

# Chương 4 : PHP

Giảng Viên: ThS. Tạ Việt Phương

# Nội dung



1. Giới thiệu PHP
2. Cơ chế hoạt động của WebServer
3. Cú pháp & Quy ước trong PHP
4. Lập trình hướng đối tượng trong php
5. Bài tập



# 1. Giới thiệu PHP



# Giới thiệu về PHP – Lịch sử phát triển

- **PHP** : Rasmus Lerdorf in 1994 (được phát triển để phát sinh các form đăng nhập sử dụng giao thức HTTP của Unix)
- **PHP 2 (1995)** : Chuyển sang ngôn ngữ script xử lý trên server. Hỗ trợ CSDL, Upload File, khai báo biến, mảng, hàm đệ quy, câu điều kiện, biểu thức,
- **PHP 3 (1998)** : Hỗ trợ ODBC, đa hệ điều hành, giao thức email (SNMP, IMAP), bộ phân tích mã PHP (parser) của Zeev Suraski và Andi Gutmans
- **PHP 4 (2000)** : Trở thành một thành phần độc lập cho các webserver. Parse đổi tên thành Zend Engine. Bổ sung các tính năng bảo mật cho PHP
- **PHP 5 (2005)** : Bổ sung Zend Engine II hỗ trợ lập trình HĐT, XML, SOAP cho Web Services, SQLite
- Phiên bản mới nhất của PHP là version PHP 7.4.5 (*www.php.net*)

# Giới thiệu về PHP – PHP là gì ?

- **PHP** viết tắt của **P**HP **H**ypertext **P**reprocessor
- Là ngôn ngữ server-side script, tương tự như ASP, JSP, ... thực thi ở phía WebServer
- Tập tin PHP có phần mở rộng là **.php**
- Cú pháp ngôn ngữ giống ngôn ngữ **C & Perl**



# Giới thiệu về PHP – Ưu điểm 1

- PHP được sử dụng làm
  - Server Side Scripting
  - Dùng để xây dựng các ứng dụng web



# Giới thiệu về PHP – Ưu điểm 2

- **Đa môi trường (Multi-Platform)**

- **Web Servers:** Apache, **Microsoft IIS**, Caudium, Netscape Enterprise Server
- **Hệ điều hành:** UNIX (HP-UX, OpenBSD, Solaris, Linux), Mac OSX, **Windows NT/98/2000/XP/2003/vista**
- **Hệ QTCSDL:** Adabas D, dBase, Empress, FilePro (read-only), Hyperwave, IBM DB2, Informix, Ingres, InterBase, FrontBase, mSQL, Direct MS-SQL, **MySQL**, ODBC, Oracle (OCI7 and OCI8), Ovrimos, PostgreSQL, SQLite, Solid, Sybase, Velocis, Unix dbm



# Giới thiệu về PHP – Ưu điểm 3

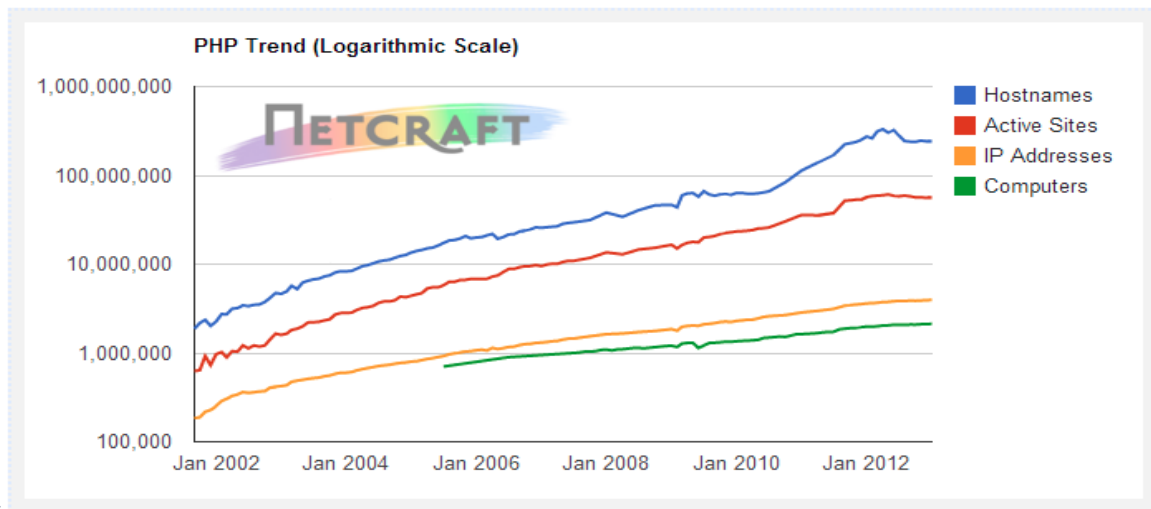
- Miễn phí

	PHP
Software	Free
Platform	Free (Linux)
Development Tools	Free ( <a href="#">PHP Coder</a> , <a href="#">jEdit</a> , ...)



