

Identifique sua apostila

Nome: _____

Turma: _____

LPII – Ap 1 PHP

Apostila 1 PHP – Disciplina LPII (PHP).

Conteúdo: Estruturas PHP. Curso:

Informática – Turmas: 72A, 72B, 72C, 72D

Prof^a Ariane Scarelli e Prof^a Kátia Zambon

2017

SUMÁRIO

1	Introdução	1
2	Editor padrão TXT	1
3	Conexão com Servidor Remoto.....	1
4	PHP	3
4.1	Tags de abertura e fechamento	3
4.2	Comentários	4
4.3	Diferenças entre maiúsculo e minúsculo (Case Sensitive)	5
4.4	Variáveis	7
4.4.1	Regras para criação e tipos de dados	7
4.4.2	Variáveis predefinidas.....	9
4.4.3	Variáveis de ambiente (\$_SERVER).....	10
4.4.4	Variáveis de formulários (\$_GET e \$_POST).....	12
4.5	Constantes.....	14
4.6	Operadores	15
4.6.1	Operadores Aritméticos.....	16
4.6.2	Operadores de Atribuição.....	16
4.6.3	Operadores de Comparação.....	17
4.6.4	Operadores Lógicos.....	17
4.6.5	Operadores de Concatenação.....	18
5	Estruturas do PHP	19
5.1	Comando if	19
5.2	Comando switch.....	21
5.3	Comando while.....	26
5.4	Comando do-while	27
5.5	Comando for	30
5.6	Comando foreach.....	31
6	Funções	35
6.1	Por que usar funções?	35
6.2	Utilizando funções	35
6.3	Retornando valores.....	37
6.4	Passagem de argumentos (valores).....	39
6.4.1	Passagem de argumentos por valor	39
6.4.2	Passagem de argumentos por referência	40
7	REFERÊNCIAS BIBLIOGRÁFICAS	42
8	SITES ÚTEIS.....	42

1 Introdução

Iniciaremos nosso estudo da linguagem de programação PHP. Neste material didático trataremos das estruturas básicas da linguagem, com exemplos e exercícios. Aprenderemos a acessar um servidor remoto com segurança.

2 Editor padrão TXT

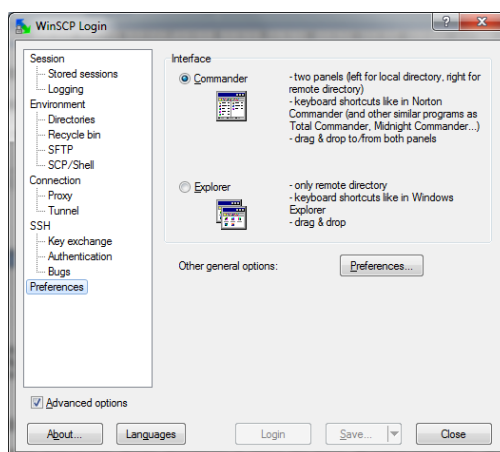
Para desenvolvermos os nossos programas PHP utilizaremos um editor do tipo texto (txt), como no HTML5. A extensão do arquivo deverá ser .php, por exemplo, "exercicio.php".

3 Conexão com Servidor Remoto

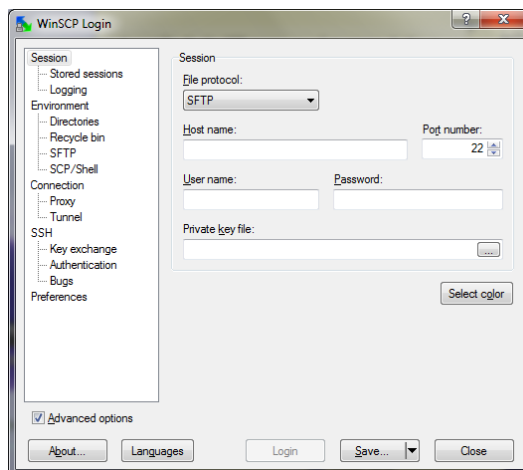
Para desenvolvermos nossas atividades utilizaremos uma conexão remota com o servidor do CTI no endereço **200.145.153.175**. No servidor, cada aluno fará o login com seu usuário e senha, que já foram criados pelo administrador.

Para ter acesso ao servidor utilizaremos o software gratuito WinSCP (**Windows Secure Copy**) que permite movimentar arquivos entre o servidor remoto e o seu computador local, como no Windows Explorer.

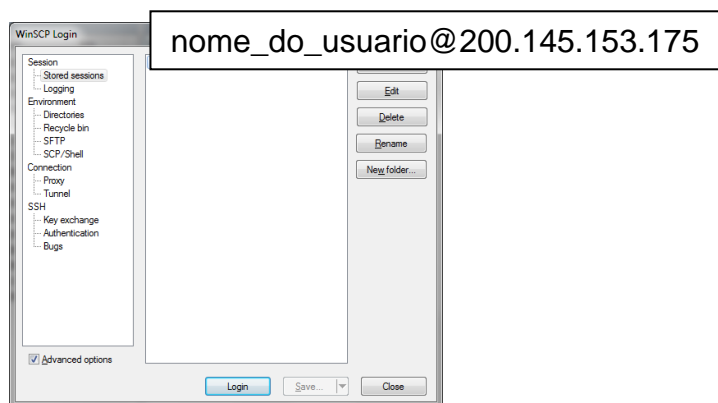
Para fazer o acesso, mantenha duas janelas abertas, uma no seu computador e a outra no servidor, deixando setada a opção "Comander" nas preferências do WinSCP. Observe na figura abaixo:



O WinSCP utiliza o protocolo de acesso seguro SSH (**Secure Shell**). Para utilizarmos o protocolo seguro, escolheremos a opção SFTP como aparece na figura abaixo:

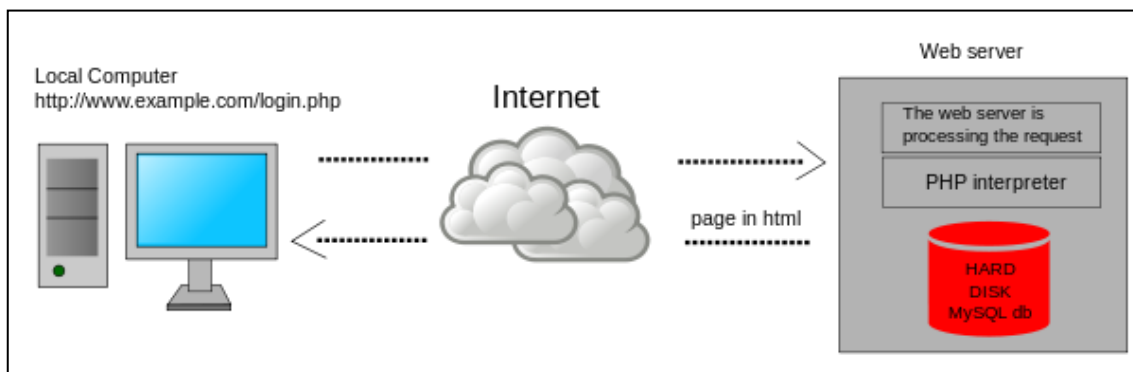


A pasta ou diretório para armazenamento dos programas de cada aluno funcionará como um disco virtual e um backup dos arquivos. Este diretório é algo do tipo: `/home/<nome_do_usuario>` com o endereço IP do servidor na tela de login:



Para que os programas sejam executados no servidor (*server side*), através da chamada pelo seu navegador local (*client side*), uma pasta diferente da *home*, com o nome *public_sites*, será o destino de armazenamento. Este diretório é algo como: `/public_sites/<nome_do_usuario>`.

A figura abaixo representa a comunicação do navegador local com o servidor. É este processo que utilizaremos para a execução dos scripts. Os scripts ficam armazenados no servidor, não nas máquinas locais.



4 PHP

PHP é um acrônimo para “PHP: Hypertext Preprocessor”, é uma linguagem de script interpretada executada no servidor, é, portanto, *server side*. É uma ferramenta *open source* poderosa para fazer páginas da Web dinâmicas e interativas rapidamente e pode ser mesclada com o HTML. É uma alternativa amplamente utilizada, livre e eficiente para concorrer com a linguagem ASP da Microsoft. Sua sintaxe lembra a do C, Java e Perl.

Scripts em PHP podem conter texto, HTML, CSS, Javascript e código PHP. A execução do programa dá-se no servidor e o resultado retorna ao navegador do usuário como um HTML simples, sendo assim, não há como saber como é o código fonte de origem. Mas o PHP não está limitado a uma saída de dados do tipo HTML, também é possível gerar saídas do tipo imagem, arquivos PDF, arquivos Flash, além de documentos XHTML e XML.

A linguagem PHP roda em várias plataformas (Windows, Linux, Unix, Mac OS X etc.), é compatível com a maioria dos servidores usados hoje em dia (Apache, IIS etc.) e suporta muitos bancos de dados.

4.1 Tags de abertura e fechamento

O script PHP pode ser inserido em qualquer parte de um documento HTML, mas todo script PHP começa com a tag **<?php** e termina com a tag **?>**.

```
<?php
    // Insira o código aqui!
?>
```

Exemplo 1 – script PHP no <body> do html5.

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>PHP aí vamos nós</title>
6   </head>
7   <body>
8     <?php
9       echo "Meu primeiro script PHP!";
10    ?>
11  </body>
12 </html>
```

Toda declaração PHP (linha de comando) deve ser terminada por ponto e vírgula (;). A tag de fechamento do PHP (?>) já implica em fechamento automático da declaração anterior que não necessita de ponto e vírgula.

