

TRƯỜNG ĐẠI HỌC NGÂN HÀNG TP. HCM



## MÔN HỌC: LẬP TRÌNH WEB

### CHƯƠNG 3 NGÔN NGỮ LẬP TRÌNH PHP



1. Giới thiệu
2. Biến, Kiểu dữ liệu và toán tử
3. Cấu trúc điều kiện
4. Cấu trúc lặp
5. Hàm
6. Nhúng PHP vào HTML

## NGÔN NGỮ LẬP TRÌNH PHP

- PHP (PHP: Hypertext PreProcessor) được tạo ra bởi Rasmus Lerdorf vào năm 1994, là một ngôn ngữ kịch bản (thông dịch) miễn phí.
- Được sử dụng kết hợp với HTML, CSS, Javascript để tạo ra các website động.
- Có thể chạy trên nhiều hệ điều hành: Windows, Linux, Unix, MacOS, . . . và hỗ trợ kết nối với nhiều hệ quản trị CSDL: Oracle, SQL Server, MySQL, MongoDB, . . .

## TẬP TIN PHP

- Nội dung trong một tập tin PHP gồm văn bản, HTML, CSS, Javascript và các đoạn mã PHP.
- Các đoạn mã PHP được thực thi, xử lý tại máy chủ (server-side) và kết quả được trả về cho trình duyệt dưới dạng HTML thuần túy.
- Tập tin PHP có phần mở rộng là **.php**

## ƯU ĐIỂM VÀ NHƯỢC ĐIỂM CỦA PHP

### ▪ Ưu điểm:

- Dễ học, dễ sử dụng.
- Có cộng đồng hỗ trợ lớn.
- Cung cấp nhiều framework và thư viện hỗ trợ.

### ▪ Nhược điểm:

- Không đóng gói được mã nguồn.
- Chỉ hỗ trợ xây dựng các ứng dụng web.

## MỘT SỐ ĐẶC ĐIỂM CỦA PHP

- Không cần kiểu dữ liệu khi thực hiện khai báo biến, biến sẽ nhận dạng kiểu dữ liệu dựa vào giá trị được gán.
- Tên biến bắt buộc phải bắt đầu bằng ký tự \$.
- Các đoạn mã PHP đặt trong cặp thẻ:

**<?php**

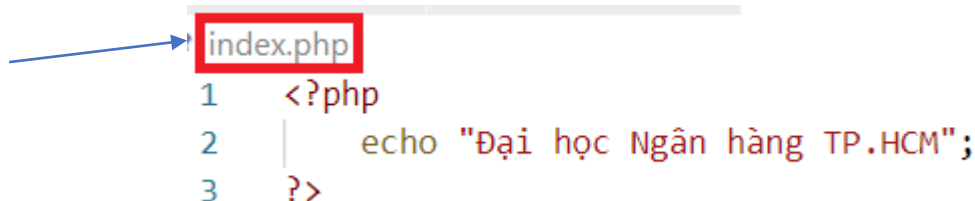
**//Các đoạn mã PHP**

**?>**

## CHẠY CHƯƠNG TRÌNH PHP ĐƠN GIẢN

- Sử dụng các trình soạn thảo văn bản (notepad, notepad++) hoặc các IDE và đặt vào các đoạn mã PHP trong cặp thẻ `<?php ?>` → Lưu file với phần mở rộng là **.php**

Tên file với phần mở rộng là **.php**

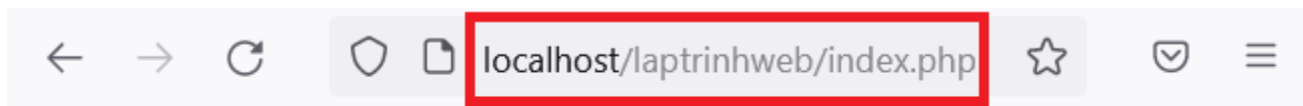


```
index.php
1  <?php
2      echo "Đại học Ngân hàng TP.HCM";
3  ?>
```

