El llenguatge PHP

??? abstract “Duració i criteris d’avaluació”

```
Duració estimada: 20 hores

<hr />

| Resultat d'aprenentatge | Criteris d'avaluació|
| -------                 | -------             |
| 2. Escriu sentències executables per un servidor Web reconeixent i aplicant procediments c
| 3. Genera pàgines web de forma dinàmica seguint especificacions rebudes. | a) S'han identi
```

1. Introducció a PHP

Què és PHP? **PHP** (Hypertext Preprocessor) és un llenguatge de programació de codi obert, especialment dissenyat per al desenvolupament web del costat del servidor. Es pot incrustar fàcilment en codi HTML, cosa que el fa una opció popular per crear aplicacions web dinàmiques i interactives. PHP s’utilitza per gestionar el contingut dinàmic, les bases de dades, el seguiment de sessions i fins i tot per crear aplicacions completes.

Història de PHP PHP (Hypertext Preprocessor) és un llenguatge de programació de codi obert creat l’any 1994 per Rasmus Lerdorf. La primera versió pública va ser PHP/FI (Personal Home Page/Forms Interpreter), que va evolucionar fins a convertir-se en PHP 3 el 1998, el qual va ser completament reescrit per Zeev Suraski i Andi Gutmans.

Actualment, PHP és mantingut per The PHP Group i la versió estable més recent (a partir de juliol de 2024) és PHP 8.3. Aquesta versió inclou millores en el rendiment, noves funcions com les propietats llegibles i escriptibles per tipus, expressió `match`, i molt més.

Importància en el desenvolupament web PHP és un dels llenguatges més utilitzats en el desenvolupament web per diverses raons:

- **Facilitat d’aprenentatge:** PHP és relativament fàcil d’aprendre per als principiants en programació web. La seva sintaxi és senzilla i similar a altres llenguatges com C i Perl.
- **Gran comunitat i suport:** PHP té una gran comunitat de desenvolupadors que proporcionen suport, biblioteques, extensions i eines. A més, hi ha nombrosos recursos en línia, tutorials i documentació.
- **Integració amb bases de dades:** PHP es pot integrar fàcilment amb una àmplia varietat de bases de dades, cosa que el fa ideal per a aplicacions web basades en dades.
- **Flexibilitat i escalabilitat:** PHP és altament flexible i es pot utilitzar per desenvolupar tant aplicacions petites com projectes grans i complexos.

També és escalable, cosa que permet gestionar un gran volum de trànsit.
- **Cost efectiu:** Com que és de codi obert, PHP és gratuït per utilitzar i distribuir. Això redueix els costos de desenvolupament per a les empreses i els desenvolupadors.

Exemple de codi PHP simple Aquí tens un exemple senzill de com funciona PHP:

=== "PHP"

```
<!DOCTYPE html>
<html>
<body>

<h1>El meu primer script PHP</h1>

<?php
echo "Hola, món!";
?>

</body>
</html>
```

En aquest exemple, el codi PHP s'incrusta dins del codi HTML i es delimina amb . Quan el servidor processa aquest fitxer, executa el codi PHP i envia el resultat al navegador, generant el contingut dinàmic "Hola, món!". Aquest és un altre exemple de com PHP pot generar HTML dinàmicament:

2. Llenguatges imbricats en HTML

PHP permet incrustar codi dins de documents HTML, permetent la generació de contingut dinàmic. El codi PHP es delimita amb `<?php ... ?>`. És comú utilitzar codi PHP dins de HTML per generar contingut dinàmic. També es pot incloure HTML dins de fitxers PHP.

```
<!DOCTYPE html>
<html>
<body>
    <h1>Benvingut a la meva web</h1>
    <p>La data d'avui és: <?= date('Y-m-d') ?></p>
</body>
</html>
```

El codi HTML pot ser generat dins d'instruccions PHP.

```
<?php
echo "<html><body>";
echo "<h1>Benvingut a la meva web</h1>";
echo "<p>La data d'avui és: " . date('Y-m-d') . "</p>";
```

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

!!! tip “Només etiquetes d’obertura” Si el nostre codi només contindrà codi PHP i res d’html, com per exemple, quan codifiquem classes o interfícies, només posarem l’etiqueta d’obertura, per a així indicar que és una arxiu de php pur.

3. Etiquetes per a inserció de codi

Per inserir codi PHP dins de HTML, utilitzem les etiquetes:

```php
<?php
// Codi PHP aquí
?>
```

Per imprimir directament:

```php
<?= $variable ?>
```

Exemple:

```php
<!DOCTYPE html>
<html>
<body>
    <h1>Benvingut a la meva web</h1>
    <p>La data d'avui és: <?= date('Y-m-d') ?></p>
</body>
</html>
```

4. Tipus de dades. Conversions entre tipus de dades

PHP té diversos tipus de dades: enter, flotant, cadena, booleà, matriu, objecte, nul.

```php
$enter = 10; // Enter
$flotant = 3.14; // Flotant
$cadena = "Hola, món!"; // Cadena
$boolea = true; // Booleà
$matriu = array(1, 2, 3); // Matriu
$objecte = new stdClass(); // Objecte
$nul = null; // Nul
```

Conversions:

```php
$integer = (int) $variable;
$float = (float) $variable;
$string = (string) $variable;
```

Exemple:

```php
$cadena = "123";
$enter = (int)$cadena; // Converteix la cadena "123" a l'enter 123
```

5. Constants

Són variables el valor dels quals no varien. Existeixen dues possibilitats:

- `define(NOMBRE, valor);`
- `const NOMBRE; // PHP > 5.3`

```php
<?php
define("PI", 3.1416);
const IVA = 0.21;

echo PI, " ", IVA; // No se pone el símbolo dolar
```

- Es declaren sempre en MAJÚSCULES
- Hi ha un conjunt de constants ja predefinides, també conegudes com *magic constants*: https://www.php.net/manual/es/language.constants.predefined.php

6. Variables. Operadors. Àmbits de les variables

- No és necessari declarar-les prèviament.
- Comencen per `$`, per exemple `$nom`. Després del `$`, el següent caràcter ha de ser una lletra en minúscula (recomanació) o guió baix `_`. Després ja es poden posar números.
- Són case *sensitive*: `$*var != $*vAR`
- No es declara el seu tipus, el tipat és dinàmic. S'assigna en temps d'execució depenent del valor assignat.
- Convenient inicialitzar-les, sinó donen error a l'utilitzar-les.

```php
$variable = "valor";
```

Operadors: - Aritmètics: `+`, `-`, `*`, `/` - Assignació: `=`, `+=`, `-=` - Comparació: `==`, `===`, `!=` - Lògics: `&&`, `||`, `!` - Concatenació: `.`

Els àmbits d'utilització d'una variable són:

Local: dins d'una funció. Global: fora de qualsevol funció. Estàtic: persisteixen el seu valor entre crides a la funció.