```
<body>
  <?php
    echo "Esta declaração deve terminar com ponto e vírgula";
    echo "Esta declaração não precisa porque a próxima linha é a tag de fechamento do PHP"
  ?>
</body>
```

Observações:

- O exemplo 1 foi salvo com o nome exemplo1.php na pasta public_sites do servidor para o usuário ariane;
- Para executar: <http://200.145.153.175/ariane/php/exemplo1.php>

4.2 Comentários

Um comentário é um código que não é executado como parte do programa PHP. Seu propósito é apenas ser lido pelo programador na edição do script. Os comentários são úteis para: a) ajudar outros programadores a entender o que você fez em cada passo do programa, especialmente quando se trabalha em equipe; b) para lembrar a você mesmo de como solucionou determinado problema.

Os seguintes modos de comentário são suportados no PHP:

```
// → comentário de uma linha
# → comentário de uma linha
/* e */ → comentário de um bloco
```

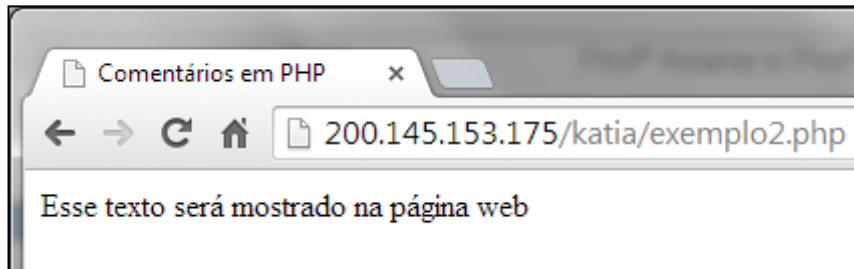
Exemplo 2 – comentários em PHP:

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8"/>
5     <title>Comentários em PHP</title>
6   </head>
7   <body>
8     <?php
9       // Este é um comentário de uma linha
10      # Este também é um comentário de uma linha
11      /* Agora, um comentário com
12        múltiplas linhas
```

```

13     echo "Esse texto não será mostrado na página web"
14     */
15     echo "Esse texto será mostrado na página web";
16     ?>
17 </body>
18 </html>

```



4.3 Diferenças entre maiúsculo e minúsculo (Case Sensitive)

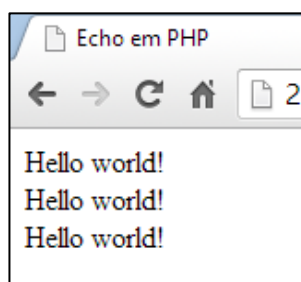
Os nomes de **funções**, **classes** e quaisquer **palavras chaves** do PHP podem ser escritos tanto em maiúsculo quanto em minúsculo, não sendo, portanto, *case sensitive*.

Exemplo 3 – comando Echo do PHP escrito de formas diferentes com o mesmo resultado:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Echo em PHP</title>
6   </head>
7   <body>
8     <?php
9       echo "Hello world! <br>";
10      ECHO "Hello world! <br>";
11      eChO "Hello world! <br>";
12    ?>
13 </body>
14 </html>

```



Os nomes de variáveis do PHP são *case sensitive*, há diferença entre letra maiúscula e minúscula.

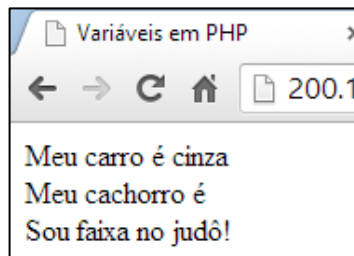
Exemplo 4 – variável do PHP escrita de formas diferentes com resultados diferentes:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Variáveis em PHP</title>
6   </head>
7   <body>
8     <?php
9       $cor="cinza";
10      echo "Meu carro é ". $cor. "<br>";
11      ECHO "Meu cachorro é ". $COR. "<br>";
12      eChO "Sou faixa ". $Cor. " no judô! <br>";
13    ?>
14  </body>
15 </html>

```

No Exemplo 4, somente a saída da linha 10 será visualizada corretamente. A variável \$cor é diferente de \$COR e de \$Cor.



EXERCÍCIOS (a numeração continua da apostila 1 – HTML)

30. Escreva um programa em PHP que contenha comentários informando o nome do autor (você) e a data da aplicação (dia de hoje). Seu programa deve apresentar com o comando ECHO seus dados pessoais: nome, cidade onde mora, e-mail e turma.
31. Escreva um programa em PHP que apresente o nome de cinco estados do Brasil e suas capitais. Cada informação deve estar em uma linha.
32. Faça um programa que apresente o número e nome do colega da posição anterior à sua na lista de chamada, seu número e seu nome e o nome do colega que ocupa a posição posterior na lista.

Exemplo:

- 09 – meu colega
- 10 – meu nome
- 11 – outro colega

33. Faça um programa que apresente o nome de um local (pode ser uma cidade ou estado no Brasil ou outro país) que você queira conhecer, uma informação geográfica deste país, uma estatística, uma comida típica, um ponto turístico e uma curiosidade.

34. Analise o script abaixo e dê como resposta (sem utilizar o interpretador PHP) o que será impresso:

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <title>Variáveis - case sensitive</title>
6  </head>
7  <body>
8      <?php
9          $valor=10;
10         $nome="teste de variáveis";
11         $selecao="Brasil";
12         echo "O time do coração de todos os brasileiros: ". $selecao. "<br>";
13         ECHO $selecao." é um país nota".$valor."?". "<br>";
14         eChO "Este é um ".$nome. "<br>";
15     ?>
16 </body>
17 </html>

```

4.4 Variáveis

Existem diversos tipos de variáveis no PHP com diferentes finalidades. Neste tópico estudaremos a respeito delas.

4.4.1 Regras para criação e tipos de dados

No PHP existem regras para a padronização da criação de variáveis, a saber:

- Uma variável sempre começa com o sinal \$ (cifrão), seguido pelo nome da variável; ex: \$Nome
- O nome da variável deve começar com uma letra ou o caractere sublinhado (_); ex: \$Cor, \$_Flor
- O nome da variável jamais deve começar com um número; ex: \$7centos
- Nomes de variáveis não podem conter espaços em branco; ex: \$minha var, \$Cod Cli

- Nomes de variáveis podem conter somente caracteres alfanuméricos e o sublinhado (A-z, 0-9, _); ex: \$Nome1, \$Nome2, \$Nome_Sobrenome
- Há diferença entre letras maiúsculas e minúsculas, é *case sensitive*; ex: \$num é diferente de \$Num
- As variáveis podem ser criadas em qualquer lugar do programa;
- As variáveis do PHP podem pertencer a três escopos diferentes: local, global e estática.

Não existe comando de declaração de variável em PHP. A variável é criada no momento do uso, quando seu primeiro valor é atribuído. O PHP determina os tipos de dados por análise de conteúdo armazenado na variável.

Exemplos:

```
<?php
    $cor = "cinza";
    $valor = 47;
    $x = 5.9;
    $existe = true;
    $nao_existe = false;
    $carros[0] = "Ferrari";
    $carros[1] = "Lamborghini";
    $carros[2] = "Aston Martin";
?>
```

Observe nos exemplos que não foi necessário dizer ao PHP que tipos de variáveis foram criados. O PHP converte para o tipo correto automaticamente, dependendo de seu conteúdo. Em outras linguagens como Pascal, C, C++, Java, o programador tem que declarar o tipo de variável antes de usá-la.

O PHP utiliza os seguintes tipos de dados implícitos:

- **Inteiro (integer)** – quaisquer números sem parte decimal, por exemplo: -5, 0, 47.
- **Ponto flutuante (float)** – números com casas decimais, também conhecidos como *double* e *real*, por exemplo: 3.14, 0.225, -8.7.
- **Caractere (string)** – quaisquer valores alfanuméricos, por exemplo: "Hello", "Rua Alpha, nº 208".
- **Lógico (boolean)** – valores *true* ou *false*, por exemplo: \$confere=true;
- **Vetor (array)** – variável que armazena múltiplos valores indexados.
- **Objeto** – uma entidade definida por propriedades (atributos) e métodos (ações); é criado pelo operador **new**.

Exemplo: class Hardware