- Đặt file \*.php trong thư mục: “**<đường dẫn thư mục cài đặt Wampp>/www**”

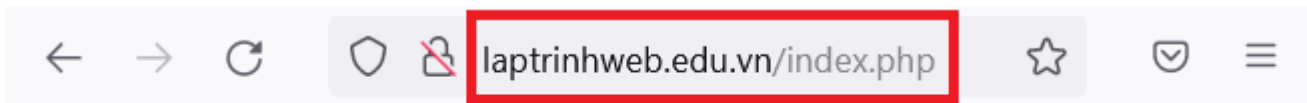
## CHẠY CHƯƠNG TRÌNH PHP ĐƠN GIẢN

- Mở trình duyệt và chạy chương trình



Đại học Ngân hàng TP.HCM

Chạy file mã nguồn [index.php](#) được đặt trong thư mục [laptrinhweb](#)



Đại học Ngân hàng TP.HCM

Chạy file mã nguồn [index.php](#) với Virtual Host



## QUY TẮC ĐẶT TÊN BIẾN

- Tên biến phải bắt đầu bằng dấu \$, theo sau là một ký tự chữ cái alphabet hoặc ký tự \_
- Tên biến chỉ chứa các ký tự a-z, A-Z, 0-9, \_
- Tên biến không được có khoảng trắng, trường hợp tên biến có nhiều từ có thể sử dụng dấu \_ để ngăn cách giữa các từ.

Ví dụ: **\$ho\_ten**

- Tên biến phân biệt chữ hoa, thường.

Ví dụ: \$diem\_so khác \$Diem\_So

## KIỂU DỮ LIỆU

- Mặc dù không cần kiểu dữ liệu khi khai báo biến nhưng biến sẽ tự nhận dạng kiểu dữ liệu dựa vào giá trị được gán.
- Một số kiểu dữ liệu trong PHP:
  - String
  - Integer
  - Float
  - Boolean
  - Array
  - Object
  - NULL

## KIỂU DỮ LIỆU STRING

- Chuỗi được đặt trong cặp dấu " " hoặc ' '.

Lệnh **echo** dùng để xuất dữ liệu ra màn hình

```
<?php $x = "Đại học Ngân hàng TP.HCM";
```

```
$y = 'Đại học Ngân hàng TP.HCM';
```

```
echo $x;
```

```
echo "<br/>";
```

Lệnh **var\_dump** dùng để xuất cấu trúc của biến ra màn hình

```
var_dump($y);
```

```
?>
```

## KIỂU DỮ LIỆU SỐ

- Integer

```
<?php  
    $year = 2022;  
    var_dump($year);  
?>
```

- Float

```
<?php  
    $x = 10.365;  
    var_dump($x);  
?>
```

## KIỂU DỮ LIỆU BOOLEAN

```
<?php
```

```
    $x = true;
```

```
    $y = false;
```

```
    echo $x;
```

```
    echo "<br/>";
```

```
    var_dump($y);
```

```
?>
```

## KIỂU DỮ LIỆU ARRAY

- Kiểu Array chứa nhiều giá trị trong một biến.
- Trường hợp không định nghĩa chỉ số cho mảng thì mặc định chỉ số bắt đầu từ **0**

Chỉ số

0

1

2



<?php

```
$cars = array("Volvo", "BMW", "Toyota");
```

```
var_dump($cars);
```

?>

## KIỂU DỮ LIỆU ARRAY

- Có thể định nghĩa chỉ số cho biến kiểu Array

```
<?php
```

```
$employee = array(  
    "Name" => "Alex",  
    "Email" => "alex_jtp@gmail.com",  
    "Age" => 21,  
    "Gender" => "Male");
```

```
?>
```

