



# **LẬP TRÌNH PHP**

**TH.S NGUYỄN ĐÌNH HOÀNG**

email: [hoangnd@itc.edu.vn](mailto:hoangnd@itc.edu.vn)

- ❑ 1. Giới thiệu
- ❑ 2. Cấu trúc PHP
- ❑ 3. Kiểu dữ liệu, hằng và biến
- ❑ 4. Các phép toán trong PHP
- ❑ 5. Các cấu trúc điều khiển
- ❑ 6. Hàm trong PHP
- ❑ 7. Mảng (array)

## ❑ 7. Mảng trong PHP

---

- **Mảng** là một danh sách các phần tử có cùng kiểu dữ liệu và được lưu cùng vị trí bộ nhớ. Mảng có thể là mảng một chiều hay nhiều chiều. Mảng có 2 thành phần là chỉ mục (key) và giá trị. Chỉ mục có thể là số nguyên hoặc là chuỗi.
- Nếu bạn muốn lưu trữ 100 số bạn dùng 1 mảng thay vì phải khai báo 100 biến.

### Có 3 loại mảng chính:

- ✓ Mảng có chỉ mục : chỉ mục là các số nguyên.
- ✓ Mảng liên kết : key của mảng là chuỗi.
- ✓ Mảng nhiều chiều : Mảng chứa mảng nhiều chiều cũng chứa nhiều phần tử, mỗi phần tử lại là một mảng một chiều.

# 7. Mảng trong PHP

## 1. Mảng được lập chỉ mục trong PHP

- Chỉ mục trong PHP được biểu thị bằng số bắt đầu từ 0. Chúng ta có thể lưu trữ số, chuỗi và đối tượng trong mảng PHP. Tất cả các phần tử mảng PHP được gán cho một số chỉ mục theo mặc định.
- Có hai cách để định nghĩa mảng được lập chỉ mục:

Ví dụ 1: Chỉ mục có thể được chỉ định tự động (chỉ mục luôn bắt đầu ở 0)

```
<?php
    $friends[0] = 'Tý';
    $friends[1] = 'Cúm';
    $friends[2] = 'Beo';
    var_dump($friends);
?>
```

# 7. Mảng trong PHP

## 1. Mảng được lập chỉ mục trong PHP

- Chỉ mục trong PHP được biểu thị bằng số bắt đầu từ 0. Chúng ta có thể lưu trữ số, chuỗi và đối tượng trong mảng PHP. Tất cả các phần tử mảng PHP được gán cho một số chỉ mục theo mặc định.
- Có hai cách để định nghĩa mảng được lập chỉ mục:

Ví dụ 2: Chỉ mục có thể được chỉ định tự động (chỉ mục luôn bắt đầu ở 0)

```
<?php
    $friends = array(Tý, 'Cúm', 'Beo');
    var_dump($friends);
?>
```

## 7. Mảng trong PHP

Để duyệt các phần tử của mảng được lập chỉ mục trong PHP, bạn có thể sử dụng **vòng lặp for, while**... như sau:

```
<?php
    $season=array("summer","winter","spring","autumn");
    // tính độ dài của mảng
    $arlength = count($season);
    // hiển thị các phần tử của mảng
    for($i = 0; $i < $arlength; $i++) {
        echo $season[$i];
        echo "<br>";
    }
?>
```

## ❑ 7. Mảng trong PHP

Để duyệt các phần tử của mảng được lập chỉ mục trong PHP, bạn có thể sử dụng **vòng lặp for, while**... như sau:

```
<?php
    $season[0]="spring";// mùa xuân
    $season[1]="summer";// mùa hạ
    $season[2]="autumn";// mùa thu
    $season[3]="winter";// mùa đông
    // tính độ dài của mảng
    $arrLength = count($season);
    // hiển thị các phần tử của mảng
    for($i = 0; $i < $arrLength; $i++) {
        echo $season[$i];
        echo "<br>";
    }
?>
```

# 7. Mảng trong PHP

---

## 2. Mảng liên kết trong PHP

- Mảng liên kết là các mảng sử dụng các khóa được đặt tên mà bạn gán cho chúng.
- Có hai cách để tạo một mảng liên kết trong PHP:

### Cách 1:

```
<?php
    $salary[ 'Ty' ] = 15000;
    $salary[ 'Cum' ] = 25000;
    $salary[ 'Beo' ] = 27000;
    var_dump($salary);
?>
```

