WEB

PHP

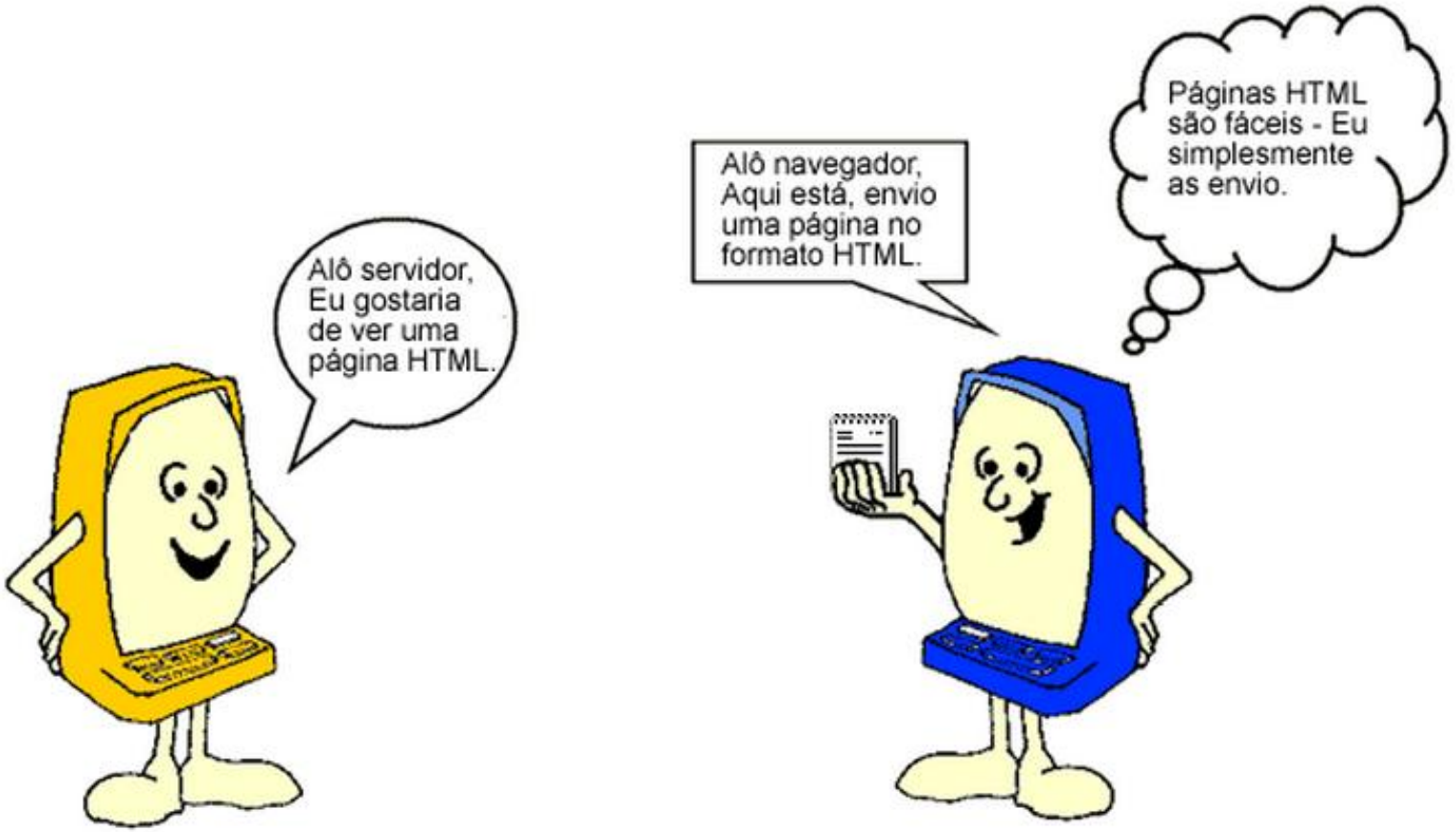
Profa. Joyce Miranda

PHP - *PHP: Hypertext Preprocessor*

- Surgiu em 1995
- É utilizada no desenvolvimento de páginas Web dinâmicas de forma simples e eficiente.
- Algumas características
  - Multiplataforma
  - *Open Source*
  - Robusta
  - *Script server-side*
  - Interpretada e não compilada
  - Permite programação estruturada e orientada a objetos