## KIỂU DỮ LIỆU NULL

- Kiểu NULL là kiểu dữ liệu đặc biệt.
- Một biến khi không được khởi tạo giá trị sẽ được gán giá trị là NULL.

```
<?php
```

```
$x = "Hello";
```

```
$x = null;
```

```
var_dump($x);
```

```
?>
```



## CHUYỂN ĐỔI KIỂU DỮ LIỆU

Cú pháp	Mô tả	Ví dụ
(int) (integer)	Chuyển sang kiểu số nguyên	<pre>&lt;?php     \$x = "1";     \$y = "1.5";     \$z = (integer) \$x;     \$t = (int) \$y;     var_dump(\$z);     echo "&lt;br/&gt;";     var_dump(\$t); ?&gt;</pre>
(float) (double) (real)	Chuyển sang kiểu số thực	<pre>&lt;?php     \$x = "1.5";     \$y = "1.7";     \$z = (float) \$x;     \$t = (double) \$y;     var_dump(\$z);     echo "&lt;br/&gt;";     var_dump(\$t); ?&gt;</pre>

## CHUYỂN ĐỔI KIỂU DỮ LIỆU

Cú pháp	Mô tả	Ví dụ
(bool) (boolean)	Chuyển sang kiểu số boolean	<pre>&lt;?php     \$x = "1";     \$y = "0";     \$z = (bool) \$x;     \$t = (boolean) \$y;     var_dump(\$z);     echo "&lt;br/&gt;";     var_dump(\$t); ?&gt;</pre>
(string)	Chuyển sang kiểu chuỗi	<pre>&lt;?php     \$x = 1;     \$z = (string) \$x;     \$t = (array) \$x;     var_dump(\$z);     echo "&lt;br/&gt;";     var_dump(\$t); ?&gt;</pre>
(array)	Chuyển sang kiểu mảng	

## HẰNG

- Hằng tương tự như biến dùng để lưu trữ một giá trị. Tuy nhiên giá trị của hằng không thể thay đổi được
- Tên hằng không cần dấu \$ và bắt đầu bằng một chữ cái hoặc dấu \_
- Sử dụng từ khóa **define** để khai báo hằng.

```
define("ROOT_LOCATION", "/usr/local/www/");
```

## TOÁN TỬ

- Được sử dụng để thực hiện các phép toán trên các biến và các giá trị.
- Trong PHP toán tử có thể chia thành các nhóm:
  - Toán tử toán học
  - Toán tử gán
  - Toán tử so sánh
  - Toán tử luận lý
  - Toán tử tăng/giảm
  - Toán tử trên chuỗi
  - Các toán tử khác

## TOÁN TỬ TOÁN HỌC

Toán tử	Mô tả	Ví dụ
+	Cộng hai toán hạng	$\$x + \$y$
-	Trừ hai toán hạng	$\$x - \$y$
*	Nhân hai toán hạng	$\$x * \$y$
/	Chia hai toán hạng	$\$x / \$y$
%	Lấy phần dư của phép chia nguyên	$\$x \% \$y$
**	Lũy thừa	$\$x ** \$y (x^y)$

## TOÁN TỬ GÁN

Toán tử	Ví dụ
=	$\$x = 5$ (gán giá trị 5 cho biến $\$x$ )
+=	$\$x += 3$ tương đương $\$x = \$x + 3$
-=	$\$x -= 3$ tương đương $\$x = \$x - 3$
*=	$\$x *= 3$ tương đương $\$x = \$x * 3$
/=	$\$x /= 3$ tương đương $\$x = \$x / 3$
%=	$\$x \% = 3$ tương đương $\$x = \$x \% 3$