# Giới thiệu về PHP – Ưu điểm 4

- Được sử dụng rộng rãi trong môi trường phát triển web
  - 244,000,000 domains (chiếm hơn 39% tên miền website)  
(01/2013 Netcraft Survey – <http://www.php.net/usage.php>)



# Giới thiệu về PHP – Ưu điểm 4

Oct 2020	Oct 2019	Programming Language	Ratings	Change
1	2	C	16.95%	+0.77%
2	1	Java	12.56%	-4.32%
3	3	Python	11.28%	+2.19%
4	4	C++	6.94%	+0.71%
5	5	C#	4.16%	+0.30%
6	6	Visual Basic	3.97%	+0.23%
7	7	JavaScript	2.14%	+0.06%
8	9	PHP	2.09%	+0.18%
9	15	R	1.99%	+0.73%
10	8	SQL	1.57%	-0.37%
11	19	Perl	1.43%	+0.40%
12	11	Groovy	1.23%	-0.16%
13	13	Ruby	1.16%	-0.16%
14	17	Go	1.16%	+0.06%
15	20	MATLAB	1.12%	+0.19%
16	12	Swift	1.09%	-0.28%
17	14	Assembly language	1.08%	-0.23%
18	10	<b>Objective-C</b>	<b>0.86%</b>	<b>-0.64%</b>
19	16	Classic Visual Basic	0.77%	-0.46%



# Một số website lớn



PHP at Yahoo!

<http://www.yahoo.com>  
The Internet's most trafficked site



vBulletin



Portal



Portal



Course Management System



Wiki



Customer Relationship Management



e-Commerce



Portal



Bulletin Board



Content Management System



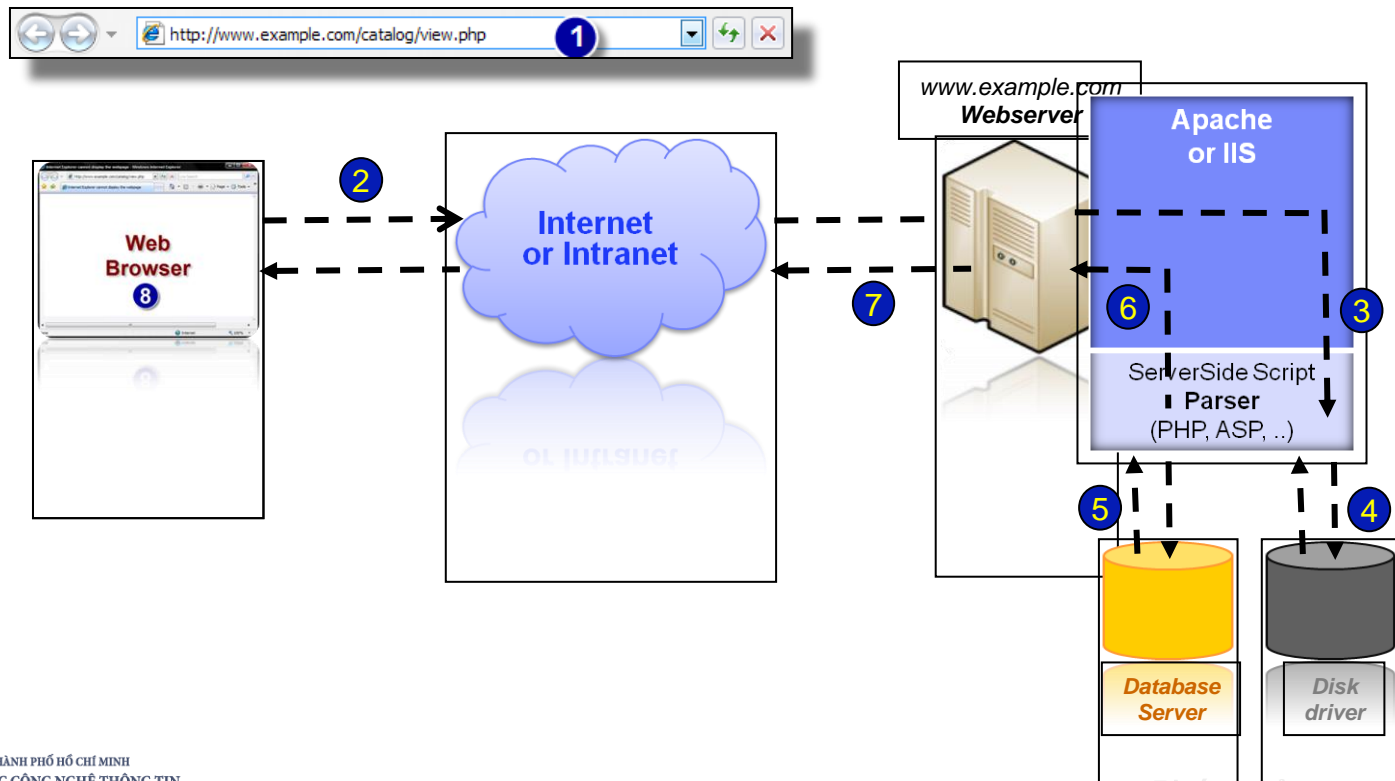
Help Desk



## 2. Cơ chế hoạt động của WebServer



# Cơ chế hoạt động của WebServer



# Chèn 1 đoạn lệnh PHP vào trang web

- Mã lệnh PHP được đặt trong các cặp thẻ sau :