### Cách 2:

```
<?php
    $salary = array( 'Ty'=>15000, 'Cum'=>25000, 'Beo'=>27000 );
    print_r($salary);
?>
```



# 7. Mảng trong PHP

## 2. Mảng liên kết trong PHP

- Để duyệt các phần tử của mảng liên kết trong PHP, bạn có thể sử dụng **vòng lặp foreach**, như sau:

```
<?php
$dbInfo = array("url"=>"http://localhost:3306", "dbName"=>"testdb",
    "username"=>"root", "password"=>"123123123");

foreach($dbInfo as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```

# 7. Mảng trong PHP

---

## 2. Mảng liên kết trong PHP

- Để duyệt các phần tử của mảng liên kết trong PHP, bạn có thể sử dụng **vòng lặp foreach**, như sau:

```
<?php
    $dbInfo ['url'] = "http://localhost:3306";
    $dbInfo ['dbName'] = "testdb";
    $dbInfo ['username'] = "root";
    $dbInfo ['password'] = "123123123";

    foreach($dbInfo as $x => $x_value) {
        echo "Key=" . $x . ", Value=" . $x_value;
        echo "<br>";
    }
?>
```

## ❑ 7. Mảng trong PHP

---

### 3. Mảng đa chiều trong PHP

- Mảng đa chiều là một mảng chứa một hoặc nhiều mảng.
- Bạn có thể sử dụng các mảng PHP đa chiều có độ sâu hai, ba, bốn, năm, hoặc nhiều hơn. Tuy nhiên, các mảng nhiều hơn ba cấp độ rất khó quản

#### ❖ Mảng 2 chiều trong PHP

- ✓ Mảng hai chiều là một mảng các mảng (mảng ba chiều là mảng mảng mảng).

## ❑ 7. Mảng trong PHP

---

### ❖ Mảng 2 chiều trong PHP

- ✓ Mảng hai chiều là một mảng các mảng (mảng ba chiều là mảng mảng mảng).
- ✓ Ta có bảng dữ liệu như sau:

Tên	Kho	Đã bán
Iphone X	22	18
Sony Z10	15	13
Samsung Note 10	5	2
Xiaomi 6	17	15

## ❑ 7. Mảng trong PHP

### ❖ Mảng 2 chiều trong PHP

Tên	Kho	Đã bán
Iphone X	22	18
Sony Z10	15	13
Samsung Note 10	5	2
Xiaomi 6	17	15

- ✓ Chúng ta có thể lưu trữ dữ liệu của bảng trên vào mảng 2 chiều như sau:

```
$hang_hoa = array  
(  
    array("Iphone X",22,18),  
    array("Sony Z10",15,13),  
    array("Samsung Note 10",5,2),  
    array("Xiaomi 6",17,15)  
);
```

## ❑ 7. Mảng trong PHP

---

### ❖ Mảng 2 chiều trong PHP

- ✓ Bây giờ mảng hai chiều `$hang_hoa` chứa bốn mảng, và nó có hai chỉ số: hàng và cột.
- ✓ Để truy cập vào các phần tử của mảng `$hang_hoa`, chúng ta phải trở đến hai chỉ mục (`hàng và cột`):
- ✓ Chúng ta cũng có thể đặt một vòng lặp for bên trong một **vòng lặp for** khác để lấy các phần tử của mảng `$hang_hoa`.

## 7. Mảng trong PHP

---

### ❖ Mảng 2 chiều trong PHP

```
<?php
    $hang_hoa = array
    (
        array("Iphone X",22,18),
        array("Sony Z10",15,13),
        array("Samsung Note 10",5,2),
        array("Xiaomi 6",17,15)
    );

    for ($row = 0; $row < 4; $row++) {
        echo "<p><b>Row number $row</b></p>";
        echo "<ul>";
        for ($col = 0; $col < 3; $col++) {
            echo "<li>".$hang_hoa[$row][$col]."</li>";
        }
        echo "</ul>";
    }
?>
```

## 7. Mảng trong PHP

---

### 4. Các hàm xử lý mảng trong PHP

PHP cung cấp các hàm xử lý mảng khác nhau để truy cập và thao tác các phần tử của mảng. Các hàm mảng PHP quan trọng được đưa ra dưới đây.

1. Hàm `array()` trong PHP
2. Hàm `array_change_key_case()` trong PHP
3. Hàm `array_chunk()` trong PHP
4. Hàm `count()` trong PHP
5. Hàm `array_reverse()` trong PHP
6. Hàm `array_search()` trong PHP
7. Hàm `array_intersect()` trong PHP
8. Hàm `sort()` trong PHP



## 7. Mảng trong PHP

---

### 4.1. Hàm array() trong PHP

- Hàm PHP array() được sử dụng để tạo và trả về một mảng. Nó cho phép bạn tạo các mảng được lập chỉ mục, mảng liên kết và mảng đa chiều. **array()**;

- Cú pháp:

```
<?php
    $season=array("summer","winter","spring","autumn");

    // tính độ dài của mảng
    $arrrlength = count($season);
    // hiển thị các phần tử của mảng
    for($i = 0; $i < $arrrlength; $i++) {
        echo $season[$i];
        echo "<br>";
    }
?>
```

## 7. Mảng trong PHP

### 4.2. Hàm array\_change\_key\_case() trong PHP

- Hàm array\_change\_key\_case() trong PHP được sử dụng để thay đổi khóa (key) của một mảng.
- Nó chỉ được sử dụng để thay đổi chữ hoa chữ thường cho khóa (key) của mảng.
- Cú pháp: `array array_change_key_case ( array $array [, int $case = CASE_LOWER ] )`

Ví dụ 1:

```
<?php
    $salary=array("Cong"=>"550000","Dung"=>"250000","Vu"=>"200000");
    print_r(array_change_key_case($salary,CASE_UPPER));
?>
```



Kết quả:

```
Array ( [CONG] => 550000 [DUNG] => 250000 [VU] => 200000 )
```

## ❑ 7. Mảng trong PHP

### 4.2. Hàm array\_change\_key\_case() trong PHP

- Hàm array\_change\_key\_case() trong PHP được sử dụng để thay đổi khóa (key) của một mảng.
- Nó chỉ được sử dụng để thay đổi chữ hoa chữ thường cho khóa (key) của mảng.
- Cú pháp: `array array_change_key_case ( array $array [, int $case = CASE_LOWER ] )`

Ví dụ 2:

```
<?php
```

```
    $salary=array("Cong"=>"550000","Dung"=>"250000","Vu"=>"200000");  
    print_r(array_change_key_case($salary, CASE_LOWER));
```

```
?>
```



Kết quả:

```
Array ( [cong] => 550000 [dung] => 250000 [vu] => 200000 )
```

## 7. Mảng trong PHP

### 4.3. Hàm array\_chunk() trong PHP

- Hàm array\_chunk() trong PHP chia mảng thành các khối. Bằng cách sử dụng phương thức array\_chunk(), bạn có thể chia mảng thành nhiều phần.  
`array array_chunk ( array $array , int $size [, bool $preserve_keys = false ] )`

- Cú pháp:

Ví dụ:

```
<?php
```

```
    $salary=array("Cong"=>"550000","Dung"=>"250000","Vu"=>"200000");
```

```
    print_r(array_chunk($salary, 2));
```

```
?>
```



Kết quả:

```
Array ( [0] => Array ( [0] => 550000 [1] => 250000 )  
        [1] => Array ( [0] => 200000 ) )
```

## 7. Mảng trong PHP

---

### 4.4. Hàm count() trong PHP

- Hàm PHP count() đếm tất cả các phần tử trong một mảng.
- Cú pháp: `int count ( mixed $array_or_countable [, int $mode = COUNT_NORMAL ] )`

Ví dụ:

```
<?php
```

```
    $season=array("summer", "winter", "spring", "autumn");
```

```
    echo count($season);
```

```
?>
```




## 7. Mảng trong PHP

### 4.5. Hàm array\_reverse() trong PHP

- Hàm array\_reverse() trong PHP trả về một mảng chứa các phần tử theo thứ tự đảo ngược.
- Cú pháp: `array array_reverse ( array $array [, bool $preserve_keys = false ] )`

```
<?php
    $season=array("summer","winter","spring","autumn");
    $reverseseason=array_reverse($season);
    foreach( $reverseseason as $s )
    {
        echo "$s<br />";
    }
?>
```



Kết quả:

```
autumn
spring
winter
summer
```

## 7. Mảng trong PHP

---

### 4.6. Hàm `array_search()` trong PHP

- Hàm `array_search()` trong PHP tìm kiếm giá trị được chỉ định trong một mảng. Nó trả về khóa nếu tìm kiếm thành công.
- Cú pháp: `mixed array_search ( mixed $needle , array $haystack [, bool $strict = false ] )`

```
<?php
    $season=array("summer", "winter", "spring", "autumn");
    $key=array_search("spring",$season);
    echo $key;
?>
```

## 7. Mảng trong PHP

### 4.7. Hàm array\_intersect() trong PHP

- Hàm array\_intersect() trong PHP trả về giao điểm của hai mảng. Nói cách khác, nó trả về các phần tử giống nhau của hai mảng.
- Cú pháp: `array array_intersect ( array $array1 , array $array2 [, array $... ] )`

