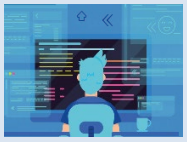


UNIDADE III

ESTRUTURAS DE REPETIÇÃO

OBJETIVOS

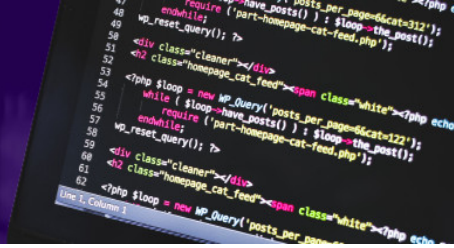
Conhecer as características das estruturas de repetição e em que situações é mais apropriado usá-las. Entender as estruturas de repetição *while, do...while* e *for*.

RESUMO

Aprenderemos nesta aula sobre como estruturar e operar as estruturas de repetição tais como: **while**, que pode manter iterações mediante uma condição lógica verdadeira; **do...while**, semelhante à estrutura while, diferenciando apenas no fato que no laço do...while, sempre será executada a primeira iteração independentemente da condição lógica ser ou não verdadeira; **for**, que executa iterações finitas.

AVALIAÇÃO

Desenvolver algoritmos utilizando a linguagem de desenvolvimento PHP para todas as questões apresentadas no tópico avaliação. Cada questão vale 2,5 ponto, totalizando 10,0 pontos.



ESTRUTURA DE REPETIÇÃO

Normalmente os blocos de códigos são executados várias vezes seguidas. E para diminuir o número de linhas (normalmente iguais e repetidas), utilizamos uma estrutura de repetição (também chamada de laço ou loops ou looping) para realizar uma tarefa. Nesta estrutura, é comum utilizar contadores para alcançar a condição esperada (verdadeira), os contadores podem ser representados de diversas maneiras, desde de números até letras ou palavras). OBS.: O comando **foreach (para cada...faça)** será exemplificado na UNIDADE IV.

Comando "*while*" **ou** "*enquanto...faça*"

while (enquanto...faça) – percorre um bloco de código APENAS enquanto a condição especificada no parênteses for verdadeira. Ou seja, é utilizada quando um determinado bloco de instruções deve ser repetido enquanto uma determinada condição for verdadeira. Para se criar um laço infinito, basta incluir o valor "true" (verdadeiro) dentro da condição (exemplo: while(true) { ... }).

Sintaxe:

```
while (condição) {
 código a ser executado;
}
```

Exemplo:

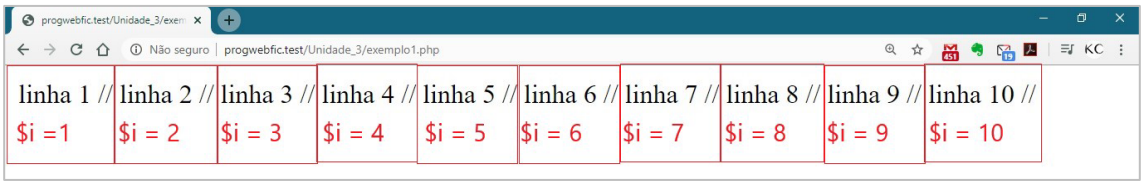
```php
<?php

$i=1; // contator

// while (condição) {
// enquanto a condição for verdadeira, o bloco continuará sendo executado
while($i <= 10) { // enquanto $i for menor ou igual que 10
    echo "linha " . $i . " // "; // saída exibida enquanto o bloco for executado
    $i++; // enquanto o bloco for executado o valor da variável $i será soamdo com +1
} // fim do bloco while

?>
```

Resultado:



Comando "*do ... while*" **ou** "*faça ... enquanto*"

do ... while (faça...enquanto) – percorre um bloco de código pelo menos uma vez e depois verifica se a condição especificada no parênteses é verdadeira. Caso seja, repete o ciclo enquanto a condição especificada for verdadeira.