Thẻ mở	Thẻ đóng
<?	?>
<?php	?>
<script language="php">	<script>

# Cơ chế hoạt động của WebServer

```
1 <html>
2 <head>
3   <title>Test Server Script Parser</title>
4 </head>
5 <body>
6
7   <h1>Server Script Parser</h1>
8   Hello world HTML
9   <br />
10  <br />
11
12  <% response.write("Hello ASP Parser !!!") %>
13  <br />
14  <br />
15
16  <?php echo "Hello PHP Parser !!!" ?>
17  <br />
18  <br />
19
20 </body>
21 </html>
22
```

**HTML Code**

**ASP Code**

**PHP Code**

# Cần gì để chạy PHP

- Download XAMPP, WAMP, LAMP
- Seminar cài đặt PHP







# 3. Cú pháp & Quy ước trong PHP



# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Xuất dữ liệu
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm



# Quy ước

- Tất cả các câu lệnh php đều cách nhau bởi dấu ";"
- Không phân biệt khoảng trắng, Tab, xuống dòng trong câu lệnh

```
<?php echo "Hello"; echo " World!"; ?>
<?php
    echo "Hello"      ;
    echo " World!";
?>
```

- Ghi chú : Theo cú pháp ghi chú của C++ & Perl

// Đây là ghi chú

# Đây là ghi chú

/\* Đây là ghi  
chú nhiều dòng\*/



# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Xuất dữ liệu
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm



# Khai báo biến

- **\$ten\_biến** = value;
- Không khai báo kiểu dữ liệu cho biến
- Biến tự động được khởi tạo ở lần đầu tiên gán giá trị cho biến
- Tên biến:
  - Có thể bao gồm các Ký tự (A..Z, a..z), Ký số (0..9), \_, \$
  - **Không** được bắt đầu bằng ký số (0..9)
  - **Phân biệt** chữ hoa – chữ thường

Ví dụ :

**\$size,**    **\$my\_drink\_size,** **\$\_drinks,** **\$drink4you;**

# Khai báo biến

- Ví dụ:

```
<?php
    $x = 5;
    $y = 4;
    echo $x + $y;
?>
```

- Hằng số - Constants

- Ví dụ:

```
define("MY_CONST", 10);
echo MY_CONST;
```

# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Xuất dữ liệu
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm



# Xuất dữ liệu

- Dùng hàm **echo**
- Dùng hàm **print**
- Dùng `var_dump`: để xuất kiểu dữ liệu kèm theo dữ liệu của biến





# Xuất dữ liệu – hàm echo

```
<?php
    $txt1 = "Learn PHP";
    $txt2 = "UIT";
    $x = 5;
    $y = 4;

    echo "<h2>" . $txt1 . "</h2>";
    echo "Study PHP at " . $txt2 . "<br>";
    echo $x + $y;
?>
```

# Xuất dữ liệu – hàm print

```
<?php
$txt1 = "Learn PHP";
$txt2 = "UIT";
$x = 5;
$y = 4;
print "<h2>" . $txt1 . "</h2>";
print "Study PHP at " . $txt2 . "<br>";
print $x + $y;
?>
```

# Xuất dữ liệu – hàm var\_dump

```
<?php  
    $x = 5985;  
    var_dump($x);  
?>
```

- Kết quả:

int(5985)

# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Xuất dữ liệu
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm



# Kiểu dữ liệu

- boolean (bool) – kiểu luận lý
- integer (int) – kiểu số nguyên
- double (float, real) – kiểu số thực
- string – kiểu chuỗi lý tự
- array – kiểu mảng
- object – kiểu đối tượng

✧ 1 Biến trong PHP có thể lưu bất kỳ kiểu dữ liệu nào



# Kiểu dữ liệu (tt)

- Chuyển kiểu dữ liệu

- Cách 1 (**automatic**)

- \$var = "100" + 15;

- \$var = "100" + 15.0;

- \$var = 39 . " Steps";

- Cách 2: (**datatype**) \$var

- Cách 3: **settype**(\$var, "**datatype**")

\$var	(int)\$var	(bool)\$var	(string)\$var
null	0	flase	""
true	1		"1"
false	0		""
"6 feet"	6	true	
"foo"	0	true	