```
<?php
```

```
$name1=array("Java","PHP","C++","VBA");  
$name2=array("PHP","HTML","CSS","Java");  
$name3=array_intersect($name1,$name2);  
foreach( $name3 as $n )  
{  
    echo "$n<br />";  
}
```

```
?>
```



Kết quả:

```
Java  
PHP
```



## ❑ 7. Mảng trong PHP

---

### 4.8. Các hàm sắp xếp trong PHP

- `sort()` - sắp xếp các mảng theo thứ tự tăng dần.
- `rsort()` - sắp xếp các mảng theo thứ tự giảm dần.
- `asort()` - sắp xếp các mảng liên kết theo thứ tự tăng dần, theo giá trị.
- `ksort()` - sắp xếp các mảng liên kết theo thứ tự tăng dần, theo khóa.
- `arsort()` - sắp xếp các mảng liên kết theo thứ tự giảm dần, theo giá trị.
- `krsort()` - sắp xếp các mảng liên kết theo thứ tự giảm dần, theo khóa.

# 7. Mảng trong PHP

## 4.8. Hàm sort() trong PHP

- Hàm PHP sort() sắp xếp tất cả các phần tử trong một mảng tăng dần.
- Cú pháp: `bool sort ( array &$array [, int $sort_flags = SORT_REGULAR ] )`

```
<?php
    $season=array("summer","winter","spring","autumn");
    sort($season);
    foreach( $season as $s )
    {
        echo "$s<br />";
    }
?>
```



Kết quả:

```
autumn
spring
summer
winter
```

## 7. Mảng trong PHP

### 4.8. Hàm rsort() trong PHP

- Hàm PHP `rsort()` sắp xếp tất cả các phần tử trong một mảng giảm dần.
- Cú pháp: `bool rsort ( array &$array [, int $sort_flags = SORT_REGULAR ] )`

```
<?php
    $cars = array("Volvo", "BMW", "Toyota");
    rsort($cars);
    foreach( $cars as $c) {
        echo "$c <br>";
    }
?>
```



Kết quả:

```
Volvo
Toyota
BMW
```


## 7. Mảng trong PHP

### 4.8. Hàm asort() trong PHP

- Hàm PHP `asort()` sắp xếp các mảng liên kết theo thứ tự tăng dần, theo giá trị.
- Cú pháp: `bool asort ( array &$array [, int $sort_flags = SORT_REGULAR ] )`

```
<?php
    $age = array("Vinh"=>"22", "Tan"=>"25", "Hoa"=>"20");
    asort($age);

    foreach($age as $x => $x_value) {
        echo "Key = " . $x . ", Value = " . $x_value;
        echo "<br>";
    }
?>
```



Kết quả:

```
Key = Hoa, Value = 20
Key = Vinh, Value = 22
Key = Tan, Value = 25
```

## 7. Mảng trong PHP

### 4.8. Hàm ksort() trong PHP

- Hàm PHP `ksort()` sắp xếp các mảng liên kết theo thứ tự tăng dần, theo khóa.  
`bool ksort ( array &$array [, int $sort_flags = SORT_REGULAR ] )`

- Cú pháp:

```
<?php
$age = array("Vinh"=>"22", "Tan"=>"25", "Hoa"=>"20");
ksort($age);
```

```
foreach($age as $x => $x_value) {
    echo "Key = " . $x . ", Value = " . $x_value;
    echo "<br>";
}
?>
```



Kết quả:

```
Key = Hoa, Value = 20
Key = Tan, Value = 25
Key = Vinh, Value = 22
```

## 7. Mảng trong PHP

### 4.8. Hàm arsort() trong PHP

- Hàm PHP `arsort()` sắp xếp các mảng liên kết theo thứ tự giảm dần, theo giá trị.
- Cú pháp: `bool arsort ( array &$amp;array [, int $sort_flags = SORT_REGULAR ] )`

```
<?php
$age = array("Vinh"=>"22", "Tan"=>"25", "Hoa"=>"20");
arsort($age);
```

```
foreach($age as $x => $x_value) {
    echo "Key = " . $x . ", Value = " . $x_value;
    echo "<br>";
}
?>
```



Kết quả:

```
Key = Tan, Value = 25
Key = Vinh, Value = 22
Key = Hoa, Value = 20
```

## 7. Mảng trong PHP

### 4.8. Hàm krsort() trong PHP

- Hàm PHP `krsort()` sắp xếp các mảng liên kết theo thứ tự giảm dần, theo giá trị.
- Cú pháp: `bool krsort ( array &$array [, int $sort_flags = SORT_REGULAR ] )`