Sintaxe:

```
do {
   código a ser executado;
} while (condição);
```

Exemplo:

```php
<?php

$i=1; // contator

// do {
// executa pelo menos uma vez e depois continua enquanto a condição for verdadeira
do { // neste exemplo o bloco continua sendo executado, pois a condição é verdadeira até $i <= 2
    echo "linha " . $i . " // "; // saída exibida enquanto o bloco for executado
    $i++; // enquanto o bloco for executado o valor da variável $i será soamdo com +1
} while ($i <= 2); // fim do bloco while

echo "<br><br><br>";

// outro exemplo
echo "Outro exemplo:<br>";

$j=1; // contator

// executa pelo menos uma vez e depois continua enquanto a condição for verdadeira
do { // neste exemplo o bloco encerra, pois a condição é falsa, mesmo somando +1
    echo "linha " . $j . " // ";
    $j++;
} while ($j == 3); // fim do bloco while

?>
```

Resultado:

Comando "*for*" **ou** "*para ... faça*"

for (para...faça) – percorre um bloco de código um determinado número de vezes, ou seja, o laço é usado quando você sabe de antemão quantas vezes o script deve ser executado.

Sintaxe:

```
for (inicialização; condição; incremento) {
 código a ser executado;
}
```

OBSERVAÇÃO: Cada um dos parâmetros ao lado pode ser vazio ou ter múltiplas expressões. Os parâmetros da estrutura são:

- *inicialização*: geralmente usado para definir um contador (inicialização da variável de controle);
- *condição*: avalia para cada iteração do loop. Se for avaliado como TRUE, o loop continuará a ser executado. Se for avaliado como FALSE, o loop termina.
- *incremento*: geralmente usado para incrementar um contador (mas pode ser qualquer código a ser executado no final do loop).

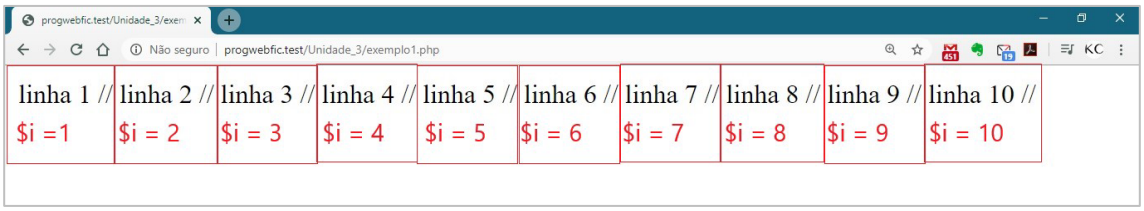
Exemplo:

```php
<?php

// for (inicialização; condição; incremento) {
// enquanto a condição for verdadeira, o bloco continuará sendo executado
for ($i = 1; $i <= 10; $i++) { // enquanto $i for menor ou igual que 10, será somado +1
    echo "linha " . $i . " // "; // saída exibida enquanto o bloco for executado
} // fim do bloco for

?>
```

Resultado:



Comandos "*break*" **e** "*continue*"

Os comandos break e continue oferecem maior controle sobre as estruturas for, foreach, while, do-while ou switch, alterando o fluxo de controle para a próxima iteração (continue) ou encerrando todo ciclo de repetições (break).

continue – é utilizado em estruturas de laço para pular o resto da iteração atual, e continuar a execução na validação da condição e, então, iniciar a próxima iteração. Ou seja, pula a iteração atual, ignorando todo o código declarado abaixo dele, dentro da estrutura de repetição. Geralmente utilizamos if/else para determinar o momento em que esse comando deverá ser executado.