# Các hàm liên quan

- Kiểm tra kiểu dữ liệu

Hàm	Ý nghĩa
gettype	Lấy kiểu dữ liệu
is_integer	Kiểm tra kiểu dữ liệu nguyên
is_double	Kiểm tra kiểu dữ liệu số thực
is_string	Kiểm tra kiểu dữ liệu chuỗi
is_array	Kiểm tra có phải kiểu dữ liệu mảng
is_object	Kiểm tra kiểu dữ liệu đối tượng
isset	Kiểm tra sự tồn tại của biến
unset	Hủy một biến
empty	Kiểm tra một biến có rỗng hay không

# Các hàm liên quan – ví dụ 1

```
<?php
    $a=9.6;
    $b=(int) $a;
    echo "b=".$b;
    if(empty($b))
        echo 'biến $b rỗng';
    else
        echo 'giá trị biến $b
        =' . $b;
?>
```

✧ Kết quả:

```
b=9
giá trị biến $b =9
```



## Các hàm liên quan – ví dụ 2

```
<?php
    $b="";
    if(empty($b))
        echo 'biến $b rỗng';
    else
        echo '<br>giá trị biến $b =' . $b;
    if(isset($b))
        echo '<br>có biến $b';
    else
        echo '<br>không có biến $b';
?>
```

✧ Kết quả:

**biến \$b rỗng  
có biến \$b**

# Các hàm xử lý trên số

- Một số hàm xử lý số

- **abs**

- pow**

- decbin**

- srand(seed)**

- **ceil**

- sqrt**

- bindec**

- rand**

- **floor**

- log**

- dechex**

- rand(min, max)**

- **round**

- log10**

- hexdec**

- **pi ...**



## Các hàm xử lý trên số \_ Ví dụ

```
<?php
```

```
echo(pi()); // returns 3.1415926535898  
echo(min(0, 150, 30, 20, -8, -200)); // returns -200  
echo(max(0, 150, 30, 20, -8, -200)); // returns 150  
echo(abs(-6.7)); // returns 6.7  
echo(sqrt(64)); // returns 8  
echo(round(0.60)); // returns 1  
echo(round(0.49)); // returns 0  
echo(rand());  
echo (rand(10, 100));
```

```
?>
```

# Kiểu chuỗi - string

- Toán tử nối chuỗi : dấu chấm .

```
$s = "Hello" . " World";          // $s = "Hello World"
```

- Phân biệt dấu nháy đơn và nháy kép

```
$user = "Bill";  
echo 'Hi $user';           // Hi $user  
echo "Hi $user";          // Hi Bill  
echo 'Hi' . $user;         // ????  
echo 'Hi' . '$user';       // ????
```

- Một số hàm xử lý chuỗi

○ <b>printf</b>	<b>trim</b>
○ <b>str_pad</b>	<b>str_replace</b>
○ <b>strlen</b>	<b>substr</b>

**strtolower**  
**strtoupper**  
**strcasecmp**

**echo**



# Ví dụ định dạng xuất số thực

```
<?php
    $a = 10.123456;
    printf( "0.2f<br>", $a );
    $a = number_format($a, 2);
    echo $a;
?>
```

➤ Kết quả

```
10.12
10.12
```

## Ví dụ nối chuỗi

```
<?php
    $x=2.5;
    $nghiem="Phương trình có nghiệm:". $x;

    echo $nghiem;
?>
```

✧ Kết quả:

**Phương trình có nghiệm:2.5**

## Các hàm liên quan đến string – ví dụ

```
<?php
```

```
echo strlen("Hello world!"); // outputs 12
echo str_word_count("Hello world!"); // outputs 2
echo strrev("Hello world!"); // outputs !dlrow olleH

echo strpos("Hello world!", "world"); // outputs 6
echo str_replace("world", "UIT", "Hello world!");
// outputs Hello UIT!
```

```
?>
```

# Mảng - array

- Khai báo

```
$a=array();
```

- Khai báo và khởi tạo

```
$words = array("Web", "Database", "Applications");
```

- Truy xuất phần tử của mảng

```
echo $words[0];
```

- Đặt tên chỉ số của mảng

```
$numbers = array(1=>"one", "two", "three", "four");
```

- ✧ echo \$numbers[1];

```
$array = array("first"=>1, "second"=>2, "third"=>3);
```

- ✧ echo \$array["second"];



# Ví dụ

```
<?php
    $a = array();
    $a[0]=1;
    $a[1]=2;
    $a[2]="Tâm";
    $a[3]="Tú";
    for ($i=0; $i<count($a); $i++)
    {
        echo $a[$i]."<br>";
    }
?>
```

1  
2  
Tâm  
Tú

# Duyệt mảng

```
<?php
    $ten=array("Tú", "Tùng", "Tâm");
    for($i=0;$i<count($ten);$i++)
        echo $ten[$i]."<br>";
?>
```

```
<?php
$age
= array("Tú"=>"35", "Tùng"=>"37", "tâm"=>"43");
foreach($age as $key => $value)
{
    echo "Key=" . $key . ", Value=" . $value;
    echo "<br>";
}
?>
```