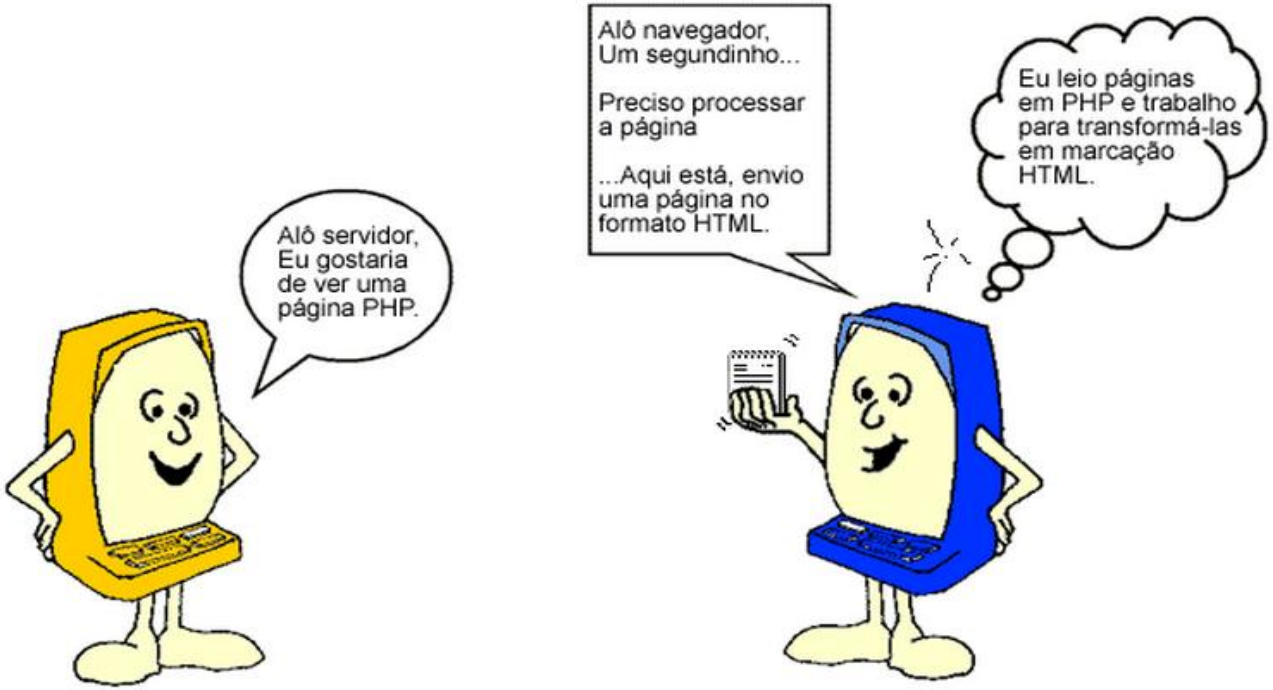
PHP

- Como funciona?
  - Requisitando uma página HTML : http://www.meusite.com/page.html



PHP

- Como funciona?
  - Requisitando uma página PHP: http://www.meusite.com/page.php



  - O servidor executa as tarefas e retorna um resultado HTML
  - Assim, não é possível visualizar os scripts PHP inseridos em uma página

PHP

- Do que você precisa para começar?
  - Navegador
  - PHP
  - Servidor WEB (Apache)
  - Banco de Dados (MySQL)
  - Pacotes pré-configurados
    - **WAMP - http://www.wampserver.com**
    - EasyPHP - http://www.easyphp.org
    - XAMPP - http://www.apachefriends.org