Exemple:

```php
$global = 10; // Variable global
function contar() {
    static $vegades = 0;
    $local = 0; // Variable local
    $local++;
    $vegades++;
```

```
    $global++;
    echo $local,$vegades,$global;
}
contar(); // Mostra 1,1,11
contar(); // Mostra 1,2,12
```

7. Sentències simples en PHP i els seus efectes

Assignacions

```
$x = 5;
$y = "Hola món";
```

Operacions aritmètiques

```
$suma = $x + 10;  // Resulta en 15
$producte = $x * 2;  // Resulta en 10
```

Operacions d'entrada i eixida:

```
echo "Hola, món!";
print "Hola, món!";
```

Operacions amb cadenes:

```
$nom = "Joan";
$salutacio = $y . ", " . $nom;  // Resulta en "Hola món, Joan"
```

Exemple complet:

```
<?php
// Assignació de valors
$x = 5;
$y = "Hola món";

// Operacions aritmètiques
$suma = $x + 10;
$producte = $x * 2;

// Concatenació de cadenes
$nom = "Joan";
$salutacio = $y . ", " . $nom;

// Impressió de resultats
echo $y;  // Hola món
echo $suma;  // 15
echo $producte;  // 10
echo $salutacio;  // Hola món, Joan
```

8. Funcions

1. **Sintaxi bàsica**:
   - La paraula clau `function` es fa servir per a definir la funció.
   - Després segueix el nom de la funció, que pot contenir lletres, nombres i guions baixos (però no pot començar amb un nombre).
   - Entre els parèntesis es poden passar arguments (opcionals).
   - El cos de la funció es defineix entre claudàtors `{}`.

```php
function nomFuncio() {
    // Codi de la funció
}
```

2. **Amb arguments**:
   - Els arguments es passen entre els parèntesis i poden ser utilitzats dins de la funció.

```php
function saluda($nom) {
    echo "Hola, $nom!";
}

saluda("Maria");  // Mostra "Hola, Maria!"
```

3. **Amb valor de retorn**:
   - La paraula clau `return` permet retornar un valor des de la funció. Es pot retornar qualsevol tipus de dades (enter, string, array, etc.).

```php
function suma($a, $b) {
    return $a + $b;
}

$resultat = suma(5, 3);  // $resultat conté 8
```

4. **Funcions amb valors per defecte**:
   - Si no es passa cap argument, la funció pot utilitzar un valor per defecte.

```php
function saludar($nom = "amic") {
    echo "Hola, " . $nom;
}

saludar();  // Mostra "Hola, amic"
saludar("Maria");  // Mostra "Hola, Maria"
```

5. **Tipus de dades en arguments i retorn** (des de PHP 7):
   - És possible especificar el tipus de dades dels arguments i el tipus de retorn. Això ajuda a assegurar que la funció rep i retorna els tipus esperats.

```php
function sumar(int $a, int $b): int {
```

```
    return $a + $b;
}

$resultat = sumar(5, 3);  // $resultat conté 8
```

9. Directives per a modificar el comportament predeterminat del codi

Include i require:

```
include 'fitxer.php';
require 'fitxer.php';
```

Include_once i require_once:

```
include_once 'fitxer.php';
require_once 'fitxer.php';
```

Exemple:

```
// contingut de fitxer.php
<?php
function saludar() {
    echo "Hola!";
}
?>

// contingut de principal.php
<?php
include 'fitxer.php';
saludar(); // Mostra "Hola!"
?>
```

10. Mecanismes de decisió (if, switch)

If, else, elseif:

```
if ($condicio) {
    // Codi si la condició és certa
} elseif ($altra_condicio) {
    // Codi si la segona condició és certa
} else {
    // Codi si cap de les condicions anteriors és certa
}
```

Switch:

```
switch ($variable) {
    case 1:
        // Codi per al cas 1
```

```
        break;
    case 2:
        // Codi per al cas 2
        break;
    default:
        // Codi per al cas per defecte
}
```

Exemple:

```
$dia = "dilluns";
switch ($dia) {
    case "dilluns":
        echo "Avui és dilluns";
        break;
    case "dimarts":
        echo "Avui és dimarts";
        break;
    default:
        echo "Avui no és dilluns ni dimarts";
}
```

Match:

```
$result = match ($variable) {
    valor1 => resultat1,
    valor2 => resultat2,
    valor3 => resultat3,
    // ...
    default => valorPerDefecte,
};
```

Diferències amb switch

```
1. **Comparació estricta**: `match` utilitza comparació estricta (===) per comparar els valo
2. **Retorna un valor**: `match` és una expressió, la qual cosa significa que retorna un val
3. **No necessita break**: A diferència de `switch`, no es necessita l'ús de `break` per evi
4. **Més concís**: Permet una sintaxi més neta i concisa.
```

Exemple simple:

```
$color = 'roig';

$resultat = match ($color) {
    'roig' => 'El color és vermell',
    'blau' => 'El color és blau',
    'verd' => 'El color és verd',
    default => 'Color desconegut',
};
```

```php
echo $resultat;  // Sortida: El color és vermell
```

Exemple expressions complexes:

```php
$edat = 20;

$categoria = match (true) {
    $edat >= 0 && $edat <= 12 => 'Nen',
    $edat >= 13 && $edat <= 17 => 'Adolescent',
    $edat >= 18 && $edat <= 64 => 'Adult',
    $edat >= 65 => 'Gent gran',
    default => 'Edat desconeguda',
};

echo $categoria;  // Sortida: Adult
```

Exemples casos multiples:

```php
$dia = 'dimecres';

$tipusDia = match ($dia) {
    'dissabte', 'diumenge' => 'Cap de setmana',
    'dilluns', 'dimarts', 'dimecres', 'dijous', 'divendres' => 'Dia laborable',
    default => 'Dia desconegut',
};
echo $tipusDia;  // Sortida: Dia laborable
```

11. Bucles (for, while, foreach)

For:

```php
for ($i = 0; $i < 10; $i++) {
    echo $i;
}
```

While:

```php
$i = 0;
while ($i < 10) {
    echo $i;
    $i++;
}
```

Foreach:

```php
$matriu = array(1, 2, 3);
foreach ($matriu as $valor) {
    echo $valor;
}
```

Exemple:

```
$fruites = array("poma", "plàtan", "maduixa");
foreach ($fruites as $fruita) {
    echo $fruita;
}
```

12. Arrays

Per a emmagatzemar dades compostes, podem utilitzar tant arrays senzills com arrays associatius (similars a un mapa). En realitat tots els arrays són mapes ordenats compostos de parells clau-valor.