Sintaxe:

```
continue; // a partir daqui, pula a iteração atual, ignorando todo o código declarado abaixo.
```

Exemplo:

```php
<?php

/** Exemplo do Comando "Continue";

Exibir os números de 1 a 10. Observação: não deve ser exibido os números cuja a divisão por 3 seja exata, ou seja, sem resto.

*/

// for (inicialização; condição; incremento/decremento) {
// enquanto a condição for verdadeira, o bloco continuará sendo executado
for ($i = 1; $i <= 10; $i++) { // enquanto $i for menor ou igual que 10, será somado +1
    if($i % 3 == 0){ // aqui verifica se o número ($i) dividido por 3 possui ou não resto
        // caso a condição seja verdadeira, o restante do código (a partir daqui) não será executado para este loop
        // ou seja, a linha 18 não será executada e nem as demais (caso existissem), porém $i será somado com +1
        // e o programa volta a analisar a linha 12 desde que a condição ($i <= 10) confinue verdadeira
        continue;
    }
    echo "$i "; // saída exibida enquanto o bloco for executado (verdadeiro)
} // fim do bloco for

// Resultado: 1 2 4 5 7 8 10

?>
```

Resultado:

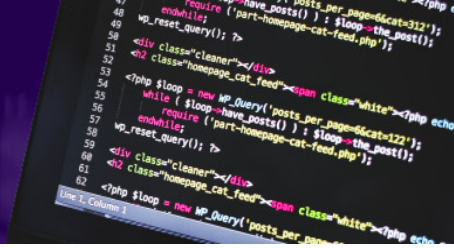
break – finaliza a execução da estrutura for, foreach, while, do-while ou switch atual, ou seja, encerra uma estrutura de repetição imediatamente. Geralmente usamos if/else para determinar o momento em que essa condição de encerramento será executada. OBSERVAÇÃO: Este comando não será exemplificado, pois tratamos dele na unidade anteior (UNIDADE II).

Sintaxe:

```
break; // a partir daqui, a execução é encerrada, ignorando todo o código declarado abaixo.
```

Prática

Refaça todos os exemplos apresentados nesta UNIDADE.



EXERCÍCIO DE FIXAÇÃO

OBSERVAÇÃO:

Faça as questões 1 a 5, abaixo, três vezes. Para cada solução utilize uma estrutura de repetição diferente (while, do...while e for);

Se preferir, poderá utilizar a estrutura de interação (entrada de dados) proposta na unidade anterior.

1) Faça um algoritmo em PHP para mostrar a tabuada (multiplicação) de 0 a 10 do número 6.

2) Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números pares.

3) Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números ímpares;

4) Faça um algoritmo em PHP que conte de 1 a 20 e some os valores cuja a divisão por 4 seja exata, ou seja, sem resto. Mostre os números que serão somados e no final exiba o valor total da soma, utilizar a quebra de linha (*tag* "
") para apresentar os valores.

5) Faça um algoritmo em PHP para ler o alfabeto (utilizar estrutura de repetição) e exibir apenas as vogais (utilizar a estrutura de seleção "switch"). OBSERVAÇÃO: A execução da estrutura de repetição deve parar quando for identificado a letra "o", avisar o usuário que o algoritmo encerrou e informar o número de consoantes registradas antes da execução parar.

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - WHILE:

1) Faça um algoritmo em PHP para mostrar a tabuada (multiplicação) de 0 a 10 do número 6.

*/

# Declaração de variáveis
$i = 0;
$tabuada = 0;

# Cálculo
while ($i <= 10) {
    $tabuada = $i * 6;
    echo "$i x 6 = $tabuada <br>";
    $i++;
}

/** RESULTADO:
0 x 6 = 0
1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - DO ... WHILE:

1) Faça um algoritmo em PHP para mostrar a tabuada (multiplicação) de 0 a 10 do número 6.

*/

# Declaração de variáveis
$i = 0;
$tabuada = 0;

# Cálculo
do {
    $tabuada = $i * 6;
    echo "$i x 6 = $tabuada <br>";
    $i++;
} while ($i <= 10);

/** RESULTADO:
0 x 6 = 0
1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - FOR:

1) Faça um algoritmo em PHP para mostrar a tabuada (multiplicação) de 0 a 10 do número 6.