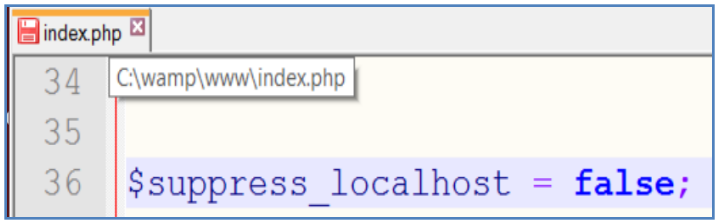
PHP

- Hello PHP!!
  - Conteúdo de uma página PHP
    - Tags HTML
    - Conteúdo
    - Código PHP

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
            echo "HELLO PHP !!";
        ?>
    </body>
</html>
```

PHP

- Hello PHP!!
  - Inicie o servidor WEB
  - Crie uma pasta para seu site dentro do servidor
  - Crie seu código
  - Salve o arquivo com a extensão “.php”. Ex: hello.php
  - Acesse plo browser o arquivo pelo endereço de seu servidor web
    - http://localhost/hello.php
    - http://127.0.0.1/hello.php




```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
            echo "HELLO PHP !!";
        ?>
    </body>
</html>
```

PHP

- Sintaxe
  - Tags de abertura e fechamento

```
1.  <?php echo 'se você quer servir documentos XHTML ou XML, faça assim'; ?>

2.  <script language="php">
        echo 'alguns editores (como o FrontPage) não
              gostam de instruções de processamento';
    </script>

3.  <? echo 'esta é a mais simples, uma instrução de processamento SGML'; ?>
    <?= expressão ?> Isto é um atalho para "<? echo expressão ?>"

4.  <% echo 'Você pode opcionalmente usar tags no estilo ASP'; %>
    <%= $variavel; # Isto é um atalho para "<% echo . . ." %>
```

PHP

- Sintaxe
  - Tags de abertura e fechamento
    - Alternativa para o echo e print
      - Grandes blocos de texto

```
<?php
  if (true){
?>
    <strong>Entrou aqui</strong>
<?php
  }else{
?>
    <strong>Não entrou aqui</strong>
<?php
  }
?>
```

PHP

- Sintaxe
  - Comentários

```php
<?php
    echo 'Isto é um teste'; // Estilo de comentário de uma linha em  c++
    /* Este é um comentário de múltiplas linhas
       ainda outra linha de comentário */
    echo 'Isto é ainda outro teste';
    echo 'Um teste final'; # Este é um comentário de uma linha no estilo shell
?>
```

PHP

- Sintaxe
  - Variáveis
    - Tipagem dinâmica
    - Tipos básicos:
      - boolean
      - integer
      - float (ou também double)
      - string

```
$a_bool = TRUE;   // um booleano
$a_str  = "foo"; // uma string
$a_str2 = 'foo'; // uma string
$an_int = 12;     // um inteiro
```

PHP

- Sintaxe
  - Operadores de Atribuição

Atribuição	O mesmo que..
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

PHP

- Sintaxe
  - Operadores Aritméticos

Operador	Nome	Exemplo
+	Adição	$x + $y
-	Subtração	$x - $y
*	Multiplicação	$x * $y
/	Divisão	$x / $y
%	Módulo	$x % $y
**	Exponenciação	$x ** $y

- Sintaxe
  - Operadores de Incremento e Decremento

Operador	Nome
++$x	Pré-incremento
$x++	Pós-incremento
--$x	Pré-decremento
$x--	Pós-decremento

- Sintaxe
  - Operadores de Comparação

Operador	Nome	Exemplo
==	Igual	$x == $y
===	Idêntico	$x === $y
!=	Diferente	$x != $y
<>	Diferente	$x <> $y
!==	Não Idêntico	$x !== $y
>	Maior	$x > $y
<	Menor	$x < $y
>=	Maior ou igual	$x >= $y
<=	Menor ou igual	$x <= $y

PHP

- Sintaxe
  - Operadores Lógicos

Operador	Nome	Exemplo
and	And	$x and $y
or	Or	$x or $y
xor	Xor	$x xor $y
&&	And	$x && $y
||	Or	$x || $y
!	Not	!$x

PHP

- Sintaxe
  - Funções Úteis
    - **gettype()**
      - Retorna o tipo de uma variável