```

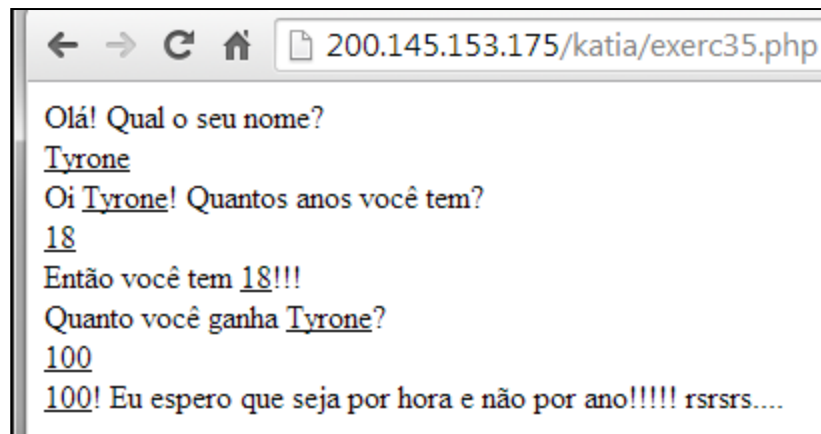
{    var $clock;
    function liga()
    {    Echo "Computador com cpu de {$this->clock";    }
    }
    $comp = new Hardware;
    $comp->clock = "2.7GHz";
    $comp->liga();

```

- **Null** – indica que uma variável não possui valor, por exemplo: \$var = NULL;

EXERCÍCIO

35. Declare três variáveis que conterão respectivamente: nome de uma pessoa, idade e um valor (pode ser de salário ou de mesada), serão valores atribuídos, não digitados. Verifique o que foi impresso na tela abaixo e reproduza isto em um programa utilizando os dados das variáveis criadas. Os itens que estão sublinhados são valores de variáveis.



4.4.2 Variáveis predefinidas

São variáveis criadas pelo próprio PHP. São chamadas de variáveis “superglobais”, que significa que podem ser acessadas de qualquer parte do script (função, classe ou arquivo).

São elas:

\$GLOBALS	\$_POST	\$_ENV
\$_SERVER	\$_GET	\$_COOKIE
\$_REQUEST	\$_FILES	\$_SESSION

4.4.3 Variáveis de ambiente (\$_SERVER)

A variável \$_SERVER é “superglobal” (ver item 4.4.2), guarda informações de cabeçalho, caminhos e localização de scripts. Os elementos descritos abaixo estarão garantidamente disponíveis somente se o servidor utilizado for o Apache, pois há variação entre os servidores.

A tabela a seguir lista os elementos mais importantes e suas descrições:

Elemento	Descrição
\$_SERVER['PHP_SELF']	Retorna o nome do script em execução no momento
\$_SERVER['GATEWAY_INTERFACE']	Retorna a versão do <i>Common Gateway Interface</i> (CGI) que o servidor esta utilizando
\$_SERVER['SERVER_ADDR']	Retorna o endereço do IP do servidor de hospedagem
\$_SERVER['SERVER_NAME']	Retorna o nome do servidor de hospedagem (como www.w3schools.com)
\$_SERVER['SERVER_SOFTWARE']	Retorna a string de identificação do servidor (como Apache/2.2.24)
\$_SERVER['SERVER_PROTOCOL']	Retorna o nome e a revisão do protocolo de informação (como HTTP/1.1)
\$_SERVER['REQUEST_METHOD']	Retorna o método de solicitação (requisição) usado para acessar a página (como POST)
\$_SERVER['REQUEST_TIME']	Retorna a data e hora do início da solicitação (como 1377687496)
\$_SERVER['QUERY_STRING']	Retorna a string de consulta se a página é acessada através de uma string de consulta
\$_SERVER['HTTP_ACCEPT']	Retorna o cabeçalho Accept da requisição atual
\$_SERVER['HTTP_ACCEPT_CHARSET']	Retorna o cabeçalho Accept_Charset da requisição atual (como utf-8,ISO-8859-1)
\$_SERVER['HTTP_HOST']	Retorna o cabeçalho do Host da requisição atual
\$_SERVER['HTTP_REFERER']	Retorna a URL completa da página atual (não confiável, porque nem todos os user-agents o suportam)
\$_SERVER['HTTPS']	É o script consultado através de um protocolo HTTP seguro

\$_SERVER['REMOTE_ADDR']	Retorna o endereço IP de onde o usuário está visualizando a página atual
\$_SERVER['REMOTE_HOST']	Retorna o nome do Host de onde o usuário está visualizando a página atual
\$_SERVER['REMOTE_PORT']	Retorna a porta que está sendo usada na máquina do usuário para se comunicar com o servidor web
\$_SERVER['SCRIPT_FILENAME']	Retorna o caminho absoluto do script atualmente em execução
\$_SERVER['SERVER_ADMIN']	Retorna o valor dado à directiva SERVER_ADMIN no arquivo de configuração do servidor web (se o seu script é executado em um host virtual, este será o valor definido para ele) (como <code>alguem@w3schools.com</code>)
\$_SERVER['SERVER_PORT']	Retorna a porta na máquina servidora utilizada pelo servidor web para comunicação (como 80)
\$_SERVER['SERVER_SIGNATURE']	Retorna a versão do servidor e nome do host virtual que é adicionado às páginas geradas no servidor
\$_SERVER['PATH_TRANSLATED']	Retorna o caminho base do sistema de arquivos para o script atual
\$_SERVER['SCRIPT_NAME']	Retorna o caminho do script atual
\$_SERVER['SCRIPT_URI']	Retorna o URI da página atual

Exemplo 5 – uso de variáveis de ambiente:

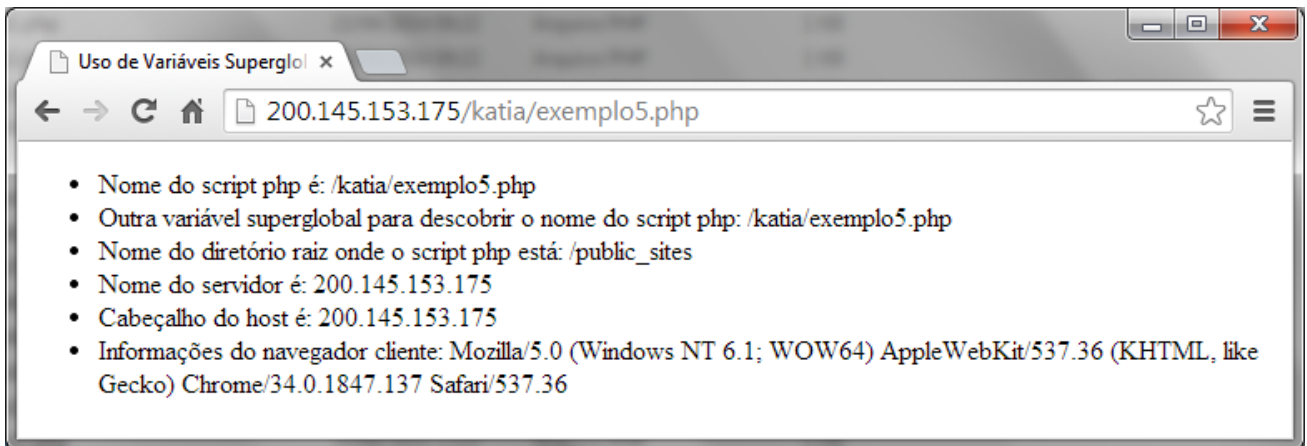
```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8" />
5     <title> Uso de Variáveis Superglobais</title>
6   </head>
7   <body>
8     <?php
9       echo"<ul>";
10        echo "<li>Nome do script php é: ".$_SERVER['PHP_SELF']."</li>";
11        echo "<li>Outra variável superglobal para descobrir o nome do script php:
12 ".$_SERVER['SCRIPT_NAME']."</li>";
13        echo "<li>Nome do diretório raiz onde o script php está:
14 ".$_SERVER['DOCUMENT_ROOT']."</li>";
15        echo "<li>Nome do servidor é: ".$_SERVER['SERVER_NAME']."</li>";
16        echo "<li>Cabeçalho do host é: ".$_SERVER['HTTP_HOST']."</li>";
17        echo "<li>Informações do navegador cliente:
18 ".$_SERVER['HTTP_USER_AGENT']."</li>";
19        echo"</ul>";
20     ?>

```

```
21 </body>
22 </html>
```

A visualização do Exemplo 5 está na imagem abaixo.



4.4.4 Variáveis de formulários (\$_GET e \$_POST)

Quando um formulário HTML é preenchido e enviado para um servidor, todas as variáveis deste formulário são automaticamente disponibilizadas dentro do script PHP. Para isso, no *action* do formulário deverá aparecer o nome do script PHP a ser executado, assim como, no *method*, informar se o método é do tipo GET ou POST. Variáveis enviadas pelo método GET ficam alojadas no vetor \$_GET do PHP; as do método POST, no vetor \$_POST. Ambos os vetores são variáveis “superglobais” (ver item 4.4.2).

Exemplo 6 (arquivo HTML) – uso de constantes predefinidas:

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8" />
5     <title>Exemplo PHP</title>
6   </head>
7   <body>
8     <h3>Demonstração método POST</h3>
9     <form method="post" action="http://200.145.153.175/ariane/processa_ex6.php">
10        <!-- com o endereço absoluto -->
11        <!-- action="processa_ex6.php"> com o endereço relativo -->
12        Primeiro nome: <input type="text" name="PrimeiroNome"> <br>
13        Último nome: <input type="text" name="UltimoNome">
14        <br><br>
15        <input type="submit" value="Enviar">
16        <input type="reset" value="Limpar">
17      </form>
18    </body>
19  </html>
```

O arquivo `processa_ex6.php` é dado abaixo:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8" />
5     <title> Uso de Formulário</title>
6   </head>
7   <body>
8     <?php
9       echo "<p><h2>Dados enviados com sucesso!</h2></p><hr>";
10      echo "Nome   : ".$_POST['PrimeiroNome']."<br>";
11      echo "Sobrenome: ".$_POST['UltimoNome']."<br><hr>";
12    ?>
13  </body>
14 </html>

```

Visualização da execução dos arquivos `exemplo6.html` e `processa_ex6.php`



EXERCÍCIOS

36. Utilizando a ideia do exercício que mostra o nome do aluno anterior, seu nome e o aluno posterior a você na lista de chamada, crie uma página em HTML que receba os 3 nomes, chame um programa PHP que mostre esta sequência. Utilize o método POST.
- Nome anterior ao meu
 - Meu nome
 - Nome posterior ao meu
37. Crie uma página HTML que receba o nome do usuário, uma senha e utilize a opção radio do elemento `<input>` (apostila HTML) para a seleção do gênero (masculino e feminino). Envie para um programa PHP que mostre estas informações. Utilize o método GET.

4.5 Constantes

Uma constante é uma variável cujo valor **não** pode ser alterado no decorrer da execução do programa. Para criar uma constante utiliza-se a função *define*("nome da constante", "valor da constante"). Seu valor não pode ser redefinido e não há o caractere \$ (cifrão) como na variável.

Exemplo: `define("PI", "3.141592");`

No PHP existem constantes predefinidas disponíveis para uso do programador. Algumas delas são:

Constante	Descrição
__LINE__	Retorna o número da linha atual do script
__FILE__	Retorna o caminho completo com o nome do arquivo
TRUE, FALSE	Não há um tipo booleano definido no PHP, então as constantes true e false, que não são <i>case sensitive</i> , são utilizadas para esse fim, por exemplo, <code>\$certo=true;</code>
PHP_VERSION	Retorna a versão do PHP
PHP_OS	Retorna o nome do sistema operacional (OS) do servidor
E_ERROR	Retorna um valor inteiro que corresponde ao erro de execução (runtime error)
E_WARNING	Retorna um valor inteiro que corresponde a erro de execução não fatal, que não aborta a execução
E_PARSE	Retorna um valor inteiro correspondente a um erro de análise (parse error)
E_NOTICE	Avisos (notices), por padrão, não são apresentados e indicam que o script encontrou alguma coisa que pode indicar um erro, mas que também pode ocorrer na execução normal do script, por exemplo, tentar acessar o valor ainda não definido de uma variável

Exemplo 7 – uso de constantes predefinidas:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8" />
5     <title> Uso de Constantes</title>
6   </head>
7   <body>
8     <?php
9       echo "<p>Esta é a linha de número: ".__LINE__."</p>";
10      echo "<p>Este é o arquivo: ".__FILE__."</p>";

