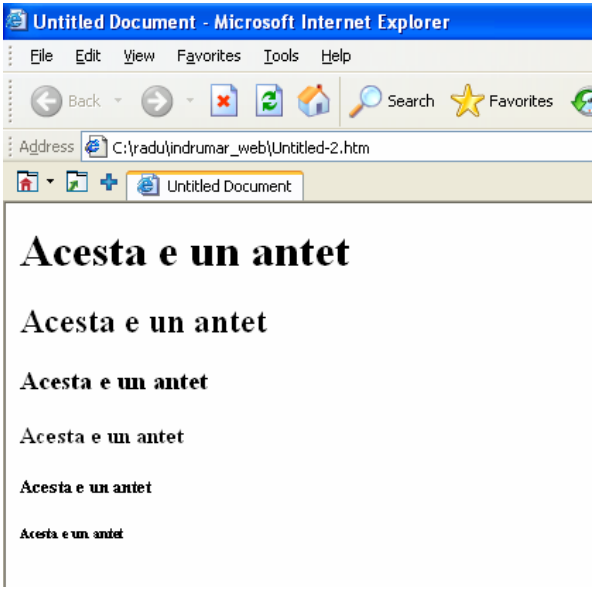
Tag	**Descriere**
<html>	Defineşte un document HTML
<body>	Defineşte corpul documentului HTML
<h1> to <h6>	Defineşte antetul 1 la 6
<p>	Defineşte un paragraf
 	Sare la rând nou
<hr>	Defineşte o linie orizontală
<!-->	Defineşte un comentariu

Headings (titlurile într-o pagină)

Headings sunt definite cu tag-urile <h1> până la <h6>,. <h1> defineşte cel mai important heading. <h6> defineşte cel mai “mic” heading.

```
<h1>Acesta e un antet</h1> <h2> Acesta e un
antet </h2> <h3> Acesta e un antet </h3> <h4>
Acesta e un antet </h4> <h5> Acesta e un
antet </h5> <h6> Acesta e un antet </h6>
```

Acest cod va genera în fereastra browserului:



Observație:
Se adăuga automat de către browser un rând liber înainte şi după declararea unui heading.

Paragrafe

Paragrafele sunt definite cu ajutorul tagului `<p>`.

```
<p>Un paragraf</p>
<p>Alt paragraf</p>
<p>Bala, bla, bla</p>
```

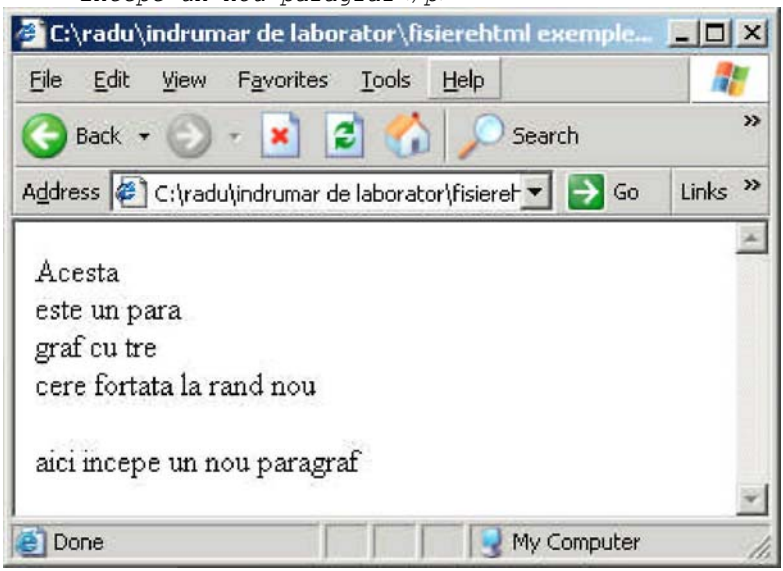
Observație: Se adăuga automat de către browser un rând liber înainte şi după declararea unui paragraf.

Rând nou

Pentru a forța trecerea la rând nou fără începerea unui paragraf nou se utilizează tag-ul `<br />`

Exemplu:

```
<p>Acesta <br> este un para<br>graf cu
tre<br>cere fortata la rand nou</p><p> aici
incepe un nou paragraf</p>
```



Observație:
Tag-ul
 nu are un tag de închidere (slash-ul de la sfârşit ține loc de tag de închidere).

Comentarii in HTML

Pentru a insera un comentariu se foloseste tag-ul <!--….>

```
<!--Un comentariu destept-->
```

Observatie:
Important este doar semnul exclamării şi este necesar doar după deschiderea parantezei unghiulare! Comentariul nu este afişat de către browser dar câteodată este bine sa-l folosim pentru lizibilitatea codului.

Vizualizarea sursei unui document HTML

Acest lucru se poate face in doua moduri:

1. Din meniul browser-ului utilizând meniul **View -> Source**
2. Cu click dreapta în fereastra browser-ului şi apoi **View Source**

Exemple

1. Centrarea unui heading în pagina: `<html> <body>`

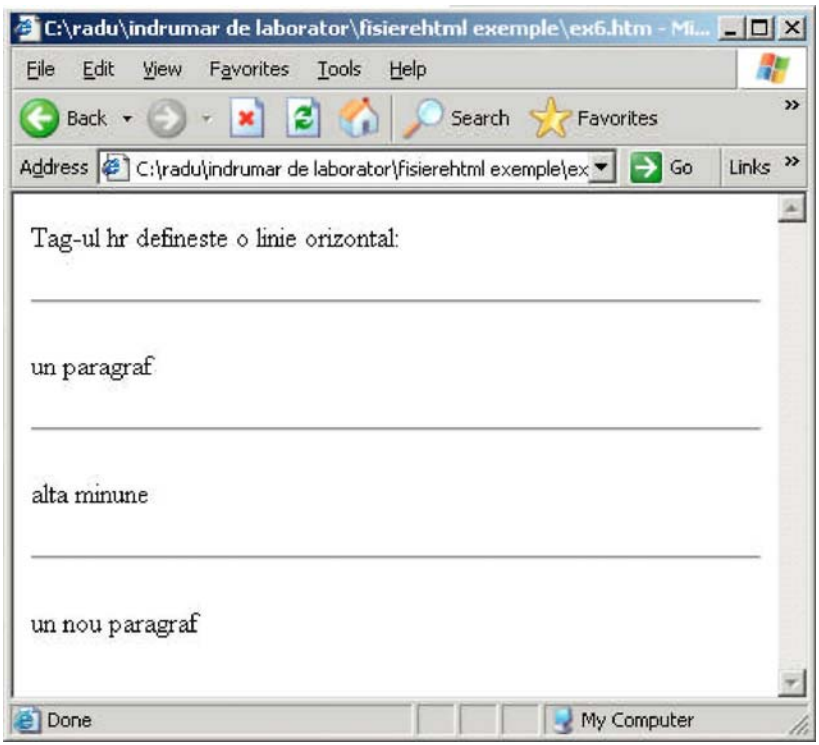
```
<h1 align="center">Acesta e un heading 1</h1>
<p>Heading-ul de sus este aliniat centrat.
Heading-ul de sus este aliniat centrat.
Heading-ul de sus este aliniat centrat.</p>
</body>
</html>
```

Codul generează următoarea afişare:

2. Linie orizontală

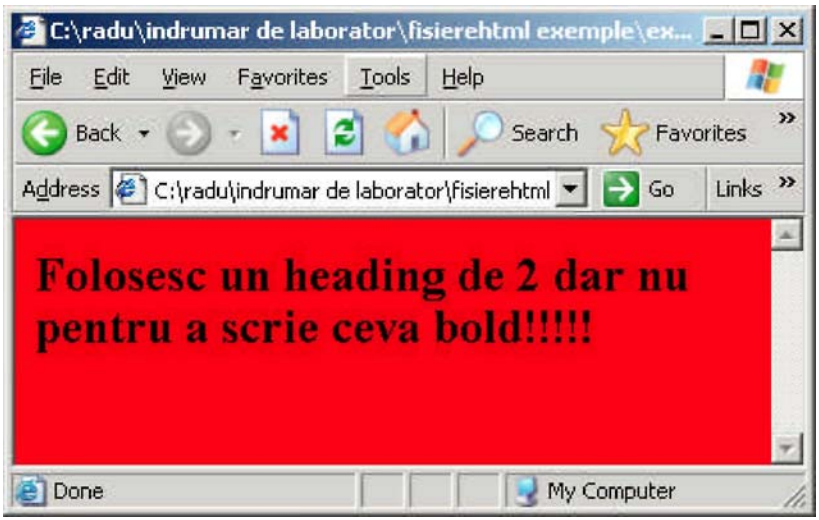
```
<html> <body>
<p>Tag-ul hr defineste o linie orizontal:</p>
<hr>
<p>un paragraf</p> <hr>
<p>alta minune</p> <hr>
<p>un nou paragraf</p> </body> </html>
```

Codul generează următoarea afişare:



3. Culoarea de fundal pentru pagina:

```
<html>
<body bgcolor="red">
<h2>Folosesc un heading de 2 dar nu pentru a
scrie ceva bold!!!!!</h2> </body> </html>
```



Tag-uri pentru formatarea textului

Tag	Descriere
<b>	Defineşte text bold
<big>	Defineşte text bold
	Defineşte text italic
<i>	Defineşte text italic
<small>	Defineşte text mic
	Defineşte text îngroşat
<sub>	Defineşte text scris la index (subscript)
<sup>	Defineşte text scris la putere (superscript)
<ins>	Defineşte text inserat
<del>	Defineşte text şters
<s>	Nu se mai utilizează. Se foloseşte<del>
<strike>	Nu se mai utilizează. Se foloseşte <del>

Tag-uri pentru "Computer Output"

Tag	Descriere
<code>	Defineşte cod text
<kbd>	Defineşte un text de tastatura
<samp>	Defineşte un exemplu de cod
<tt>	Defineşte text teletype
<var>	Defineşte o variabila

Tag-uri pentru citări, indentări şi definiri

Tag	Descriere
<abbr>	Defineşte o abreviere
<acronym>	Defineşte un acronim
<address>	Defineşte un element HTML de tip adresa
<bdo>	Defineşte direcția de afişare a textului
<blockquote>	Defineşte o intentare
<cite>	Defineşte o citare
<dfn>	Defineşte un element HTML de tip definiție

Caractere speciale

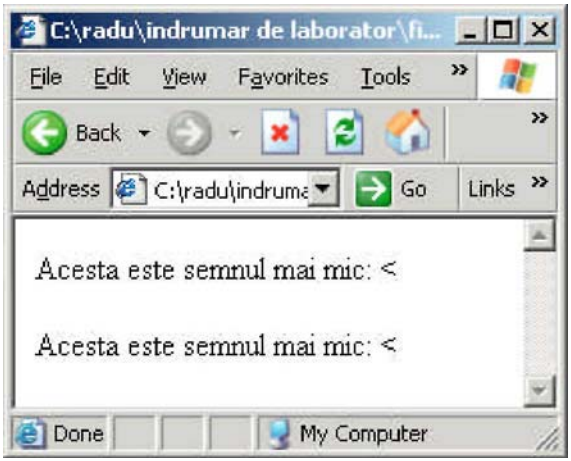
Unele caractere cum ar fi < au în HTML un rol special şi de aceea nu pot fi folosite în cadrul textului. Pentru a putea totuşi utiliza semnul “mai mic ca” vom folosi entitățile de caractere (caractere speciale)

Caracterul special are trei părți:

1. începe cu &
2. nume sau un număr precedat de semnul #
3. la final au semnul (;).

Exemplu:

```
<p> Acesta este semnul mai mic: &lț</p> <p>
Acesta este semnul mai mic: &#60;</p>
```



Observație: Caracterele speciale sunt “case sensitive”.

Caracterele speciale cele mai des utilizate:

Semnul afişat	Descrierea	Numele entității	Numărul entității
	non-breaking space		
<	mai mic ca	&lț	<
>	mai mare ca	&gț	>
&	ampersand	&	&
"	ghilimele	&quoț	"
'	apostrof	'	'
¢	cent	&cenț	¢
£	lira	£	£
¥	yen	¥	¥
§	secțiune	&secț	§
©	copyright	©	©

®	marca înregistrată	®	®
X	înmulțire	×	×
÷	împărțire	÷	÷

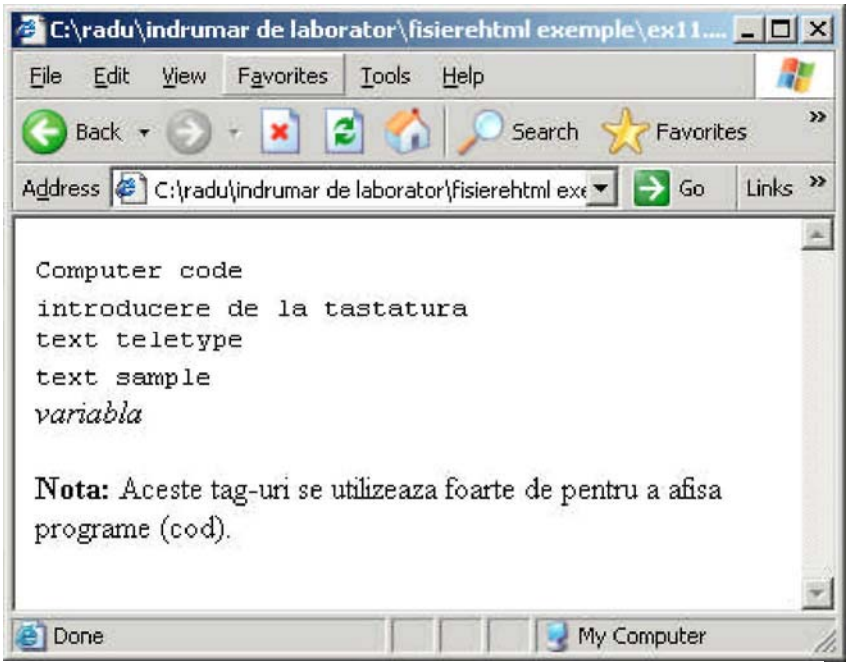
Exemple diverse:

1. Formatarea textului

```
<html>
<body>
<b>text bold</b> <br>
<strong> text strong </strong> <br>
<big> text big </big> <br>
<em>
text emphasized </em> <br>
<i>
text italic </i> <br>
<small>
text small
</small>
<br>text
<sub>
subscript
</sub>
<br>
text
<sup>
superscript
</sup>
</body>
</html>
```

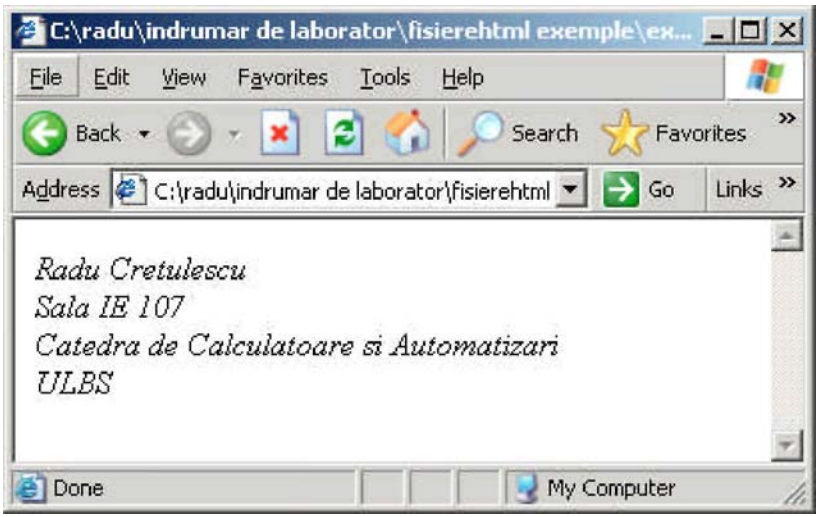
2. Vizualizarea unui cod de program

```
<html> <body>
<code>Computer code</code>
<br>
<kbd>introducere de la tastatura</kbd>
<br>
<tt>text teletype </tt>
<br>
<samp> text sample</samp>
<br>
<var> variabla</var>
<br>
<p>
<b>Nota:</b> Aceste tag-uri se utilizeaza
foarte de pentru a afisa
programe (cod).
</p>
</body>
</html>
```