*/

# Declaração de variáveis
$tabuada = 0;

# Cálculo
for ($i=0; $i <= 10 ; $i++) {
    $tabuada = $i * 6;
    echo "$i x 6 = $tabuada <br>";
}

/** RESULTADO:
0 x 6 = 0
1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - WHILE:

2)  Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números pares.

*/

# Declaração de variáveis
$i = 0;

# Cálculo
while ($i++ <= 10) { // aqui estou utilizando o contador na condição ao invés de utilizar no final do bloco
    if($i % 2 == 0) { // A divisão por 2 que não possuir resto será par.
        echo "número par: $i <br>";
    }
}

/** RESULTADO:

número par: 2
número par: 4
número par: 6
número par: 8
número par: 10

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - DO ... WHILE:

2)  Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números pares.

*/

# Declaração de variáveis
$i = 1;

# Cálculo
do {
    if($i % 2 == 0) { // A divisão por 2 que não possuir resto será par.
        echo "número par: $i <br>";
    }
} while ($i++ <= 10); // aqui estou utilizando o contador na condição ao invés de utilizar no final do bloco

/** RESULTADO:

número par: 2
número par: 4
número par: 6
número par: 8
número par: 10

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - FOR:

2)  Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números pares.

*/

# Declaração de variáveis

# Cálculo
for ($i=1; $i <= 10 ; $i++) {
    if($i % 2 == 0) { // A divisão por 2 que não possuir resto será par.
        echo "número par: $i <br>";
    }
}

/** RESULTADO:

número par: 2
número par: 4
número par: 6
número par: 8
número par: 10

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - WHILE:

3)  Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números ímpares;

*/

# Declaração de variáveis
$i = 1;

# Cálculo
while ($i <= 10) {
    if($i % 2 > 0) { // A divisão por 2 que possuir resto será ímpar.
        echo "número ímpar: $i <br>";
    }
    $i++; // aqui estou utilizando o contador no final do bloco
}

/** RESULTADO:

número ímpar: 1
número ímpar: 3
número ímpar: 5
número ímpar: 7
número ímpar: 9

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - DO ... WHILE:

3)  Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números ímpares;

*/

# Declaração de variáveis
$i = 1;

# Cálculo
do {
    if($i % 2 > 0) { // A divisão por 2 que possuir resto será ímpar.
        echo "número ímpar: $i <br>";
    }
    $i++; // aqui estou utilizando o contador no final do bloco
} while ($i <= 10);

/** RESULTADO:

número ímpar: 1
número ímpar: 3
número ímpar: 5
número ímpar: 7
número ímpar: 9

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - FOR:

3)  Faça um algoritmo em PHP que conte de 1 a 10 e mostre os números ímpares;

*/

# Declaração de variáveis

# Cálculo
for ($i=1; $i <= 10 ; $i++) {
    if($i % 2 > 0) { // A divisão por 2 que possuir resto será ímpar.
        echo "número ímpar: $i <br>";
    }
}

/** RESULTADO:

número ímpar: 1
número ímpar: 3
número ímpar: 5
número ímpar: 7
número ímpar: 9

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - WHILE:

4)  Faça um algoritmo em PHP que conte de 1 a 20 e some os valores cuja a divisão por 4 seja exata, ou seja, sem resto. Mostre os números que serão somados e no final exiba o valor total da soma, utilizar a quebra de linha (tag "<br>") para apresentar os valores.