```

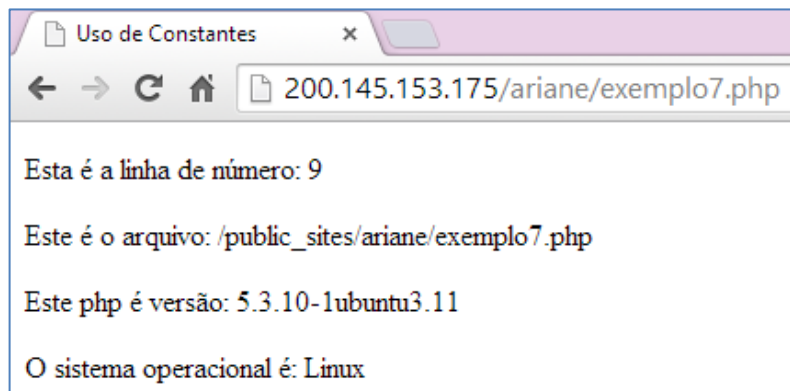


```

11     echo "<p>Este php é versão: ".PHP_VERSION."</p>";
12     echo "<p>O sistema operacional é: ".PHP_OS."</p>";
13     ?>
14     </body>
15 </html>

```

A visualização do Exemplo 7 está na imagem abaixo.



EXERCÍCIOS

38. Faça um programa PHP que defina uma constante PI com valor 3.141592. Crie uma variável, atribua um valor que represente o raio de uma esfera. Calcule e apresente sua área e seu volume, dados pelas fórmulas:

$$AREA = 4\pi R^2$$

$$VOLUME = \frac{4}{3}\pi R^3$$

39. Faça um script PHP que receba de um formulário HTML5 uma variável com o ano de nascimento de uma pessoa, crie uma constante com o ano atual, calcule e mostre:
- a idade dessa pessoa em anos;
 - quantos dias esta pessoa já viveu;
 - quantos anos essa pessoa terá em 2025

4.6 Operadores

Os operadores da linguagem PHP são: aritméticos, de atribuição, de comparação, lógicos e de concatenação.

4.6.1 Operadores Aritméticos

São os operadores comuns em outras linguagens como C++ e Java. Veja através dos exemplos a seguir:

Exemplo	Nome	Resultado
$-\$a$	Negação	Oposto de $\$a$.
$\$a + \b	Adição	Soma de $\$a$ e $\$b$.
$\$a - \b	Subtração	Diferença entre $\$a$ e $\$b$.
$\$a * \b	Multiplicação	Produto de $\$a$ e $\$b$.
$\$a / \b	Divisão	Quociente de $\$a$ por $\$b$.
$\$a \% \b	Módulo	Resto de $\$a$ dividido por $\$b$.
$-\$a$	Troca Sinal	Trocar o sinal de $\$a$
$++\$a$	Pré-incremento	Incrementa $\$a$ em um, e então retorna $\$a$.
$\$a++$	Pós-incremento	Retorna $\$a$, e então incrementa $\$a$ em um.
$--\$a$	Pré-decremento	Decrementa $\$a$ em um, e então retorna $\$a$.
$\$a--$	Pós-decremento	Retorna $\$a$, e então decrementa $\$a$ em um.

4.6.2 Operadores de Atribuição

São operadores que modificam o valor da variável. Também são comuns em outras linguagens como C++ e Java.

Exemplo	Efeito
$\$a = \b	$\$a$ recebe o valor de $\$b$
$\$a += \b	Equivalente a $\$a = \$a + \$b$
$\$a -= \b	Equivalente a $\$a = \$a - \$b$
$\$a *= \b	Equivalente a $\$a = \$a * \$b$
$\$a /= \b	Equivalente a $\$a = \$a / \$b$
$\$a \% = \b	Equivalente a $\$a = \$a \% \$b$
$\$a .= \b	Concatenação: equivalente a $\$a = \$a . \$b$

4.6.3 Operadores de Comparação

São operadores que executam comparações entre duas variáveis, ou uma variável e um texto ou uma variável e um número.

Exemplo	Nome	Resultado
<code>\$a == \$b</code>	Igual	Verdadeiro (TRUE) se <code>\$a</code> é igual a <code>\$b</code> .
<code>\$a === \$b</code>	Idêntico	Verdadeiro (TRUE) se <code>\$a</code> é igual a <code>\$b</code> , e eles são do mesmo tipo (introduzido no PHP4).
<code>\$a != \$b</code>	Diferente	Verdadeiro se <code>\$a</code> não é igual a <code>\$b</code> .
<code>\$a <> \$b</code>	Diferente	Verdadeiro se <code>\$a</code> não é igual a <code>\$b</code> .
<code>\$a !== \$b</code>	Não idêntico	Verdadeiro se <code>\$a</code> não é igual a <code>\$b</code> , ou eles não são do mesmo tipo (introduzido no PHP4).
<code>\$a < \$b</code>	Menor que	Verdadeiro se <code>\$a</code> é estritamente menor que <code>\$b</code> .
<code>\$a > \$b</code>	Maior que	Verdadeiro se <code>\$a</code> é estritamente maior que <code>\$b</code> .
<code>\$a <= \$b</code>	Menor ou igual	Verdadeiro se <code>\$a</code> é menor ou igual a <code>\$b</code> .
<code>\$a >= \$b</code>	Maior ou igual	Verdadeiro se <code>\$a</code> é maior ou igual a <code>\$b</code> .

4.6.4 Operadores Lógicos

São operadores que retorna verdadeiro ou falso.

Exemplo	Nome	Resultado
<code>\$a and \$b</code>	E	Verdadeiro (TRUE) se tanto <code>\$a</code> quanto <code>\$b</code> são verdadeiros.
<code>\$a or \$b</code>	OU	Verdadeiro se <code>\$a</code> ou <code>\$b</code> são verdadeiros.
<code>\$a xor \$b</code>	XOR	Verdadeiro se <code>\$a</code> ou <code>\$b</code> são verdadeiros, mas não ambos.
<code>! \$a</code>	NÃO	Verdadeiro se <code>\$a</code> não é verdadeiro.
<code>\$a && \$b</code>	E	Verdadeiro se tanto <code>\$a</code> quanto <code>\$b</code> são verdadeiros.
<code>\$a \$b</code>	OU	Verdadeiro se <code>\$a</code> ou <code>\$b</code> são verdadeiros.

4.6.5 Operadores de Concatenação

Operador	Nome	Exemplo	Resultado
.	Concatenação	\$txt1 = "Hello" \$txt2 = \$txt1 . " world!"	\$txt2 contem "Hello world!"
.=	Concatenação e atribuição	\$txt1 = "Hello" \$txt1 .= " world!"	\$txt1 contem "Hello world!"

EXERCÍCIOS

40. Faça um programa PHP que crie duas variáveis e atribua dois valores inteiros. Apresente todas as operações: adição, subtração, divisão, multiplicação e resto da divisão utilizando os operadores de atribuição.

Para o exercício 41 consulte o comando IF e os laços de repetição For, While.

41. Crie uma página que receba o ISBN (International Standard Book Number) que pode se referir a um livro ou a uma revista ou periódico científico. Envie para um programa PHP que validará o número e apresentará uma mensagem indicando se é válido ou não.

Um ISBN consistente possui 10 dígitos e é válido se:

$$1 * \text{dig1} + 2 * \text{dig2} + 3 * \text{dig3} + 4 * \text{dig4} + \dots + 10 * \text{dig10} \text{ for um múltiplo de } 11.$$

Exemplo: Número digitado: 0201314525

$$\text{Cálculo: } 1 * 5 + 2 * 2 + 3 * 5 + 4 * 4 + 5 * 1 + 6 * 3 + 7 * 1 + 8 * 0 + 9 * 2 + 10 * 0 = 88$$

Como 88 é múltiplo de 11, o ISBN é válido!

Observação: o dígito inicial do cálculo é o último da sequência, dígito 5, e o último do cálculo, o primeiro da sequência, dígito 0.

Alguns comandos serão úteis:

trim – retira espaços em branco à esquerda e à direita da string;

strlen – retorna quantos caracteres a string tem.

Exemplo: \$texto=" teste PHP ";

strlen(trim(\$texto)) => retira os os espaços em branco antes e depois da string e retorna o valor 9

Para transformar uma string em um vetor utilize a função str_split:

\$vet = str_split(trim(\$texto));

Array ([0] => t [1] => e [2] => s [3] => t [4] => e [5] => [6] => P [7] => H [8] => P)

5 Estruturas do PHP

As estruturas do PHP que estudaremos a seguir, são: if, switch, while, do-while, for e foreach.