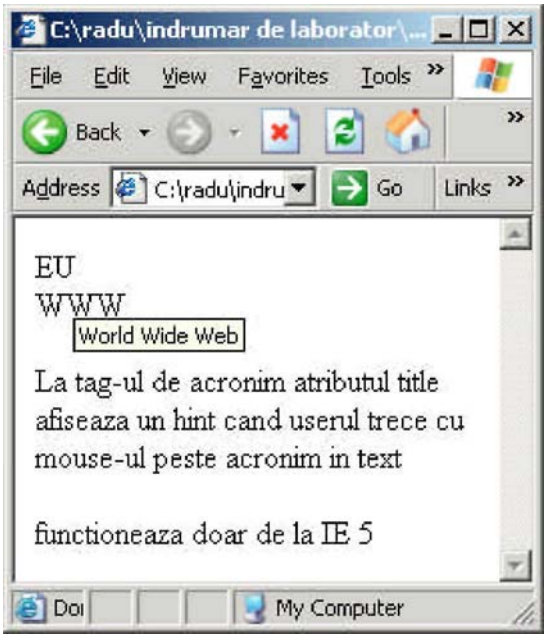
3. Inserarea unei adrese

```
<html> <body>
<address> Radu Cretulescu<br> Sala  IE
107<br>
Catedra de Calculatoare si Automatizari<br>
ULBS
</address>
</body>
</html>
```



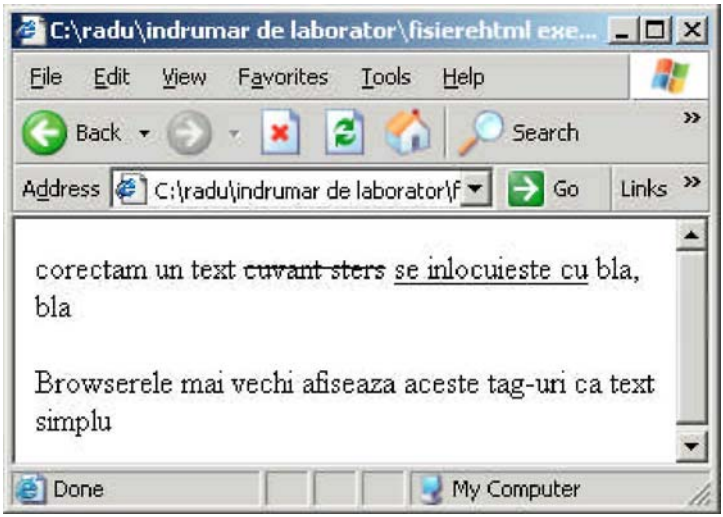
4. Acronime si abrevieri

```
<html> <body>
<abbr title="Uniunea Europeana">EU</abbr>
<br>
<acronym title="World Wide Web">WWW</acronym>
<p>La tag-ul de acronim atributul title
afiseaza un hint cand userul trece cu mouse-
ul peste acronim in text</p> <p>functioneaza
doar de la IE 5 </p> </body> </html>
```



5. Text inserat şi text şters

```
<html> <body> <p>
corectam un text <del>cuvant sters</del>
<ins>se inlocuieste cu</ins> bla, bla </p>
<p>
Browserele mai vechi afiseaza aceste tag-uri
ca text simplu </p> </body> </html>
```


C:\radu\indrumar de laborator\fisierehtml exe...
File Edit View Favorites Tools Help
Back
Search
Address C:\radu\indrumar de laborator\f
Go
Links
corectam un text cuvant sters se inlocuieste cu bla, bla
Browserele mai vechi afiseaza aceste tag-uri ca text simplu
Done
My Computer

3 Hiperlink

Ancora

Tag	Descriere
<a>	Defineşte o ancora (legătură)

Pentru a crea o legătură spre un alt document WEB, se utilizează tag-ul **<a>** (ancora). O ancora poate indica spre orice resursă din web: o pagina web, o imagine, un fişier de sunet, un film etc.

Tag-ul **<a>** poate primi atribute.

Atributul href

În cazul acesta ancora va crea o legătură spre un alt document.

Sintaxa pentru crearea unei legături:

```
<a href="url">textul meu care se va afisa si care
este subliniat albastru</a>
```

T
ag-ul <a> primeşte ca atribut pe **href** care ia ca valoare adresa resursei spre care ancora face legătura. Cuvintele scrise între cele doua tag-uri <a> ..... /a> vor fi afişate de către browser ca hiperlink.

Exemplu:

```
<a href="http://www.csac.ulbsibiu.ro/">Situl
Catedrei de Calculatoare si Automatizari</a>
```

Atributul target

Acest atribut se foloseşte pentru a defini „locul” unde se va deschide documentul spre care se face legătura. Daca acest atribut nu se specifică atunci pagina se va deschide în fereastra browserului în locul acelei care a făcut legătura.
Pentru deschiderea documentului spre care se face legătura într-o nouă fereastră se va folosi atributul target cu valoarea **_blank**

Exemplu

```
<a href=http://www.csac.ulbsibiu.ro/
target=”_blank”>Situl Catedrei de
Calculatoare si Automatizari</a>
```

Atributul name

Atributul name este utilizat pentru a crea o ancora cu nume. Aceste ancore sunt foarte des utilizate în cadrul documentelor mari. Pentru a ajunge la un anumit punct din document acela se marchează ca fiind o ancora cu nume. Având aceasta ancora din orice document se poate “sări” direct la secțiunea din document marcată de ea. Acest lucru uşurează enorm navigarea prin document.

Sintaxa pentru crearea unei ancore cu nume este:

```
<a name="nume eticheta">textul de afisat</a>
```

Exemplu

```
<a name="cap1">Capitolul 1</a>
```

Observație:
O ancora cu nume nu se va afişa de către browser într-un mod anume.
Pentru a face o legătură spre o ancoră cu nume se adaugă la sfârşitul numelui documentului spre care se face legătura semnul #

urmat de numele ancorei. (Daca suna prea complicat urmăriți exemplul):

Exemplu
Pas 1.

În documentul ex1.htm avem mai multe denumiri de capitole. Pentru a sări mai uşor la începutul fiecărui capitol definim câte o ancora cu nume:

```
<html>
<head>
<body>
<a name="cap1"> Capitolul 1</a>
<p> In capitolul 1 povestim basme si
intamplari basme si intamplari basme
si intamplari basme si intamplari basme si
intamplari basme si intamplari
basme si intamplari basme si intamplari basme
si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari
basme si intamplari basme si intamplari basme
si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari
basme si intamplari basme si intamplari basme
si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari
basme si intamplari basme si intamplari basme
si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari
basme si intamplari basme si intamplari basme
si intamplari basme si
intamplari ..... </p>
<a name="cap2"> Capitolul 2</a>
<p> In capitolul 2 povestim iarasi basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
```

```
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari basme si
intamplari basme si intamplari ..... </p>
</body> </html>
```

Pas 2.
Definim legătura direct spre o ancora cu nume. Presupunem ca avem un document web care conține cuprinsul...

```
<html>
<head>
</head>
<body>
<p> Cuprinsul</p>
<a href="ex1.htm#cap1"
target="_blank">Capitolul 1</a><br>
<a href="ex1.htm#cap2"
target="_blank">Capitolul 2</a><br>
</body>
</html>
```

Observație:
Daca se doreşte saltul în cadrul aceluiaşi document atunci la definirea legăturii se poate utiliza doar numele ancorei precedat de # fără a mai specifica numele fişierului.

Exemplu

```
<a href="#cap1" target="_blank">Capitolul
1</a><br>
```

Observație generală:

Întotdeauna este bine să adăugați la sfârşit un / când dați adresa spre un subfolder:

Ex corect:

```
href="www.csac.ulbsibiu.ro/orar/
```

Daca nu puneți si / la sfârşit atunci browserul va face doua cereri: prima în care va adăuga automat / iar a doua când va face cererea cu / adăugat.

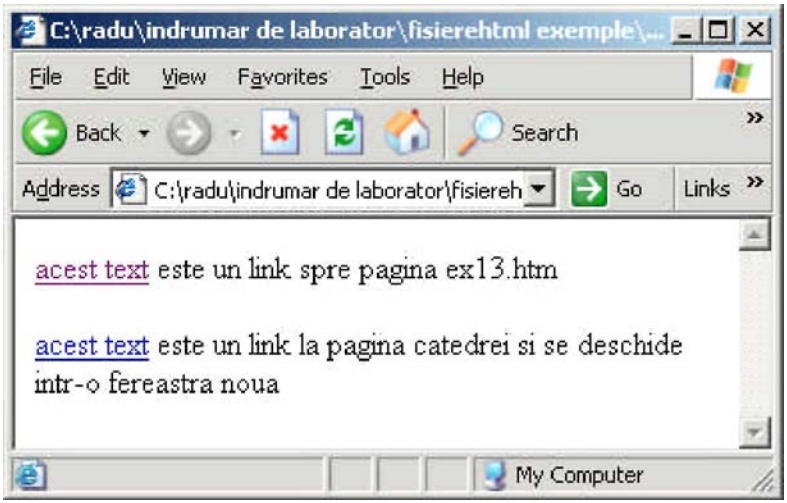
Ex. incorect:

```
href="www.csac.ulbsibiu.ro/orar
```

Exemple diverse

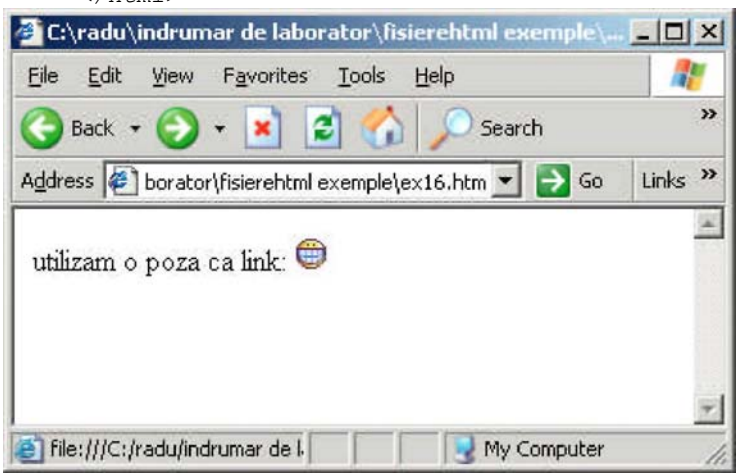
1. Crearea unei legături

```
<html> <body> <p>
<a href="ex13.htm">
acest text</a> este un link spre pagina
ex13.htm </p> <p>
<a href="http://www.csac.ulbsibiu.ro/"
target="_blank"> acest text</a> este un link
la pagina catedrei si se deschide intr-o
fereastra noua </p> </body> </html>
```



2. Crearea unei legături dintr-o poza

```
<html> <body> <p>
utilizam o poza ca link: <a href="ex15.htm">
<img border="0" src="04.gif"> </a>
</p>
</body>
</html>
```



Observaţie: Aici am folosit un tag <img> care îl vom parcurge mai târziu. Atributul src indică sursa pozei adică numele fişierului care trebuie să fie în acelaşi director cu fişierul html.

3. Un link spre o adresa de mail

```
<html> <body> <p>
un link catre o adresa de mail - daca aveti
un client de mail instalat la click pe acest
link se deschide fereastra clientului de
mail: <a
href="mailto:radu.kretzulescu@ulbsibiu.ro?sub
ject=salutare"> trimite-ma</a> </p> </body>
</html>
```

Observaţie: Se observă pe status bar linkul făcut!

4 -Tabele

Tag-uri specifice tabelelor

Tag	Descriere
<table>	Defineşte un tabel
<th>	Defineşte header-ul unui tabel
<tr>	Defineşte un rând (linie) în tabel
<td>	Defineşte o celula în tabel
<caption>	Defineşte caption-ul tabelului
<colgroup>	Defineşte un grup de coloane ale tabelului
<col>	Defineşte valorile atributelor a unei sau mai multor coloane dintr-un tabel
<thead>	Defineşte un rând de head
<tbody>	Defineşte corpul tabelului
<tfoot>	Defineşte footer-ul tabelului

Tabele

Tabelele se definesc cu ajutorul tag-ului <table>.
Tabelele se împart in linii care se definesc cu ajutorul tag-ului <tr> (table row).
Liniile tabelului se împart in celule cu ajutorul tag-ului <td> (table data)
O celulă dintr-un tabel poate conține: text, imagini, paragrafe, forme, tabele, liste etc.

Exemplu:

```
<table border="1">
<tr>
<td>rand 1, celula 1</td>
<td>rand 1, celula 2</td>
</tr>
<tr>
<td>rand 2, celula 1</td>
<td>rand 2, celula 2</td>
</tr>
</table>
```

Va fi afişat în browser astfel:

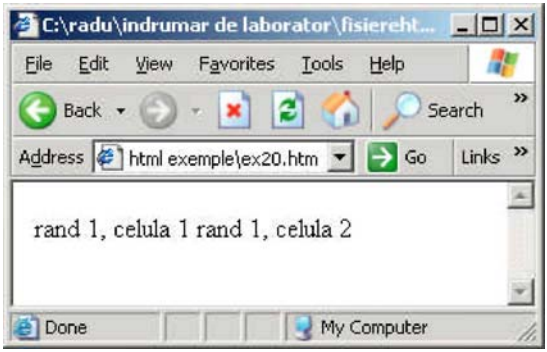
Atributul borders

Daca nu se specifica acest atribut tabelul va fi afişat fără linii. În unele cazuri acest lucru este bun deoarece cu ajutorul tabelelor se poate asigura “împărțirea” paginii web astfel încât informația să poată fi structurată.
Ex. În partea stânga se va afişa o celula care conține un meniu de navigare iar în dreapta conținutul .....

Pentru a afişa liniile tabelului atributul borders se utilizează astfel:

```
<table border="1"> <tr>
<td>rand 1, celula 1</td> <td>rand 1, celula
2</td> </tr> </table>
```

Acelaşi tabel dar fără atributul `border`

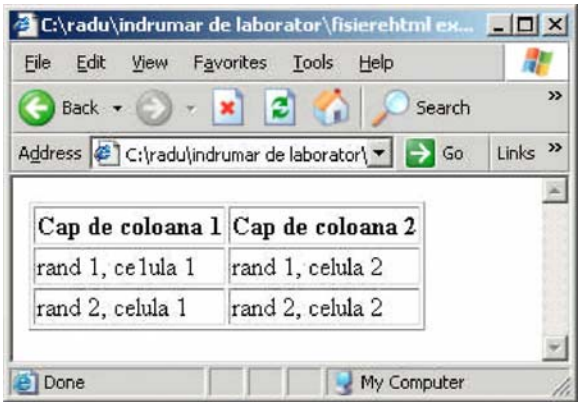


Capul de tabel

Capul de table sau heading se definesc într-un table cu tag-ul <th>.

```
<table border="1"> <tr>
<th>Cap de coloana 1</th> <th>Cap de coloana
2</th> </tr>
<tr>
<td>rand 1 <td>rand 1 </tr> <tr>
<td>rand 2 <td>rand 2
celula K/td> celula 2</td>
celula K/td> celula 2</td>
</tr> </table>
```

Arata în browser astfel:

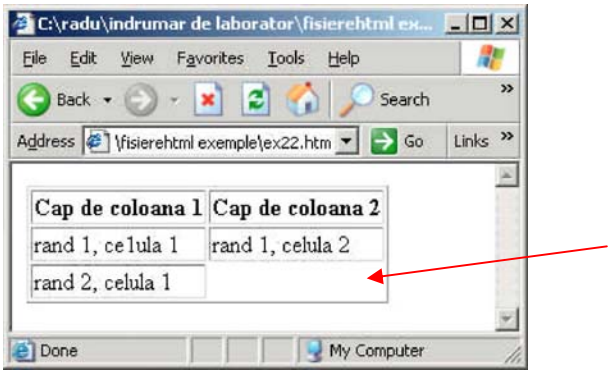


Observație: Capul de tabel apare bold fără a fi nevoie de editare specială!