Key=Tú, Value=35  
Key=Tùng, Value=37  
Key=Tâm, Value=43

# Ví dụ khởi tạo mảng và dùng chỉ số phần tử

```
<?php
$age = array();
$age["Tâm"]=32;
$age["Tú"]=22;
foreach ($age as $key=>$value)
{
    echo $key."\t".$value."<br>";
}
?>
```

Tâm 32  
Tú 22

# Mảng 1 chiều – single arrays

- Một số hàm xử lý trên mảng

- |                |                      |              |               |               |               |
|----------------|----------------------|--------------|---------------|---------------|---------------|
| ○ <b>count</b> | <b>is_array</b>      | <b>sort</b>  | <b>asort</b>  | <b>ksort</b>  | <b>usort</b>  |
| ○ <b>min</b>   | <b>array_reverse</b> | <b>rsort</b> | <b>arsort</b> | <b>krsprt</b> | <b>uasort</b> |
| ○ <b>max</b>   | <b>uksort</b>        |              |               |               |               |

- **Ví dụ:**

```
<?php
    $ten=array("Tú","Tùng","Tâm");
    sort($ten);
    for($i=0;$i<count($ten);$i++)
    echo $ten[$i]."<br>";
?>
```

Tâm  
Tùng  
Tú

# Ví dụ về sắp xếp giảm

```
<?php
    $num=array(1,2,3,4,5);
    rsort($num);
    for($i=0;$i<count($num);$i++)
        echo
$num[$i]."<br>";
?>
```

➤ Kết  
quả:

5  
4  
3  
2  
1

# Một số hàm liên quan đến mảng

- `array_push(array, elements)` : Thêm elements vào cuối mảng
- `array_pop(array)` : Lấy phần tử cuối ra khỏi mảng
- `array_unshift(array, elements)` : Thêm elements vào đầu mảng
- `array_shift(array)` : Lấy phần tử đầu ra khỏi mảng
- `array_merge(array, array)` : kết 2 mảng lại và trả ra mảng mới
- `shuffle(array)` : Sort random mảng
- `sort(array, flag)` : `flag = {sort_regular, sort_numeric, sort_string, sort_locale_string}`

# Mảng 2 chiều

- Khai báo mảng 2 chiều

- Khai báo và tạo mảng 2 chiều có 2 dòng, mỗi dòng chứa 3 cột

```
$a=array();
```

```
for($i=0;$i<2;$i++)
```

```
    $a[$i]=array();
```

## Ví dụ mảng 2 chiều – cách 1

```
<?php
```

```
$lop=array();  
$a1=array("1125001","Tuấn",7.5);  
$a2= array("1125002","Tùng",8.0);  
$a3 = array("1125003","Âm",7.0);  
$a4 = array("1125003","Tú",6.5);  
$lop[0] =$a1;  
$lop[1] =$a2;  
$lop[2] =$a3;  
$lop[3] =$a4;
```

```
?>
```



# Ví dụ mảng 2 chiều – cách 1

```
<?php echo "<table border='1' bordercolor='#CC00FF'
cellspacing='0'>";
echo "<tr><th>Mã số</th><th>Tên</th><th>Điểm trung bình</th>";
for ($i=0; $i<count($lop); $i++)
{
    echo "<tr>";
    for ($j=0; $j<count($lop[$i]); $j++) {
        echo "<td>".$lop[$i][$j]."</td>";
    }
    echo "</tr>";
}
echo "</table>";
?>
```

Mã số	Tên	Điểm trung bình
1125001	Tuấn	7.5
1125002	Tùng	8
1125003	Tâm	7
1125003	Tú	6.5

# Mảng 2 chiều – cách 2

```
<?php $lop=array(  
    array("1125001","Tuấn",7.5),  
    array("1125002","Tùng",8.0),  
    array("1125003","Tâm",7.0),  
    array("1125003","Tú",6.5) );  
echo "<table border='1' bordercolor='#CC00FF' cellspacing='0'>";  
echo "<tr><th>Mã số</th><th>Tên</th><th>Điểm trung bình</th>";  
for($i=0;$i<count($lop);$i++){  
    echo "<tr>";  
    for($j=0;$j<count($lop[$i]);$j++){  
        echo "<td>".$lop[$i][$j]."</td>";  
    }  
    echo "</tr>";  
}  
echo "</table>";  
?>
```

Mã số	Tên	Điểm trung bình
1125001	Tuấn	7.5
1125002	Tùng	8
1125003	Tâm	7
1125003	Tú	6.5

# Ví dụ mảng 2 chiều

```
<?php
$lop= array(array("1125001", "Tuấn", 7.5),
             array("1125002", "Tùng", 8.0),
             array("1125003", "Tâm", 7.0),
             array("1125003", "Tú", 6.5)           );
echo "<table border='1' bordercolor='#CC00FF' cellspacing='0'>"
echo "<tr><th>Mã số</th><th>Tên</th><th>Điểm trung bình</th>";
for($i=0;$i<count($lop);$i++){
    echo "<tr>";
    foreach($lop[$i] as $key=>$value) {
        echo "<td>".$value."</td>";
        echo "</tr>";
    }
echo "</table>";
```