## TOÁN TỬ SO SÁNH

Toán tử	Mô tả	Ví dụ	Kết quả
<code>==</code>	Kiểm tra bằng nhau	<code>\$x == \$y</code>	Trả về True nếu \$x bằng \$y (\$x và \$y có thể khác kiểu dữ liệu)
<code>===</code>	Kiểm tra bằng nhau đồng nhất	<code>\$x === \$y</code>	Trả về True nếu \$x bằng \$y và \$x cùng kiểu dữ liệu với \$y
<code>!= (&lt;&gt;)</code>	Kiểm tra khác nhau	<code>\$x != \$y</code>	Trả về True nếu \$x không bằng \$y
<code>!==</code>	Kiểm tra không đồng nhất	<code>\$x !== \$y</code>	Trả về True nếu \$x không bằng \$y hoặc \$x khác kiểu dữ liệu với \$y

## TOÁN TỬ SO SÁNH

Toán tử	Mô tả	Ví dụ	Kết quả
$>$	Kiểm tra lớn hơn	$\$x > \$y$	Trả về True nếu $\$x$ lớn hơn $\$y$
$<$	Kiểm tra nhỏ hơn	$\$x < \$y$	Trả về True nếu $\$x$ nhỏ hơn $\$y$
$>=$	Kiểm tra lớn hơn hoặc bằng	$\$x >= \$y$	Trả về True nếu $\$x$ lớn hơn hoặc bằng $\$y$
$<=$	Kiểm tra nhỏ hơn hoặc bằng	$\$x <= \$y$	Trả về True nếu $\$x$ nhỏ hơn hoặc bằng $\$y$



## TOÁN TỬ TĂNG/GIẢM

Toán tử	Mô tả	Ví dụ	Kết quả
<code>++\$x</code>	Tăng giá trị biến <code>\$x</code> lên một đơn vị và trả về kết quả	<code>\$x = 0;</code> <code>\$t = ++\$x;</code>	<code>\$t = 1;</code>
<code>\$x++</code>	Trả về giá trị <code>\$x</code> và tăng giá trị biến <code>\$x</code> lên một đơn vị	<code>\$x = 0;</code> <code>\$t = \$x++;</code>	<code>\$t = 0;</code>
<code>--\$x</code>	Giảm giá trị biến <code>\$x</code> lên một đơn vị và trả về kết quả	<code>\$x = 0;</code> <code>\$t = --\$x;</code>	<code>\$t = -1;</code>
<code>\$x--</code>	Trả về giá trị <code>\$x</code> và giảm giá trị biến <code>\$x</code> lên một đơn vị	<code>\$x = 0;</code> <code>\$t = \$x--;</code>	<code>\$t = 0;</code>

## TOÁN TỬ LOGIC

Toán tử	Mô tả	Ví dụ
&&	Trả về kết quả là True khi cả hai biểu thức đều True	<pre>&lt;?php     \$x = 10;    \$y = 5;     if (\$x == 10 &amp;&amp; \$y == 5)         echo "Hello world!"; ?&gt;</pre>
	Trả về kết quả là True nếu một trong hai biểu thức là True	<pre>&lt;?php     \$x = 10;    \$y = 8;     if (\$x == 10    \$y == 5)         echo "Hello world!"; ?&gt;</pre>
!	Đảo ngược giá trị toán hạng, từ True sang False và ngược lại	<pre>&lt;?php     \$x = 8;     if (!( \$x == 7 ))         echo "Hello world!"; ?&gt;</pre>

## TOÁN TỬ LOGIC

Toán tử	Mô tả	Ví dụ
and	Trả về kết quả là True khi cả hai biểu thức đều True	<pre>&lt;?php     \$x = 10;    \$y = 5;     if (\$x == 10 and \$y == 5)         echo "Hello world!"; ?&gt;</pre>
or	Trả về kết quả là True nếu một trong hai biểu thức là True	<pre>&lt;?php     \$x = 10;    \$y = 8;     if (\$x == 10 or \$y == 5)         echo "Hello world!"; ?&gt;</pre>
xor	Trả về kết quả là True khi hai biểu thức có kết quả khác nhau	<pre>&lt;?php     \$x = 10;    \$y = 5;     if (\$x == 10 xor \$y == 5)         echo "Hello world!";     else         echo "ĐHNNH"; ?&gt;</pre>