Celule goale într-un tabel

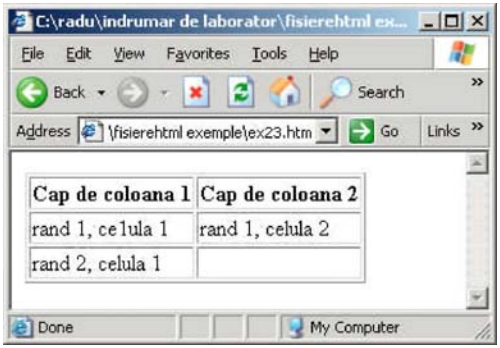
Tabelele care conțin celule goale nu sunt afişate corect de toate browser-ele. Problema care apare este lipsa chenarului în jurul celulei care nu conține nimic.

```
<html>
<body>
<table border="1">
<tr>
<th>Cap de coloana 1</th>
<th>Cap de coloana 2</th>
</tr>
<tr>
<td>rand 1, celula 1</td>
<td>rand 1, celula 2</td>
</tr>
<tr>
<td>rand 2, celula 1</td>
<td></td>
</tr>
</table>
</body>
</html>
```



Pentru a rezolva această problemă adăugăm un caracter special de spațiu ** **

```
<html> <body>
<table border="1"> <tr>
<th>Cap de coloana 1</th> <th>Cap de coloana
2</th> </tr> <tr>
<td>rand 1, celula 1</td> <td>rand 1, celula
2</td> </tr> <tr> <td>rand 2, celula 1</td>
<td> </td>
</tr>
</table>
</body>
</html>
```

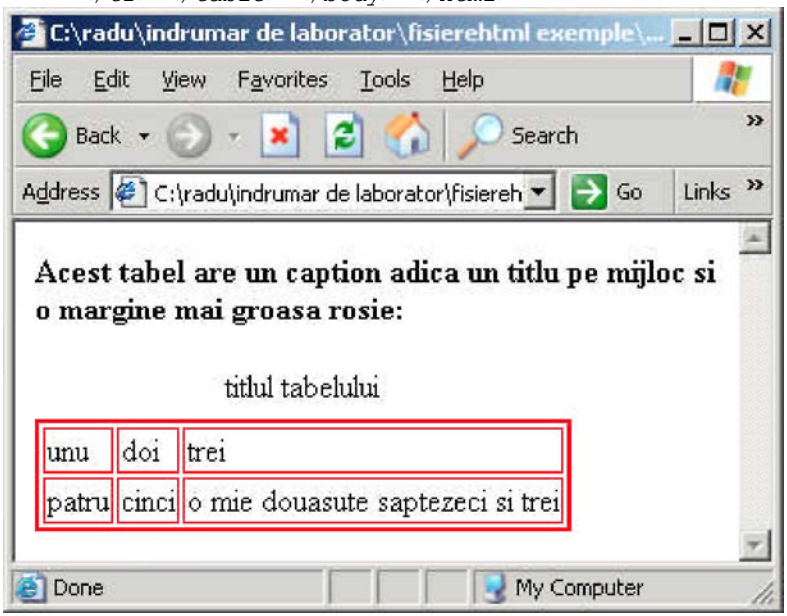


Observație: Alta viata!

Exemple diverse

1. Tabel cu caption (cu titlu)

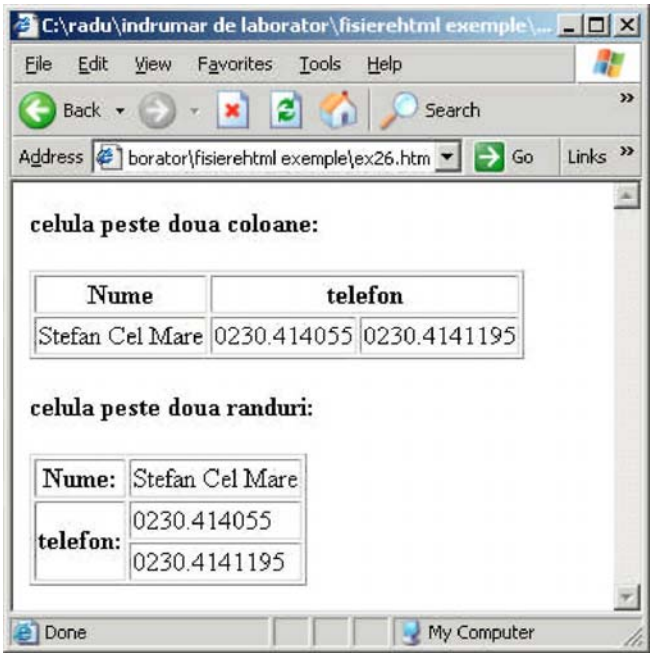
```
<html> <body> <h4>
Acest tabel are un caption adica un titlu pe
mijloc si o margine mai groasa rosie: </h4>
<table border="2" bordercolor="red">
<caption>titlul tabelului</caption> <tr>
<td>unu</td>
<td>doi</td>
<td>trei</td> </tr> <tr>
<td>patru</td>
<td>cinci</td>
<td>o mie douasute saptezeci si trei</td>
</tr> </table> </body> </html>
```



2. Celule care sunt unite peste 2 coloane/rânduri

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

```
<h4>celula peste doua coloane:</h4> <table
border="1"> <tr>
<th>Nume</th>
<th colspan="2">telefon</th> </tr> <tr>
<td>Stefan Cel Mare</td>
<td>0230.414055</td>
<td>230.4141195</td> </tr> </table>
<h4>celula peste doua randuri:</h4> <table
border="1"> <tr>
<th>Nume:</th>
<td>Stefan Cel Mare</td> </tr> <tr>
<th rowspan="2">telefon:</th>
<td>0230.414055</td> </tr> <tr>
<td>230.4141195</td> </tr> </table> </body>
</html>
```



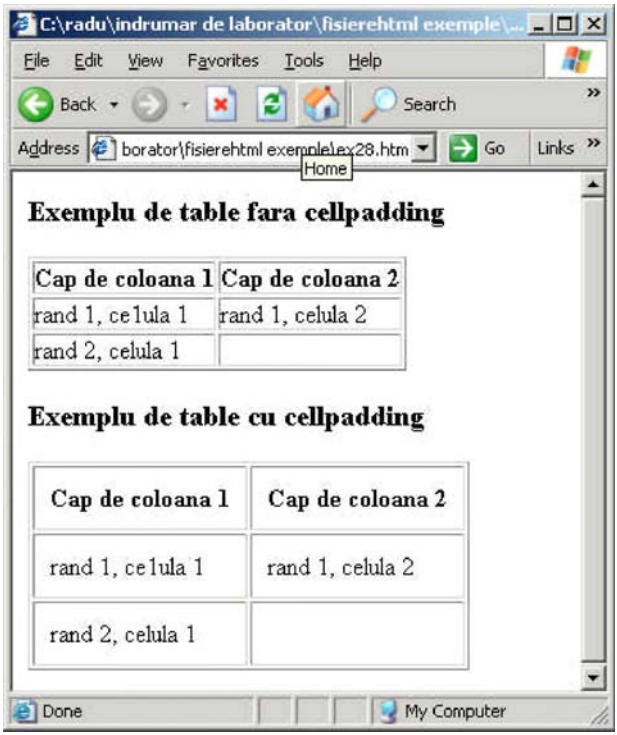
3. Alte tag-uri în cadrul celulelor:

```
<html> <body>
<table border="1"> <tr> <td> <p>un
```

```
paragraf</p> <p>un alt paragraf</p> </td>
<td>celula aceasta contine un tabel: <table
border="1" bordercolor="blue"> <tr>
<td>A</td> <td>B</td> </tr> <tr>
<td>C</td> <td>D</td> </tr> </table> </td>
</tr> <tr>
<td>celula contine o lista <ul> <li>unu</li>
<li>doi</li> <li>trei</li> </ul> </td>
<td>salut</td> </tr> </table> </body> </html>
```

4. Crearea cu ajutorul atributului de cellpadding de spațiu între text şi marginea celulei

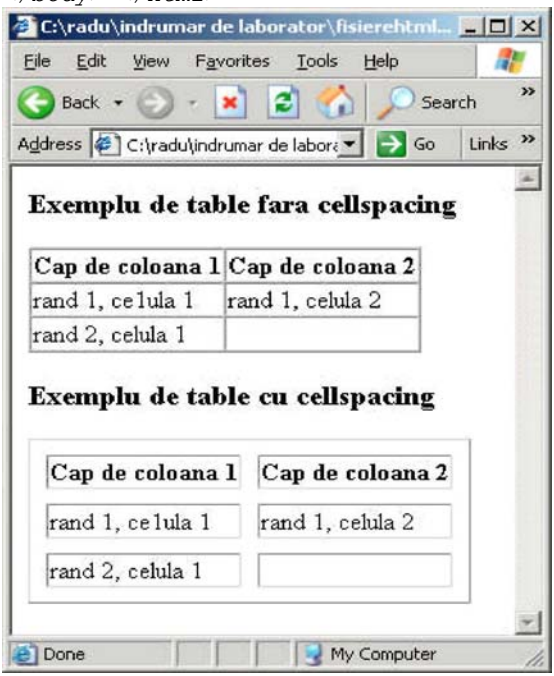
```
<html> <body>
<h3> Exemplu de table fara cellpadding</h3>
<table border="1" cellpadding="0">
<tr>
  <th>Cap de coloana 1</th>
  <th>Cap de coloana 2</th> </tr>
<tr>
  <td>rand 1, <td>rand 1, </tr>
<tr>
  <td>rand 2, <td> </td> </tr>
</table>
<h3> Exemplu de table cu cellpadding</h3>
<table border="1" cellpadding="10">
<tr>
  <th>Cap de coloana 1</th>
  <th>Cap de coloana 2</th>
</tr>
<tr>
  <td>rand 1, celula 1<td>
  <td>rand 1, celula 2</td>
</tr>
<tr>
 <td>rand 2, celula 1</td>
 <td> </td>
</tr>
</table>
</body>
</html>
```



5. Crearea cu ajutorul atributului de cellspacing de spațiu între celule

```
<html>
<body>
<h3> Exemplu de table fara cellspacing</h3>
<table border="1" cellspacing="0">
<tr>
<th>Cap de coloana 1</th>
<th>Cap de coloana 2</th>
</tr>
<tr>
<td>rand 1, celula 1</td>
<td>rand 1, celula 2</td>
</tr>
<tr>
<td>rand 2, celula 1</td>
```

```
<td> </td>
</tr>
</table>
<h3> Exemplu de table cu cellspacing</h3>
<table border="1" cellspacing="10">
<tr>
<th>Cap de coloana 1</th>
<th>Cap de coloana 2</th>
</tr>
<tr>
<td>rand 1, ce1ula 1</td>
<td>rand 1, celula 2</td>
</tr>
<tr>
<td>rand 2, celula 1</td>
<td> </td>
</tr>
</table>
</body> </html>
```



6. Adăugarea de culoare sau imagine pe fundalul tabelului cu ajutorul atributelor **bgcolor** si **background**.

```
<html>
<body>
<h4>culoare de fungdal pe intreg
tabelul:</h4>
<table border="1"
bgcolor="green">
<tr>
<td>pam</td>
<td>pim</td> </tr> <tr>
<td>pim</td>
<td>pam</td> </tr> </table>
<h4>o imagine pe fundalul tabelului:</h4>
<table border="1" background="04.gif"> <tr>
<td>pam</td>
<td>pim</td> </tr> <tr>
<td>pim</td>
<td>pam</td> </tr> </table> </body> </html>
```

6. Adăugare de culoare sau imagine pe fundalul unei celule cu ajutorul atributelor **bgcolor** si **background**

```
<html>
<body>
<h4>culoare de fungdalul unei celule:</h4>
<table border="1">
<tr>
<td bgcolor="green">pam</td>
<td>pim</td> </tr> <tr>
<td>pim</td>
<td>pam</td> </tr> </table>
<h4>o imagine pe fundalul unei celule:</h4>
<table border="1"> <tr>
<td>pam</td>
<td background="04.gif">pim</td> </tr> <tr>
<td>pim</td>
<td>pam</td> </tr> </table> </body> </html>
```

7. Alinierea conținutului unei celule cu ajutorul atributului **align**

```
<html>
<body>
<table border="1">
<tr>
<th align="left">pam</td>
<th>numere</td> </tr> <tr>
<td align="left">pim</td>
<td align="right">150.000 lei</td> </tr> <tr>
<td align="left">pam</td>
<td align="right">1.000 lei</td> </tr> <tr>
<td align="left">pum</td>
<td align="right">12.000 lei</td> </tr>
</table> </body> </html>
```

4 - Liste

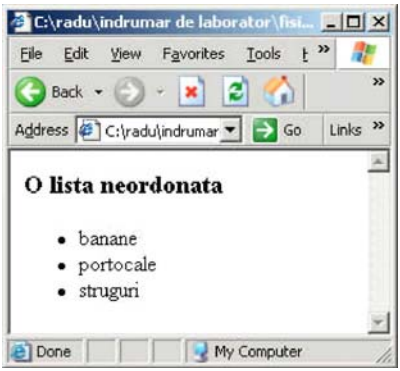
Tag-uri pentru liste

Tag	Descriere
	Defineşte o lista ordonata
	Defineşte o lista neordonata
	Defineşte un element al listei
<dl>	Defineşte o lista de definiții
<dt>	Defineşte un termen al listei de definiții
<dd>	Defineşte o descriere
<dir>	Nu se mai utilizează. Se foloseşte
<menu>	Nu se mai utilizează. Se foloseşte

Liste neordonate

O lista neordonată este o listă în care elementele listei sunt marcate cu o bulină la început - de obicei un cerc mic negru.
O lista neordonata începe cu tag-ul <ul>. Fiecare element al listei începe cu <li>.

```
<ul>
<li>banane</li>
<li>portocale</li>
<li>struguri</li>
</ul>
```

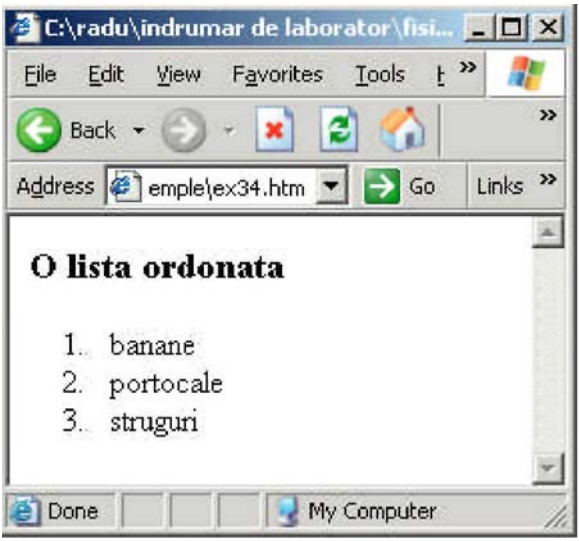


În interiorul fiecărui element al unei liste pot fi puse paragrafe, imagini, link-uri, alte liste etc.

Liste ordonate

O lista ordonata este o lista în care elementele listei sunt numerotate. O lista ordonata începe cu <ol> iar fiecare element cu tag-ul

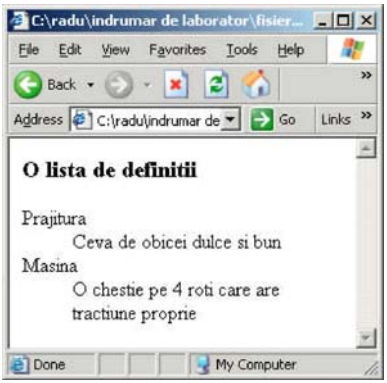
```
<ol>
<li>banane</li>
<li>portocale</li>
<li>struguri</li>
</ol>
```



Liste de definiții

Lista de definiții este o lista de termeni si explicarea acestora. O lista de definiții începe tag-ul <dl>. Fiecare termen din lista începe cu tag-ul <dt> . Fiecare definiție a unei liste de definiții începe cu tag-ul <dd>.