!!! caution "Compte amb mesclar tipus" Com el tipat és dinàmic, nostres arrays poden contenir dades de diferents tipus. No es recomana mesclar els tipus.

De la mateixa manera que Java, es defineixen mitjançant claudàtors, són 0-index, i es pot assignar un valor a un posició determinada:

```
<?php
$frutas = array("naranja", "pera", "manzana");

$frutas2 = ["naranja", "pera", "manzana"];

$frutas3 = [];
$frutas3[0] = "naranja";
$frutas3[1] = "pera";
$frutas3[] = "manzana"; // lo añade al final
```

Podem obtindre la grandària del array mitjançant la funció `count(array)`. Per a recórrer el array farem ús d'un bucle `for`:

```
<?php
$tam = count($frutas); // tamaño del array

for ($i=0; $i<count($frutas); $i++) {
    echo "Elemento $i: $frutas[$i] <br />";
}
```

Una altra manera de recórrer els arrays, fins i tot més elegant, és fer ús de `foreach`. La seua sintaxi és `foreach (array as element)`:

```
<?php
// Mitjançant foreach no necessitem saber la grandària del array
foreach ($frutas as $fruta) {
    echo "$fruta <br />";
}
```

13. Arrays associatius

Cada element és un parell clau-valor. En comptes d'accedir per la posició, el fem mitjançant una clau. Així doncs, per a cada clau s'emmagatzema un valor.

A l'hora de recórrer aquest tipus de arrays, mitjançant `foreach` separem cada element en una parella `clau => valor`:

```php
<?php
$capitales = ["Italia" => "Roma",
    "Francia" => "Paris",
    "Portugal" => "Lisboa"];
$capitalFrancia = $capitales["Francia"]; // se accede al elemento por la clave, no la posic

$capitales["Alemania"] = "Berlín"; // añadimos un elemento

echo "La capital de Francia es $capitalFrancia <br />";
echo "La capital de Francia es {$capitales["Francia"]} <br />";

$capitales[] = "Madrid"; // se añade con la clave 0 !!! ¡¡¡No asignar valores sin clave!!!

foreach ($capitales as $valor) { // si recorremos un array asociativo, mostraremos los valo
    echo "$valor <br />";
}

foreach ($capitales as $pais => $ciudad) { // separamos cada elemento en clave => valor
    echo "$pais : $ciudad <br />";
}
```

14. Com utilitzar les cometes en PHP

En PHP, les cometes dobles (") i les cometes simples (') s'utilitzen per definir cadenes de caràcters, però tenen comportaments diferents a l'hora de processar variables:

- **Cometes dobles (")**: Interpolen variables i seqüències d'escapament especials. És a dir, el contingut de la variable es mostrarà dins de la cadena.

  ```php
  $nom = "Maria";
  echo "Hola, $nom!";  // Sortida: Hola, Maria!
  ```

- **Cometes simples (')**: No interpolen variables ni seqüències d'escapament especials. La cadena es mostrarà exactament com es defineix.

  ```php
  $nom = "Maria";
  echo 'Hola, $nom!';  // Sortida: Hola, $nom!
  ```

Exemple d'ús de cometes dobles per imprimir variables

```php
$color = "blau";
$frase = "El meu color preferit és $color.";
echo $frase;  // Sortida: El meu color preferit és blau.
```

15. Comentaris en el codi

Comentaris d'una línia:

```php
// Això és un comentari d'una línia
```

Comentaris de diverses línies:

```php
/* Això és un comentari
   de diverses línies */
```

16. Variables de servidor

PHP emmagatzema la informació del servidor i de les peticions HTTP en sis arrays globals:

- `$_ENV`: informació sobre les variables d'entorn
- `$_GET`: paràmetres enviats en la petició GET
- `$_POST`: paràmetres enviats en el envio POST
- `$_COOKIE`: conté les cookies de la petició, les claus del array són els noms de les cookies
- `$_SERVER`: informació sobre el servidor
- `$_FILES`: informació sobre els fitxers carregats via upload

Si ens centrem en el array `$_SERVER` podem consultar les següents propietats:

- `PHP_SELF`: nom del script executat, relatiu al document root (p.ej: `/tenda/carret.php`)
- `SERVER_SOFTWARE`: (p.ej: Apatxe)
- `SERVER_NAME`: domini, àlies DNS (p.ej: `www.elche.es`)
- `REQUEST_METHOD`: GET
- `REQUEST_URI`: URI, sense el domini
- `QUERY_STRING`: tot el que va després de `?` en la URL (p.ej: `heroe=Batman&nomene=Bruce`)

Més informació en https://www.php.net/manual/es/reserved.variables.server.php

```php
<?php
echo $_SERVER["PHP_SELF"]."<br>"; // /u4/401server.php
echo $_SERVER["SERVER_SOFTWARE"]."<br>"; // Apache/2.4.46 (Win64) OpenSSL/1.1.1g PHP/7.4.9
echo $_SERVER["SERVER_NAME"]."<br>"; // localhost

echo $_SERVER["REQUEST_METHOD"]."<br>"; // GET
echo $_SERVER["REQUEST_URI"]."<br>"; // /u4/401server.php?heroe=Batman
echo $_SERVER["QUERY_STRING"]."<br>"; // heroe=Batman
```

Altres propietats relacionades:

- `PATH_INFO`: ruta extra després de la petició. Si la URL és `http://www.php.com/php/pathinfo.php/algo` llavors `$_SERVER['PATH_INFO']` serà `/alguna cosa/cosa`.
- `REMOTE_HOST`: hostname que va fer la petició
- `REMOTE_ADDR`: IP del client
- `AUTH_TYPE`: tipus d'autenticació (p.ej: Basic)
- `REMOTE_USER`: nom de l'usuari autenticat

Apatxe crea una clau per a cada capçalera HTTP, en majúscules i substituint els guions per subratllats:

- `HTTP_USER_AGENT`: agent (navegador)
- `HTTP_REFERER`: pàgina des de la qual es va fer la petició

```php
<?php
echo $_SERVER["HTTP_USER_AGENT"]."<br>"; // Mozilla/5.0 (Windows NT 10.0; Win64; x64) Apple
```