```php
$a_bool = TRUE;   // um booleano
$a_str  = "foo";  // uma string
$a_str2 = 'foo';  // uma string
$an_int = 12;     // um inteiro
```

```php
echo gettype($a_bool); // mostra:  boolean
echo gettype($a_str);  // mostra:  string
```

PHP

- Sintaxe
  - Funções Úteis
    - **var_dump()**
      - Retorna o tipo e valor de uma **expressão**

```
$b = 3.1;
$c = true;
var_dump($b,$c);
```

```
float(3.1)
bool(true)
```

```
$an_int = 1 ;
```

```
echo var_dump($an_int==13); //boolean false
```

PHP

- Sintaxe
  - Funções Úteis
    - **is_tipo()**
      - Verifica se uma variável é de um certo tipo

```
$an_int = 12;
```

```
// Se ele é um inteiro, incrementa-o com quatro
if (is_int($an_int)) {
    $an_int += 4;
}
```

```
$a_bool = TRUE;
```

```
// Se $bool é uma string, mostre-a
// (não imprime nada)
if (is_string($a_bool)) {
    echo "String: $a_bool";
}
?>
```

PHP

- Sintaxe
  - **Conversão de Tipos**
    - **cast**

(int), (integer)
(bool), (boolean)
(float), (double), (real)
(string)
(binary)
(array)
(object)
(unset)

```
$y = 2.5;
$y = (int) $y;
var_dump($y);
```

```
int 2
```

PHP

- Sintaxe
  - Strings
    - **Conversão**
      - Na conversão para string basta incluir aspas duplas

```
$foo = 10;              // $foo é um interio
$str = "$foo";          // $str é uma string
```

PHP

- Sintaxe
  - Operadores de String

Operador	Nome	Exemplo	Exemplo
.	Concatenação	$txt1 = "Hello" $txt2 = $txt1 . " world!"	"Hello world!"
.=	Atribuição de Concatenação	$txt1 = "Hello" $txt1 .= " world!"	"Hello world!"

PHP

- Sintaxe
  - Strings
    - **Concatenando**

```php
$num = 100;
$numDesc = "cem";
echo "O número é $num : $numDesc ";
//O número é 100 : cem
```

```php
echo "O número é ".$num." : ".$numDesc;
```

PHP

- Sintaxe
  - Constantes
    - Uma vez definida, jamais alterada.

```
define($nomedaconstante,$valordaconstante,$case_sensitive);
```

```php
define('CONSTANTE','Ola mundo');
echo CONSTANTE; //resposta sera: Ola Mundo
echo Constante; // erro
```

PHP

- Sintaxe
  - Constantes
    - Verificando existência

```php
define('CONSTANTE','10');

if(defined('CONST'))
{
    echo 'esta definida';
}
else
{
    echo 'nao esta definida';
}
```

PHP

- Sintaxe
  - Funções

PHP

- Sintaxe
  - Funções

```php
// Função sem parâmetros e sem retorno
function exibe_frase () {
    echo 'Eu sou uma função que executa a ação.';
}

// Chama a função
exibe_frase();
```

PHP

- Sintaxe
  - Funções

```php
// Criei uma função para imprimir o nome de alguém,
// então também criei o parâmetro $nome, para passar
// o nome de alguém para dentro da função.
function ola($nome) {
    print 'oi ' . $nome . '!';
}

// Agora eu executo a função passando o nome de alguém para dentro dela.
// Aqui a função imprime 'oi Eduardo!'
ola('Eduardo');
```

PHP

- Sintaxe
  - Funções

```php
<?php

function retornaMaior($valor1, $valor2){
        if( $valor1 > $valor2){
            return $valor1;
        }else{
            return $valor2;
        }
}
echo retornaMaior(10, 5);

?>
```

PHP

- Sintaxe
  - Funções

```php
<?php

function retornaMaior($valor1, $valor2){
        if( $valor1 > $valor2){
            return $valor1;
        }else{
            return $valor2;
        }
}
echo retornaMaior(10, 5);

?>
```

PHP

- Pratique!
  - Crie uma função que vai receber como parâmetro o ano de nascimento de uma pessoa e vai retornar a idade da pessoa.
    - *calculaIdade(1915)*
    - *Idade: 100 anos*
  - Execute a função criada.