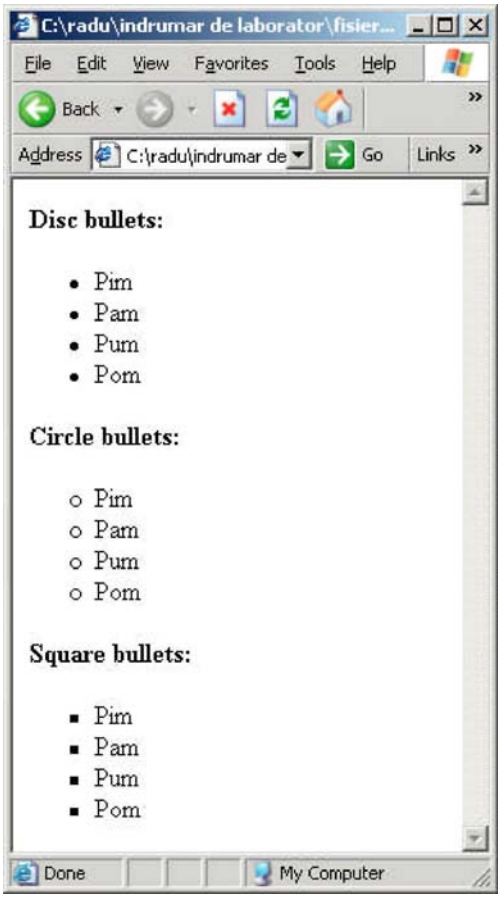
```
<dl>
<dt>Prajitura</dt>
<dd>Ceva de obicei dulce si bun</dd>
<dt>Masina</dt>
<dd>O chestie pe 4 roti care are tractiune
proprie</dd>
</dl>
```



Exemple

1. Alte tipuri de liste neordonate

```
<html> <body>
<h4>Disc bullets:</h4> <ul type="disc">
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ul>
<h4>Circle bullets:</h4> <ul type="circle">
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ul>
<h4>Square bullets:</h4> <ul type="square">
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ul>
</body> </html>
```



2. Alte tipuri de liste ordonate.

```
<html> <body> <h4>lista <ol>
numerotata:</h4>
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ol>
<h4>lista numerotata cu litere mari:</h4> <ol
type="A">
<li>Pim</li>
```

```
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ol>
<h4> lista numerotata cu litere mici:</h4>
<ol type="a">
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ol>
<h4> lista numerotata cu cifre romane
mari:</h4> <ol type="I">
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ol>
<h4> lista numerotata cu cifre romane
mici:</h4>
<ol type="i">
<li>Pim</li>
<li>Pam</li>
<li>Pum</li>
<li>Pom</li> </ol>
</body> </html>
```

5 - Imagini

Tag-uri pentru imagini

Tag	Descriere
<img>	Defineşte o imagine
<map>	Defineşte o harta a imaginii
<area>	Defineşte o zona în cadrul harții unei imagini

Pentru a defini o imagine în HTML avem nevoie de tag-ul <img>. Pentru a afişa o imagine trebuie să specificam valoarea atributului src (sursa). Sursa este chiar URL-ul imaginii pe care vrem sa o afişam. Sintaxa este:
`<img src="url">` Browserul pune întotdeauna imaginea în locul în care întâlneşte tag-ul <img>.

Exemplu:

```
<html>
<body>
<p> pam pam</p>
<img src="04.gif">
</body>
</html>
```

Atributul alt

Acest atribut se foloseşte pentru a afişa un text alternativ pentru o imagine. Adică în cazul în care browserul nu poate afişa imaginea atunci în spațiul rezervat imaginii apare textul specificat de atributul alt.

Foarte important!!!

Fără acest atribut validatorul de cod html nu va valida documentul (http://validator.w3c.org)

Sintaxa:

`<img src="04.gif" alt`="Smiley"> Utilizarea acestui atribut este o practică buna mai ales pentru cei ce navighează fără să-şi afişeze si pozele.

```
<html>
<body>
<p> pam pam</p>
<img src="05.gif" alt="Smiley">
</body>
</html>
```

Exemple

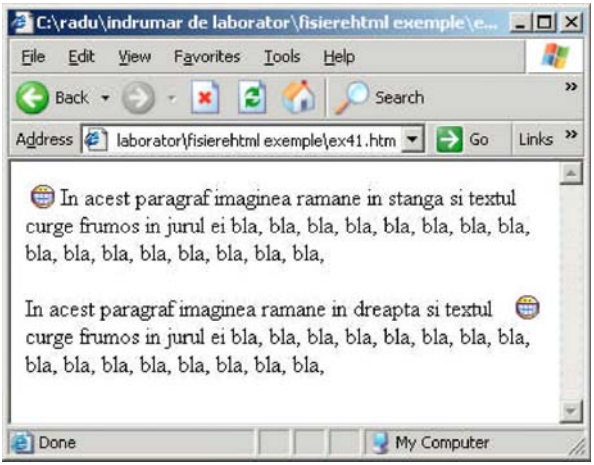
1. Alinierea imaginii față de text

```
<html> <body> <p>
imagine
<img src="04.gif" align="bottom"> in text
aliniata jos </p> <p>
imagine
<img src="04.gif" align="middle"> in text
aliniata la mijloc </p> <p>
imagine
<img src="04.gif" align="top"> in text
aliniata sus </p> <p>
<img src ="04.gif"> inaintea textului </p>
<p>dupa text <img src ="04.gif"> </p> </body>
</html>
```

2. Alinierea imaginii în cadrul unui paragraf:

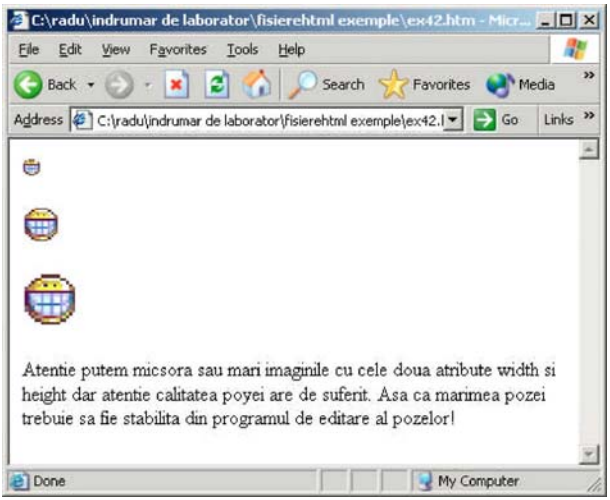
```
<html> <body> <p>
<img src ="04.gif" align ="left">
```

```
In acest paragraf imaginea ramane in stanga
si textul curge frumos in jurul ei bla, bla,
bla, bla, bla, bla, bla,bla,
bla,bla,bla,bla,bla,bla,bla,bla, </p> <p>
<img src ="04.gif" align ="right">
In acest paragraf imaginea ramane in dreapta
si textul curge frumos in jurul ei
bla,bla,bla,bla,bla,bla,bla,bla,
bla,bla,bla,bla,bla,bla,bla,bla, </p> </body>
</html>
```



3. Mărimea pozelor:

```
<html> <body> <p>
<img src="04.gif" width="15" height="15">
</p> <p>
<img src="04.gif" width="30" height="30">
</p> <p>
<img src="04.gif" width="45" height="45">
</p> <p>
Atentie putem micsora sau mari imaginile cu
cele doua atribute width si height dar
atentie calitatea pozei are de suferit. Asa
ca marimea pozei trebuie sa fie stabilita din
programul de editare al pozelor! </p> </body>
</html>
```


C:\radu\indrumar de laborator\fisierehtml exemple\ex42.htm - Micr...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\radu\indrumar de laborator\fisierehtml exemple\ex42.l Go Links
Atentie putem micsora sau mari imaginile cu cele doua atribute width si height dar atentie calitatea poyei are de suferit. Asa ca marimea pozei trebuie sa fie stabilita din programul de editare al pozelor!
Done
My Computer

6 - Culori HTML

Culorile se obțin utilizând modelul RGB (RED, GREEN, BLUE)

Valorile culorilor

Culorile sunt definite cu ajutorul notației hexazecimale pentru fiecare dintre cele trei culori de baza: roşu, verde, albastru. Cea mai mica valoare care se poate da unei culori este #00 iar cea mai mare este #FF Tabelul de mai jos arata unele combinații de culori
Culoare

Numele culorilor

Doar 16 **nume de culori** sunt recunoscute de către standardul HTML 4.0. Acestea sunt: *aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white şi yellow*. Pentru toate celelalte nuanțe se impune folosirea codului hexa.

	Color HEX	Color RGB
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

Culoare	Color HEX	Nume
	#F0F8FF	AliceBlue
	#FAEBD7	AntiqueWhite
	#7FFFD4	Aquamarine
	#000000	Black
	#0000FF	Blue
	#8A2BE2	BlueViolet
	#A52A2A	Brown

000000	000033	000066	000099	0000CC	0000FF
003300	003333	003366	003399	0033CC	0033FF
006600	006633	006666	006699	0066CC	0066FF
009900	009933	009966	009999	0099CC	0099FF
00CC00	00CC33	00CC66	00CC99	00CCCC	00CCFF
00FF00	00FF33	00FF66	00FF99	00FFCC	00FFFF
330000	330033	330066	330099	3300CC	3300FF
333300	333333	333366	333399	3333CC	3333FF
336600	336633	336666	336699	3366CC	3366FF
339900	339933	339966	339999	3399CC	3399FF
33CC00	33CC33	33CC66	33CC99	33CCCC	33CCFF
33FF00	33FF33	33FF66	33FF99	33FFCC	33FFFF
660000	660033	660066	660099	6600CC	6600FF

Daca am dori să folosim toate nuanțele o să ajungem la 16 milioane de culori (256x256x256)

663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF
669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF
FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

7 – Poziționarea conținutului unei pagini html

7.1 Frame-uri

Taguri

Tag-ul	**Descrierea**
<frameset>	Defineşte un set de frame-uri

<frame>	Defineşte un frame (cadre)
<noframes>	Defineşte o secțiune în care se pune cod pentru browserele care nu suporta tag-ul de frame
<iframe>	Defineşte o fereastra interioara paginii

Cu ajutorul frame-urilor se pot afişa în aceeaşi fereastră a browserului mai multe fişiere htm. Fiecare fişier este independent de celelalte. Probleme care apar sunt:

- Browserele vechi nu suporta acest tag
- Tipărirea întregii ferestre fiind compusa din mai multe cadre este mai complicata
- Motoarele de căutare returnează doar urn-ul unui singur frame ceea ce duce in multe situații la “pierderea” sistemului de navigare.

Pentru a împărți o fereastra a browser-ului in mai multe frame-uri avem nevoie
de 2 tag-uri:
<frameset> care descrie modul cum se împarte fereastra
<frame> care defineşte ce fişier se pune intr-un cadru
De obicei frame-urile se utilizează la crearea sistemului de navigare care sa fie “vizibil” tot timpul. Astfel într-un frame se încărca fişierul care conține doar sistemul de navigare iar in celalalt(e) conținutul fişierelor spre care indica sistemul de navigare.

Avantaje:
Nu se mai reîncărca la fiecare click pe un link tot sistemul de navigare.

Tagul frameset
Tag-ul **<frameset>** defineşte modul in câte linii **sau** coloane se împarte fereastra browserului şi cât de mari să fie acestea în comparație cu fereastra browserului. Atenție nu se pot defini simultan linii si coloane!

Exemplu:

```
<frameset cols="25%,75%">
```

```
… </frameset>
```

Primul tag indica faptul ca fereastra browserului nostru se împarte în două coloane prima având o lățime de 25% din lățimea ferestrei iar a doua de 75%.

Atributele tag-ului frameset

Atribut	Valoare	Descriere
cols	pixels % *	Defineşte numărul şi mărimea coloanelor
rows	pixels % *	Defineşte numărul şi mărimea liniilor

Tagul Frame

Tag-ul <frame> defineşte ce fişier HTML se va pune in fiecare cadru descris în <frameset>. În exemplul de mai sus am definit două coloane. Pentru a defini care fişiere să se afişeze în aceste coloane se va proceda astfel: Presupunem că avem două fişiere cu numele **1.htm** si **2.htm.** Pentru a afişa fişierul 1.htm pe prima coloana şi pe 2.htm pe a doua coloană completăm codul prezentat în exemplul de mai sus astfel:

```
<frameset cols="25%,75%">
<frame src="1.htm">
<frame src="2.htm"> </frameset>
```

Atributele tag-ului frame

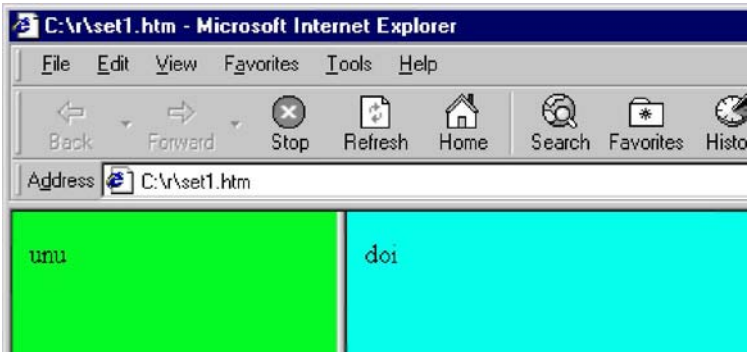
Atribut	Valoare	Descriere
frameborder	0 1	Specifica faptul ca se afişează sau nu marginea frame-ului
marginheight	pixels	Defineşte marginile de sus si jos in cadrul frame-ului
marginwidth	pixels	Defineşte marginile din ştanga şi dreapta in cadrul frame-ului
name	*frame_name*	Defineşte un nume unic atribuit frame-ului (se foloseşte în script-uri)

noresize	noresize	Când ia valoarea noresize atunci utilizatorul nu poate modifica lățimea coloanei sau liniei
scrolling	yes no auto	Determina afişarea unui scrollbar
src	URL	Defineşte URL-ul documentului de afişat

Exemple diverse

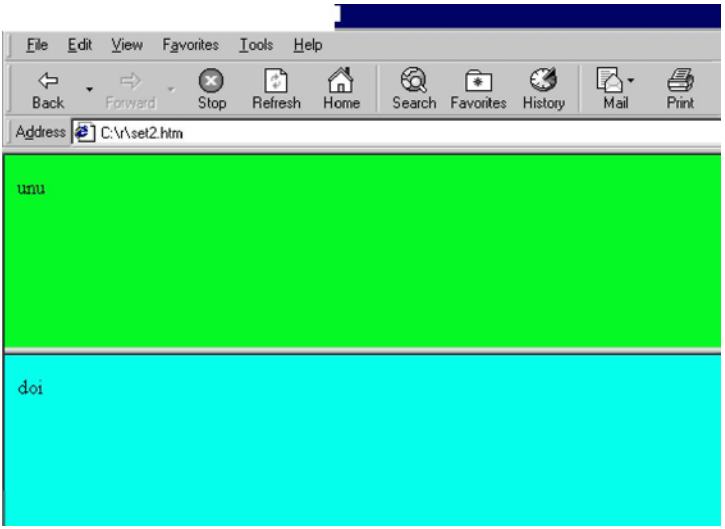
1. Frame-uri verticale

```
<html>
<head>
</head>
<frameset cols="25%,75%"> <frame src="1.htm">
<frame src="2.htm">
</frameset>
</html>
```



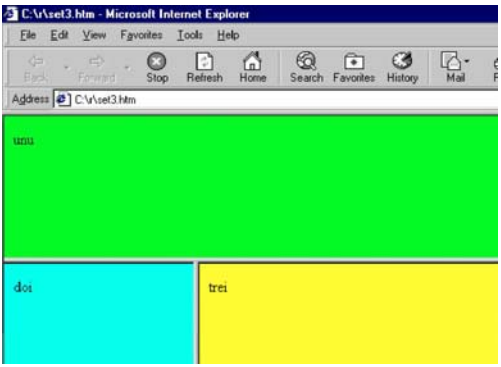
2. Frame-uri orizontale

```
<html>
<head>
</head>
<frameset rows="25%,75%"> <frame src="1.htm">
<frame src="2.htm">
</frameset>
</html>
```



3. Frame-uri mixte:

```
<html>
<head>
</head>
<frameset rows="25%,75%"> <frame src="1.htm">
<frameset cols="200, *"> <frame src="2.htm">
<frame src="3.htm"> </frameset>
</frameset>
</html>
```



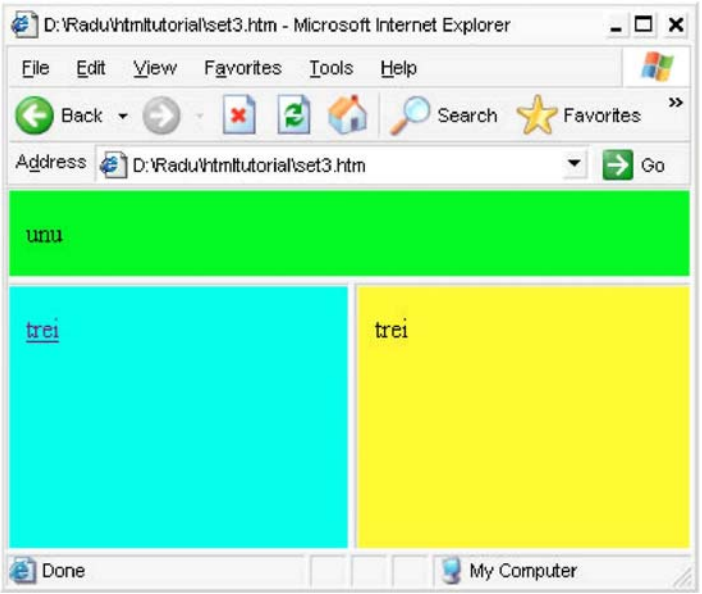
Observație: la toate exmplele de mai sus utilizatorul poate modifica lățimea frame-urilor. Pentru a bloca acest lucru utilizam atributul de **noresize.**

```
<html>
<head>
</head>
<frameset rows="25%,75%">
<frame noresize="noresize" src="1.htm">
<frameset cols="200, *">
<frame noresize="noresize" src="2.htm">
<frame noresize="noresize" src="3.htm">
</frameset> </frameset> </html>
```