17. Formularis

A l'hora d'enviar un formulari, hem de tindre clar quan usar GET o POST

- GET: els paràmetres es passen en la URL
  - <2048 caràcters, només ASCII
  - Permet emmagatzemar la direcció completa (marcador / historial)
  - Idempotent: dues crides amb les mateixes dades sempre ha de donar el mateix resultat
  - El navegador pot cachejar les cridades
- POST: paràmetres ocults (no encriptats)
  - Sense límit de dades, permet dades binàries.
  - No es poden escorcollar
  - No idempotent → actualitzar la BBDD

Així doncs, per a recollir les dades accedirem al array depenent del mètode del formulari que ens ha invocat:

```php
<?php
$par = $_GET["parametro"]
$par = $_POST["parametro"]
```

A l'hora d'enviar un formulari, hem de tindre clar quan usar GET o POST. Per als següents apartats ens basarem en el següent exemple:

Validació

Respecte a la validació, és convenient sempre fer *validació doble*:

- En el client mitjançant JS
- En servidor, abans de cridar a la capa de negoci, és convenient tornar a validar les dades.

```php
<?php
if (isset($_GET["parametro"])) {
    $par = $_GET["parametro"];
    // comprobar si $par tiene el formato adecuado, su valor, etc...
}
```

!!! info "Llibreries de validació" Existeixen diverses llibreries que faciliten la validació dels formularis, com són respect/validation o particle/validator. Quan estudiem Laravel aprofundirem en la validació de manera declarativa.

Parámetre multivalor

Existeixen elements HTML que envien diversos valors:

- `select multiple`
- `checkbox`

Per a recollir les dades, el nom de l'element ha de ser un array.

```php
<select name="lenguajes[]" multiple="true">
    <option value="c">C</option>
    <option value="java">Java</option>
    <option value="php">PHP</option>
    <option value="python">Python</option>
</select>

<input type="checkbox" name="lenguajes[]" value="c" /> C<br />
<input type="checkbox" name="lenguajes[]" value="java" /> Java<br />
<input type="checkbox" name="lenguajes[]" value="php" /> Php<br />
<input type="checkbox" name="lenguajes[]" value="python" /> Python<br />
```

De manera que després en recollir les dades:

```php
<?php
$lenguajes = $_GET["lenguajes"];

foreach ($lenguajes as $lenguaje) {
    echo "$lenguaje <br />";
}
```

Tornant a emplenar un formulari

Un *sticky form* és un formulari que recorda els seus valors. Per a això, hem d'emplenar els atributs `value` dels elements HTML amb la informació que contenien:

```php
<?php
if (!empty($_POST['modulos']) && !empty($_POST['nombre'])) {
  // Aquí se incluye el código a ejecutar cuando los datos son correctos
} else {
```

```
  // Generamos el formulario
  $nombre = $_POST['nombre'] ?? "";
  $modulos = $_POST['modulos'] ?? [];
  ?>
  <form action="<?php echo $_SERVER['PHP_SELF'];?>" method="POST">
   <p><label for="nombre">Nombre del alumno:</label>
    <input type="text" name="nombre" id="nombre" value="<?= $nombre ?>" />
   </p>
   <p><input type="checkbox" name="modulos[]" id="modulosDWES" value="DWES"
    <?php if(in_array("DWES",$modulos)) echo 'checked="checked"'; ?> />
    <label for="modulosDWES">Desarrollo web en entorno servidor</label>
   </p>
   <p><input type="checkbox" name="modulos[]" id="modulosDWEC" value="DWEC"
    <?php if(in_array("DWEC",$modulos)) echo 'checked="checked"'; ?> />
    <label for="modulosDWEC">Desarrollo web en entorno cliente</label>
   </p>
   <input type="submit" value="Enviar" name="enviar"/>
  </form>
<?php } ?>
```

Pujant arxius

S'emmagatzemen en el servidor en el array `$_FILES` amb el nom del camp del tipus `file` del formulari.

```
<form enctype="multipart/form-data" action="<?php echo $_SERVER['PHP_SELF']; ?>"  method="P
    Archivo: <input name="archivoEnviado" type="file" />
    <br />
    <input type="submit" name="btnSubir" value="Subir" />
</form>
```

Configuració en `php.ini`

- `file_uploads`: on / off
- `upload_max_filesize`: 2M
- `upload_tmp_dir`: directori temporal. No és necessari configurar-ho, agafarà el predeterminat del sistema
- `post_max_size`: grandària màxima de les dades POST. Ha de ser major a upload_max_filesize.
- `max_file_uploads`: nombre màxim d'arxius que es poden carregar alhora.
- `max_input_estafe`: temps màxim emprat en la càrrega (GET/POST i upload → normalment es configura en 60)
- `memory_limit`: 128M
- `max_execution_estafe`: temps d'execució d'un script (no té en compte el upload)

Per a carregar els arxius, accedim al array `$_FILES`:

```php
<?php
if (isset($_POST['btnSubir']) && $_POST['btnSubir'] == 'Subir') {
    if (is_uploaded_file($_FILES['archivoEnviado']['tmp_name'])) {
        // subido con éxito
        $nombre = $_FILES['archivoEnviado']['name'];
        move_uploaded_file($_FILES['archivoEnviado']['tmp_name'], "./uploads/{$nombre}");

        echo "<p>Archivo $nombre subido con éxito</p>";
    }
}
```

Cada arxiu carregat en `$_FILES` té:

- `name`: nom
- `tmp_name`: ruta temporal
- `size`: grandària en bytes
- `type`: tipus ACARONE
- `error`: si hi ha error, conté un missatge. Si ok → 0.

Capçaleres de resposta

Ha de ser el primer a retornar. Es retornen mitjançant la funció `header(cadena)`. Mitjançant les capçaleres podem configurar el tipus de contingut, temps d'expiració, redirigir el navegador, especificar errors HTTP, etc.

```php
<?php header("Content-Type: text/plain"); ?>
<?php header("Location: http://www.ejemplo.com/inicio.html");
exit();
```

Es pot comprovar en les eines del desenvolupador dels navegadors web mitjançant *Developer Tools –> Network –> Headers*.