5.1 Comando if

O comando if em PHP tem as mesmas características de execução implementadas em outras linguagens, mas possui diferentes formas para sua utilização:

Formato 1	Formato 2	Formato 3
if (expressão) comando;	if (expressão) { comando1; comando2; comando-n; }	if (expressão): comando1; comando2; comando-n; endif;

Formato 4	Formato 5	Formato 6	Formato 7
if (expressão) comando; else comando;	if (expressão): comando1; comando-n; else: comando2; comando-n; endif;	if (expressão1) comando1; else if (expressão2) comando2; else if (expressão3) comando3;	if (expressão1) { comando1; comando-n; } elseif (expressão2) { comando2; comando-n; } else comando;

Exemplo 8 – uso do comando if (formato 6):

```

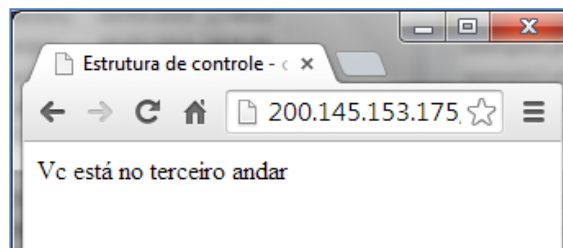
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
```

```

5     <title>Estrutura de controle - comando if</title>
6     </head>
7     <body>
8         <?php
9             $andar=3;
10            if ($andar == 1)
11                echo "Vc est&aacute; no primeiro andar";
12            else if ($andar == 2)
13                echo "Vc est&aacute; no segundo andar";
14            else if ($andar == 3)
15                echo "Vc est&aacute; no terceiro andar";
16            else
17                echo "Andar inv&aacute;lido";
18        ?>
19     </body>
20 </html>

```

A visualização do Exemplo 8 está na imagem abaixo.



EXERCÍCIOS:

42. Utilize o exemplo 8 e faça outro programa em PHP utilizando o formato 7 descrito para o comando if.
43. Faça um programa em PHP que defina uma variável com o peso e a altura de uma pessoa. Calcule e mostre o IMC.
44. Crie uma página em HTML5 na qual a pessoa possa digitar seu peso e sua altura e um programa PHP para o cálculo do IMC da pessoa. Defina se a pessoa está acima ou abaixo do peso. Procure referências sobre este índice (o que é considerado normal, abaixo ou acima do peso). Inclua a referência (página de acesso) para que a pessoa leia sobre este assunto.
45. Faça uma página em HTML5 que leia o placar de um jogo de futebol (os gols de cada time, um valor em cada variável do tipo int). Informe se houve empate ou se a vitória foi do 1º ou do 2º time. A entrada pode ter o nome dos dois times.
46. Crie uma página que receba uma temperatura em Fahrenheit, transforme para Celsius e classifique-a.

- se temp. em Celsius for menor ou igual a zero, imprimir “Frio ártico!”;
- se temp. em Celsius de 01 a 12 graus, imprimir “Muito frio!”;
- se temp. em Celsius de 13 a 23 graus, imprimir “Clima ameno!”;
- se temp. em Celsius de 24 a 32 graus, imprimir “Calor!”;
- se temp. em Celsius de 33 a 40 graus, imprimir “Calor tórrido!”;
- Qualquer outro valor, imprimir “Sem registro!”;

47. Faça uma página em HTML que receba a velocidade máxima permitida em uma avenida e a velocidade com que o motorista estava dirigindo nela e calcule e apresente a multa que uma pessoa vai receber, sabendo que são pagos:

- R\$ 50,00 se o motorista estiver ultrapassar em até 10km/h a velocidade permitida (ex.: velocidade máxima: 50km/h; motorista a 60km/h ou a 56km/h);
- R\$ 100,00 se o motorista ultrapassar de 11 a 30 km/h a velocidade permitida.
- R\$ 200,00 se estiver acima de 31km/h da velocidade permitida.

5.2 Comando switch

A execução do comando switch é semelhante ao comando if. No entanto, sua forma deixa a comparação explicitada pelos valores que as variáveis podem assumir.

Formato 1
<pre> switch (variável) { case valor1: comando; comando; break; case valor-n: comando; comando; break; default: comando; break; } </pre>

Formato 2	
switch (variável) :	
case valor1: comando;	
.....	
comando;	
break;	
.....	
case valor-n: comando;	
.....	
comando;	
break;	
default:	
comando;	
break;	
endswitch;	

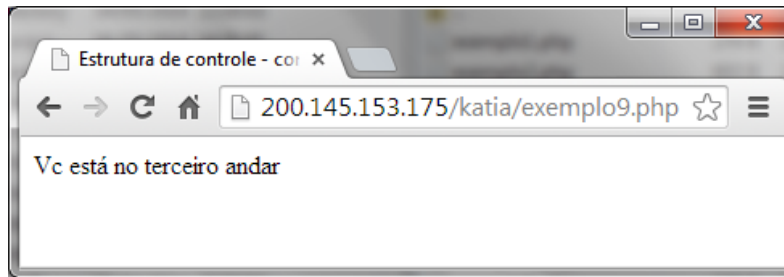
Exemplo 9 – uso do comando switch (formato 1):

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Estrutura de controle - comando switch</title>
6   </head>
7   <body>
8     <?php
9       $andar=3;
10      switch ($andar){
11        case 1:
12          echo "Vc est&aaacute; no primeiro andar";
13          break;
14        case 2:
15          echo "Vc est&aaacute; no segundo andar";
16          break;
17        case 3:
18          echo "Vc est&aaacute; no terceiro andar";
19          break;
20        default:
21          echo "Andar inv&aaacute;lido";
22          break;
23      }
24     ?>
25   </body>
26 </html>

```


Execução do exemplo 9:



Exemplo 9_a – uso do comando switch (formato 2) e case com dois valores testados:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Estrutura de controle - comando switch</title>
6   </head>
7   <body>
8     <?php
9       $andar=3;
10      switch ($andar):
11        case 1:
12          echo "Você est&aaacute; no primeiro andar";
13          break;
14        case 2:
15          echo "Você est&aaacute; no segundo andar";
16          break;
17        case 3:
18        case 4:
19          echo "Você est&aaacute; no terceiro ou quarto andar";
20          break;
21        default:
22          echo "Andar inv&aaacute;lido";
23          break;
24      endswitch;
25    ?>
26  </body>
27 </html>

```

Em outras linguagens a variável deve ser do tipo int ou do tipo char. No PHP é possível utilizar variáveis do tipo string para os testes. No exemplo abaixo, o método *post* foi utilizado e o comando switch executado com o conteúdo da variável do tipo string:

Exemplo 10 (arquivo HTML) – comando switch com variável tipo string

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8" />
5     <title>Exemplo PHP com switch</title>

```

```

6 </head>
7 <body>
8   <h1>Utilização metodo POST e comando switch</h1>
9   <form method="post" action="http://200.145.153.175/katia/processa_ex10.php">
10    Digite a cor da casa: <input type="text" name="casa"> <br>
11    <br><br>
12    <input type="submit" value="Enviar">
13    <input type="reset" value="Limpar">
14  </form>
15 </body>
16 </html>

```

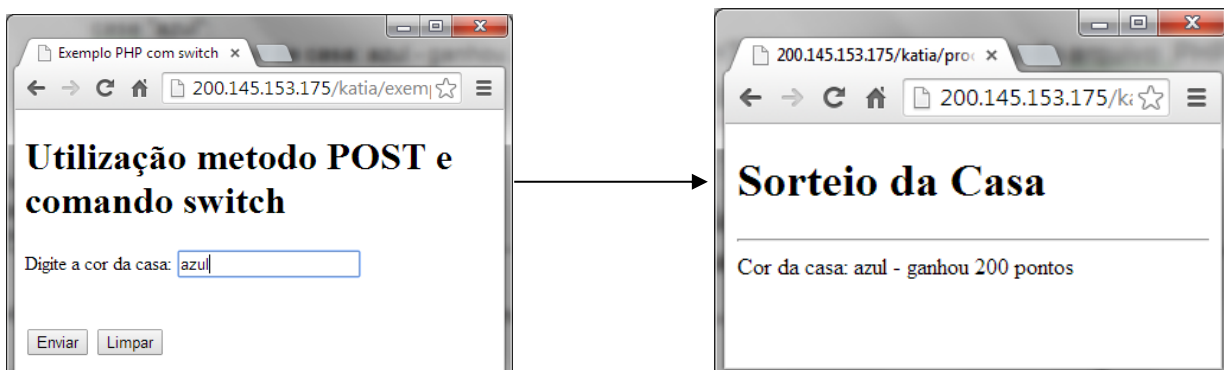
Conteúdo do arquivo processa_ex10.php:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <body>
4     <?php
5       echo "<p><h1>Sorteio da Casa</h1></p><hr>";
6       switch ($_POST['casa']){
7         case "amarela":
8           echo "Cor da casa: amarela - ganhou 100 pontos<br>";
9           break;
10        case "azul":
11          echo "Cor da casa: azul - ganhou 200 pontos<br>";
12          break;
13        default:
14          echo "Perdeu todos os pontos";
15        }
16      ?>
17   </body>
18 </html>

```

A página HTML e o processamento do arquivo .PHP estão nas imagens abaixo:



EXERCÍCIOS

48. Faça uma página em HTML que receba o número de integrantes de uma equipe e o nome do líder da equipe. Envie pelo método post ou get este valor para um programa PHP que apresentará (utilizando o comando switch):

- ✓ 0 : não forma equipe
- ✓ 1: não forma equipe
- ✓ 2, 3 ou 4: equipe formada com sucesso
- ✓ demais valores: inválidos

Se formar a equipe, apresente o nome do líder.

Dica: faça o teste, o comando switch aceita: case 1: case 2: case 3:

49. Faça uma página em HTML que mostre a bandeira de três países distintos. Quando o usuário clicar sobre a bandeira, envie o nome do país para um programa PHP que lá fará a distinção mostrando o nome de duas cidades do país selecionado e a língua que se fala neste país.