## TOÁN TỬ LOGIC

Toán tử	Mô tả	Ví dụ
and	Trả về kết quả là True khi cả hai biểu thức đều True	<pre>&lt;?php     \$x = 10;    \$y = 5;     if (\$x == 10 and \$y == 5)         echo "Hello world!"; ?&gt;</pre>
or	Trả về kết quả là True nếu một trong hai biểu thức là True	<pre>&lt;?php     \$x = 10;    \$y = 8;     if (\$x == 10 or \$y == 5)         echo "Hello world!"; ?&gt;</pre>
xor	Trả về kết quả là True khi hai biểu thức có kết quả khác nhau	<pre>&lt;?php     \$x = 10;    \$y = 5;     if (\$x == 10 xor \$y == 5)         echo "Hello world!";     else         echo "ĐHNNH"; ?&gt;</pre>

## TOÁN TỬ TRÊN CHUỖI

Toán tử	Mô tả	Ví dụ
.	Nối chuỗi	<pre>&lt;?php     \$text1 = "Hello ";     \$text2 = "world";     \$text = \$text1.\$text2; ?&gt;</pre>
.=	<p>\$txt1 .= txt2 tương đương \$txt1 = \$txt1. \$txt2</p>	<pre>&lt;?php     \$text1 = "Hello ";     \$text2 = "world";     \$text1 .= \$text2; ?&gt;</pre>

## TOÁN TỬ KHÁC

- Toán tử 3 ngôi
- Cú pháp:

**condition ? result1 : result2**

- Nếu condition là True thì trả về giá trị result1, ngược lại trả về giá trị result2
- Ví dụ:

```
<?php
    $x = 5;
    $y = 10;
    $max = $x > $y ? $x : $y;
?>
```

## Câu lệnh: **if**

- Cú pháp:

```
if (<biểu thức điều kiện>) {  
    <các câu lệnh nếu biểu thức điều kiện đúng>  
}
```

## Câu lệnh: if

■ Ví dụ: 

```
<?php
    $x = 15; $y = 10;
    if($x > $y){
        echo "Giá trị lớn hơn là ".$x;
    }
?>
```

```
<?php
    $x = 15; $y = 10; $z = 5;
    if($x > $y && $x > $z)
    {
        echo "Giá trị lớn nhất là ".$x;
    }
?>
```



## Câu lệnh: **if - else**

- Cú pháp:

```
if (<biểu thức điều kiện>) {  
    <các câu lệnh nếu biểu thức điều kiện đúng>  
}  
  
else  
{  
    <các câu lệnh nếu biểu thức điều kiện sai>  
}
```

## Câu lệnh: if - else

- Ví dụ:

```
<?php
    $x = 15; $y = 10;
    if($x > $y){
        echo "Giá trị lớn hơn là ".$x;
    }
    else
    {
        echo "Giá trị lớn hơn là ".$y;
    }
?>
```

## Câu lệnh: if - elseif - else

- Cú pháp:

if (<biểu thức điều kiện 1>) {

    <các câu lệnh nếu biểu thức điều kiện 1 **đúng**>

}

elseif (<biểu thức điều kiện 2>) {

    <các câu lệnh nếu biểu thức điều kiện 1 **sai** và biểu thức điều kiện 2 **đúng**>

}

else {

    <các câu lệnh nếu biểu thức điều kiện 1 **sai** và biểu thức điều kiện 2 **sai**>

}

## Câu lệnh: if - elseif - else

- Ví dụ:

```
<?php
    $x = 15; $y = 10;
    if($x > $y){
        echo "Giá trị lớn hơn là ".$x;
    }
    elseif($x < $y){
        echo "Giá trị lớn hơn là ".$y;
    }
    else{
        echo "Hai giá trị bằng nhau";
    }
?>
```