```
<?php
    $age = array("Vinh"=>"22", "Tan"=>"25", "Hoa"=>"20");
    krsort($age);

    foreach($age as $x => $x_value) {
        echo "Key = " . $x . ", Value = " . $x_value;
        echo "<br>";
    }
?>
```



Kết quả:

```
Key = Vinh, Value = 22
Key = Tan, Value = 25
Key = Hoa, Value = 20
```

## ❑ 7. Mảng trong PHP

---

### Các hàm liên quan đến mảng

- `var_dump($array)` : xuất nội dung thông tin mảng
- `is_array(array)` : kiểm tra mảng
- `min(array)` : phần tử nhỏ nhất trong mảng
- `max(array)` : phần tử lớn nhất trong mảng
- `reset(array)` : khởi tạo lại mảng
- `array_push(array, elements)` : thêm phần tử cuối mảng
- `array_pop(array)` : lấy phần tử cuối mảng
- `array_unshift(array, elements)` : thêm phần tử đầu mảng
- `array_shift(array)` : lấy phần tử đầu mảng
- `array_merge(array, array)` : trộn 2 mảng
- `shuffle(array)` : sắp xếp mảng ngẫu nhiên



## ❑ 8. Chuỗi trong PHP

---

**String trong PHP là một chuỗi các ký tự**, như "Hello world!". Mỗi ký tự trong PHP có độ dài là 1 byte. Điều này có nghĩa là php hỗ trợ  $2_8 = 256$  ký tự.

### Cú pháp:

Thông thường, một chuỗi ký tự trong PHP có thể được định nghĩa theo 2 cách sau:

- Dấu nháy đơn. Ví dụ `$str1 = 'Hello PHP String!'`
- Dấu nháy kép. Ví dụ `$str1 = "Hello PHP String!"`

## ❑ 8. Chuỗi trong PHP

---

### Cách viết một chuỗi

- Trong PHP, một chuỗi phải được đặt bên trong cặp dấu nháy kép " " hoặc cặp dấu nháy đơn ' '.
- Ví dụ:

```
<?php
```

```
$a = "HTML và CSS"; //Biến a có giá trị là chuỗi HTML và CSS
```

```
$b = 'Tài liệu PHP'; //Biến b có giá trị là chuỗi Tài liệu PHP
```

```
$c = ""; //Biến c có giá trị là một chuỗi rỗng
```

```
?>
```

- Hai dấu dùng để đặt xung quanh chuỗi phải cùng một loại, nếu khác loại sẽ dẫn đến sai cú pháp làm chương trình bị lỗi.

## ❑ 8. Chuỗi trong PHP

---

- Hai dấu dùng để đặt xung quanh chuỗi phải cùng một loại, nếu khác loại sẽ dẫn đến sai cú pháp làm chương trình bị lỗi.

```
<?php
    $a = "HTML và CSS"; //SAI
    $b = 'Tài liệu PHP'; //SAI
?>
```

- Nếu chuỗi được đặt bên trong cặp dấu nháy kép thì chuỗi đó không được chứa ký tự là dấu nháy kép (*tuy nhiên nó có thể chứa ký tự là dấu nháy đơn*)
- Nếu chuỗi được đặt bên trong cặp dấu nháy đơn thì chuỗi đó không được chứa ký tự là dấu nháy đơn (*tuy nhiên nó có thể chứa ký tự là dấu nháy kép*)

## ❑ 8. Chuỗi trong PHP

---

- Nếu chuỗi được đặt bên trong cặp dấu nháy kép thì chuỗi đó không được chứa ký tự là dấu nháy kép (*tuy nhiên nó có thể chứa ký tự là dấu nháy đơn*)
- Nếu chuỗi được đặt bên trong cặp dấu nháy đơn thì chuỗi đó không được chứa ký tự là dấu nháy đơn (*tuy nhiên nó có thể chứa ký tự là dấu nháy kép*)

```
<?php
```

```
$a = "Tài liệu học " PHP"; //SAI
```

```
$b = 'Tài liệu học ' PHP'; //SAI
```

```
$c = "Tài liệu học ' PHP"; //ĐÚNG
```

```
$d = 'Tài liệu học " PHP'; //ĐÚNG
```

```
?>
```

## ❑ 8. Chuỗi trong PHP

---

- Để giải quyết trường hợp bạn muốn chuỗi được đặt bên trong cặp dấu nháy kép có thể chứa ký tự là dấu nháy kép thì bạn phải đặt một dấu gạch chéo ngược phía trước ký tự là dấu nháy kép đó (*trường hợp dấu nháy đơn cũng tương tự*).