```
//recuperar ano atual
$anoAtual = date("Y");
```

- Sintaxe
  - Escopo das Variáveis
    - As variáveis utilizadas dentro da função são por default limitada dentro do escopo local da função.

```php
// Uma variável fora da função
$variavel_externa = 'Valor';

// Uma função
function erro () {
    // Variável local
    $variavel_local = 'Valor';

    // Tenta acessar a variável externa
    echo $variavel_externa;
}

// Chama a função e obtém um erro:
// Undefined variable: variavel_externa
erro();

// Tenta acessar a variável local da função erro()
// Gera outro erro: Undefined variable: variavel_local
echo $variavel_local;
```

PHP

- Sintaxe
  - Escopo das Variáveis
    - As variáveis globais precisam ser declaradas globais dentro de uma função se for utilizada naquela função..

```php
// Uma variável fora da função
$variavel_externa = 'Valor externo';

// Uma função
function certo () {
    // A variável abaixo estará acessível dentro e fora da função
    global $variavel_global;

    // A variável externa também estará acessível na função
    global $variavel_externa;

    // Define o valor da variável global
    $variavel_global = 'Valor local';

    // Exibe a variável externa
    echo $variavel_externa;
}

// Chama a função e exibe a variável externa
certo();

// Acessa a variável criada dentro da função
echo $variavel_global;
```

PHP

- Sintaxe
  - Escopo das Variáveis
    - Usando **$GLOBALS** no lugar de global

```php
<?php
$a = 1;
$b = 2;

function Soma()
{
    $GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b'];
}

Soma();
echo $b;
?>
```

PHP

SUPERGLOBAIS	
$GLOBALS	Referencia todas as variáveis disponíveis no escopo global
$_SERVER	Informação do servidor e ambiente de execução
$_GET	HTTP GET variáveis
$_POST	HTTP POST variáveis
$_FILES	HTTP File Upload variáveis
$_REQUEST	Variáveis de requisição HTTP
$_SESSION	Variáveis de sessão
$_ENV	Variáveis de ambiente
$_COOKIE	HTTP Cookies variáveis
$php_errormsg	A mensagem de erro anterior
$argc	O número de argumentos passado para o script
$argv	Array de argumentos passados para o script

PHP

- Sintaxe
  - Qual será o resultado?

```php
function andar ( $distancia ) {
    // Define que a variável é estática
    $soma;

    // Soma a distancia
    $soma += $distancia;

    // Retorna a distancia
    return $soma;
}
```

```php
echo andar(10);
echo andar(100);
echo andar(1000);
```

- Sintaxe
  - Variáveis estáticas
    - Existe somente no escopo local da função, mas ela não perde seu valor quando o nível de execução do programa deixa o escopo.

```php
function andar ( $distancia ) {
    // Define que a variável é estática
    static $soma;

    // Soma a distancia
    $soma += $distancia;

    // Retorna a distancia
    return $soma;
}
```

```php
echo andar(10);
echo andar(100);
echo andar(1000);
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Letras Maiúsculas e Minúsculas
    - **strtoupper**
      - Converte as letras de um texto para letras maiúsculas.

```php
$nome = 'Eduardo Monteiro';
$nome= strtoupper($nome);

// imprime EDUARDO MONTEIRO
print $nome;
```

PHP

- Sintaxe
  - Funções Internas - Strings
  - Letras Maiúsculas e Minúsculas
    - **strtolower**
      - Converte as letras de um texto para letras minúsculas.

```php
$nome = 'Eduardo Monteiro';
$nome = strtolower($nome);

// imprime eduardo monteiro
print $nome;
```

PHP

- Sintaxe
  - Funções Internas - Strings
  - Letras Maiúsculas e Minúsculas
    - **ucwords**
      - Converte a primeira letra de cada palavra de um texto para maiúscula.

```php
$nome = 'eduardo monteiro';
$nome = ucwords($nome);

// imprime Eduardo Monteiro
print $nome;
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Retirar espaços
    - **ltrim**
      - Retira os espaços em branco localizados no lado esquerdo de um texto.