*/

# Declaração de variáveis
$i = 0;
$soma = 0;

# Cálculo
while ($i++ <= 20) {
    if($i % 4 == 0) {
        echo "O número que será somado: $i <br>";
        $soma += $i; // soma por atribuição, é o mesmo que: $soma = $soma + $i.
    }
}

echo "------------------------------------ <br>";
echo "O valor total da soma é: $soma <br>";

/** RESULTADO:

O número que será somado: 4
O número que será somado: 8
O número que será somado: 12
O número que será somado: 16
O número que será somado: 20
------------------------------------
O valor total da soma é: 60

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - DO ... WHILE:

4)  Faça um algoritmo em PHP que conte de 1 a 20 e some os valores cuja a divisão por 4 seja exata, ou seja, sem resto. Mostre os números que serão somados e no final exiba o valor total da soma, utilizar a quebra de linha (tag “<br>”) para apresentar os valores.

*/

# Declaração de variáveis
$i = 1;
$soma = 0;

# Cálculo
do {
    if($i % 4 == 0) {
        echo "O número que será somado: $i <br>";
        $soma += $i; // soma por atribuição, é o mesmo que: $soma = $soma + $i.
    }
} while ($i++ <= 20);

echo "------------------------------------ <br>";
echo "O valor total da soma é: $soma <br>";

/** RESULTADO:

O número que será somado: 4
O número que será somado: 8
O número que será somado: 12
O número que será somado: 16
O número que será somado: 20
------------------------------------
O valor total da soma é: 60

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - FOR:

4)  Faça um algoritmo em PHP que conte de 1 a 20 e some os valores cuja a divisão por 4 seja exata, ou seja, sem resto. Mostre os números que serão somados e no final exiba o valor total da soma, utilizar a quebra de linha (tag “<br>”) para apresentar os valores.

*/

# Declaração de variáveis
$soma = 0;

# Cálculo
for ($i=1; $i < 21 ; $i++) {
    if($i % 4 == 0) {
        echo "O número que será somado: $i <br>";
        $soma += $i; // soma por atribuição, é o mesmo que: $soma = $soma + $i.
    }
}

echo "------------------------------------ <br>";
echo "O valor total da soma é: $soma <br>";

/** RESULTADO:

O número que será somado: 4
O número que será somado: 8
O número que será somado: 12
O número que será somado: 16
O número que será somado: 20
------------------------------------
O valor total da soma é: 60

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - WHILE:

5)  Faça um algoritmo em PHP para ler o alfabeto (utilizar estrutura de repetição) e exibir apenas as vogais (utilizar a estrutura de seleção "switch"). OBSERVAÇÃO: A execução da estrutura de repetição deve parar quando for identificado a letra “o”, avisar o usuário que o algoritmo encerrou e informar o número de consoantes registradas antes da execução parar.

*/

# Declaração de variáveis
$abc = 'a'; // declaração da variável que vai receber as letras do alfabeto
$nConsoante = 0; // declaração da variável para contar o número de consoantes

# Cálculo
while ($abc <= 'z') { // Laço de repetição que percorre as letras, de A a Z
    switch ($abc) { // Para cada letra ($abc) esta estrutura será executda e analisada
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u': // Para identificar as vogais, isto é, caso seja 'a', 'e', 'i', 'o' ou 'u'
            echo "Vogal: $abc <br>"; // Informa que é vogal
            break; // Este "bleak" finaliza a estrutura de seleção "switch"

        default: // Caso não seja vogal, irá consedirar que seja consoante
            $nConsoante++; // Conta o número de consoantes
            break; // Este "bleak" finaliza a estrutura de seleção "switch"
    } // fim do bloco

    if($abc == "o") { // Localiza a letra "o"
        echo "<br>### Execução ENCERRADA! ### <br><br>"; // Exige a mensagem que o sistema parou
        break; // Este "bleak" finaliza a estrutura de repetição "while"
    } // fim da estrutura de comporação

    $abc++; // incremento da letra: a, b, c, d ... z

} // fim da estrutura de repetição


echo "Número de consoantes registradas: $nConsoante";

/** RESULTADO:

Vogal: a
Vogal: e
Vogal: i
Vogal: o

### Execução ENCERRADA! ###

Número de consoantes registradas: 11

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - DO ... WHILE:

5)  Faça um algoritmo em PHP para ler o alfabeto (utilizar estrutura de repetição) e exibir apenas as vogais (utilizar a estrutura de seleção "switch"). OBSERVAÇÃO: A execução da estrutura de repetição deve parar quando for identificado a letra “o”, avisar o usuário que o algoritmo encerrou e informar o número de consoantes registradas antes da execução parar.