4. Utilizarea atributului **name**
Acest atribut se foloseşte pentru a identifica unic un frame. Se foloseşte in cazurile în care într-un frameset avem un frame care conține şi stemul de navigare iar afişajul se face în alt frame. În acest caz frame-ul în care se face afişarea trebuie identificat cu ajutorul atributului **name.**

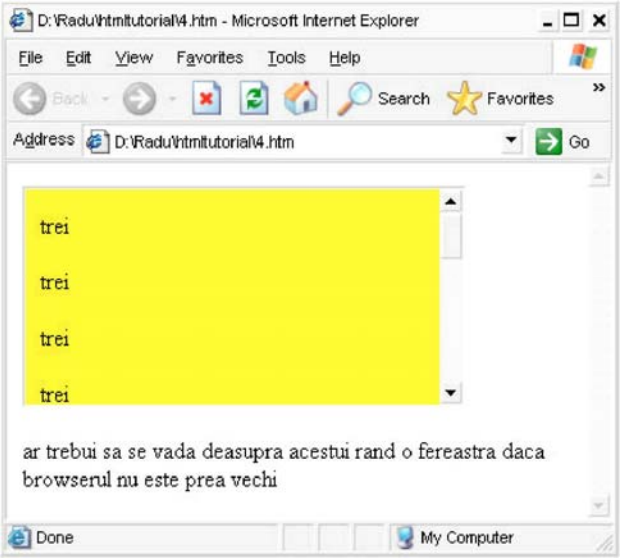
```
<html>
<head>
</head>
<frameset rows="25%,75%">
<frame noresize="noresize" src="1.htm">
<frameset cols="200, *">
<frame noresize="noresize" src="2.htm">
<frame name="principal" noresize="noresize"
src="3.htm">
</frameset> </frameset> </html>
Prezentam mai jos codul modificat al
fisierului 2.htm
<html>
<head>
</head>
<body bgcolor="00FFFF">
<a href="3.htm" target="principal"> doi </a>
</body>
</html>
```

Observăm faptul că la tag-ul **<a>** am adăugat atributul ***target*** care a primit valoarea atributului name din frame-ul 3 declarat în frameset.



5. Inline frame

```
<html>
<body>
<iframe src="3.htm"></iframe>
<p>ar trebui sa se vada deasupra acestui rand
o fereastra daca
browserul nu este prea vechi</p>
</body>
</html>
```



O altă variantă e a aranja conținutul unei pagini html este utilizarea tabelelor sau a diviziunilor (tag-ul div).
Practic aceste sunt cele mai bune variante deoarece motoarele de căutare vor returna întotdeauna şi meniul de navigare al sitului.
Lucrul cu tabele este puțin mai greoi față de cel cu diviziuni. La utilizarea diviziunilor se impune aranjarea în pagină a acestora cu ajutorul css-ului

8 – CSS

Tag-urile de style

Tag	Descriere
<style>	Defineşte o definire de stil
<link>	Defineşte o resursă a unei referințe
	Defineşte o secțiune a unui document
	Defineşte o secțiune în document
<font>	Depreciat. Se foloseşte styles
<basefont>	Depreciat. Se foloseşte styles
	Depreciat. Se foloseşte styles

În capitolele anterioare am menționat faptul că în standardul HTML 4 se recomandă iar în standardul XHTML se impune formatarea textului unui document html în afara sa cu ajutorul unei “foi de stil”. Declararea unui stil se poate face în trei moduri:

- în foaie externă documentului HTML
- în foaie internă
- în cadrul instrucțiunii html

Foaia de stil externă

utilizarea unei foi de stil externe (un fişier css) este utilă atunci când se doreşte modificare formatării unui sit întreg. Practic orice modificare a unui stil în fişierul css duce la modificarea formatării paginilor sit-ului cu condiția ca paginile sit-ului să fie legate de această foaie de stil. Fiecare pagină din sit se va lega de foaia de stil prin tag-ul <link> . Acest tag trebuie specificat în zona de head a fiecărei pagini care va folosi stilurile declarate în foaia de stiluri.

Exemplu:

```
<head>
<link rel="stylesheet" type="text/css"
href="silul_meu.css">
</head>
```

Foaia de stil internă

Acest tip de foaie de stil se foloseşte atunci când stilul de formatare se va utiliza doar în cadrul unui singur document HTML. Ca şi în cazul foii de stil externe definirea unei foi de stil interne se face în secțiunea de head cu ajutorul tag-ului <style>

Exemplu

```
<html>
<head>
<style type="text/css">
body {background-color: #ff0000}
h1 {font-face: Verdana}
p {margin-left: 30px}
</style>
</head>
<body>
</body>
</html>
```

Declararea stilului în cadrul instrucțiunii html

Acest tip de declarare se foloseşte doar în cazurile în care se doreşte aplicarea stilului doar unei singure instrucțiuni html. Atributul style poate conține în acest caz orice valoare a unei proprietăți CSS.

```
<p style="color: red">
Propozitia mea
</p>
```

În continuare vom prezenta principalele elemente ale CSS

Ce este CSS?

- **CSS** este prescurtarea de la **C**ascading **S**tyle **S**heets (foi de stiluri cascadate)
- Stilurile definesc modul de afişare a elementelor HTML
- Stilurile sunt conținute în foi de stiluri - **Style Sheets**
- Stilurile au fost adăugate în HTML pentru a rezolva o problemă şi anume separarea formei de conținut
- Foile de stil externe sunt salvate în fişiere **CSS**

Definirea multiplă de stiluri se va **cascada** în una singură.
Problema pe care o rezolvă utilizarea stilurilor de editare este faptul că se separă conținutul unui document HTML, marcat prin diverse tag-uri care îl împart în paragrafe, tabele, titluri etc., de forma sa. Pentru motoarele de căutare viitoare şi chiar pentru web-ul semantic această separare este esențială. Astfel deja în standardul XHTML nu se mai acceptă tag-uri de formatare a textului cum ar fi <font> cu atributele sale de size, color,....
Problema formatării „externe" a unui document HTML a fost rezolvată cu ajutorul CSS-urilor care sunt recunoscute deja de către Netscape Navigator 4 şi IE 4.
CSS-urile salvează foarte multă muncă. Dacă doriți să modificați modul cum arată un sit întreg este de ajuns să se modifice fişierul CSS. Ar fi foarte incomod să modificați de sute de ori tag-ul <font>

Ordinea de cascadare a stilurilor.
La începutul acestui capitol am prezentat mai multe modalități de a declara un stil: în cadrul unui tag, în partea de <head> a documentului, într-un fişer extern CSS sau chiar în mai multe fişiere externe. Toate aceste stiluri se vor cascada într-unul nou, "virtual" după următoarea regulă - unde 1 are cea mai mică prioritate iar 4 cea mai mare:

1. Setări default ale browser-ului
2. Foaie de stil externă - fişier CSS
3. Foaie de stil internă - declarare în interiorul tag-ului <head>
4. Declarare în interiorul elementului HTML

Practic o declarare de stil în cadrul unui element HTML va suprascrie orice stil declarat în partea de <head> şi orice stil declarat într-un

fişier extern CSS.
În continuare ne vom ocupa de sintaxa declarării unui stil şi de tipurile de stiluri

Sintaxa CSS

Sintaxa CSS este alcătuită din 3 părți:
un selector
o proprietate
o valoare atribuită proprietății

Mod de redactare:

```
selector {proprietate: valoare}
```

Selectorul este de obicei chiar elementul/tag-ul HTML pe care dorim să-l definim
Proprietatea este atributul elementului HTML pe care dorim să-l modificăm
Valoarea proprietății este chiar valoarea pe care dorim să o atribuim atributului elementului HTML
Trebuie să reținem faptul că în sintaxă proprietatea este separată de valoare prin “două puncte”. Dacă o valoare este compusă din mai multe cuvinte atunci în cadrul sintaxei acestea se pun între ghilimele. Dacă se doreşte stabilirea mai multor proprietăți pentru un singur selector acestea se vor separa cu “punct şi virgulă”
Exemple:

Ex1

```
body {color: red}
```

Ex2

```
p {font-family: "sans serif"}
```

Ex3

```
p{
font-family: verdana;
color: green;
text-align: right }
```

Exemplul 1 stabileşte pentru tag-ul <body> atributul color cu valoarea red adică textul delimitat de acest tag se va scrie cu culoarea roşie

Exemplul 2 stabileşte pentru tag-ul de paragraf <p> atributul font-family cu valoarea “sans serif” adică textul delimitat de acest tag va fi scris utilizând fontul “sans serfi”
Exemplul 3 stabileşte mai multe proprietăți pentru tag-ul <p>. Observăm că acestea sunt separate de „;“

Există 3 tipuri de selectori:

- selector html
- selector class
- selector ID

HTMLSelector {Property:Value;}

```
<style type="text/css">
p {font-family:arial; font-size:14px;
color:red}
</style>
```

.

ClassSelector {Property:Value;}

```
<style type="text/css">
.headline {font-family:arial; font-size:14px;
color:red}
</style>
```

#IDSelector {Property:Value;}

```
<style type="text/css">
#layer1 {position:absolute; left:100;top:100;
z-Index:0}
#layer2 {position:absolute; left:140;top:140;
z-Index:1}
</style>
```

Gruparea selectorilor

Selectorii pot fi grupați. Separarea lor se face cu virgulă.
Exemplu

```
h1,h2,h3
{
color: blue
}
```

Acest stil stabileşte faptul că textul delimitat de cele 3 tag-uri <h1>,

<h2> şi <h3> se vor scrie cu culoare albstră.

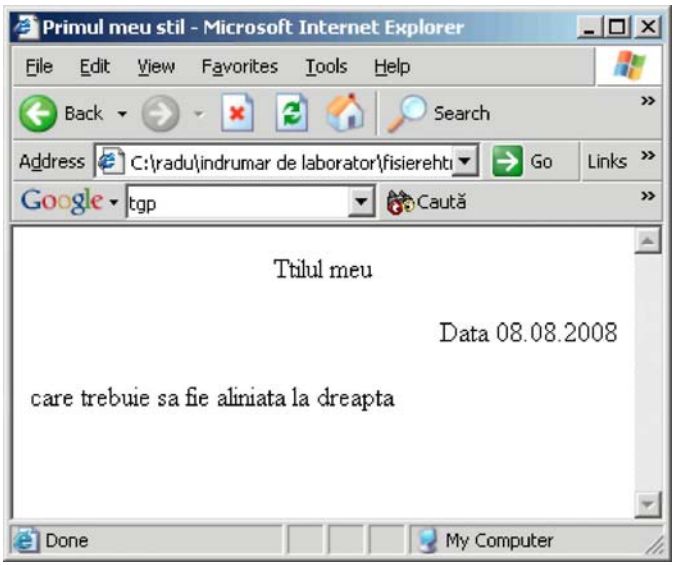
Selectorul de tip clasă

Cu acest tip de selector pot fi stabilite stiluri diferite pentru acelaşi tip de tag

Exemplu:
Dacă am dori să avem două tipuri de paragrafe: unul aliniat la centru iar celălalt aliniat la dreapta am proceda astfel:
Pas 1
Ne declarăm ambele stiluri cu ajutorul unui selector de tip clasă astfel:

```
p.dreapta {text-align: right}
p.centrat {text-align: center}
```

Aceste clase (le numim astfel în continuare) vor trebui aplicate ca atribute paragrafelor.
Afişarea ar fi astfel:



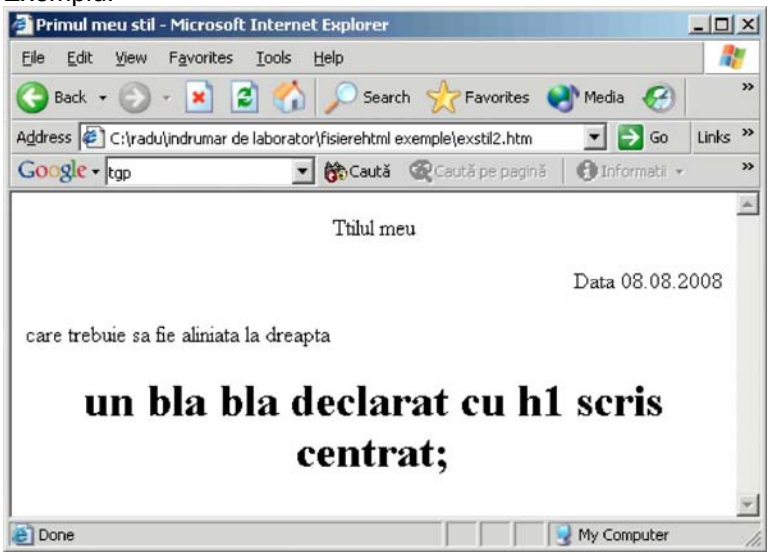
Codul sursă ar arăta astfel:

```
<html> <head>
<title>Primul meu stil</title>
<style type="text/css">
p.centrat {text-align:center} p.dreapta{text-
align:right}
</style> </head>
<body>
<p class="centrat">Ttilul meu</p>
<p class="dreapta">Data 08.08.2008 </p>
<p>care trebuie sa fie aliniata la dreapta
</p>
</body>
</html>
```

Ceea ce este foarte important este faptul că ulterior se mai pot face modificări iar acestea se vor face în toate paragrafele care au ca atribut clasa respectivă. ATENȚIE: Un element HTML poate primi un singur atribut class.

În declararea unui selector de clase se poate omite numele tag-ului, selectorul fiind acum unul general

Exemplu:



Codul sursă arată astfel:

```
<html> <head>
<title>Primul meu stil</title>
<style type="text/css">
.centrat {text-align:center}
p.dreapta{text-align:right}
</style> </head>
<body>
<p class="centrat">Titlul meu</p>
<p class="dreapta">Data 08.08.2008 </p>
<p>care trebuie sa fie aliniata la dreapta
</p>
<h1 class="centrat">un bla bla declarat cu h1
scris centraţ</h1>
</body>
</html>
```

Observaţie:
Nu s-a modificat modul de afişare a primului paragraf.

Proprietăţile CSS pentru background (fundal)

Aceste proprietăţi definesc modul de afişaj al fundalului unui element html.
În tabelul de mai jos în coloana a doua sunt trecute cu bold valorile explicite care pot fi atribuite proprietăţilor din prima coloană.

Proprietate	**Valoare**	**Scurtă descriere**
background	*background-color* *background-image* *background-repeat* *background-attachment* *background-position*	O proprietate cu care se pot seta toate valorile de background
background-attachment	**scroll fixed**	Stabile§te faptul că o poză de pe fundal rămâne fixă sau se mişcă odată conţinutul

background-color	*color-rgb color-hex color-nume* **transparent**	Setează culoarea de fundal a unui element HTML
background-image	*url* **none**	Setează o imagine pe fundalul elementului HTML
background-position	**top left top center top right center left center center center right bottom left bottom center bottom right** *x-% y-% x-pos y-pos*	Setează locul de unde începe poziționarea pozei de pe fundal
background-repeat	**repeat repeat-x repeat-y no-repeat**	Setează modul cum/dacă se repetă o imagine pe fundal

Exemple:

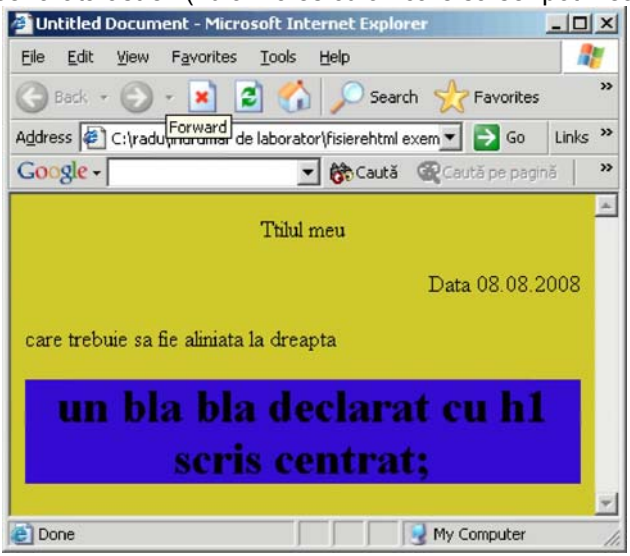
1. Stabilirea unor fundaluri…..