És molt comú configurar les capçaleres per a evitar consultes a la cache o provocar la seua renovació:

```php
<?php
header("Expires: Sun, 31 Jan 2021 23:59:59 GMT");
// tres horas
$now = time();
$horas3 = gmstrftime("%a, %d %b %Y %H:%M:%S GMT", $now + 60 * 60 * 3);
header("Expires: {$horas3}");
// un año
$now = time();
$anyo1 = gmstrftime("%a, %d %b %Y %H:%M:%S GMT", $now + 365 * 86440);
header("Expires: {$anyo1}");
// se marca como expirado (fecha en el pasado)
$pasado = gmstrftime("%a, %d %b %Y %H:%M:%S GMT");
header("Expires: {$pasado}");
```

```php
// evitamos cache de navegador y/o proxy
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT");
header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT");
header("Cache-Control: no-store, no-cache, must-revalidate");
header("Cache-Control: post-check=0, pre-check=0", false);
header("Pragma: no-cache");
// redirigimos a otra página
header("Location: http://www.ejemplo.com/inicio.html");
exit();
// devolvemos un error 404
header("HTTP/1.0 404 Not Found");
// enviamos un archivo para descargar
header("Content-Type: application/force-download");
// enviamos un archivo json
header("Content-Type: application/json");
```

18. Referències

Llibres Recomanats

1. **PHP and MySQL Web Development** de Luke Welling i Laura Thomson
   - Aquest llibre proporciona una visió completa sobre el desenvolupament web amb PHP i MySQL, des de conceptes bàsics fins a tècniques avançades.
2. **Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5** de Robin Nixon
   - Ideal per a aquells que busquen una introducció pràctica al desenvolupament web modern utilitzant PHP juntament amb altres tecnologies web.
3. **Modern PHP: New Features and Good Practices** de Josh Lockhart
   - Aquest llibre explora les novetats i bones pràctiques en les versions modernes de PHP, incloent-hi temes com la programació orientada a objectes i l'ús de frameworks.

Documentació Oficial

- **Documentació Oficial de PHP**
  - La documentació oficial de PHP és una referència essencial per a qualsevol desenvolupador que utilitzi PHP. Inclou guies de llenguatge, referències de funcions i exemples de codi.

Cursos en Línia

- **Curso de PHP en W3Schools**
  - Un recurs en línia gratuït amb tutorials i exercicis interactius que cobreixen els conceptes bàsics i avançats de PHP.

- **PHP for Beginners - Become a PHP Master - CMS Project** a Udemy
  - Un curs complet de PHP en Udemy que inclou la creació d'un sistema de gestió de continguts (CMS) com a projecte final.

Recursos Addicionals

- **PHP: The Right Way**
  - Un recurs comunitari que ofereix bones pràctiques, consells i pautes per al desenvolupament amb PHP.
- **Laravel Documentation**
  - Documentació oficial de Laravel, un dels frameworks de PHP més populars per al desenvolupament web.

Exemples i Tutorials

- **TutorialsPoint PHP Tutorial**
  - Proporciona una introducció detallada a PHP amb exemples de codi i explicacions pas a pas.
- **GeeksforGeeks PHP Tutorials**
  - Una col · lecció d'articles i tutorials sobre diversos temes relacionats amb PHP, des de conceptes bàsics fins a avançats.

Repositoris de Codi

- **GitHub PHP**
  - Una col · lecció de repositoris de codi font en PHP, on pots trobar projectes de codi obert i exemples pràctics.

19. Exercicis

Bateria d'Exercicis Solucionats per a la Unitat de PHP

Exercici 1: Introducció a PHP

1. Crea un fitxer que imprimeixi "Hola, món!" a la pantalla.
2. Modifica el fitxer per tal que imprimeixi el teu nom utilitzant una variable.

Solució

```
```php
<?php
// Exercici 1
$name = 'Ignasi':
echo "Hola, món!" . "Hola, " .$name;
?>
```
```

Exercici 2: Ús de cometes

1. Crea un fitxer que definisca una variable `$name` amb el teu nom i imprimeixi la frase "Hola, [nom]!" utilitzant cometes dobles.
2. Fes-ho amb cometes simples i compara els resultats.

Solució

```php
<?php
// Exercici 2
$name = 'Ignasi':
echo "Hola, $name!";
echo 'Hola, $name!';
?>
```

Exercici 3: Funcions bàsiques

1. Crea una funció `suma` que sume dos números i retorni el resultat. Invoca la funció amb els números 5 i 3 i imprimeix el resultat.
2. Crea una funció `multiplicacio` que multipliqui dos números i retorni el resultat. Invoca la funció amb els números 4 i 7 i imprimeix el resultat.

Solució

```php
<?php
// Exercici 2
function suma($a, $b) {
    return $a + $b;
}
function multiplicacio($a, $b) {
    return $a * $b;
}
echo suma(5, 3);  // Sortida: 8
echo multiplicacio(4, 7);  // Sortida: 28
?>
```

Exercici 4: Control de flux - Condicionals

1. Crea un fitxer que definisca una variable `$edat`. Si `$edat` és major o igual a 18, imprimeix "Ets major d'edat"; en cas contrari, imprimeix "Ets menor d'edat".
2. Modifica el fitxer `edat.php` per tal que imprimeixi "Edat invàlida" si `$edat` és un número negatiu.

Solució

```php
<?php
// Exercici 4
$edat = 20;
if ($edat < 0) {
    echo "Edat invàlida";
} else{
    if ($edat >= 18) {
        echo "Ets major d'edat";
    } else {
        echo "Ets menor d'edat";
    }
}
?>
```
</details>

Exercici 5: Control de flux - Bucles
1. Crea un fitxer que utilitzi un bucle `for` per imprimir els números del 0 al 9.
2. Fes-ho també amb un bucle `while` que faci el mateix.

<details>
<summary>Solució</summary>

```php
<?php
// Exercici 5
// Bucle for
for ($i = 0; $i < 10; $i++) {
    echo $i . "<br>";
}
// Bucle while
$i = 0;
while ($i < 10) {
    echo $i . "<br>";
    $i++;
}
?>
```
</details>

Exercici 6: Treballar amb arrays
1. Crea un fitxer que definisca un array `$fruites` amb tres elements: "poma", "plàtan" i "
2. Afegeix un quart element "taronja" a l'array i imprimeix tots els elements utilitzant un

<details>
<summary>Solució</summary>

```php
<?php
// Exercici 6
// Punt 1
$fruites = array("poma", "plàtan", "maduixa");
echo $fruites[0] . "<br>";
// Punt 2
$fruites[] = "taronja";
foreach ($fruites as $fruita) {
    echo $fruita . "<br>";
}
?>
```
</details>

Exercici 7: Cometes dobles i variables
1. Crea un fitxer que definisca una variable `$color` amb el valor "blau". Utilitza cometes
2. Ara, utilitza cometes simples i concatenació.