```php
$texto = " texto com espaços";
$texto = ltrim($texto);

// imprime "texto com espaços"
print $texto;
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Retirar espaços
    - **rtrim**
      - Retira os espaços em branco localizados no lado direito de um texto.

```
$texto = "texto com espaços ";
$texto = rtrim($texto);

// imprime "texto com espaços"
print $texto;
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Retirar espaços
    - **trim**
      - Retira os espaços encontrados em ambos os lados de um texto.

```
$texto = " texto com espaços ";
$texto = trim($texto);

// imprime "texto com espaços"
print $texto;
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Acrescentar caracteres
    - **str_repeat**
      - Retorna um texto criado a partir de outro texto repetido várias vezes.

```
// imprime =-==-==-==-==-==
print str_repeat('=-=', 5);
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Genéricas
    - **strlen**
      - Retorna um inteiro que indica o tamanho de um texto.

```php
$texto = "meu texto qualquer";

print strlen($texto); // imprime 18
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Genéricas
    - **strpos**
      - Retorna a posição da primeira ocorrência de um texto dentro de outro.

$texto = “maria antonia augusta de oliveira”
$pos = strpos($texto, “augusta”)

```
print $pos; // imprime 15
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Genéricas
    - **strrpos**
      - Retorna a posição da última ocorrência de um texto dentro de outro.

$texto="maria antonia augusta de oliveira"

```
$pos = strrpos($texto, "i");

print $pos; // imprime 31
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Genéricas
    - **substr**
      - Retorna parte de um texto.
      - Argumentos
        - 1º : Texto
        - 2º : Posição Inicial
        - 3º : Tamanho do texto extraído
      - Valores negativos iniciam o recorte do final

```
$texto = "abcdefghij";

print substr($texto, 5) . "<br>";
print substr($texto, 6, 4) . "<br>";
print substr($texto, -1) . "<br>";
print substr($texto, -3, 3) . "<br>";
```

```
fghij
ghij
j
hij
```

PHP

- Sintaxe
  - Funções Internas – Strings
  - Genéricas
    - **str_replace**
      - Substitui um conjunto de caracteres dentro de uma string

```php
$texto = "Eu vou para a escola";

// imprime "Eu não vou para a escola"
print str_replace("vou", "não vou", $texto);

// imprime "Você vai para a escola"
print str_ireplace("eu vou", "Você vai", $texto);
```

    - **str_ireplace**
      - Semelhante ao *str_replace,* só que não faz distinção entre letras maiúsculas e letras minúsculas.

PHP

- Sintaxe
  - Strings - Funções – Diversas funcionalidades
    - **strrev**
      - Reverte uma string, deixando-a ao contrário.

```
$texto = "oi mundo!!";

// imprime "!!odnum io"
print strrev($texto);
```

PHP

- Pratique!
  - Crie uma função que receba uma data no formato “dd/mm/yyyy” e imprima o dia, o mês e o ano de forma isolada.
    - *printDate(“02/10/2015”)*
      - *Dia: 02*
      - *Mês: 10*
      - *Ano: 2015*

PHP

- Sintaxe
  - Arrays
    - Criado com o construtor de linguagem array()

```
$vetor = array("x", "y", "z", "w");
print_r($vetor);
```

```
Array ( [0] => x [1] => y [2] => z [3] => w )
```

PHP

- Sintaxe
  - Arrays
    - Reconhece pares separados por vírgula ***chave => valor*** .

```
$vetor = array(0=>"a",1=>"b",2=>"c",3=>"d");
print_r($vetor);
```

```
Array ( [0] => a [1] => b [2] => c [3] => d )
```

PHP

- Sintaxe
  - Arrays
    - Bidimensional

```php
$matriz = array(0=>array(0=>"a", 1=>"b"),
                1=>array(0=>"c", 1=>"d"));

echo "matriz[0][0] = ".$matriz[0][0]."<br>";
echo "matriz[0][1] = ".$matriz[0][1]."<br>";
echo "matriz[1][0] = ".$matriz[1][0]."<br>";
echo "matriz[1][1] = ".$matriz[1][1]."<br>";
```