Codul sursă:

```
<html>
<head>
<title>Alt stil</title>
<style type="text/css">
body{background-color:#CCCC00}
.centrat {text-align:center}
p.dreapta{text-align:right}
h1{background-color:#3300FF}
</style> </head>
<body>
<p class="centrat">Ttilul meu</p>
<p class="dreapta">Data 08.08.2008 </p>
<p>care trebuie sa fie aliniata la dreapta
</p>
<h1 class="centrat">un bla bla declarat cu h1
```

```
scris
centrat</h1>
</body> </html>
```

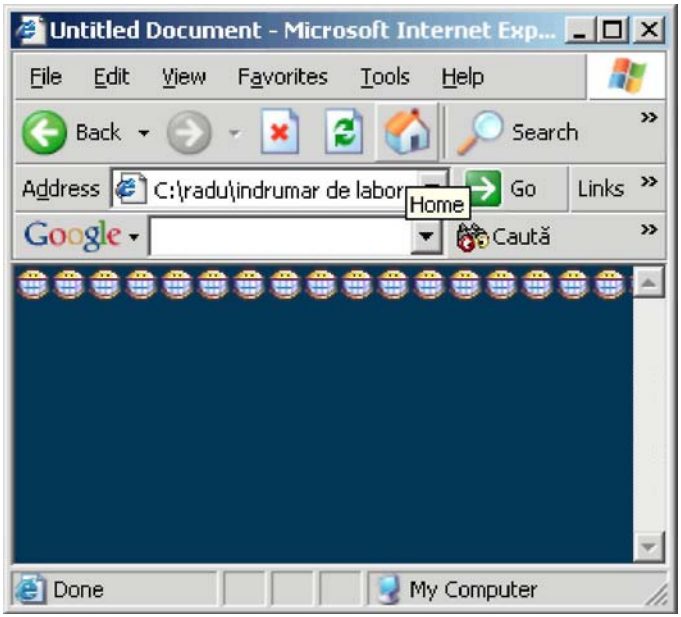
În browser arată astfel: (nu am ales culori care să se “potrivească”)



2. Imagine pe fundal şi repetarea ei doar pe axa x

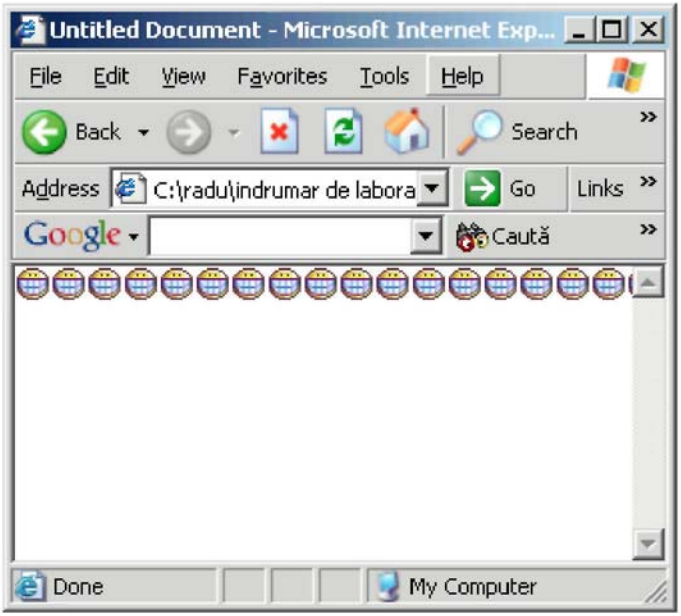
```
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
body {background-image:url(04.gif);
background-repeat:repeat-x} </style> </head>
<body>
</body> </html>
```

Arată astfel:



3. Toate proprietățile într-o singură declarație

```
<html> <head>
<title>Untitled Document</title> <style
type="text/css">
body {background:url(04.gif) #003366 repeat-x
}
</style> </head>
<body>
</body>
</html>
```



Proprietățile CSS pentru text

Cu aceste proprietăți se pot modifica: modul cum se afişează textul, distanța dintre caractere, dintre cuvinte etc.

Proprietate	**Valori**	**Descriere**
color	*color*	Setează culoarea unui text
direction	**ltr rtl**	Setează direcția de scriere ltr=stânga la dreapta rtl=dreapta la stânga
letter-spacing	**normal** *length*	Măreşte sau micşorează spațiul dintre litere

text-align	**left right center justify**	Aliniază textul dintr-un lelement html
text-decoration	**none underline overline line-through blink**	Adaugă „decorări” textului
text-indent	*Lungime px %*	Indentează prima linie dintr-un element HTML
text-transform	**none capitalize uppercase lowercase**	Controlează modul de afişare a literelor unui text: cu MAJUSCULE .....
word-spacing	**normal** *length*	Măre§te sau micşorează spațiul dintre cuvinte

Exemple:

1. Mărirea spațiului dintre litere: în pixeli şi centimetri

```
<html>
<head>
<title>CSS Text</title>
<style type="text/css">
hi {letter-spacing:10px}
h2{letter-spacing:1cm}
</style> </head>
<body>
<hl>Primul titlu</hl>
<h2>Al doilea titlu</h2>
</body>
</html>
```

Proprietățile CSS pentru fonturi

Proprietate	**Valori**	**Descriere**
font	*font-style* *font-variant* *font-weight* *font-size/line-height* *font-family* **caption** **icon** **menu** **message-box** **small-caption** **status-bar**	O proprietate cu care se pot defini într-o singură declarație toate proprietățile fontului
font-family	*family-name* *generic-family*	O listă de nume de fonturi care se vor utiliza dacă există pe maşina pe care rulează browserul în ordinea stabilită

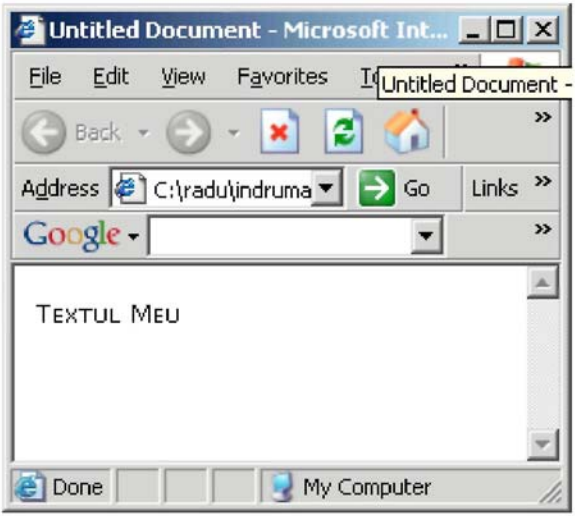
font-size	**xx-small** **x-small** **small** **medium** **large** **x-large** **xx-large** **smaller** **larger** *mărime %*	Setează dimensiunea fontului
font-stretch	**normal wider** **narrower** **ultra-condensed** **extra-condensed** **condensed** **semi-condensed** **semi-expanded** **expanded** **extra-expanded** **ultra-expanded**	Specifică modul de condensare a textului
font-style	**normal** **italic** **oblique**	Setează stilul de scriere
font-variant	**normal small-caps**	Afişează textul cu "CAPITALE MICI" sau normal.

font-weight	**normal** **bold** **bolder** **lighter** **100** **200** **300** **400** **500** **600** **700** **800** **900**	Setează grosimea fontului

Exemplu:

1. Setarea tipului de font, fel, mărime

```
<html>
<head>
<title>Untitled Document</title>
<style type="text/css">
p{font-family:Verdana, Arial, Helvetica,
sans-serif;
font-variant:small-caps;
font-size:small}
</style> </head>
<body>
<p> Textul Meu</p>
</body>
</html>
```



Pseudoclase

a:link
Defineşte stilul pentru link-uri nevizitate

a:visited
Defineşte stilul pentru link-uri vizitate

a:active
Defineşte stilul pentru link-uri active
Link-ul devine activ la click

a:hover
Defineşte stilul pentru link-uri “plutitoare”
Un link pluteşte când mouse-ul se află deasupra link-ului

Exemplu:

```
<style type="text/css">
a:link {text-decoration: none}
a:visited {text-decoration: none}
a:active {text-decoration: none}
a:hover {text-decoration: underline overline;
color: red;}
</style>
```

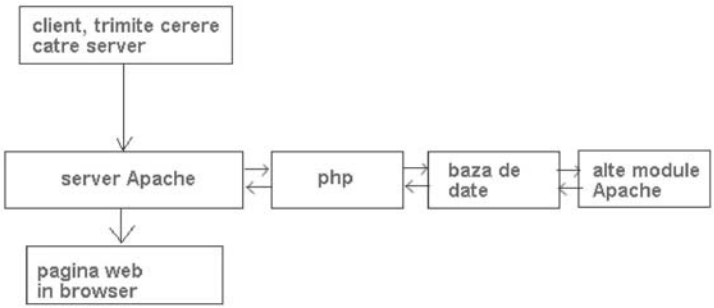
Cu ajutorul acestor pseudoclase se pot realiza foarte uşor meniuri de navigare astfel încât putem renunța la meniurile tip flash sau javascript care sunt ignorate de către motoarele de căutare.

9- PHP

Observație generală:

Scripturile php incluse în codul html sunt executate pe server. Asta are ca efect faptul că dacă încercați să vizualizați codul sursă unei pagini web dinamice o să observați că el conține doar tag-uri html, acestea fiind rezultatul obținut în urma interpretării codului php.

PHP (se pronunta pe-haş-pe) este un limbaj de programare ce ruleaza server, proiectat special pentru WEB. Intr-o pagina HTML puteti ingloba cod PHP care va fi executat la fiecare vizitare a paginii.
Codul dumneavoastră PHP este interpretat pe serverul WEB si generează un cod HTML care va fi văzut de utilizator (clientului (browserului) fiindu-i transmis numai cod interpretat ca si HTML).



PHP a fost conceput in anul 1994

Sintaxa php

Un fişier php conține tag-uri normale de html şi cod php. Marcarea începutului codului php se face cu **„<?php”** iar sfârşitul blocului de cod php cu **„?>”**

Exemplu:
Clasicul exemplu cu „Hellou lume” transformat în script php:

```
<html> <body>
<?php echo "Hellou lume"; ?>
</body>
</html>
```

Observații:
Fişierul de mai sus trebuie salvat cu extensia php pentru ca severul de web să transmită fişierul interpretorului php

Rezultatul va fi:

iar codul sursa returnat (vizibil de altfel cu View Source) este următorul:

```
<html> <body>
Hellou lume</body> </html>
```

Un bloc de script php poate fi inserat în orice loc a unui document html. Fiecare instrucțiune php trebuie separata de o alta instrucțiune cu **;**

Afişarea în php

1. Afişarea cu ajutorul lui “echo”
2. Elemente de formatare a codului sursa
3. Afişarea elementelor HTML
4. Escape-ul pentru anumite semne
5. Afişarea semnelor speciale din PHP

1. Afişarea cu ajutorul lui “echo“

Pentru a afişa ceva în PHP (adică codul html pentru o pagină) se pot utiliza mai multe funcţii: **echo**, **print** şi **printf**. Funcţia **printf()** se utilizează asemănător ca în C în sensul că permite formatarea afişărilor. Noi vom folosi funcţia **echo** care este puţin mai rapida ca **print**. Diferenţa dintre echo si print este ca print returnează întotdeauna valoarea 1 după ce a fost executata si deci poate fi utilizata în construcţii mai complexe.

Exemplu 1

```
<?php
echo "Acesta este textul care se va afişa";
?>
```

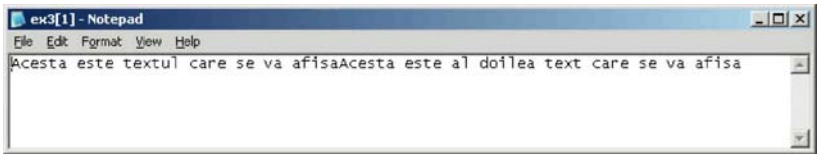
După cum se observă lipsesc parantezele specifice funcţiilor. Asta se întâmpla deoarece echo nu este o funcţie ci doar un constructor al limbajului PHP. În schimb dacă am folosi printf() atunci trebuie utilizate parantezele.

Exemplu 2

```
<?php
echo "Acesta este textul care se va afişa";
echo "Acesta este al doilea text care se va
afişa"; ?>
```

Daca salvam acest fişier şi îl salvăm pe server, obţinem următorul rezultat:

Codul sursă generat de php este:



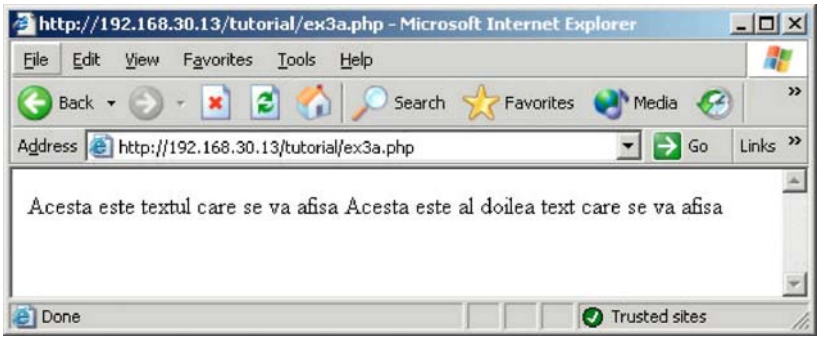
Observăm ca al doilea string a fost adăugat primului fără salt la rând nou cu toate ca am avut doua echo în codul php. Problema este ca după afişarea primului string cursorul imaginar al PHP-ului a rămas după primul string, al doilea echo fiind afişat începând cu acea poziție.
Pentru a avea un salt la linie noua trebuie sa utilizam elemente de formatare

2. Elemente de formatare
Pentru a forța salt la rând nou în php utilizam string-ul **\n**

Exemplul 3

```
<?php
echo "Acesta este textul care se va afişa\n";
echo "Acesta este al doilea text care se va
afişa"; ?>
```

Rezultatul este



Adică cam tot ca la exemplul anterior. Totuşi dacă vedem codul generat de php observăm modificarea.

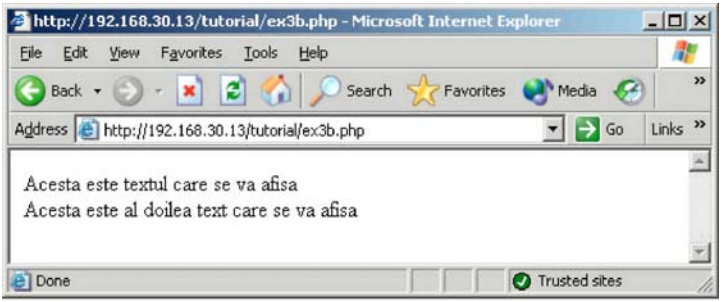
În principiu utilizăm elementele de formatare pentru lizibilitatea codului.
Totuşi pentru a putea afişa si în fereastra browser-ului cele două string-uri pe doua rânduri mai utilizam si elemente html.

3. Afişarea elementelor html cu ajutorul lui php
Pentru a forța un salt la rând nou utilizam elementul html

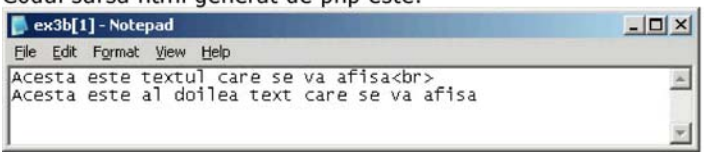
Exemplul 4

```
<?php
echo "Acesta este textul care se va
afişa<br>\n";
echo "Acesta este al doilea text care se va
afişa"; ?>
```

Rezultatul este:



Codul sursa html generat de php este:



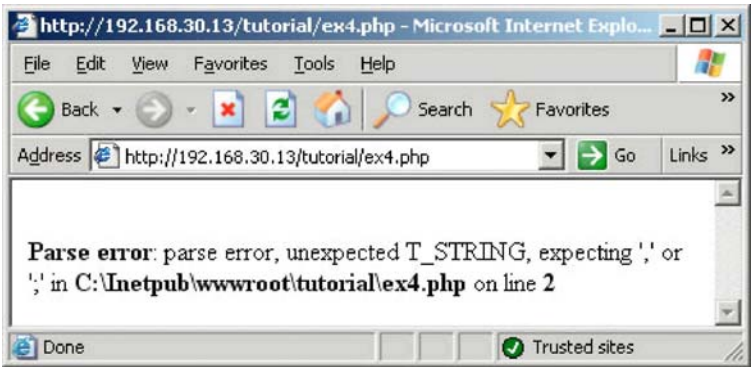
4. Escape-ul anumitor semne

Titlul este ales nefericit. Ce vrea sa spună este faptul că anumite semne cum sunt ghilimelele, slash-ul etc. sunt interpretate de către php. Adică dacă avem un text care ar conține ghilimele am avea probleme la afişarea sa.