Mã số	Tên	Điểm trung bình
1125001	Tuấn	7.5
1125002	Tùng	8
1125003	Tâm	7
1125003	Tú	6.5

# Đặt và lấy múi giờ

- Đặt múi giờ
  - `date_default_timezone_set("Asia/Ho_Chi_Minh");`
- Lấy múi giờ
  - `$timezone = date_default_timezone_get();`



# Các hàm thời gian

- Lấy ngày hệ thống: **date**("định dạng")

Định dạng	Mô tả	Giá trị
d	Ngày trong tháng, lấy 2 ký tự	01-31
D	Ngày trong tuần, lấy 3 ký tự	Mon-Sun
j	Ngày trong tháng, lấy hai ký tự, không lấy số 0 đứng trước	1-31
S	Hậu tố của ngày trong tháng	st,nd,rd
z	Ngày trong năm	0-365
w	Ngày trong tuần theo số	0: Sunday " 6: Saturday
W	Tuần trong năm	1 – 52
F	Tháng trong năm dạng chuỗi	January
m	Tháng trong năm, lấy 2 ký tự	01 – 12
M	Tháng trong năm, lấy 3 ký tự	Jan – Dec
n	Tháng trong năm không lấy số 0	1 – 12
t	Số ngày tối đa trong tháng	28 - 31
L	Năm nhuận có giá trị 1, thường có giá trị 0	0, 1
Y	Năm viết đầy đủ	2014
y	Lấy 2 ký tự cuối của năm	14

# Định dạng giờ, phút, giây

Định dạng	Mô tả	Giá trị
a	Buổi trong ngày chữ thường	am-pm
A	Buổi trong ngày chữ hoa	AM-PM
g	Định dạng 12h không có số 0 đứng đầu	1-12
G	Định dạng 24h không có số 0 đứng đầu	0-24
h	Định dạng 12h có số 0 đứng đầu	01-12
H	Định dạng 24h có số 0 đứng đầu	00-23
i	Định dạng phút	00-59
s	Định dạng giây	00-59

# Ví dụ

```
<?php  
    echo date(" g:m:s a d-m-y");  
?>
```

**Kết quả:**

2:10:28 am 27-10-14

5

# Lấy ngày hiện tại

- Dùng hàm : array `getdate()`;
- Hàm trả về một mảng gồm 10 phần tử có chỉ số là chuỗi chứa các giá trị của ngày hiện tại. Các chỉ số của mảng theo định dạng sau:



# Các chỉ số của mảng

Chỉ số	Ý nghĩa
[seconds]	Chỉ số lấy giá trị giây
[minutes]	Chỉ số lấy giá trị phút
[hours]	Chỉ số lấy giá trị giờ
[mday]	Ngày trong tháng
[wday]	Ngày trong tuần 0 -> chủ nhật,..
[weekday]	Thứ trong tuần
[mon]	Tháng 1-12
[month]	Tháng dạng chuỗi
[year]	Năm
[yday]	Ngày trong năm

## Ví dụ

```
<?php
    $m=getdate();
    $thu=array("Monday"=>"Thứ 2","Tuesday"=>"Thứ 3",
    "Wednesday"=>"Thứ 4", "Thursday"=>"Thứ 5",
    "Friday"=>"Thứ 6", "Saturday"=>"Thứ 7", "Sunday"=>"Chủ
    nhật");
    $thutrongtuan=$m['weekday'];
    echo "hôm nay là: ".$thu[$thutrongtuan]." ngày
    ".$m[mday]." Tháng ".$m[mon]." Năm ".$m[year];
?>
```

**Kết quả:**

**hôm nay là: Chủ nhật ngày 26 Tháng 10 Năm 2014**

# Tổng số giây từ 1/1/1970

- Dùng hàm : `long time()`
- Ví dụ:

```
<?php  
    echo time();  
?>
```

Kết quả: 1414296069

# Chuyển chuỗi thành thời gian

- `long strtotime(chuỗi);`

```
<?php
    echo "Tổng số giây từ 1-1-1970 đến ";
    //Lấy ngày sau khi đổi chuỗi ra ngày
    $t=getdate(strtotime("October 14 2014"));
    echo "Ngày ".$t[mday]."Tháng ".$t[mon]. " Năm
    ".$t[year]."là";
    //in tổng số ngày từ 1-1-1970 đến ngày cần chuyển
    echo strtotime("October 14 2014");
?>
```

## **Kết quả:**

Tổng số giây từ 1-1-1970 đến Ngày 14 Tháng 10 Năm 2014  
là 1413244800

## Lấy ngày từ control “date”

01/06/2015

2015-01-06

Ngày 6 Tháng 1 Năm 2015

# Code lấy ngày từ control

```
<body>
<form method="GET">
<input type="date" name="date"></input>
<input type="Submit" value="GetDate" name='getdate'>
</form>
<?php
if(isset($_GET['getdate']) && ($_GET['getdate']=='GetDate'))
{
    $date=$_GET['date'];
    echo $date."<br>";
    $m=getdate(strtotime($date));
    echo "Ngày " . $m['mday'] . " Tháng " . $m['mon'] . " Năm " . $m['year'];
}
}?>
</body>
```

# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Kiểu dữ liệu
- **Toán tử**
- Cấu trúc điều khiển
- Hàm



# Toán tử

Loại	Toán tử	Ghi chú
	new .	
	. [] ()	
Toán học	+, -, *, **, /, %, ++, --	
So sánh	< > <= >= != == === !==	
Luận lý	&&    ?: ,	
Xử lý bit	! ~ << >> >>> AND OR XOR	
Gán	= += -= *= /= %= >>= <<= &=  = ^= .=	
Ép kiểu	(kiểu dữ liệu)	(int) (double) (string)...



## Ví dụ

```
<?php
```

```
    $x = 2;
```

```
    $y = 3;
```

```
    echo $x ** $y; // 2*2*2=8
```

```
?>
```

```
<?php
```

```
    $x = 100;
```

```
    $y = "100";
```

```
    if ($x === $y)
```

```
        echo "true";
```

```
    else
```

```
        echo "false"; // out put false
```

```
?>
```

# Ví dụ

```
<?php
    $x = 100;
    $y = "100";
    if ($x !== $y)
        echo "true"; //vì 2 kiểu dữ liệu khác
nhau
    else
        echo "false";
?>
```

```
<?php
    echo $user = $_GET["user"] ?? "no user"; //no
user
    $size=40;
    echo $s = $size ?? "41"; //40
?>
```

# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm



# Cấu trúc điều khiển

- Điều kiện **if, if...else**
- Điều khiển **switch**
- Vòng lặp **for**
- Vòng lặp **while**
- Vòng lặp **do.. while**
- Vòng lặp **foreach**



# Điều kiện if

**if** (*condition*)

{

*statement[s] if true*

}

**else** //(condition)

{

*statement[s] if false*

}

Ví dụ:

```
$x = 5;
```

```
if ($x < 4)
```

```
    echo "$x is less than 4";
```

```
else
```

```
    print '$x isn't less than  
4';
```

**\$x isn't less than 4**

# Điều khiển switch

switch (*expression*)

```
{  
    case label 1 :  
        statementlist  
        break;  
    case label 2 :  
        statementlist  
        break;  
    ...  
    default :  
        statementlist  
}
```

You picked threeYou picked four

Ví dụ:

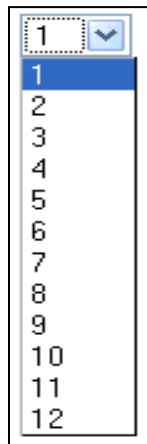
```
$menu = 3;  
switch ($menu) {  
    case 1:  
        echo "You picked one";  
        break;  
    case 2:  
        echo "You picked two";  
        break;  
    case 3:  
        echo "You picked three";  
        //break;  
    case 4:  
        echo "You picked four";  
        break;  
    default:  
        echo "You picked another option";  
}
```

# Vòng lặp for

```
for([initial expression];[condition];[update expression])  
{  
    statement[s] inside loop  
}
```

## ▪ Ví dụ:

```
echo "<select>";  
for ($i = 1; $i <= 12; $i++) {  
    echo "<option>$i</option>";  
}  
echo "</select>";
```



1	▼
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	

# Vòng lặp while, do...while

```
while (expression)
{
    statements
}
```

```
do
{
    statements
}while (expression);
```

Ví dụ:

```
$i = 1; $j = 9;

while ($i <= 10)
{

    $temp = $i * $j;

    echo "$j * $i = $temp<br>";

    $i++;

}
```

```
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
9 x 10 = 90
```



# Vòng lặp foreach

```
foreach (array as variable)
{
    statements
}
```

Ví dụ:

```
<?php    $sp = array('Mã sản phẩm' => 'SP001',
                  'Tên sản phẩm' => 'NOKIA 720',
                  'Giá bán' => 5000000);
echo "<table border='1' bordercolor='#CC00FF' cellspacing='0'>";
foreach ($sp as $key => $value) {
    echo "<tr><td>$key</td><td>$value</td></tr>";
}
echo "</table>";
```

<?>

Mã sản phẩm	SP001
Tên sản phẩm	NOKIA 720
Giá bán	5000000

# Cú pháp & Quy ước trong PHP

- Quy ước
- Khai báo biến
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển
- Hàm



# Hàm - function

```
function functionName  
([parameter1]...[,parameterN])  
{  
    statement[s] ;  
}
```

```
function functionName  
([parameter1]...[,parameterN])  
{  
    statement[s] ;  
    return .... ;  
}
```



# Hàm – ví dụ

```
<?php
    function Tong ($a, $b)
    {
        $s=$a+$b;
        return $s;
    }
    $t=Tong (5, 6) ;
    echo "Tổng =" . $t;
?>
```