<details>
<summary>Solució</summary>

```php
<?php
// Exercici 7
$color = "blau";
echo "El meu color preferit és $color.";
echo 'El meu color preferit és ' . $color . '.';
?>
```
</details>

Exercici 8: Combinació de funcions i arrays
1. Crea una funció `afegir_element` que prenga un array i un element com a arguments, afegei
2. Crea un fitxer on defnisques un array `$animals` amb els elements "gat" i "gos". Utilitza

<details>
<summary>Solució</summary>

```php
<?php
// Exercici 8
function afegir_element($array, $element) {
    $array[] = $element;
    return $array;
```

```
    }
    $animals = array("gat", "gos");
    $animals = afegir_element($animals, "conill");
    foreach ($animals as $animal) {
        echo $animal . "<br>";
    }
    ?>
    ```
</details>

Exercici 9: Utilitzant `match` per a categoritzar

Crea un fitxer que utilitze la instrucció `match` per categoritzar una variable `$nota` seg
     - Si la nota és 10, imprimir "Excel·lent".
     - Si la nota és 8 o 9, imprimir "Molt bé".
     - Si la nota és 5, 6 o 7, imprimir "Bé".
     - Per qualsevol altra nota, imprimir "Insuficient".

<details>
<summary>Solució</summary>

```php
$nota = 8;

$resultat = match (true) {
    $nota === 10 => 'Excel·lent',
    $nota >= 8 && $nota <= 9 => 'Molt bé',
    $nota >= 5 && $nota <= 7 => 'Bé',
    default => 'Insuficient',
};

echo $resultat;  // Sortida: Molt bé
```

Exercici 10: Llista de preus amb `match` Crea un fitxer que utilitze la instrucció `match` per assignar un preu a una variable `$producte`. Els productes i preus són: - “pa” => 1.00 - “llet” => 0.80 - “formatge” => 2.50 - Qualsevol altre producte => 0.00

Solució

```php
$producte = 'formatge';

$preu = match ($producte) {
    'pa' => 1.00,
    'llet' => 0.80,
    'formatge' => 2.50,
```

```
    default => 0.00,
};

echo "El preu de $producte és $preu euros.";  // Sortida: El preu de formatge és 2.5 euros.
```

Exercici 11: Calculadora simple amb `match` Crea un fitxer que utilitze la instrucció `match` per fer operacions matemàtiques bàsiques (`+`, `-`, `*`, `/`). La variable `$operacio` ha de determinar l'operació a realitzar i les variables `$a` i `$b` seran els operands.

Solució

```
$a = 10;
$b = 5;
$operacio = '+';

$resultat = match ($operacio) {
    '+' => $a + $b,
    '-' => $a - $b,
    '*' => $a * $b,
    '/' => $a / $b,
    default => 'Operació desconeguda',
};

echo "El resultat de $a $operacio $b és $resultat.";  // Sortida: El resultat de 10 + 5 és
```

Exercici 12: Tractament de formulari

1. Crea un formulari en HTML que permeti als usuaris introduir el seu nom i edat. Després de l'enviament del formulari, mostra una pàgina PHP que processi les dades introduïdes i mostri un missatge de benvinguda personalitzat.

Solució

```
<!DOCTYPE html>
<html lang="ca">
<head>
    <meta charset="UTF-8">
    <title>Formulari de Benvinguda</title>
</head>
<body>
    <h2>Formulari de Benvinguda</h2>
    <form action="benvinguda.php" method="post">
        <label for="nom">Nom:</label>
        <input type="text" id="nom" name="nom" required><br><br>
        <label for="edat">Edat:</label>
        <input type="number" id="edat" name="edat" required><br><br>
```

```
        <input type="submit" value="Enviar">
    </form>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="ca">
<head>
    <meta charset="UTF-8">
    <title>Benvinguda</title>
</head>
<body>
    <?php
        if ($_SERVER["REQUEST_METHOD"] == "POST") {
            $nom = htmlspecialchars($_POST['nom']);
            $edat = htmlspecialchars($_POST['edat']);
            echo "<h2>Benvingut/da, $nom!</h2>";
            echo "<p>Tens $edat anys.</p>";
        } else {
            echo "<p>Si us plau, completa el formulari.</p>";
        }
    ?>
</body>
</html>
```

Exercici 13: Formulari en la mateixa pàgina

1. Crea un formulari en HTML que permeta als usuaris introduir la seva adreça de correu electrònic i un missatge. Després de l'enviament del formulari, crea una pàgina PHP que processe les dades introduïdes, comprove que l'adreça de correu electrònic és vàlida i mostre el missatge introduït per l'usuari.

Solució

```
<!DOCTYPE html>
<html lang="ca">
<head>
    <meta charset="UTF-8">
    <title>Formulari de Contacte</title>
</head>
<body>
    <?php
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $email = htmlspecialchars($_POST['email']);
        $missatge = htmlspecialchars($_POST['missatge']);
```

```
        if (filter_var($email, FILTER_VALIDATE_EMAIL)) {
            echo "<h2>Gràcies per contactar-nos!</h2>";
            echo "<p>El teu correu electrònic: $email</p>";
            echo "<p>El teu missatge: $missatge</p>";
        } else {
            echo "<p>Adreça de correu electrònic no vàlida. Si us plau, torna-ho a intentar
        }
    } else {
        ?>
        <h2>Formulari de Contacte</h2>
        <form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>" method="post">
            <label for="email">Correu electrònic:</label>
            <input type="email" id="email" name="email" required><br><br>
            <label for="missatge">Missatge:</label><br>
            <textarea id="missatge" name="missatge" rows="4" cols="50" required></textarea><
            <input type="submit" value="Enviar">
        </form>
        <?php
    }
    ?>
</body>
</html>
```

Exercici 14: Validació de formulari amb `match`

1. Crea un fitxer que utilitze la instrucció `match` per validar un formulari amb camps per a nom, correu electrònic i edat. Si algun camp està buit, ha de retornar un missatge d'error corresponent.

Solució

```
$nom = 'Joan';
$correu = 'joan@example.com';
$edat = '';

$validacio = match (true) {
empty($nom) => 'El camp nom és obligatori.',
!filter_var($correu, FILTER_VALIDATE_EMAIL) => 'El correu electrònic no és vàlid.',
empty($edat) => 'El camp edat és obligatori.',
default => 'Formulari vàlid.',
};

echo $validacio;  // Sortida: El camp edat és obligatori.
```

Exercici 15: Variables de servidor

1. Mostra en un fitxer les variables de servidor que conegues

Solució