Exemplu:

```
<?php
echo "Acesta este "textul" care se va
afişa<br>\n";
echo "Acesta este al doilea text\scris care
se va afişa"; ?>
```

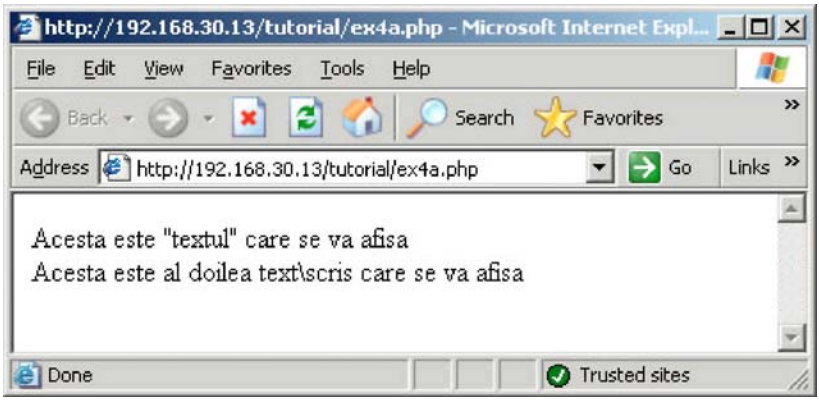
Dacă vom lăsa php-ul să interpreteze acest script vom obține un rezultat nesatisfăcător:



Linia a doua conține mai mult de 2 ghilimele şi acest lucru duce la eroarea afişata. Pentru a evita astfel de probleme trebuie să „marcam” ghilimelele care sunt necesare în text şi care nu fac parte din sintaxă. Acest lucru se face cu ajutorul escape-ului – se pune un \ înaintea ghilimelelor:

Codul ar trebui să arate astfel:

```
<?php
echo "Acesta este \"textul\" care se va
afişa<br>\n";
echo "Acesta este al doilea text\\scris care
se va afişa"; ?>
```



Observăm faptul că pentru a afişa un \ a trebuit să adăugam în codul sursa un al doilea \.

5. Afişarea semnelor speciale din PHP
Sunt cazuri în care am dori să afişăm semnul $. În PHP acesta este un semn rezervat pentru variabile în sensul că orice sir de caractere care începe cu $ este considerat nume de variabilă.
Pentru a rezolva această problemă utilizăm tot semnul de escape \

Exemplu:

```
<?php
echo "Semnul de dolar \$ este o unitate
monetara"; ?>
```

Variabile in PHP

1. Declararea variabilelor
2. Atribuirea de valori
2.1 Şiruri de caractere
2.1.1Operator de concatenare
2.2 Numere

1. Declararea variabilelor
Toate variabilele încep cu semnul de dolar $ urmat de _ sau o litera.
Variabilele sunt case-sensitive astfel încât $ID nu este aceeaşi

variabila ca $Id.

2. Atribuirea de valori unei variabile
Atribuirea unei valori (indiferent ca este vorba de string, integer, float, sau bool) se face cu operatorul de atribuire =.

2.1 Şiruri de caractere
Exemplu 1:

```
<?php
$xy = "Salutare!"; $ 10 = "Studenti";
?>
```

In ambele cazuri variabilelor $xy si $_10 li se atribuie cate un string. Acesta poate fi ulterior afişat cu echo.

2.1.1 Operator de concatenare
Pentru a uni doua şiruri de caractere se utilizează operatorul “**.**” (punct).
Exemplu 2:

```
<?php
$variable = "Buna"." ziua!";
echo $variable;
echo "Salutare"."studenti"; ?>
```

// va afişa "Buna ziua!"
// va afişa "Salutarestudenti"

Se pot uni inclusiv o variabila de tip string cu un string sau două variabile de tip string.

Exemplu 3:

```
<?php
$bla = "Mama";
$name = $bla." mea"; // va uni conținutul lui
$bla + " mea" (cu spațiu)
echo $name;         // va afișa "Mama mea"
echo $name." si a ta";// va afișa "Mama mea
si a ta" ?>
```

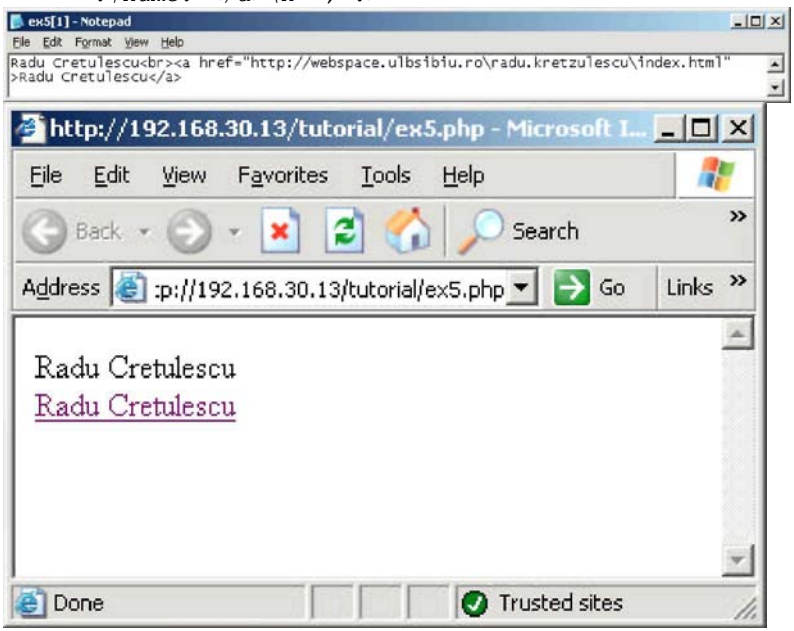
Într-o expresie se pot utiliza mai mulți operatori de concatenare. Oricum nu se va ține cont dacă se unesc string-uri sau variabile.

Exemplu 4:

```
<?php
$prenume = "Radu"; $nume = "Cretulescu";
$Nume = $prenume." ".$nume;  // Variabilele
conţin şiruri de caractere si
//trebuie sa adăugam manual un spaţiu echo
$name;                          //
```

Va afişa "Radu Cretulescu"

```
echo "<a
href=\"http://webspace.ulbsibiu.ro\\radu.kret
zulescu\\index.html\">" .$prenume."
".$nume."</a>\n" ; ?>
```



2.2 Numere

2.2.1 Tipuri de variabile
2.2.2 Numere int
2.2.3 Numere float

2.2.1 Tipuri de variabile

In PHP se cunosc doua tipuri de numere: întregi (int) şi în virgulă

mobilă (float sau double)

2.2.2 Numere întregi

La atribuirea valorii unei variabile se stabileşte automat tipul variabilei. Daca dorim sa definim o variabila int atunci ii atribuim direct o valoare întreaga.

Exemplu 1:

```
<?php
$valoare = 3;
echo "Numarul ales este ".$valoare."\n";
echo 100; ?>
```

2.2.3 Numere float

Numerele care conțin parte zecimală sunt de tip float

```
<?php
$g = 9.81;
echo "Constanta gravitaționala g= ".$g."
unitatea de masura\n"; ?>
```

La acest tip de numere este important să reținem că separatorul zecimal este semnul punct, semnul virgula se utilizează pentru separarea parametrilor unei funcții

Exemplu 2:

```
<?php
xyz(5.6);  // 5.6 este un parametru de tip
float al funcției xyz
abc(5,6);  // 5 si 6 sunt doi parametric ai
funcției abc
abc(5, 6); // ca si mai sus ?>
```

Mai apare încă o problemă: Punctul este şi operatorul de concatenare. Daca am dori să unim două numere avem următoarele posibilități:

1. Atribuim mai întâi numerele unor variabile si apoi unim acestea cu operatorul punct.

```
<?php
$unitati = 4;
$zeci = 5;
echo $zeci.$unitati; ?>
```

2. Dacă dorim să unim două valori întregi cu ajutorul operatorului punct atunci trebuie să inseram câte un spațiu înainte si după operator.

```
<?php
echo 4 . 5; // Afiseaza 45
echo 4.5; // Afiseaza 4.5 ?>
```

Operatori PHP

Operatori aritmetici

Operator	Descriere	Exemplu	Rezultat
+	Adunare	x=2 x+2	4
-	Scădere	x=2 5-x	3
*	Înmulțire	x=4 x*5	20
/	Împărțire	15/5 5/2	3 2.5
%	Modul (rest)	5%2 10%8 10%2	1 2 0
++	Incrementare	x=5 X++	x=6
--	Decrementare	x=5 x--	x=4

Operatori de atribuire

Operator	Exemplu	Corespunde
=	x=y	x=y
+=	x+=y	x=x+y
- =	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Operatori de comparare

Operator	Descriere	Exemplu
==	este egal cu	5==8 returnează false
!=	nu este egal cu	5!=8 returnează true
>	este mai mare ca	5>8 returnează false
<	este mai mic ca	5<8 returnează true
>=	este mai mare sau egal ca	5>=8 returnează false
<=	este mai mic sau egal ca	5<=8 returnează true

Operatori logici

Operator	Descriere	Exemplu
&&	si	x=6 y=3 (x < 10 && y > 1) returnează true
||	sau	x=6 y=3 (x==5 || y==5) returnează false

!	not	x=6 y=3 !(x==y) returnează true

Probleme:

1. Dați un exemplu de script în care declarăm un string
2. Dați un exemplu de utilizare a escape-ului pentru anumite semne

10. Formulare

Elementul FORM

```
<form [action=url] [method=get/post]
[enctype=MIMEType] [onsubmit=script]
[onreset=script] [accept-
charset=set_caractere] [core] [international]
[events]>
</form>
```

Elementul INPUT

```
<input
[type=text|password|checkbox|radio|submit|ima
ge|reset|button|hidden|file] [name=nume]
[value=valoare] [checked] [disabled]
[readonly] [size=latime]
[maxlength=cuvinte_maxime] [src=url]
[alt=altText] [usemap=url]
[align=left|center|right|justify]
[tabindex=numar] [accesskey=keyCombo]
[onfocus=script] [onblur=script]
[onselect=script] [onchange=script]
[accept=set_caractere] [core] [international]
[events]>
```

Acest element input este cel mai important in utilizarea formularelor.

Explicarea valorilor Type ale elementului INPUT

Exemplu:

```
<input type="submit" name="Buton"
value="Buton">
```

Casete de validare

Exemplu:

```
<input type="checkbox" name="nume"
value="valoare">
```

Exemplu:

```
 <input type="file" name="nume"
value="valoare">
```

Exemplu:

```
<input type="hidden" name="nume"
```

```
value="valoare">
```

Exemplu:

```
<input type="image" name="Buton"
src="poza_buton.gif">
```

Casete de introducere a parolei

```
<input type="password" name="nume"
value="valoare">
```

Butoane radio

Exemplu:

```
<input type="radio" name="nume"
value="valoare">
```

Buton reset

Exemplu:

```
<input type="reset" name="Reseteaza"
value="Reseteaza">
```

Exemplu:

```
<input type="button" name="Trimite"
value="Trimite">
```

Exemplu cu valoare initiala:

```
<input type="text" name="nume"
value="valoare">
```

Elementul SELECT

Acest element este folosit pentru crearea unei liste de opțiuni, fie ca un meniu care se desfăşoară, fie ca o caseta cu lista. Fiecare din optiunile din lista reprezintă un element OPTION.

```
<select [name=nume] [size=latime] [multiple]
[disabled] [tabindex=numar] [onfocus=script]
[onblur=script] [onchange=script] [core]
[international] [events]>
```

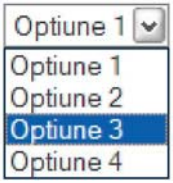
Elementele din select

```
</select>
<option [selected] [disabled] [value=valoare]
[core] [international] [events]>Nume</option>
```

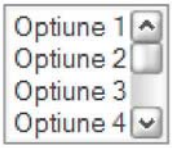
Exemplu select:

```
<select name="test">
<option value="optiune 1">Optiune 1</option>
<option value="optiune 2">Optiune 2</option>
<option value="optiune 3">Optiune 3</option>
<option value="optiune 4">Optiune 4</option>
</select>
```

Arată astfel:




```
<select name="test" multiple size="3">
<option value="optiune 1">Optiune 1</option>
<option value="optiune 2">Optiune 2</option>
<option value="optiune 3">Optiune 3</option>
<option value="optiune 4">Optiune 4</option>
<option value="optiune 5">Optiune 5</option>
<option value="optiune 6">Optiune 6</option>
</select>
```



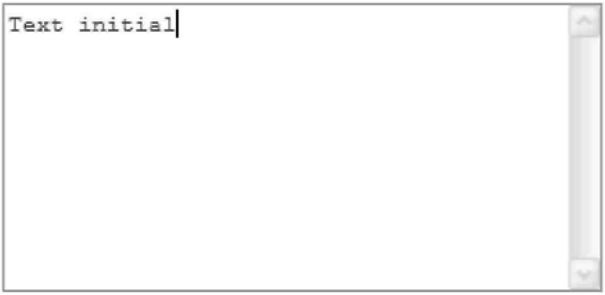
Elementul TEXTAREA

Acest element este asemănător cu cel text numai ca aici se poate tasta într-o sectiune mult mai mare decât în cazul text.

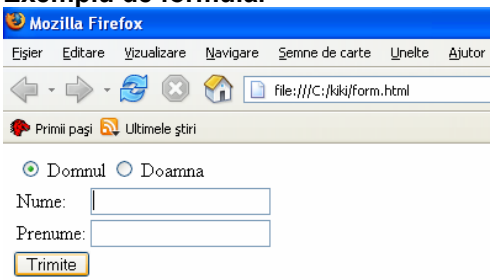
```
<textarea [name=nume] [rows=nr_randuri]
[cols=nr_coloane] [disabled] [readonly]
[tabindex=numar] [onfocus=script]
[onblur=script] [onselect=script]
[onchange=script] [core] [international]
[events]</textarea>
```

Exemplu textarea:

```
<textarea name=”nume” cols=”40”
rows=”10”>Text initial</textarea>
```



Exemplu de formular




```
<form action=“formular.php” method="get">
<table>
<tr>
<td colspan=2>
<input type="radio" name="a" value="h">
domnul
<input type="radio" name="a" value="f">
doamna
<tr>
<td>name:
<td><input type="text" name="nu" size="20">
<tr>
<td>vorname:
<td><input type="text" name=“pre" size="20">
```

```
</table>
<input type="submit" value="trimite">
</form>
```

Formularul prezentat în exemplul de mai sus este pur cod html, deci este static. Importante sunt datele pe care le trimite. În cazul nostru din formularul de mai sus datele vor fi trimise, conform cu atributul action la scrptul *"formular.php".* Acesta le va prelua în funcâe de metoda cu care au fost trimise datele în variabila globală $_POST[] sau $_GET[]. În cazul nostru datele au fost trimise cu ajutorul metodei get. Scriptul formular.php arată astfel.

```
<?php
$nume=$_GET['Nu'];
$prenume=$_GET['Pre'];
$titlu=$_GET['A'];
if ($titlu=="Do")
echo "Domnul";
else
echo "Doamna";
echo "Numele este:" $nume;
echo "Prenumele este:" $prenume;

?>
```

11. Instructiuni php

IF

Pentru a lua o decizie, in scriptul nostru PHP, putem folosi instrucțiunea if. Acestei instrucțiuni trebuie sa îi oferim o condiție pe care să o folosească, iar daca acea condiție este adevărata, va fi executat blocul de cod de după ea. Condițiile din instrucțiunea if trebuie să fie trecute intre paranteze rotunde ( )

```
<?php
$a = 12;
$b = 8;
$rezultat = $a + $b;
if($rezultat == '20') {
echo 'Rezultatul este perfect';
}
?>
```

