Funções PHP

https://www.php.net/manual/pt_BR/functions.user-defined.php

Funções definidas pelo usuário

- Uma função pode ser definida usando a seguinte sintaxe:

```php
<?php
	function foo($arg_1, $arg_2, /* ..., */ $arg_n)
	{
		echo "Exemplo de função.\n";
		return $valor_retornado;
	}
?>
```

Qualquer código PHP válido pode aparecer dentro de uma função, mesmo outras funções e definições de classes.

- Nomes de funções seguem as mesmas regras que outros nomes no PHP.
- Um nome de função válido começa com uma letra ou um sublinhado, seguido, seguido por qualquer número de letras, números ou sublinhado.
- Com uma expressão regular, seria expressado com:

^[a-zA-Z_\x80-\xff][a-zA-Z0-9_\x80-\xff]*$.

Criação de uma função

- As funções não precisam ser criadas antes de serem referenciadas.
- Uma função que é condicionalmente definida só pode ser usada após a sua validação.
- As regras são aplicadas em qualquer parte do seu código.

Funções definidas condicionalmente

```php
<?php

$criarsalgado = true;

salgado();
doce();

if ($criarsalgado) {
  function salgado()
  {
    echo "salgado é uma função pelo PHP";
  }
}


function doce()
{
  echo "doce é uma função criada pelo PHP";
}

?>
```

Funções dentro de funções e recursividade

```php
<?php
function foo()
{
  function bar()
  {
    echo "Eu não existo até foo() ser chamada.\n";
  }
}

/* Nós não podemos chamar bar() ainda
   porque ela ainda não foi definida. */

foo();

/* Agora nós podemos chamar bar(),
   porque o processamento de foo()
   tornou a primeira acessivel */

bar();

?>
```

```php
<?php
function recursion($a)
{
    if ($a < 20) {
        echo "$a\n";
        recursion($a + 1);
    }
}
?>
```

Criando a função exibir()

```php
/* Criando uma função, você pode criar uma função com parametros, ou
    uma função vazia, sem parametros. o nome da sua função deve sempre
    estar relacioanado ao que fua função vai fazer para um melhor entendimento. */
function exibir()
{  /*as chaves demostram o que vai ser executado na função.  */
    echo 'exibir texto na tela. <br/><hr/>';
}
```

Usando a função exibir dentro de um for

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

Verificando a função!

```
function retorna()
{
    return 'Dias de Glória <br /><hr />';
}
```

Função que retorna um valor calculado.

```php
function calc()
{
    $calc = (3000 / 100) * 80;
    return $calc;
}
echo calc().'<br /><hr />';
```

Argumentos de funções

- Informações podem ser passadas para funções através da lista de argumentos, que é uma lista de expressões delimitados por vírgulas.
- Os argumentos são avaliados da esquerda para a direita, e antes que a função seja efetivamente chamada (avaliação antecipada eager).

- O PHP suporta a passagem de argumentos por valor (o padrão), passagem por referência, e valores padrões de argumentos.
- lista de argumentos de tamanho variável e argumentos nomeados também são suportadas.

Passando arrays para funções

```php
<?php
     function takes_array($input)
     {
     echo "$input[0] + $input[1] = ",
     $input[0]+$input[1];
     }
?>
```

A partir do PHP 8.0.0

- a lista de argumentos de uma função pode incluir uma vírgula final, que será ignorada.
- Isto é particularmente útil nos casos que a quantidade de argumentos é longa ou contém nomes longos, sendo conveniente listar os argumentos verticalmente.

Argumentos de função com uma vírgula final

```php
<?php
     function takes_many_args(
          $primeiro,
          $segundo,
          $uma_variavel_com_nome_longo,
          $parametro_com_default = 5,
          $de_novo = 'argumento padrão', // Essa
vírgula final não era permitida antes do PHP 8.0.0.
)
{
// ...
}
?>
```

https://www.php.net/manual/pt_BR/functions.arguments.php

Calculo com argumento

```php
function calculum($valor, $porcento)
{
    $calc = ($valor / 100) * $porcento;
    return $calc;
}
echo calculum(2000,30).'<br /><hr />';
```

Exercício

Criar uma calculadora utilizando funções com as 4 operações matemáticas

Será utilizado dois argumentos, antes de passar os argumentos você deverá fazer a verificação se a variável dos argumentos existe e se ela não está vazia, está verificação deverá ser feita por uma função.