50. Construa uma página HTML na qual o usuário deverá entrar com uma das opções do menu para a escolha do tipo de filme que gosta de assistir (utilize uma entrada do tipo radio ou outra a de seleção) e a idade do usuário. Faça esta pesquisa para 5 usuários (apresente na página HTML a opção de digitação para as cinco pessoas):

- (a) Comédia
- (b) Romance
- (c) Terror
- (d) Aventura
- (e) Ação
- (f) Ficção

Apresente no final cada gênero de filme e a média das idades dos usuários para cada seleção. Utilize o comando switch.

5.3 Comando while

É um comando de repetição que executa um ou mais comandos até que a condição ou expressão testada resulte em *false*. Assim como o comando *if* em PHP, possui formas alternativas para a sua sintaxe.

Formato 1	Formato 2	Formato 3
while (expressão) comando;	while(expressão) { comando1; comando2; comando-n; }	while (expressão): comando1; comando2; comando-n; endwhile;

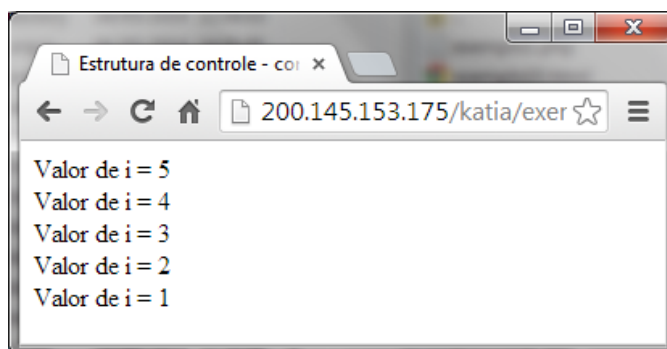
Exemplo 11 – uso do comando while no formato 3

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Estrutura de controle - comando while</title>
6   </head>
7   <body>
8     <?php
9       $i=10;
10      while ($i > 0):
11        echo "Valor de i = ". $i. "<br>";
12        $i--;
13      endwhile;
14    ?>
15  </body>
16 </html>

```

A visualização do Exemplo 11 está na imagem abaixo.



EXERCÍCIOS

51. Faça um programa em PHP que mostre todos os números inteiros de 100 a 200 com incremento de 2 em 2.
52. Faça um programa em PHP que apresente todos os valores ímpares no intervalo de 500 a 1000.
53. Através de uma página HTML receba o valor inicial e final de um intervalo. Acione um programa PHP que verificará se o inicial for menor ou igual ao final e apresentará uma mensagem para encerrar o programa. Caso contrário, apresente com o comando while todos os valores do intervalo.
54. Escreva um programa em PHP para somar todos os números pares num intervalo de 10 a 500 e apresentar a soma.

5.4 Comando do-while

A execução deste comando é parecida com a do comando while, apenas com a diferença que o teste da condição ou expressão é realizado no final. Com isto, o bloco de comandos é executado uma vez e depois é realizado o teste. Possui apenas um formato, como segue.

Formato 1
do { comando1; comando-n; }while (expressão);

Exemplo 12 – utilização do comando do-while. A execução é a mesma do exemplo 11.

1	<!DOCTYPE html>
2	<html lang="pt-br">
3	<head>
4	<meta charset="UTF-8">
5	<title>Estrutura de controle - comando do-while</title>
6	</head>
7	<body>
8	<?php
9	\$i=5;

```

10         do {
11             echo "Valor de i = ". $i. "<br>";
12             $i--;
13         } while ($i > 0);
14     ?>
15 </body>
16 </html>

```

Exemplo 13 (arquivo HTML) – página para entrada de valores que serão validados no PHP

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3     <head>
4         <meta charset="UTF-8" />
5         <title>Formulário de Contato</title>
6     </head>
7     <body>
8         <form action="valida_exemplo13.php" method="post">
9             <label><b>Nome:</b><br>
10            <input type="text" name="nome" id="nome"><br><br>
11            </label>
12            <label><b>Idade:</b><br>
13            <input type="text" name="idade" id="idade"><br><br>
14            </label>
15            <label><b>Assunto:</b><br>
16            <input type="text" name="assunto" id="assunto"><br><br>
17            </label>
18            <label><b>Mensagem:</b><br>
19            <textarea name="mensagem" id="mensagem" cols="30" rows="3">
20            </textarea><br><br>
21            </label>
22            <input type="submit" name="button" value="Enviar">
23        </form>
24    </body>
25 </html>

```

Conteúdo do arquivo valida_exemplo13.php:

```

1 <?php
2     $nome = "$_POST[nome]"; //nome
3     $idade = "$_POST[idade]"; //idade
4     $mensagem = "$_POST[mensagem]"; //mensagem
5     $assunto = "$_POST[assunto]"; //assunto
6     /*condições de envio. Se os campos nome, idade, assunto e mensagem
7     não forem preenchidos será mostrada uma mensagem de erro.*/
8     if (($nome == "") || ($idade == "") || ($assunto == "") || ($mensagem == ""))
9         echo "Atencao! Todos os campos do formulario devem ser preenchidos, clique em voltar";
10    /*caso todos os campos estejam preenchidos, testar se a idade está
11    no intervalo de 0 a 100*/
12    else
13        if ($idade < 0 || $idade > 100)
14            echo "Idade invalida, fora do intervalo 0-100, clique em voltar e redigite";
15    ?>
16 <br>
17 <a href="exemplo13.html"> Voltar</a>

```

As simulações da página HTML e o processamento do arquivo .PHP estão nas imagens abaixo:

Formulário de Contato

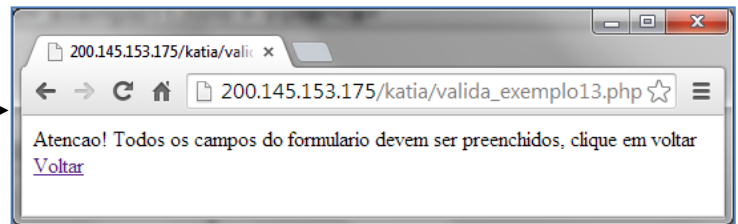
Nome:

Idade:

Assunto:

Mensagem:

Enviar



Formulário de Contato

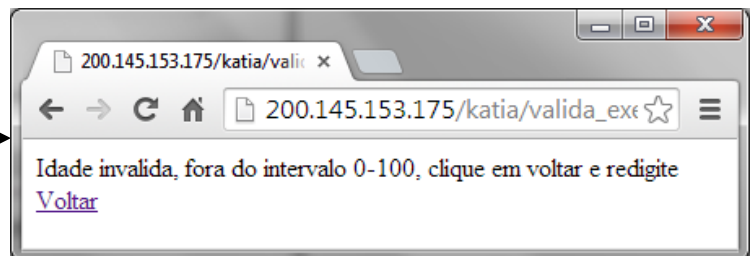
Nome:

Idade:

Assunto:

Mensagem:

Enviar



EXERCÍCIO

55. Desenvolva uma página HTML na qual o usuário deve digitar dois valores, um que representa o limite inferior e o outro o limite superior de um intervalo. Caso o limite inferior seja maior ou igual ao superior, dê uma mensagem e retorne à página de origem para redigitação. Caso contrário mostre todos os valores de lim-inferior até lim-superior utilizando o do-while.

5.5 Comando for

Este comando permite que um bloco de comandos seja repetido por um número de vezes pré-definido. Neste caso não é possível utilizar condições de parada que dependam do usuário, como por exemplo, que o bloco de comandos seja executado enquanto a condição seja verdadeira. O comando for possui três diferentes formas para sua utilização.

Formato 1	for (inicialização;condição;incremento ou decremento) comando;
Formato 2	for (inicialização;condição;incremento ou decremento) { comando1; comando-n; }
Formato 3	for (inicialização;condição;incremento ou decremento) : comando1; comando-n; endfor;

Exemplo 14a – utilização do comando for (formato 1 ou 2) com o mesmo exemplo do while e do-while.