## Câu lệnh: **switch - case**

- Cú pháp: **<?php**

```
switch (<mã trường hợp>) {  
    case <mã 1>:  
        <câu lệnh 1>  
        break;  
    case <mã 2>:  
        <câu lệnh 2>  
        break;  
    case <mã N>:  
        <câu lệnh N>  
        break;  
    default:  
        <câu lệnh mặc định>  
}
```

**?>**

## Câu lệnh: **switch - case**

### ■ Ví dụ:

```
<?php
```

```
    $favcolor = "red";
```

```
    switch ($favcolor) {
```

```
        case "red":
```

```
            echo "Your favorite color is red!";
```

```
            break;
```

```
        case "blue":
```

```
            echo "Your favorite color is blue!";
```

```
            break;
```

```
        default:
```

```
            echo "Your favorite color is neither red nor blue!";
```

```
    }
```

```
?>
```

## Câu lệnh: **while**

- Cú pháp:

```
while (<biểu thức điều kiện>) {  
    <các câu lệnh nếu biểu thức điều kiện đúng>  
}
```

- Lưu ý: Trong vòng lặp **while** phải có câu lệnh thay đổi giá trị của biểu thức điều kiện để tránh việc lặp vô hạn

## Câu lệnh: **while**

### ▪ Nguyên tắc thực thi

(1) Chương trình kiểm tra biểu thức điều kiện

(2) Nếu điều kiện là true

- Thực thi câu lệnh
- Quay lại kiểm tra điều kiện tại bước (2) => Trong câu lệnh thực thi phải có phép toán thay đổi biểu thức điều kiện để chương trình không lặp vô hạn.


(3) Nếu điều kiện false chương trình đi đến câu lệnh theo sau cấu trúc while



## Câu lệnh: **while**

- Ví dụ:

```
<?php
    $x = 1;
    while ($x <= 5) {
        echo "The number is: ".$x."<br>";
        $x++;
    }
?>
```



Thay đổi giá trị trong biểu thức điều kiện

## Câu lệnh: **do - while**

- Cú pháp:

```
do {  
    <các câu lệnh>  
} while (<biểu thức điều kiện>)
```

- Lưu ý: Các câu được thực hiện ít nhất một lần. Trong vòng lặp **do - while** phải có câu lệnh thay đổi giá trị của biểu thức điều kiện để tránh việc lặp vô hạn.

## Câu lệnh: **do - while**

### ▪ Nguyên tắc thực thi

(1) Chương trình thực thi ngay **<các câu lệnh>**

- Do đó, **<các câu lệnh>** được thực thi ít nhất 01 lần
- Sau khi thực thi xong, chương trình đánh giá biểu thức điều kiện và kiểm tra

(2) Nếu điều kiện là **true**

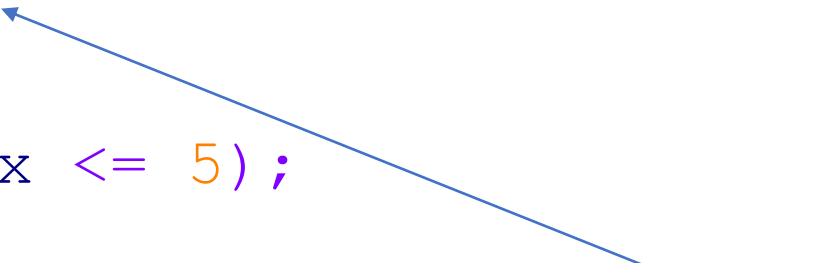
- Đi đến bước (1) để thực thi câu lệnh

(3) Nếu điều kiện false chương trình đi đến câu lệnh theo sau cấu trúc **do while**

## Câu lệnh: do - while

- Ví dụ:

```
<?php
    $x = 1;
    do {
        echo "The number is: ".$x."<br>";
        $x++;
    } while ($x <= 5);
?>
```