PHP

- Sintaxe
  - Funções Internas – Arrays
  - Genéricas
    - **in_array**
      - Verifica se um determinado valor existe em alguma posição do array.

```php
$array = array('pêra', 'uva', 'melão');

if(in_array('uva', $array)){
    echo 'Valor encontrado';
}
//Escreve: "Valor encontrado"
```

- Sintaxe
  - Funções Internas – Arrays
  - Genéricas
    - **array_count_values**
      - Retorna um novo array onde os índices são os valores do array passado como parâmetro e os valores são o número de ocorrências de cada valor.

```php
$array = array('laranja', 'banana', 'maçã', 'laranja');
print_r(array_count_values($array));
/* Retorna
(
    [laranja] => 2
    [banana] => 1
    [maçã] => 1
)
*/
```

PHP

- Sintaxe
  - Funções Internas – Arrays
  - Genéricas
    - **array_merge**
      - Agrega o conteúdo dos dois arrays passados como parâmetro e gera um novo array com todos os valores.

```php
$array1 = array('maça', 'melão', 'goiaba');
$array2 = array('laranja', 'banana', 'melancia');

print_r(array_merge($array1, $array2));
/* Retorna
(
    [0] => maça
    [1] => melão
    [2] => goiaba
    [3] => laranja
    [4] => banana
    [5] => melancia
)
*/
```

PHP

- Sintaxe
  - Funções Internas – Arrays
  - Genéricas
    - **array_combine**
      - Mescla os dois arrays e gera um novo array onde as chaves serão os valores do primeiro array e os valores os valores do segundo array.

```php
$keys = array('nome', 'email', 'site');
$values = array('Rafael', 'rafaelwendel@hotmail.com', 'www.rafaelwendel.com');
print_r(array_combine($keys, $values));
/* Retorna
(
    [nome] => Rafael
    [email] => rafaelwendel@hotmail.com
    [site] => www.rafaelwendel.com
)
*/
```

PHP

- Sintaxe
  - Funções Internas – Arrays
  - Genéricas
    - **explode**
      - Divide um texto em pedaços, produzindo um *array* como resultado da operação.

```php
$texto = "item1;item2;item3;item4;item5";
$pedacos = explode(";", $texto);

print_r($pedacos);
```

```
Array
(
    [0] => item1
    [1] => item2
    [2] => item3
    [3] => item4
    [4] => item5
)
```

PHP

- Sintaxe
  - Funções Internas – Arrays
  - Genéricas
    - **implode**
      - Junta todos os elementos de um *array* num único texto.

```php
$lista = array("uva","melancia","banana","maçã");
$texto = implode(" - ", $lista);

// imprime "uva - melancia - banana - maçã"
print $texto;
```

PHP

- Sintaxe
  - Estruturas de controle
    - IF

```php
<?php
if ($a > $b) {
  echo "a is bigger than b";
  $b = $a;
}
?>
```

```php
<?php if ($a == 5): ?>
A is equal to 5
<?php endif; ?>
```

PHP

- Sintaxe
  - Estruturas de controle
    - ELSE

```php
<?php
if ($a > $b) {
  echo "a is greater than b";
} else {
  echo "a is NOT greater than b";
}
?>
```

```php
<?php if (condition): ?>

html code to run if condition is true

<?php else: ?>

html code to run if condition is false

<?php endif ?>
```

PHP

- Sintaxe
  - Estruturas de controle
    - ELSE IF / ELSEIF

```php
<?php
if ($a > $b) {
    echo "a is bigger than b";
} elseif ($a == $b) {
    echo "a is equal to b";
} else {
    echo "a is smaller than b";
}
?>
```

```php
if($a > $b):
    echo $a." is greater than ".$b;
elseif($a == $b): // Note the combination of the words.
    echo $a." equals ".$b;
else:
    echo $a." is neither greater than or equal to ".$b;
endif;
```