A execução é a mesma do Exemplo 11.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Estrutura de controle - comando for</title>
6   </head>
7   <body>
8     <?php
9       for ($i=10; $i > 0; $i--)
10        echo "Valor de i = ". $i. "<br>";
11     ?>
12   </body>
13 </html>

```


5.6 Comando foreach

Esta estrutura de programação tem o propósito de percorrer arrays ou lista de objetos de forma mais fácil que outras estruturas de repetição. Ela não pode ser utilizada para outros tipos de dados e que não foram inicializadas.

Formato 1	foreach (\$variavel_array as \$valor) { comando; }
Formato 2	foreach (\$variavel_array as \$chave => \$valor) { comando; }
Formato 3	foreach (\$variavel_array as \$valor): comando; endforeach;

No primeiro formato a cada iteração o valor do elemento corrente (\$variavel_array) é atribuído à variável \$valor e um ponteiro interno avança uma posição (para o próximo elemento). No segundo formato, adicionalmente o valor da posição do array é atribuído à variável \$chave a cada iteração.

Exemplo 14b¹ – utilização do comando foreach (formato 1).

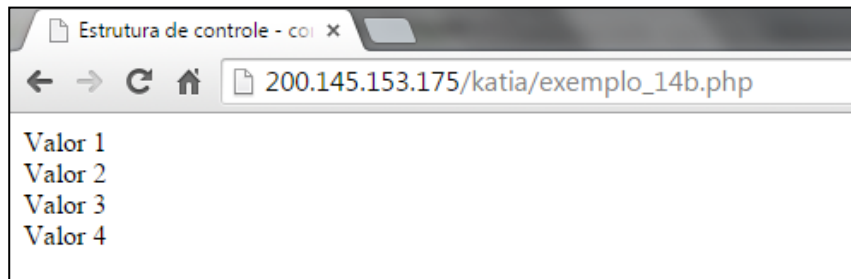
```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Estrutura de controle - comando foreach - Array</title>
6   </head>
7   <body>
8     <?php
9       $meu_array = array('Valor 1','Valor 2','Valor 3', 'Valor 4');
10      /* O laço foreach simples */
11      foreach ( $meu_array as $seu_valor )
12      {
13          echo $seu_valor . '<br>';
14      }
15      ?>
16   </body>
17 </html>

```

¹ Adaptado de http://php.net/manual/pt_BR/control-structures.foreach.php, acesso em 25/04/2015.

A visualização do Exemplo 14b está na imagem abaixo.



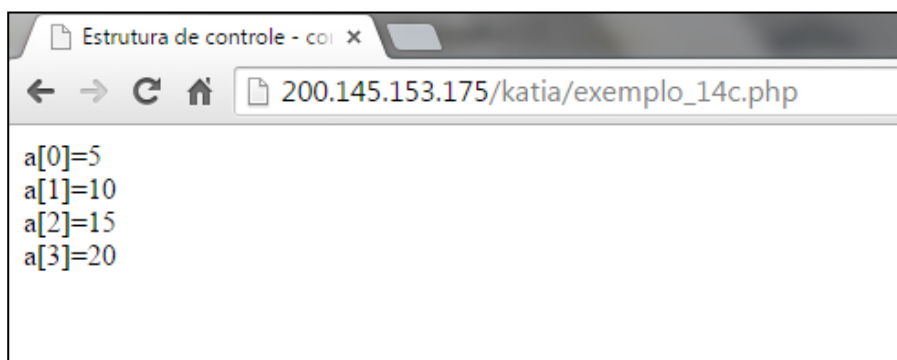
Exemplo 14c – utilização do comando foreach (formato 2) com apresentação da posição do Array.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="UTF-8">
5     <title>Estrutura de controle - comando foreach - Array</title>
6   </head>
7   <body>
8     <?php
9       $a = array(5, 10, 15, 20);
10      /* O laço foreach simples */
11      foreach ($a as $k => $v)
12        {
13          echo "a[" . $k. "]=" . $v. "<br>";
14        }
15      ?>
16   </body>
17 </html>

```

A imagem a seguir corresponde à execução do Exemplo 14c.



EXERCÍCIOS

56. Faça uma página HTML na qual o usuário deve um valor no intervalo de 5 a 50. Chame um programa PHP que verifique se o valor está no intervalo. Caso não esteja, retorne à página de origem para redigitação. Se estiver correto, apresente a tabuada do número no formato apresentado no exemplo:

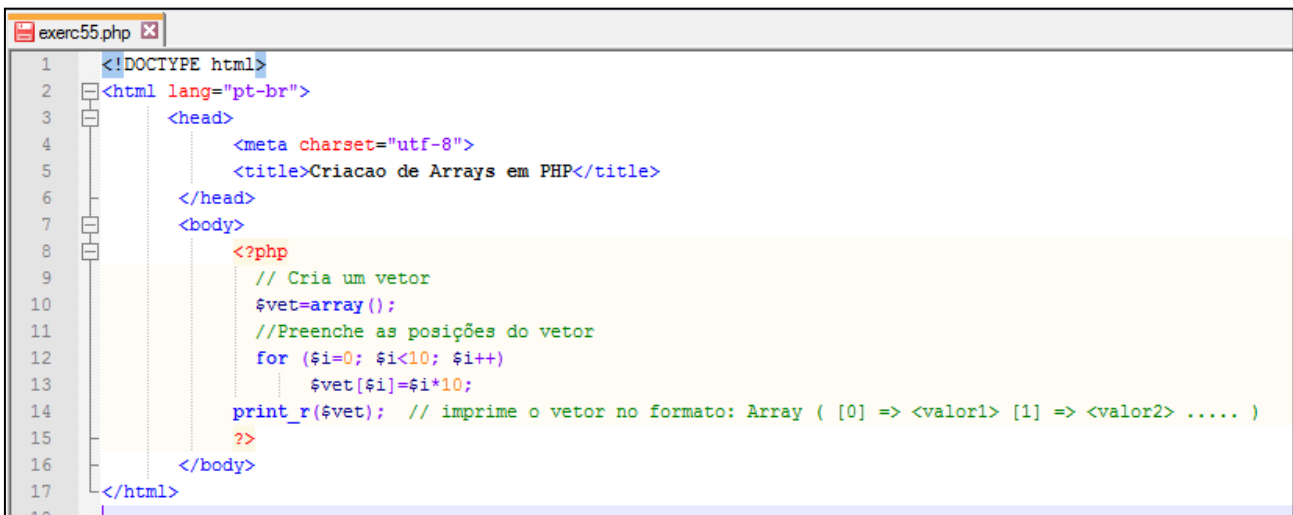
Tabuada do número: 7

```
7 * 0 = 0
7 * 1 = 7
....
7 * 10 = 70
```

57. Faça uma página que receba um valor positivo e apresente a saída:

```
Enter a number:5
----1
---22
--333
-4444
55555
```

58. Verifique o código abaixo e dê como resposta quais são os elementos do vetor criado no formato apresentado por `print_r`. Após a verificação, digite o exemplo e veja sua execução.



```
exerc55.php
1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Criação de Arrays em PHP</title>
6   </head>
7   <body>
8     <?php
9       // Cria um vetor
10      $vet=array();
11      //Preenche as posições do vetor
12      for ($i=0; $i<10; $i++)
13        $vet[$i]=$i*10;
14      print_r($vet); // imprime o vetor no formato: Array ( [0] => <valor1> [1] => <valor2> ..... )
15      ?>
16    </body>
17  </html>
```

59. Faça um programa que declare e atribua a um vetor o nome de 10 pessoas. Peça ao usuário para digitar um nome a ser procurado no vetor (página HTML). Faça a busca e apresente quantos nomes são iguais ao digitado.

Uma forma de criar e atribuir valores ao vetor é:

```
// Cria um vetor e atribui valores
```

```
$vet=array("Pablo","Tyrone","Uniqua","Tasha");
```

```
Array ( [0] => Pablo [1] => Tyrone [2] => Uniqua [3] => Tasha )
```

60. Encontre três valores consecutivos no vetor e retorne Verdadeiro ou Falso para o usuário. Para isto, receba 5 valores e armazene-os em um vetor. Se houver casos como 4,5,6, ou seja, 3 valores consecutivos, apresente uma mensagem.

61. Escreva um programa para trocar a primeira metade com a segunda metade da string. Em caso de comprimento da palavra ser ímpar, a primeira metade deve ter menos caracteres. E se o comprimento de palavra é um, então nenhuma mudança ocorrerá.

Exemplo1:

string digitada: "abcde" (tamanho 5 => ímpar)

primeira metade: ab segunda metade: cde

- o que deve ser apresentado: cdeab

Exemplo2:

string digitada: "Teste de PHP" (tamanho 12 => par)

primeira metade: Teste segunda metade: de PHP

- o que deve ser apresentado: de PHPTeste

62. Quando um autor de artigo científico publica, ele segue as normas da ABNT para a composição do nome. Se a pessoa se chama João Marcondes Ferracine, suas publicações terão o formato: Ferracine, J. M.

Faça uma página que receba o nome completo da pessoa (sem abreviações) e mostre-o no formato de publicação.

63. Desenvolva uma página que receba duas strings que corresponderão ao login e a senha de um pessoa. Criptografe a senha utilizando o método de troca de vogais por números. Apresente o login, a senha original e a senha criptografada.

Método simples de criptografia:

A	E	I	O	U	a	e	i	o	u
0	1	2	3	4	5	6	7	8	9

6 Funções

Uma função é um pedaço de código ou bloco de código reutilizável que executa uma ação específica em um script PHP.

6.1 Por que usar funções?

Funções são extremamente úteis na criação dos códigos de programação pelas seguintes razões:

- Modularização – funções permitem agrupar blocos de códigos relacionados que executam uma tarefa específica juntos, dando melhor organização.
- Reutilização – uma vez definida, a função pode ser chamada por um número indefinido de vezes, em diferentes programas, poupando tempo do programador para ser aplicado em outras tarefas.
- Fácil manutenção – atualizações ficam localizadas em um só bloco.

6.2 Utilizando funções

A construção básica de uma função em PHP é mostrada abaixo:

```
function nome_da_função ($arg_1, $arg_2, ..., $arg_n)
{
    instrução 1;
    instrução 2;
    ...
    instrução n;
    return $resultado; //variável de retorno
}
```

Diferentemente de outras linguagens de programação – por exemplo, o Pascal – no PHP as variáveis globais não são acessíveis dentro das funções, exceto se apresentarem uma definição interna na própria função. Isso está relacionado ao **escopo das variáveis**, ou seja, o contexto em que uma variável foi criada e as circunstâncias em que ela pode ser acessada. No PHP, por definição, **todas as variáveis são consideradas locais**, a menos que sejam explicitamente ditas globais.