Thay đổi giá trị trong  
biểu thức điều kiện

## Câu lệnh: **for**

- Cú pháp:

```
for (<khởi tạo>; <điều kiện>; <thay đổi giá trị>) {  
    <câu lệnh 1>  
    <câu lệnh 2>  
    <câu lệnh N>  
}
```

## Câu lệnh: **for**

### ▪ Nguyên tắc thực thi:

(1) Chương trình sẽ khai báo và khởi tạo các biến trong trong <khởi tạo> và kiểm tra biểu thức điều kiện

(2) Nếu <điều kiện> là true

- Thực hiện câu lệnh <câu lệnh>
- Thực thi các thay đổi trong <thay đổi giá trị>
- Kiểm tra lại điều kiện ở Bước (2) ở trên

(3) Ngược lại

- Đi đến câu lệnh theo sau câu lệnh lặp này

## Câu lệnh: **for**

- Ví dụ:

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: ".$x." <br>";
}
?>
```

```
<?php
for ($x = 0; $x <= 100; $x+=10) {
    echo "The number is: ".$x." <br>";
}
?>
```

## Câu lệnh: **foreach**

### ■ Cú pháp 1:

```
foreach ($array as $value)
{
    <câu lệnh 1>
    <câu lệnh 2>
    <câu lệnh N>
}
```

### ■ Cú pháp 2:

```
foreach ($array as $key => $value)
{
    <câu lệnh 1>
    <câu lệnh 2>
    <câu lệnh N>
}
```



## Câu lệnh: **foreach**

### ▪ Nguyên tắc thực thi:

- (1) Giá trị (chỉ số) đầu tiên của mảng **\$array** sẽ gán vào biến **\$value** (**\$key**) ở vòng lặp đầu tiên.
- (2) Qua mỗi vòng lặp thì biến **\$value** (**\$key**) sẽ được gán giá trị (chỉ số) của phần tử tiếp theo của mảng.
- (3) Vòng lặp sẽ lặp đi lặp lại đến khi duyệt qua tất cả các phần tử trong mảng **\$array**.

## Câu lệnh: **foreach**

### ■ Ví dụ 1:

```
$cars = array("Volvo", "BMW", "Toyota");  
foreach ($cars as $value) {  
    echo $value;  
}
```

### ■ Ví dụ 2:

```
$employee = array(  
    "Name" => "Alex",  
    "Email" => "alex_jtp@gmail.com",  
    "Age" => 21,  
    "Gender" => "Male");  
foreach ($employee as $key => $value) {  
    echo $key . ": " . $value . "<br/>";  
}
```

## Từ khóa **break**

- Khi các câu lệnh trong vòng lặp thực thi đến lệnh **break**, thì vòng lặp sẽ dừng ngay lập tức.
- Ví dụ: Từ khóa **break** trong vòng lặp **for**

```
<?php
    for ($x = 0; $x < 10; $x++) {
        if ($x == 4) {
            break;
        }
        echo "The number is: ".$x."<br>";
    }
?>
```

## Từ khóa **break**

- Ví dụ: Từ khóa **break** trong vòng lặp **while**

```
<?php
    $x = 0;
    while ($x < 10) {
        if ($x == 4) {
            break;
        }
        echo "The number is: ".$x." <br>";
        $x++;
    }
?>
```

## ĐỊNH NGHĨA HÀM TRONG PHP

- Sử dụng từ khóa **function** để định nghĩa theo cú pháp sau:

```
function functionName() {  
    <các câu lệnh>  
    [return <giá trị trả về>]  
}
```

Ví dụ:

```
<?php  
    function writeMsg() {  
        echo "Hello world!";  
    }  
    writeMsg(); // gọi hàm  
?>
```

## ĐỊNH NGHĨA HÀM TRONG PHP

- Định nghĩa hàm có tham số và truyền tham số cho hàm.
- Ví dụ:

```
<?php
```

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

```
$a = 5;
```

```
$b = 10;
```

```
$ret = sum($a, $b); // 15
```

```
?>
```

## HÀM ĐỊNH NGHĨA SẴN TRONG PHP

- PHP có hơn 1000 hàm định nghĩa sẵn (built-in functions).
- Các hàm có sẵn được gọi trực tiếp và thực hiện một chức năng cụ thể nào đó.

## MỘT SỐ HÀM ĐỊNH NGHĨA SẴN TRONG PHP

Tên hàm	Mô tả	Ví dụ
<code>abs(number)</code>	Trị tuyệt đối	<code>echo abs(-1);</code>
<code>date(format, timestamp)</code>	Định dạng ngày giờ	<code>echo date('d-m-Y');</code> <code>echo date('Y');</code>
<code>explode(separator, string, limit)</code>	Tách chuỗi dựa vào <code>separator</code>	<code>\$str = "02/05/2022";</code> <code>\$ret = explode("/", \$str);</code> <code>var_dump(\$ret);</code>
<code>round(number, precision, mode);</code>	Làm tròn	<code>echo round(1.75, 1);</code>
<code>strlen(string)</code>	Trả về độ dài của chuỗi	<code>echo strlen("Hello");</code>
<code>strtolower(string)</code>	Trả về chuỗi in thường	<code>echo strtolower("Hello");</code>
<code>strtoupper(string)</code>	Trả về chuỗi in hoa	<code>echo strtoupper("Hello");</code>
<code>substr(string, start, length)</code>	Trả về chuỗi con	<code>echo substr("Hello", 0, 2);</code>



- Sử dụng cặp thẻ `<?php ?>` để chèn mã PHP vào HTML
- File mã nguồn phải có phần mở rộng là `.php`

`*.php`

```
<!DOCTYPE html>
```

```
<html>
```

```
    <head>
```

```
        <title>Kết hợp HTML và PHP</title>
```

```
    </head>
```

```
    <body>
```

```
        <?php $hk = "I/2022-2023"; ?>
```

```
        <h1>Đại học Ngân hàng TP.HCM</h1>
```

```
        <h1>Học kỳ:<?php echo $hk; ?></h1>
```

```
    </body>
```

```
</html>
```

## XỬ LÝ DỮ LIỆU GỬI TỪ FORM HTML

- Tạo form nhập dữ liệu như sau:

Khi nhấn nút **Gửi**, dữ liệu trong form sẽ gửi đến file **age.php** theo phương thức **Post**

```
<form action="age.php" method="post">  
Năm sinh: <input type="text" name="nam_sinh"  
           placeholder = "Nhập năm sinh" /><br/>  
           <input type="submit" value="Gửi">  
</form>
```

Năm sinh:

## XỬ LÝ DỮ LIỆU GỬI TỪ FORM HTML

- Tạo file **age.php** để xử lý dữ liệu nhận được.

Lấy dữ liệu được gửi bằng  
**\$\_POST** và giá trị thuộc tính **name**

<?php

```
$nam = $_POST["nam_sinh"];  
$tuoi = date('Y') - $nam;  
echo "Tuổi của bạn là: ".$tuoi;
```

?>

**age.php**

1. Viết chương trình giải phương trình bậc hai  $ax^2 + bx + c$ . Thiết kế form nhập vào ba hệ số  $a, b, c$
2. Viết chương trình cho biết số nguyên  $n$  có phải là số nguyên tố không. Thiết kế form để nhập  $n$ .
3. Viết chương trình tìm số lớn nhất trong ba số  $a, b, c$ . Thiết kế form nhập  $a, b, c$
4. Viết chương trình tính tổng các số nguyên. Thiết kế form nhập vào các số nguyên, mỗi số ngăn cách với nhau bởi dấu phẩy
5. Viết chương trình kiểm tra ngày có hợp lệ hay không? Thiết kế form để nhập ngày, ngày có định dạng dd/mm/yyyy

# HỎI & ĐÁP