*/

# Declaração de variáveis
$abc = 'a'; // declaração da variável que vai receber as letras do alfabeto
$nConsoante = 0; // declaração da variável para contar o número de consoantes

# Cálculo
do{
    switch ($abc) { // Para cada letra ($abc) esta estrutura será executda e analisada
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u': // Para identificar as vogais, isto é, caso seja 'a', 'e', 'i', 'o' ou 'u'
            echo "Vogal: $abc <br>"; // Informa que é vogal
            break; // Este "bleak" finaliza a estrutura de seleção "switch"

        default: // Caso não seja vogal, irá consedirar que seja consoante
            $nConsoante++; // Conta o número de consoantes
            break; // Este "bleak" finaliza a estrutura de seleção "switch"
    } // fim do bloco

    if($abc == "o") { // Localiza a letra "o"
        echo "<br>### Execução ENCERRADA! ### <br><br>"; // Exige a mensagem que o sistema parou
        break; // Este "bleak" finaliza a estrutura de repetição "while"
    } // fim da estrutura de comporação

    $abc++; // incremento da letra: a, b, c, d ... z

} while ($abc <= 'z'); // Laço de repetição que percorre as letras, de A a Z


echo "Número de consoantes registradas: $nConsoante";

/** RESULTADO:

Vogal: a
Vogal: e
Vogal: i
Vogal: o

### Execução ENCERRADA! ###

Número de consoantes registradas: 11

*/

?>
```

```php
<?php

/** UNIDADE III - Exercício de Fixação

# ESTRUTURA DE REPETIÇÃO - WHILE:

5)  Faça um algoritmo em PHP para ler o alfabeto (utilizar estrutura de repetição) e exibir apenas as vogais (utilizar a estrutura de seleção "switch"). OBSERVAÇÃO: A execução da estrutura de repetição deve parar quando for identificado a letra “o”, avisar o usuário que o algoritmo encerrou e informar o número de consoantes registradas antes da execução parar.

*/

# Declaração de variáveis
$nConsoante = 0; // declaração da variável para contar o número de consoantes

# Cálculo

// $abc='a' declaração da variável que vai receber as letras do alfabeto
// $abc++ incremento da letra: a, b, c, d ... z
for ($abc='a'; $abc <= 'z' ; $abc++) { // Laço de repetição que percorre as letras, de A a Z
    switch ($abc) { // Para cada letra ($abc) esta estrutura será executda e analisada
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u': // Para identificar as vogais, isto é, caso seja 'a', 'e', 'i', 'o' ou 'u'
            echo "Vogal: $abc <br>"; // Informa que é vogal
            break; // Este "bleak" finaliza a estrutura de seleção "switch"

        default: // Caso não seja vogal, irá consedirar que seja consoante
            $nConsoante++; // Conta o número de consoantes
            break; // Este "bleak" finaliza a estrutura de seleção "switch"
    } // fim do bloco

    if($abc == "o") { // Localiza a letra "o"
        echo "<br>### Execução ENCERRADA! ### <br><br>"; // Exige a mensagem que o sistema parou
        break; // Este "bleak" finaliza a estrutura de repetição "for"
    } // fim da estrutura de comporação

} // fim da estrutura de repetição

echo "Número de consoantes registradas: $nConsoante";

/** RESULTADO:

Vogal: a
Vogal: e
Vogal: i
Vogal: o

### Execução ENCERRADA! ###

Número de consoantes registradas: 11

*/

?>
```