```
<!DOCTYPE html>
<html lang="ca">
<head>
    <meta charset="UTF-8">
    <title>Informació del Servidor</title>
</head>
<body>
    <h2>Informació del Servidor</h2>
    <?php
        echo "<p><strong>Nom del servidor:</strong> " . $_SERVER['SERVER_NAME'] . "</p>";
        echo "<p><strong>Adreça IP del servidor:</strong> " . $_SERVER['SERVER_ADDR'] . "</p
        echo "<p><strong>Software del servidor:</strong> " . $_SERVER['SERVER_SOFTWARE'] . "
        echo "<p><strong>Agent d'usuari del client:</strong> " . $_SERVER['HTTP_USER_AGENT']
        echo "<p><strong>Mètode de la sol·licitud:</strong> " . $_SERVER['REQUEST_METHOD'] .
        echo "<p><strong>URL de la sol·licitud:</strong> " . $_SERVER['REQUEST_URI'] . "</p>
        echo "<p><strong>Referent:</strong> " . (isset($_SERVER['HTTP_REFERER']) ? $_SERVER[
        echo "<p><strong>Protocol utilitzat:</strong> " . $_SERVER['SERVER_PROTOCOL'] . "</p
        echo "<p><strong>Port utilitzat:</strong> " . $_SERVER['SERVER_PORT'] . "</p>";
    ?>
</body>
</html>
```

Exercici 16: Pujar fitxers al servidor

1. Crea un formulari en HTML que permeti als usuaris pujar un fitxer i seleccionar una opció d'un checkbox. Les opcions del checkbox han de ser carregades des d'un array predefinit en PHP. Després de l'enviament del formulari, el fitxer pujat ha de ser processat i mogut a una ubicació específica del servidor, i s'ha de mostrar la informació del fitxer i l'opció seleccionada.

Solució

```
<!DOCTYPE html>
<html lang="ca">
<head>
    <meta charset="UTF-8">
    <title>Pujar Fitxer i Selecció Opció</title>
</head>
<body>
    <?php
    // Definim les opcions per al checkbox
    $opcions = ["Opció 1", "Opció 2", "Opció 3"];
```

```php
// Comprovem si el formulari ha estat enviat
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Comprovem si el fitxer ha estat pujat sense errors
    if (isset($_FILES["fitxer"]) && $_FILES["fitxer"]["error"] == 0) {
        $nom_fitxer = $_FILES["fitxer"]["name"];
        $tipus_fitxer = $_FILES["fitxer"]["type"];
        $mida_fitxer = $_FILES["fitxer"]["size"];
        $ubicacio_temporal = $_FILES["fitxer"]["tmp_name"];

        // Movem el fitxer a una ubicació permanent
        $ubicacio_destinacio = "uploads/" . basename($nom_fitxer);
        if (move_uploaded_file($ubicacio_temporal, $ubicacio_destinacio)) {
            echo "<p>El fitxer <strong>$nom_fitxer</strong> ha estat pujat correctament.
            echo "<p>Tipus de fitxer: $tipus_fitxer</p>";
            echo "<p>Mida del fitxer: " . ($mida_fitxer / 1024) . " KB</p>";
            echo "<p>Ubicació del fitxer: $ubicacio_destinacio</p>";
        } else {
            echo "<p>Error al moure el fitxer a la ubicació final.</p>";
        }
    } else {
        echo "<p>Error al pujar el fitxer.</p>";
    }

    // Comprovem si una opció del checkbox ha estat seleccionada
    if (isset($_POST['opcio'])) {
        $opcio_seleccionada = $_POST['opcio'];
        echo "<p>Has seleccionat: $opcio_seleccionada</p>";
    } else {
        echo "<p>No has seleccionat cap opció.</p>";
    }
} else {
    ?>
    <h2>Formulari per Pujar Fitxer i Selecció d'Opció</h2>
    <form action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>" method="post" e
        <label for="fitxer">Selecciona un fitxer:</label>
        <input type="file" id="fitxer" name="fitxer" required><br><br>

        <label for="opcio">Selecciona una opció:</label><br>
        <?php
        foreach ($opcions as $opcio) {
            echo '<input type="checkbox" id="' . $opcio . '" name="opcio" value="' . $op
            echo '<label for="' . $opcio . '"> ' . $opcio . '</label><br>';
        }
        ?><br>

        <input type="submit" value="Enviar">
```

```
        </form>
        <?php
    }
    ?>
</body>
</html>
```

Exercicis proposats

Exercici 1: Manipulació de Variables i Operadors Assigna múltiples variables i utilitza operadors aritmètics i lògics. Mostra el resultat de cada operació.

Exercici 2: Control de Flux amb Bucles Utilitza un bucle `for` per imprimir els números parells del 0 al 20. Fes-ho també amb un bucle `while`.

Exercici 3: Treballar amb Arrays i Funcions Escriu una funció que prenga un array de números, calculi la mitjana i retorni el resultat. Utilitza aquesta funció per imprimir la mitjana d'un array de cinc números.

Exercici 4: Manipulació de Strings Escriu un script que prenga una cadena de text i compti el nombre de vocals. Imprimeix el resultat.

Exercici 5: Arrays Multidimensionals Crea un array multidimensional que represente una taula de multiplicar del 1 al 5 i imprimeix els resultats en forma de taula.

Exercici 6: Utilitzant `match` per a categoritzar Crea un fitxer que utilitze la instrucció `match` per categoritzar una variable `$nota` segons el següent criteri: - Si la nota és 10, imprimir "Excel · lent". - Si la nota és 8 o 9, imprimir "Molt bé". - Si la nota és 5, 6 o 7, imprimir "Bé". - Per qualsevol altra nota, imprimir "Insuficient".

Exercici 7: Validació de Formularis Crea un formulari en HTML que permetis als usuaris introduir el seu nom, el correu electrònic i un password (dues vegades). Després de l'enviament del formulari, valida que tots els camps han estat completats i que el correu electrònic és vàlid, que el password tinga complexitat i que coincidixen. Mostra un missatge d'error si alguna validació falla, i si tot és correcte, mostra un missatge confirmant que l'usuari s'ha registrat correctament.

Exercici 8: Processament de Formularis amb Select i Radio Buttons A partir del formulari newBook.php, fes que el mòdul i el estat els agafe de valors introduïts en arrays. Mostra el resultat en una taula.

Exercici 9: Pujar imatges des de formulari A partir del formulari anterior fes que es puga pujar una imatge. Mostra la imatge en la taula.

Solucions

20. Enunciats dels Projectes

Projecte "L'Ofegat"

Enunciat: Implementa una versió simplificada del joc "L'Ofegat" utilitzant HTML i PHP. El joc ha de permetre als jugadors endevinar les lletres d'una paraula predefinida i mostrar l'estat actual de les lletres endevinades. No és necessari mantenir l'estat del joc entre sol · licituds ni comprovar si s'han esgotat els intents en aquesta fase inicial.

Requisits:

1. **Paraula a Endevinar:**
   - Defineix una paraula predefinida a endevinar.
2. **Inicialització de les Lletres Endevinades:**
   - Crea un array amb guions baixos per representar les lletres que l'usuari ha de trobar.
   - Crea una funció per imprimir este array i que serà el que ens mostre el progres del joc. **(les funcions aniran en un fitxer a part)**
3. **Funció per a Comprovar Intents:**
   - Crea una funció en PHP que prenga la paraula a endevinar, la lletra introduïda per l'usuari i l'array de lletres endevinades (passat per referència).
   - La funció ha de comprovar si la lletra introduïda per l'usuari forma part de la paraula.
   - Si la lletra és correcta, la funció substituirà els guions baixos per la lletra corresponent.
   - La funció retornarà un valor booleà que indique si la lletra és errònia o no.
   - Fes proves de la funció per comprovar que funciona correctament.
4. **Comprovació d'Intents:**
   - Utilitza la funció creada per comprovar si la lletra introduïda per l'usuari forma part de la paraula.
   - Mostra un missatge d'error si la lletra no és correcta.
5. **Interfície d'Usuari:**
   - Mostra les lletres introduïdes per l'usuari fins al moment en color verd si són encertades o roig si són errades.
   - Mostra les errades de l'usuari.
   - Crea un formulari HTML que permeta als jugadors introduir una lletra.

Codi CSS Proporcionat:

```
.correct { color: green; }
.incorrect { color: red; }
```

Projecte "4 en Ratlla"

Enunciat: Implementa una versió simplificada del joc "4 en Ratlla" utilitzant HTML i PHP. El joc ha de permetre als jugadors introduir els seus moviments i mostrar l'estat actual de la graella. No és necessari mantenir l'estat del joc entre sol · licituds ni comprovar si hi ha un guanyador en aquesta fase inicial.

Requisits:

1. **Inicialització de la Graella**:
   - Crea una funció `inicialitzarGraella()` que inicialitze una graella buida de 6 files i 7 columnes.
2. **Pintar la Graella**:
   - Crea una funció `pintarGraella($graella)` que pinte la graella en HTML. Utilitza diferents colors per a les fitxes dels jugadors, aplicant el CSS adjunt.
3. **Realitzar Moviments**:
   - Crea una funció `ferMoviment(&$graella, $columna, $jugadorActual)` que realitze un moviment en la columna especificada pel jugador actual.
4. **Interfície d'Usuari**:
   - Crea un formulari HTML que permeta als jugadors introduir la columna on volen posar la seua fitxa. Aquest formulari ha de mantenir el jugador actual entre sol · licituds.

Codi CSS Proporcionat:

```
table { border-collapse: collapse; }
td {
    width: 50px;
    height: 50px;
    border-radius: 50%;
    border: 10px dotted #fff; /* Cercle amb punts blancs */
    background-color: #000; /* Fons negre o pot ser un altre color */
    display: inline-block;
    margin: 10px;
    color: white;
    font-size: 2rem;
    text-align: center ;
    vertical-align: middle;
}
.player1 {
    background-color: red; /* Color vermell per un dels jugadors */
}
.player2 {
```

```
    background-color: yellow; /* Color groc per l'altre jugador */
}
.buid {
    background-color: white; /* Color blanc per cercles buits */
    border-color: #000; /* Puntes negres per millor visibilitat */
}
```

Rúbrica

Criteri	Puntuació Total	Complet(2)	A mitjes(1)	No(0)
Paraula a Endevinar	1	Paraula definida correctament	Paraula definida però amb errors	No s'ha definit la paraula
Inicialització de les Lletres	3	Array creat correctament	Array creat però amb errors	No s'ha creat l'array
Funció per a Comprovar Intents	5	Funció creada i funcional	Funció creada però amb errors	No s'ha creat la funció
Comprovació d'Intents	3	Comprovació realitzada correctament	Comprovació amb errors	No s'ha realitzat la comprovació
Inicialització de la Graella	4	Funció creada correctament	Funció creada però amb errors	No s'ha creat la funció
Pintar la Graella	4	Funció creada i funcional	Funció creada però amb errors	No s'ha creat la funció
Realitzar Moviments	5	Funció creada i funcional	Funció creada però amb errors	No s'ha creat la funció
Interfícies d'Usuari	6	Interfície correcta i funcional	Interfície creada però amb errors	No s'ha creat la interfície
Estil CSS	2	CSS aplicat correctament	CSS aplicat però amb errors	No s'ha aplicat el CSS
Comentaris i Claredat del Codi	1	Codi ben comentat i clar	Codi amb alguns comentaris	Codi sense comentaris o desordenat

21. Autoavaluació: Conceptes Bàsics de PHP

Exercici 1: Sintaxi Bàsica de PHP

Pregunta: Quina és la manera correcta d'iniciar i finalitzar un bloc de codi PHP dins d'un fitxer HTML?

Opcions: a) `<?php ... ?>` b) `<php> ... </php>` c) `<?php ?> ... <?php end ?>` d) `<script php> ... </script>`

Exercici 2: Variables i Tipus de Dades

Pregunta: Quina opció mostra el tipus de dades assignat a la variable `$nom`?

Opcions: a) `echo gettype($nom);` b) `print datatype($nom);` c) `echo typeof($nom);` d) `print type($nom);`

Exercici 3: Estructures de Control

Pregunta: Quina és la sortida de l'estructura de control següent?

```
$numero = 10;
if ($numero > 5) {
    echo "Major que 5";
} else {
    echo "Menor o igual a 5";
}
```

Opcions a) Major que 5 b) Menor o igual a 5 c) No imprimeix res d) Error de sintaxi

Exercici 4: Operadors en PHP

Pregunta: Quin serà el valor de `$resultat` després d'executar el següent codi?

```
$resultat = 5 + 2 * 3;
```

Opcions: a) 21 b) 11 c) 17 d) 13

Exercici 5: Arrays en PHP

Pregunta: Com es pot afegir un element al final d'un array en PHP?

Opcions: a) array_add($array, $element); b) $array[] = $element;< /label >< br >< inputtype = "radio"id = "q5c"name = "question5"value = "c" >< labelfor = "q5c" > c)append(array, element);< /label >< br >< inputtype = "radio"id = "q5d"name = "question5"value = "d" >< labelfor = "q5d" > d)add_to_array(array, $element);

Exercici 6: Funcions en PHP

Pregunta: Com es defineix una funció en PHP?

Opcions: a) function: myFunction() { … } b) def myFunction() { … } c) function myFunction() { … } d) fn myFunction() => { … }

Exercici 7: Instrucció switch

Pregunta: Quina instrucció s'utilitza per detenir l'execució d'un cas dins d'un switch en PHP?

Opcions: a) stop b) exit c) halt d) break