PHP

- Sintaxe
  - Estruturas de repetição
    - WHILE

```php
$i = 1;
while ($i <= 10) {
    echo $i++;  /* the printed value would be
                   $i before the increment
                   (post-increment) */
}
```

```php
$i = 1;
while ($i <= 10):
    echo $i;
    $i++;
endwhile;
?>
```

PHP

- Sintaxe
  - Estruturas de repetição
    - WHILE

```php
$arr = array("orange", "banana", "apple", "raspberry");

$i = 0;
while ($i < count($arr)) {
    $a = $arr[$i];
    echo $a ."\n";
    $i++;
}
```

PHP

- Sintaxe
  - Estruturas de repetição
    - WHILE

```php
<?php
$myArray=array('aa','bb','cc','dd');
while (list ($key, $val) = each ($myArray) ) echo $val;
reset($myArray);
while (list ($key, $val) = each ($myArray) ) echo $val;
?>
```

PHP

- Sintaxe
  - Estruturas de repetição
    - DO WHILE
      - Não há forma alternativa

```php
<?php
$i = 0;
do {
    echo $i;
} while ($i > 0);
?>
```

PHP

- Sintaxe
  - Estruturas de repetição
    - FOR

```
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
```

PHP

- Sintaxe
  - Estruturas de repetição
    - FOREACH
      - Forma fácil de iteragir sobre arrays

```
foreach (array_expression as $value)
    statement
foreach (array_expression as $key => $value)
    statement
```

PHP

- Sintaxe
  - Estruturas de repetição
    - FOREACH

```php
$arr = array(1, 2, 3, 4);
foreach ($arr as $value) {
    $value = $value * 2;
}
```

Array ([0] => 2 [1] => 4 [2] => 6 [3] => 8)

PHP

- Sintaxe
  - Estruturas de repetição
    - FOREACH

```php
$a = array(
    "one" => 1,
    "two" => 2,
    "three" => 3,
    "seventeen" => 17
);

foreach ($a as $k => $v) {
    echo "\$a[$k] => $v.\n";
}
```

PHP

- Sintaxe
  - BREAK
    - Encerra a execução de um laço de repetição

```
$arr = array('one', 'two', 'three', 'four', 'stop', 'five');
while (list(, $val) = each($arr)) {
    if ($val == 'stop') {
        break;    /* You could also write 'break 1;' here. */
    }
    echo "$val<br />\n";
}
```

PHP

- Sintaxe
  - CONTINUE
    - É usado dentro de estruturas de loop para pular o resto da iteração do loop e continuar a execução a partir da próxima iteração do loop.

```php
$stack = array('first', 'second', 'third', 'fourth', 'fifth');

foreach($stack AS $v){
    if($v == 'second')continue;
    if($v == 'fourth')break;
    echo $v.'<br>';
}
```

PHP

- Sintaxe
  - INCLUDE
    - Insere o conteúdo de um arquivo PHP dentro de outro arquivo PHP.
    - No caso de erro, NÃO GERA interrupção da execução do script.

```
vars.php
<?php

$color = 'green';
$fruit = 'apple';

?>
```

```
test.php
<?php

echo "A $color $fruit"; // A

include 'vars.php';

echo "A $color $fruit"; // A green apple

?>
```

PHP

- Sintaxe
  - REQUIRE
    - Insere o conteúdo de um arquivo PHP dentro de outro arquivo PHP.
    - No caso de erro, GERA interrupção da execução do script.

```php
<?php
require 'somefile.php';
?>
```

PHP

- Pratique!
  - Crie uma função que receba um *array* de produtos.
  - A função deve contabilizar a freqüência de cada produto e retornar o produto de maior freqüência.
    - *$carrinho = array("radio", "tv", "geladeira", "tv");*
    - *produtoMaiorFreq($carrinho)*
    - *TV: 2 unidades*

PHP

- Pratique!
  - Crie uma função que receba o nome completo de uma pessoa.
  - A função deve retornar o nome e o sobrenome da pessoa.
    - *nomeSobrenome(“Joyce Miranda dos Santos”)*
    - *Joyce Santos*