Puneți codul intr-un fisier if.php, salvați si apoi vizualizați în explorer accesând http://localhost/...
Observați, condiția noastră, si anume aceea ca valoarea rezultata in urma adunării dintre variabila a (12) si variabila b (8) sa fie egala cu numărul 20, este adevărata si in acest caz, codul de după { si respectiv } a fost executat.
Daca valoarea adunării dintre variabila a si variabila b nu era 20, atunci afişarea in browser era nula.

```
<?php
$a = 155;
$b = 8;
$rezultat = $a + $b;
if($rezultat == '20') {
echo 'Rezultatul este perfect';
}
?>
```

In conditiile noastre, dupa cum vedeti, ne folosim de operatorii din PHP pe care i-am scris mai sus

Instructiunea ELSE

De multe ori, pe lângă decizia de a executa o acțiune, atunci când condiția este adevărata, doriți sa executați o alta care in caz contrar (in cazul in care condiția nu este adevărata) sa returneze o alta bucata de cod.

```
<?php
$a = 20;
$b = 8;
$rezultat = $a + $b;
if($rezultat == '20') {
echo 'Rezultatul este perfect';
} else {
echo 'Rezultatul nu este egal cu cel din
conditie';
}
?>
```

Instructiunea ELSEIF

Aceasta instrucțiune este (dupa cum vedeti) o combinație dintre instrucțiunea if si cea else.
Aceasta poate verifica fiecare condiție pana in momentul in care una dintre condițiile găsite returnează o valoare adevărata.

```
<?php
$a = 20;
$b = 1;
$rezultat = $a + $b;
if($rezultat == '20') {
echo 'Rezultatul este egal cu 20';
} elseif ($rezultat == '21') {
echo 'Rezultatul este egal cu 21';
} else {
echo 'Rezultatul nu este egal cu cel din
conditie';
}
?>
```

Instructiunea SWITCH

Aceasta instrucțiune funcționează asemănător cu cea if, insă permite condițiilor sa aibă mai mult de 2 valori. Intr-o instrucțiune if, condiția poate fi adevărată sau falsă, insă intr-o instrucțiune **switch** condiția poate lua orice număr de valori diferite. Aceasta instrucțiune trebuie sa conțină o instrucțiune **case** care sa manevreze fiecare valoare pe care o doriti.

```
<?php
if(!isset($_GET['modul'])) $_GET['modul'] =
''; switch($_GET['modul']) {
case '':
echo 'Pagina switch.php';
break;
case 'pagina1':
echo 'Pagina switch.php?modul=pagina1';
break;
case 'pagina2':
echo 'Pagina switch.php?modul=pagina2';
break; }
?>
```

Bucla WHILE

Cel mai simplu tip de bucla PHP este **while.** Asemenea instrucțiunii if, ea se bazează pe o acțiune. Diferența dintre if si while este aceea ca instrucțiunea if, daca găseşte adevărata condiția, afişează o singura data bucata de cod din ea, insa in condiția while, daca rezultatul este adevărat, bucata de cod din ea se va repeta atâta timp cat conditia este adevarata.

```
<?php
$numar = 1;
while($numar <= 5) echo $numar.'<br>';
$numar++; }
?>
```

Structura FOR

O alternativa cu o funcționalitate mai ridicata pentru utilizarea buclelor este structura repetitiva **for**. Sintaxa este foarte asemănătoare cu cea din limbajele *C/C++* şi anume:

```
for(expresie1; condiţie; expresie2) {
//instructiune
}
```

Prima expresie este evaluata o singura data, înainte de începerea execuției ciclului.
Expresia condiție este testata înaintea fiecărei repetări a buclei.
Daca expresia returnează fals, repetarea se opreşte.
Expresia 2 este executata la sfarsitul fiecarei repetari.

Instructiunea se executa la fiecare repetare a buclei.
Oricare dintre cele trei expresii poate lipsi; in cazul in care o expresie lipseste, se considera ca ea are valoarea **true**.
Bucla WHILE si FOR sunt identice din puct de vedere functional insa bucla FOR este putin mai complexa.

```
<?php
for ($variabila = 1; $variabila <= 10;
$variabila++) {
echo $variabila.'<br>'; }
?>
```

Sa mai luam un exemplu de lucru cu bucla for.
Creati o pagina cu numele **for.php**, introduceti codul urmator apoi testati in browser.

```
<?php
echo "<table
border=\"1\">\n<tr><td>Celula</td></tr>\n";
$culoare = "yellow";
for ($variabila = 1; $variabila <= 10;$variabila++)
{
if($culoare == "yellow") $culoare = "red";
else $culoare = "yellow";
echo "<tr><td
bgcolor=".$culoare.">".$variabila."</td></tr>
\n"; }
echo "</table>";
?>
```

12-13. Conectarea la MySQL folosind PHP

In acest capitol vom învăța cum să ne conectam la o baza de date, cum sa citim informații din ea, cum sa le ştergem/modificăm sau cum sa adăugam noi informații cu ajutorul scripturilor PHP.
Creati un folder in directorul rădăcina a serverului dumneavoastra apache (www) cu numele **tutorial**. Creați un fişier cu numele config.php in care puneti urmatorul cod:

```
<?php
// Informatii baza de date
$host = "localhost";
$user = "root";
$passw = "parola_mysql";
$db_name = "teste";
$conexiune = mysql_connect($host,$user,$passw) or
die("Nu ma pot conecta la MySQL!");
mysql_select_db($db_name,$conexiune) or die("Nu
gasesc baza de date!");
?>
```

Acesta este fisierul de configurare cu care vom realiza conexiunea la baza noastra de date.
Modificați valoarea variabilei $passw cu parola pe care ati setat-o dumneavoastra bazei de date.

Variabila $host este definita cu valoarea localhost deoarece aceasta este adresa serverului. (Adica, serverul Apache+PHP este instalat pe acelasi calculator ca si pachetul MySQL)
Variabila $user este definita cu valoarea root, acesta fiind utilizatorul cu toate drepturile de acces asupra bazei de date, administratorul.
Variabila $db_name este definita cu valoarea teste, aceasta fiind numele bazei de date asupra careia lucram.
Salvati, si inchideti fisierul.

Instrucțiunea SELECT

Creati un nou fisier cu numele **extragere_date.php** si introduceti in

el urmatorul cod:

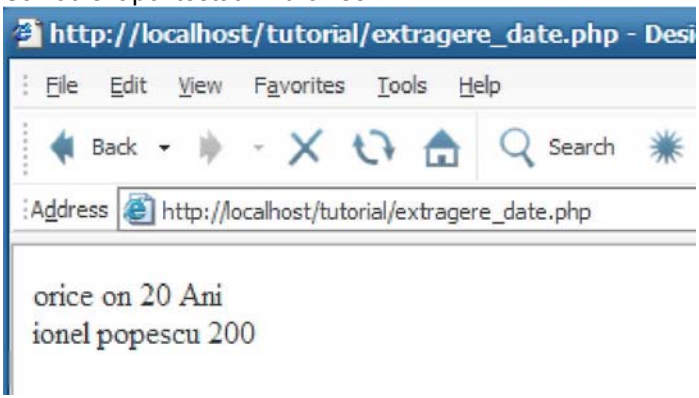
```
<?php
require_once('config.php');
// Selectare dare din baza de date
$cerereSQL = 'SELECT * FROM `formular`';
$rezultat = mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {
echo $rand['nume'];
}
?>
```

Salvati fisierul si apoi accesati in browser

Observati ca daca ati introdus o singura valoare in câmpul nume din tabela formular, aceasta se va afişa pe pagina, daca sunt mai multe intrări ele se vor afisa una in continuarea celeilalte.
Pentru a le afisa una sub alta modificați linia `echo $rand['nume'];` in `echo $rand['nume'].'<br>';`
Si acum sa analizam codul PHP.
Prima linie din script trebuie sa fie **functia require_once**, funcție care include, o singura data, in pagina noastra extragere_date.php pagina config.php; adică datele din config.php vor fi transmise in pagina noastră.
$cerereSQL = 'SELECT * FROM `formular`'; - In aceasta linie avem variabila $cerereSQL cu valoarea cererii SQL pentru a extrage datele din tabela formular. Ea se interpretează cam asa: SELECTEAZA tot DIN formular.
$rezultat = mysql_query($cerereSQL); - In aceasta linie avem variabila $rezultat cu valoarea functiei **mysql_query**, functie care realizeaza deschiderea conexiunii.
while($rand = mysql_fetch_array($rezultat)) {
echo $rand['nume'];
}
In aceasta parte de cod observam bucla while, bucla care am invatat cum se foloseşte…
echo $rand['nume']; - In acest echo apar valorile coloanei nume din tabelul formular.

nume
orice
ionel

In interiorul acestei bucle putem afisa si prenumele, si varsta, modificand constructia echo, si anume: echo $rand['nume'].' '.$rand['prenume'].' '.$rand['varsta'].'
';
Salvati si apoi testati in browser.



In cazul meu, am 2 intrari in tabela formular si anume:

←T→	id	nume	prenume	varsta
	1	orice	on	20 Ani
	2	ionel	popescu	200

In cererea noastra SQL am selectat * adica tot din tabela formular, insa se poate selecta numai un camp sau mai multe.. si anume:
$cerereSQL = 'SELECT `nume,prenume` FROM `formular`';

Sintaxa SELECT:

```
SELECT
[ALL | DISTINCT | DISTINCTROW ]
[HIGH_PRIORITY]
[STRAIGHT_JOIN]
[SQL_SMALL_RESULT] [SQL_BIG_RESULT]
[SQL_BUFFER_RESULT]
```

```
[SQL_CACHE | SQL_NO_CACHE]
[SQL_CALC_FOUND_ROWS] select_expr, ... [INTO
OUTFILE 'file_name' export_options
| INTO DUMPFILE 'file_name'] [FROM
table_references
[WHERE where_definition]
[GROUP BY {col_name | expr | position} [ASC |
DESC], ... [WITH ROLLUP]]
[HAVING where_definition]
[ORDER BY {col_name | expr | position} [ASC |
DESC] , ...]
[LIMIT {[offset,] row_count | row_count
OFFSET offset}]
[PROCEDURE procedure_name(argument_list)]
[FOR UPDATE | LOCK IN SHARE MODE]]
```

Sa luam urmatorul exemplu, pentru a selecta tot din tabela formular, si sa ii punem o conditie WHERE.

```
<?php
require_once('config.php');
// Selectare dare din baza de date
$cerereSQL = 'SELECT * FROM formular WHERE
nume="orice" '; $rezultat =
mysql_query($cerereSQL);
while($rand = mysql_fetch_array($rezultat)) {
echo $rand['nume'].' '.$rand['prenume'].'
'.$rand['varsta'].' <br>';
}
?>
```

Salvati si apoi vizualizati in browser.
Observati ca avem conditia WHERE nume="orice", iar rezultatul o sa fie numai prima linie din tabela formular,
deoarece primul nume de acolo este egal cu numele dat de noi in conditie.

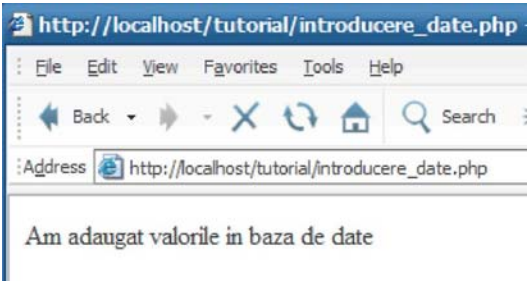
INSERT

Sintaxa insert se foloseşte pentru a adăuga date in baza de date.
Sa luam următorul exemplu:

```
<?php
require_once('config.php');
```

```
$cerereSQL = "INSERT INTO `formular` (`nume`
, `prenume`, `varsta`)
VALUES ('Ivascu', 'Valentin', '20')";
mysql_query($cerereSQL);
echo 'Am adaugat valorile in baza de date';
?>
```

Creați un fişier nou in folderul tutorial si numiți-l **introducere_date.php**, apoi introduceți codul PHP de mai sus, salvati si vizualizati in browser.



Apoi accesati baza de date prin scriptul phpMyAdmin, intrati la tutorial, dupa care apasati pe tabela formular si sus pe butonul “Navigare”.
Observati ca au fost adaugate datele in tabela formular.

Sintaxa INSERT:

```
INSERT [LOW_PRIORITY | DELAYED |
HIGH_PRIORITY] [IGNORE] [INTO] tbl_name
[(col_name,...)] VALUES ({expr |
DEFAULT},...),(...),... [ ON DUPLICATE KEY
UPDATE col_name=expr, ... ]
sau:
INSERT [LOW_PRIORITY | DELAYED |
HIGH_PRIORITY] [IGNORE] [INTO] tbl_name
SET col_name={expr | DEFAULT}, ... [ ON
DUPLICATE KEY UPDATE col_name=expr, ... ]
```

sau:

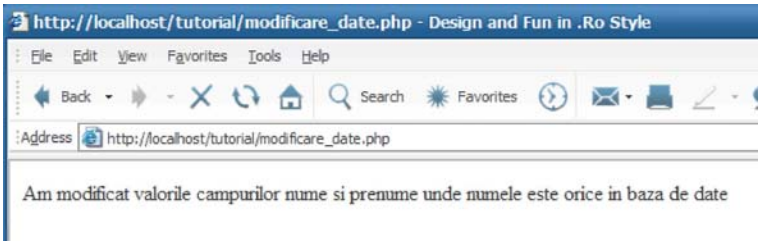
```
INSERT [LOW_PRIORITY | HIGH_PRIORITY]
[IGNORE] [INTO] tbl_name
[(col_name,...)] SELECT ... [ ON
DUPLICATE KEY UPDATE col_name=expr, ...
]
```

UPDATE

Sintaxa update se foloseşte pentru a modifica datele existente din baza de date.
Sa luam urmatorul exemplu:

```
<?php
require_once('config.php');
$cerereSQL = "UPDATE `formular` SET
nume='nume', prenume='prenume' WHERE
nume='orice' "; mysql_query($cerereSQL);
echo 'Am modificat valorile campurilor nume
si prenume unde numele este orice in baza de
date';
?>
```

Creati un fisier nou in folderul tutorial, numiti-l **modificare_date.php** si introduceti codul PHP de mai sus, dupa care salvati si vizualizati in browser.



Accesati baza de date prin phpMyAdmin si observati ca prima coloana a fost modificata.

Sintaxa UPDATE

Sintaxa simpla

```
UPDATE [LOW_PRIORITY] [IGNORE] SET
col_name1=expr1 [, col [WHERE
where_definition] [ORDER BY ...] [LIMIT
row_count]
```

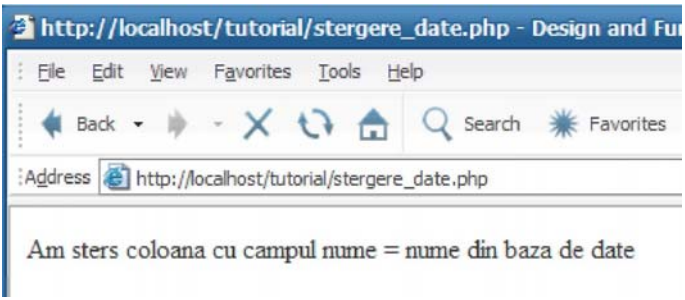
```
Sintaxa multipla
UPDATE [LOW_PRIORITY] [IGNORE]
table_references SET col_name1=expr1 [,
col_name2=expr2 ...] [WHERE where_definition]
tbl_name name2=expr2 ...]
```

DELETE

Sintaxa delete se foloseşte pentru a şterge datele existente din baza de date.
Sa luam urmatorul exemplu:

```
<?php
require_once('config.php');
$cerereSQL = "DELETE FROM `formular` WHERE
nume='nume'"; mysql_query($cerereSQL);
echo 'Am sters coloana cu campul nume = nume
din baza de date';
?>
```

Creati un fisier nou in folderul tutorial si numiti-l **stergere_date.php,** troduceti codul PHP de mai sus, salvati si vizualizati in browser.



Accesati baza de date prin phpMyAdmin si observati ca prima coloana a fost stearsa.
Sintaxa DELETE
:
Sintaxa simpla

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM
tbl_name [WHERE where_definition] [ORDER BY
```

```
...] [LIMIT row_count]
Sintaxa multipla
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
tbl_name[.*] [, tbl_name[.*] ...] FROM
table_references [WHERE where_definition]
```

sau:

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
FROM tbl_name[.*] [, tbl_name[.*] ...] USING
table_references [WHERE where_definition]
```

Exerciții:

1. Realizați un script care se conectază la baza de date teste făcând legătura între tabela profi şi tabela materii pe baza unei chei primare id_profi.
2. Selectați profeorii care predau o anumită materie