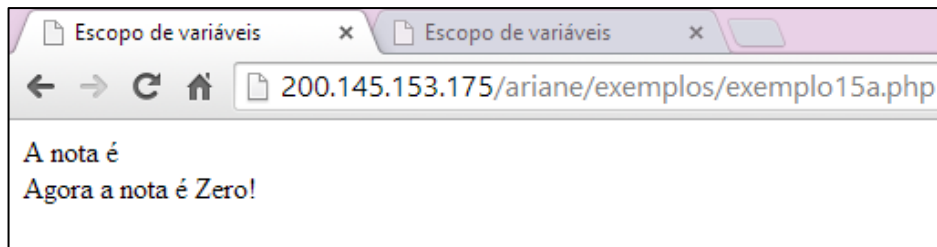
Exemplo 15a² – escopo das variáveis.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Escopo de variáveis</title>
6   </head>
7   <body>
8     <?php
9       $nota = "Dez!";
10
11      function ImprimeNotaLocal()
12      {
13        echo("A nota é ");
14        echo $nota."<br>";
15
16        $nota = "Zero!";
17        echo("Agora a nota é ");
18        echo $nota."<br>";
19      }
20
21      ImprimeNotaLocal();
22    ?>
23  </body>
24 </html>

```

Execução do exemplo 15a é:



Exemplo 15b³ – escopo das variáveis.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Escopo de variáveis</title>
6   </head>
7   <body>
8     <?php
9       $nota = "Dez!<br>";
10
11      function ImprimeNotaGlobal()
12      {
13        global $nota; //observe a diretiva global utilizada aqui

```

² Adaptado de Bruno, Estrozi e Batista Neto (2010, p. 46):

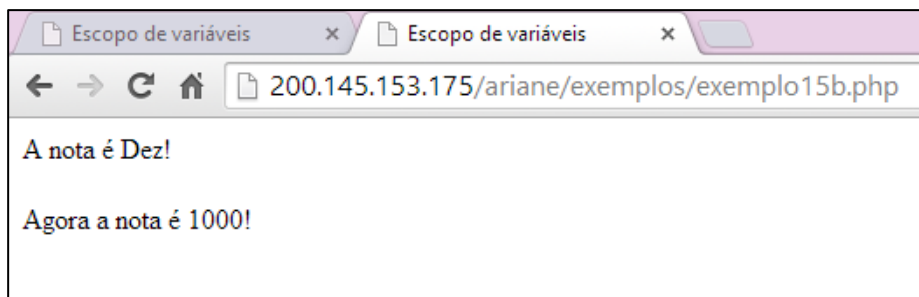
³ Adaptado de Bruno, Estrozi e Batista Neto (2010, p. 47):

```

14     echo("A nota é ");
15     echo($nota)."<br>";
16
17     $nota = "1000!";
18     echo("Agora a nota é ");
19     echo $nota."<br>";
20 }
21
22     ImprimeNotaGlobal();
23     ?>
24 </body>
25 </html>

```

Execução do exemplo 15b é:



Observe no exemplo 15b, na linha 13, o uso de uma diretiva chamada *global*. É essa diretiva que torna a variável *\$nota* visível em qualquer parte do script PHP.

6.3 Retornando valores

As funções podem retornar valores e para isso utiliza-se a instrução *return*. Apenas um argumento (valor) pode ser retornado com o *return*, podendo ser de qualquer tipo, uma lista de elementos ou um objeto.

Exemplo 16⁴ – retornando valores da função.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Retornando um valor</title>
6   </head>
7   <body>
8     <?php
9       function nome_completo ($nome, $sobrenome)
10      {
11        $ncompleto = $nome." ".$sobrenome;
12        return $ncompleto;
13      }
14

```

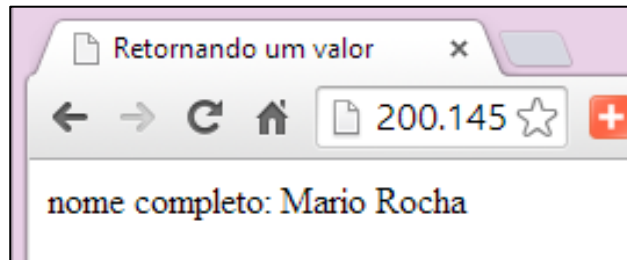
⁴ Adaptado de Bruno, Estrozi e Batista Neto (2010, p. 92):

```

15     $completo=nome_completo("Mario","Rocha");
16     echo "nome completo: ".$completo;
17     ?>
18 </body>
19 </html>

```

Execução do exemplo 16 é:



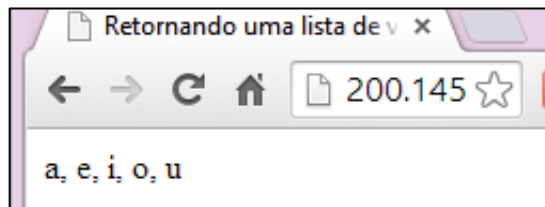
Exemplo 17⁵ – retornando lista de valores da função.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Retornando uma lista de valores</title>
6   </head>
7   <body>
8     <?php
9       function vogais ()
10      {
11        return array ("a", "e", "i", "o", "u");
12      }
13
14      list ($a, $b, $c, $d, $e) = vogais();
15      echo "$a, $b, $c, $d, $e";
16    ?>
17  </body>
18 </html>

```

Execução do exemplo 17 é:



⁵ Adaptado de Bruno, Estrozi e Batista Neto (2010, p. 93):

6.4 Passagem de argumentos (valores)

Argumentos ou valores são variáveis passadas para as funções na linguagem PHP. A função pode receber uma lista de valores e estes devem ser separados por vírgula. Em PHP podemos passar os argumentos por valor ou por referência.

6.4.1 Passagem de argumentos por valor

Este é o padrão na linguagem PHP e é feita através de uma atribuição de valores. Um argumento passado por valor tem uma cópia de seu conteúdo enviado à função. Qualquer alteração feita dentro da função não alterará a variável de origem. É como uma expressão de atribuição, por exemplo, uma variável \$a recebe o conteúdo de uma variável \$b; em seguida, a variável \$a recebe um novo valor, contido agora na variável \$x. O novo valor atribuído a \$a não provocará nenhuma alteração na variável \$b que continuará contendo o mesmo valor.

```
$a = $b;
```

```
$a = $x;
```

Exemplo 18⁶ – passagem de argumento por valor à função.

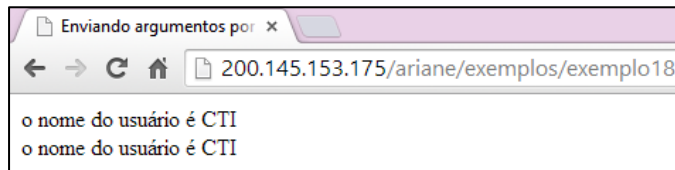
```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Enviando argumentos por valor</title>
6   </head>
7   <body>
8     <?php
9       function NomeUsuario($nome)
10        {
11          echo "o nome do usuário é ".$nome;
12          $nome="Unesp";
13        }
14        $user="CTI";
15        NomeUsuario($user); //poderia ser também NomeUsuario("CTI");
16        echo "<br>o nome do usuário é ".$user;
17      ?>
18   </body>
19 </html>

```

⁶ Adaptado de Bruno, Estrozi e Batista Neto (2010, p. 94):

Execução do exemplo 18 é:



Observe que ocorre uma repetição do valor "CTI". Isso acontece porque o valor "Unesp" atribuído à variável \$nome dentro da função não alterará o conteúdo da variável \$user, já que ela foi enviada por valor.

6.4.2 Passagem de argumentos por referência

Na passagem de argumentos por referência qualquer alteração feita dentro da função, na variável que foi enviada, causará uma alteração também na variável de origem. Isso acontece porque é o endereço da variável que é enviado à função, não apenas uma cópia de seu valor.

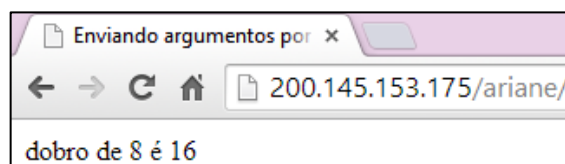
Exemplo 19⁷ – passagem de argumento por referência à função.

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3   <head>
4     <meta charset="utf-8">
5     <title>Enviando argumentos por referência</title>
6   </head>
7   <body>
8     <?php
9       function dobro(&$num) //observe o & na frente do nome da variável
10      {
11        $num*=2;
12      }
13      $a=8;
14      echo "dobro de $a é ";
15      dobro($a); // $a passa a valer 16
16      echo $a;
17    ?>
18  </body>
19 </html>

```

Execução do exemplo 19 é:



⁷ Adaptado de Bruno, Estrozi e Batista Neto (2010, p. 95):

No exemplo 19 a variável \$a teve seu conteúdo alterado porque foi enviada a função por referência. Para que isso acontecesse foi inserido o caractere & (E comercial) na frente do nome da variável que recebeu o valor na função (&\$num).

EXERCÍCIOS

64. Faça uma página HTML na qual o usuário deve um valor no intervalo de 5 a 50. Chame um programa PHP que verifique se o valor está no intervalo. Caso não esteja, retorne à página de origem para redigitação. Se estiver correto, chame uma função para que calcule e apresente a tabuada do número no formato apresentado no exemplo:

Tabuada do número: 7

$$7 * 0 = 0$$

$$7 * 1 = 7$$

.....

$$7 * 10 = 70$$

65. Refaça todos os exercícios de 51 a 54 escrevendo funções para solucionar o que se pede.

7 REFERÊNCIAS BIBLIOGRÁFICAS

BRUNO, O. M.; ESTROZI, L. F.; BATISTA NETO, J. E. S. **Programando para a internet com PHP**, Rio de Janeiro: Brasport, 2010.

FLATSCHART, F. **HTML5: embarque imediato**, Rio de Janeiro: Brasport, 2011.

SAVOIA, H. R. **HTML e CSS + PHP e MySQL**, Ribeirão Preto, SP: IELD, 2013.

The PHP Group. Disponível em <<http://www.php.net/>>. Acesso em: 18 dez. 2013.

W3C. Disponível em <<http://www.w3.org/>>. Acesso em: 18 dez. 2013.

8 SITES ÚTEIS

<http://www.w3c.br/Home/WebHome>

<http://www.w3schools.com/>

http://www.w3schools.com/html/html5_intro.asp

http://validator.w3.org/#validate_by_input

<http://jigsaw.w3.org/css-validator/>

<http://www.php.net/>

http://www.php.net/manual/pt_BR/

<http://codex.wiki.br/Html/Html>

<http://tableless.com.br/html5/>

<http://pt.wikipedia.org/wiki/HTML5>

Boas práticas em PHP: <http://br.phptherightway.com/>