```
<?php
    $a = "Tài liệu học \" PHP";
    $b = 'Tài liệu học \' PHP';
?>
```

## ❑ 8. Chuỗi trong PHP

### Điểm khác nhau giữa " " và ' '

- Đối với chuỗi được đặt bên trong cặp dấu nháy kép, nếu chuỗi đó có chứa cụm từ *"gọi tên biến"* thì khi ta hiển thị chuỗi này lên màn hình, nó sẽ hiển thị luôn cả giá trị của biến.
- Đối với chuỗi được đặt bên trong cặp dấu nháy đơn, nếu chuỗi đó có chứa cụm từ *"gọi tên biến"* thì khi ta hiển thị chuỗi này lên màn hình, nó sẽ KHÔNG hiển thị giá trị của biến.

```
<?php
    $name = "Nguyễn Thành Tài";
    $text_1 = "Tên của tôi là $name";
    $text_2 = 'Tên của tôi là $name';
    echo $text_1;
    echo $text_2;
?>
```

## ❑ 8. Chuỗi trong PHP

---

### Cách nối các chuỗi lại với nhau

- Ta có thể nối hai hoặc nhiều chuỗi lại với nhau thành một chuỗi bằng cách đặt dấu chấm ở giữa hai chuỗi cần nối.

Giá trị của biến \$text là một chuỗi được nối từ ba chuỗi.

```
<?php
    $text = "Tài liệu học "."ngôn ngữ lập trình"." PHP";
    echo $text;
?>
```

## ❑ 8. Chuỗi trong PHP

---

### Các hàm xử lý chuỗi trong PHP

Hàm PHP **strlen()** trả về độ dài của một chuỗi.

Ví dụ dưới đây trả về độ dài của chuỗi "Hello world!":

```
<?php
    echo strlen("Hello world!");
?>
```

Kết quả:

2

Hàm PHP **strrev()** đảo ngược một chuỗi:

```
<?php
    echo strrev("Hello world!");
?>
```

Kết quả:

!dlrow olleH



## ❑ 8. Chuỗi trong PHP

### Các hàm xử lý chuỗi trong PHP

- Hàm PHP **strpos()** tìm kiếm một văn bản cụ thể trong một chuỗi.
- Nếu tìm thấy một kết quả phù hợp, hàm sẽ trả về vị trí ký tự đầu tiên khớp nhau. Nếu không tìm thấy kết quả phù hợp, nó sẽ trả về FALSE.
- Ví dụ bên dưới tìm kiếm chuỗi "World" và "PHP" trong chuỗi "Hello World!":

```
<?php
$str = "Hello world!";
echo strpos($str, "world");
echo "<br>";
if (strpos($str, "world")) {
    echo "\"$str\" chứa chuỗi " . "\"world\"";
} else {
    echo "\"$str\" không chứa chuỗi " . "\"world\"";
}
echo "<br>";

if (strpos($str, "PHP")) {
    echo "\"$str\" chứa chuỗi " . "\"PHP\"";
} else {
    echo "\"$str\" không chứa chuỗi " . "\"PHP\"";
}
?>
```

Kết quả:

```
6
"Hello world!" chứa chuỗi "world"
"Hello world!" không chứa chuỗi "PHP"
```

## ❑ 8. Chuỗi trong PHP

---

### Các hàm xử lý chuỗi trong PHP

Hàm PHP **str\_replace()** thay thế một số ký tự bằng một số ký tự khác trong một chuỗi.

Ví dụ bên dưới thay thế chuỗi ký tự "World" bằng "PHP":

```
<?php  
    echo str_replace("World", "PHP", "Hello World!");  
?>
```

Kết quả:

```
Hello PHP!
```

## ❑ 8. Chuỗi trong PHP

---

### Các hàm xử lý chuỗi trong PHP:

#### 1. addslashes (\$str, \$charList)

Hàm này sẽ thêm dấu gạch chéo () đằng trước những ký tự trong chuỗi \$str mà ta liệt kê ở \$charList.

#### 2. addslashes ( \$str )

Hàm này sẽ thêm dấu gạch chéo trước những ký tự (‘, “, \) trong chuỗi \$str.

#### 3. stripslashes (\$str)

Hàm này ngược với hàm addslashes, nó xóa các ký tự \ trong chuỗi \$str.

#### 4. crc32 ( \$str )

Hàm này sẽ chuyển chuỗi \$str thành một dãy số nguyên (có thể âm hoặc dương tùy theo hệ điều hành).

#### 5. explode ( \$delimiter , \$string)

Hàm này sẽ chuyển một chuỗi \$string thành một mảng các phần tử với ký tự tách mảng là \$delimiter.

## ❑ 8. Chuỗi trong PHP

---

### Các hàm xử lý chuỗi trong PHP:

#### 6. `implode($delimiter, $piecesarray);`

Hàm này ngược với hàm `explode`, nó chuyển một mảng `$piecesarray` thành chuỗi và mỗi phần tử cách nhau bởi chuỗi `$delimiter`.

#### 7. `ord ( $string )`

Hàm này trả về mã ASCII của ký tự đầu tiên trong chuỗi `$string`.

#### 8. `strlen($string)`

Hàm này đếm số ký tự của chuỗi `$string`.

#### 9. `str_word_count($str)`

Hàm này trả về số từ trong chuỗi `$str`.

#### 10. `str_repeat( $str, int $n )`

Hàm này lặp chuỗi `$str` `$n` lần.

## ❑ 9. Câu lệnh include và require

---

- Câu lệnh **include** và **require** trong PHP được sử dụng để chèn nội dung của file php này vào file php khác.
- Việc chèn nội dung file là rất hữu ích khi bạn muốn chèn các dòng code php, html giống nhau vào nhiều trang khác nhau.
- **Câu lệnh include và require là giống nhau, trừ trường hợp bị lỗi:**
  - ✓ **require**: sẽ tạo ra lỗi nghiêm trọng (E\_COMPILE\_ERROR) và dừng tập lệnh.
  - ✓ **include**: sẽ chỉ tạo cảnh báo (E\_WARNING) và tập lệnh sẽ tiếp tục.
- Vì vậy, nếu bạn muốn chương trình tiếp tục được thực thi và hiển thị đến người dùng, ngay cả khi file được chèn vào bị thiếu, hãy sử dụng câu lệnh include. Nếu không, trong trường hợp Framework, CMS hoặc ứng dụng PHP phức tạp, hãy luôn sử dụng câu lệnh require để chèn một file là bắt buộc tới luồng thực thi. Điều này sẽ giúp tránh ảnh hưởng đến tính bảo mật và tính toàn vẹn của ứng dụng.

## ❑ 9. Câu lệnh include và require

**Cú pháp**      `include 'filename';` hoặc `require 'filename';`

Ví dụ 1: Ta có một file gọi là "footer.php" như sau:

```
<?php
    echo "<p>Copyright © 2021-" . date("Y") . "itc.edu.vn </p>";
?>
```

Sử dụng câu lệnh include để chèn chân trang vào trang page1.php:

```
<html>
<body>
    <h1>Welcome to ITC!</h1>
    <p>Learning PHP.</p>
    <p>Hello CD20CT.</p>
    <?php include 'footer.php';?>
</body>
</html>
```

← → ↻ ⓘ localhost/demo/page.php

# Welcome to ITC!

Learning PHP.

Hello CD20CT.

Copyright © 2019 - 2021 itc.edu.vn

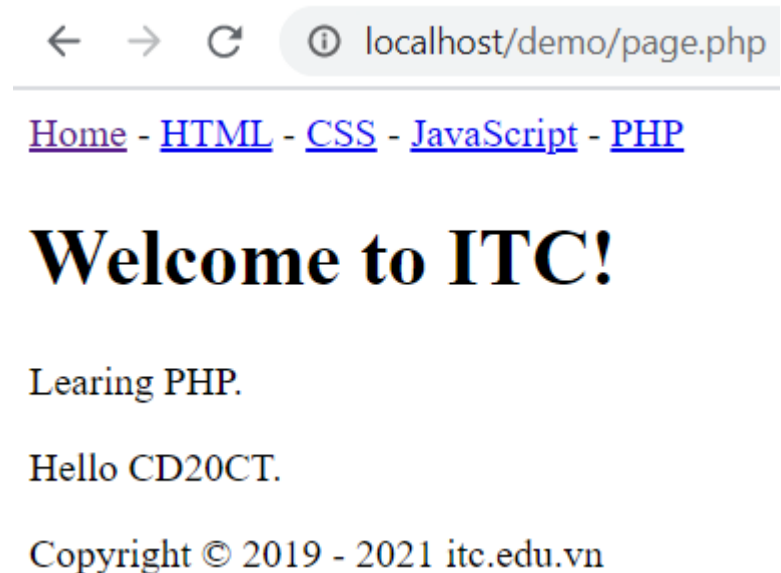
## ❑ 9. Câu lệnh include và require

Ví dụ 2: Ta có một file gọi là "menu.php" như sau:

```
<?php
    echo '<a href="">Home</a> -
    <a href="/html">HTML</a> -
    <a href="/css">CSS</a> -
    <a href="/javascript">JavaScript</a> -
    <a href="/php">PHP</a>';
?>
```

Sử dụng câu lệnh include để chèn chân trang vào trang page2.php:

```
<html>
<body>
    <?php include 'menu.php';?>
    <h1>Welcome to ITC!</h1>
    <p>Learning PHP.</p>
    <p>Hello CD20CT.</p>
    <?php include 'footer.php';?>
</body>
</html>
```



## ❑ 9. Câu lệnh include và require

---

Ví dụ 3: Ta có một file gọi là "cars.php" với một số biến định nghĩa như sau:

```
<?php
    $color='đỏ';
    $car='BMW';
?>
```

Sử dụng câu lệnh include để chèn chân trang vào trang page3.php:

```
<html>
<body>

    <?php include 'cars.php';
    echo "Xe $car có màu $color.";

</body>
</html>
```



## ❑ 9. Câu lệnh include và require

---

### Phân biệt giữa hàm include() và hàm require()

Khi có lỗi xảy ra(file được chèn không tồn tại):

- Hàm **include()** sẽ hiển thị ra một thông báo lỗi dạng warning và đoạn mã vẫn tiếp tục được thực thi.
- Hàm **require()** sẽ hiển thị ra một thông báo lỗi dạng fatal và đoạn mã sẽ bị dừng xử lý ngay sau đó.
- Nếu chúng ta làm ví dụ tương tự bằng cách sử dụng lệnh require, câu lệnh echo sẽ không được thực thi vì việc thực thi tập lệnh chết sau khi lệnh require trả về một lỗi nghiêm trọng:

## ❑ 9. Câu lệnh include và require

---

Ví dụ: Chúng ta không tạo file “file\_khong\_ton\_tai.php” khi chạy chương trình sẽ không hiển thị nội dung của echo.

```
<html>
<body>
  <h1>Welcome to ITC!</h1>
  <?php require 'file_khong_ton_tai.php';
  echo "Xe $car có màu $color.";
  ?>
</body>
</html>
```

## ❑ 10. Câu lệnh `include_once` và `require_once` trong PHP

### Lệnh `include_once()` trong PHP

Lệnh `require_once()` có thể được sử dụng để chèn một tập tin php trong một số tập tin khác, khi bạn có thể cần phải bao gồm các tập tin được gọi nhiều hơn một lần. Nếu nó đã được chèn vào rồi, thì những vị trí chèn sau sẽ bỏ qua.

**Cú pháp:** `include_once('filename');`

**Ví dụ:** File: x.php

```
<?php
echo "Hôm nay là:".date("Y-m-d");
?>
```

File: y.php

```
<?php
echo "Chèn x.php lần 1: ";
include_once('x.php');
echo "</br>";
echo "Chèn x.php lần 2: ";
include_once('x.php');
?>
```

## ❑ 10. Câu lệnh `include_once` và `require_once` trong PHP

---

Kết quả:

```
Chèn x.php lần 1: Hôm nay là:2018-09-01  
Chèn x.php lần 2:
```

- File x.php được sử dụng 2 lần với lệnh `include_once()` để chèn vào file y.php. Nhưng file thứ 2 sẽ bị bỏ qua.

## ❑ 10. Câu lệnh `include_once` và `require_once` trong PHP

---

### Lệnh `require_once()` trong PHP

- Lệnh `require_once()` có thể được sử dụng để chèn một tập tin php trong một số tập tin khác, khi bạn có thể cần phải bao gồm các tập tin được gọi nhiều hơn một lần. Nếu nó đã được chèn vào rồi, thì những vị trí chèn sau sẽ bỏ qua.
- Cú pháp:** `require_once('filename');`

**Ví dụ:** File: x.php

```
<?php  
echo "Hôm nay là:".date("Y-m-d");  
?>
```

File: y.php

```
<?php  
echo "Chèn x.php lần 1: ";  
require_once('x.php');  
echo "</br>";  
echo "Chèn x.php lần 2: ";  
require_once('x.php');  
?>
```

## ❑ 10. Câu lệnh `include_once` và `require_once` trong PHP

---

Kết quả:

```
Chèn x.php lần 1: Hôm nay là:2018-09-01  
Chèn x.php lần 2:
```

- Nếu lệnh `require_once()` không tìm thấy file được chèn thì chương trình sẽ bị dừng lại.

**KẾT THÚC**