**Tổng =11**



# Hàm – Phạm vi biến

```
<?php
function doublevalue($var=10)
{
    global $temp;
    $temp = $var * 2;
}

$temp = 5;
doublevalue();
echo "\$temp is: $temp";
?>
```

**\$temp is: 5**

**\$temp is:  
20**

# Hàm – Tham trị và Tham biến

```
<?php
function doublevalue( $var)
{
    $var = $var * 2;
}

$a = 5;
doublevalue($a);
echo "\$a is: $a";
?>
```

**\$variable is: 5**

**\$a is: 10**

# Hàm – include & require

✧ khaibao.php

```
<?php
function InLuong($ten, $luong)
{
    echo "<table bordercolor='#CC00FF'
        border='1' cellpadding='0'>";    echo
        "<tr><th>Tên</th><th>Lương</th>";
    echo "<tr><td>".$ten."</td>";
    echo "<td>".$luong."</td>";
    echo "</tr>";
    echo "</table>";
}
?>
```

✧ goiInclude.php

```
<?php
    $ten="Nguyễn Xuân Tú";
    $luong=7000000;
    include "khaibao.php";
    InLuong($ten,$luong);

?>
```



## 4. Lập trình hướng đối tượng





# Khai báo lớp

```
<?php
```

```
class TênLớp
```

```
{
```

```
    Phạm vi truy xuất Thuộc tính;
```

```
    Phạm vi truy xuất Phương thức của lớp;
```

```
}
```

```
?>
```

# Khai báo lớp

```
<?php
```

```
class TênLớp
```

```
{
```

```
    Phạm vi truy xuất Thuộc tính;
```

```
    Phạm vi truy xuất Phương thức của  
lớp;
```

```
}
```

```
?>
```

# Tạo ra đối tượng thuộc lớp

```
<?php
```

```
    $đổitượng = new TênLớp;
```

```
    //Truy xuất đến các thành phần của đối tượng
```

```
    $đổitượng ->Tên Phương Thức;
```

```
?>
```

# Tạo ra đối tượng thuộc lớp

```
<?php
```

```
    $đổitượng = new TênLớp;
```

```
    //Truy xuất đến các thành phần của đối tượng
```

```
    $đổitượng ->Tên Phương Thức;
```

```
?>
```

# Lớp – Ví dụ

## ✧ Sinhvien.php

```
<?php
    class SV{
        private $mssv;
        private $ten;
        private $dtb;
        public function Set($ms,$t,$d){
            $this->mssv=$ms;
            $this->ten=$t;
            $this->dtb=$d;
        }
    ?>
```

## Lớp – Ví dụ

```
public function In() {  
    echo $this->mssv;  
    echo "<br>";  
    echo $this->ten;  
    echo "<br>";  
    echo $this->dtb;  
    echo "<br>";  
}
```

?>

# Lớp – Ví dụ (tt)

## ✧ xulysinhvien.php

```
<?php
    include "sinhvien.php";
    $sv1= new SV();
    $sv1->Set("001","Nguyễn Xuân Tú",9.5);
    $sv1->In();
    $sv1->mssv="009";
    echo $sv1->mssv;
?>
```

# Phạm vi truy xuất mặc định

- Nếu không chỉ định phạm vi truy xuất của các thành phần trong lớp khi khai báo thì PHP sẽ gán **mặc định** phạm vi truy xuất là **public**
- Ví dụ

```
class PhanSo{  
    $tuso;  
    $mauso;  
}
```

- **\$tuso**, **\$mauso** có phạm vi truy xuất là public



# Phương thức khởi tạo

- Dùng để khởi tạo các giá trị ban đầu cho các thuộc tính của đối tượng
- Có 2 cách dùng

❖ Cách 1:

```
public function __construct($name,$age,$color);
```

**Hoặc:**

```
public function __construct();
```

❖ Cách 2:

```
public function TênLớp($name,$age,$color);
```

**Hoặc:**

```
public function TênLớp();
```

# Phương thức hủy

- Thường hủy các giá trị của biến được khởi tạo trong session
- Cách dùng:

```
public function __destruct()  
  
{  
  
    echo "Destruct được gọi";  
  
}
```

# Ví dụ đối tượng

```
<?php
class ConMeo{

    private $name;
    private $age;
    private $color;
    public function ConMeo()
    {
        $this->name = 'Mimi';
        $this->age = 1;
        $this->color = 'Vàng';
    }
}
```

# Ví dụ đối tượng

```
/*
public function __construct($name,$age,$color)
{
    $this->name = $name;
    $this->age = $age;
    $this->color = $color;
}
*/
/*public function ConMeo($name,$age,$color)
{
    $this->name = $name;
    $this->age = $age;
    $this->color = $color;
} */
```

# Ví dụ đối tượng

```
public function __construct()  
{  
    $this->name = 'Mimi';  
    $this->age = 2;  
    $this->color = 'Trang';  
}  
public function Show()  
{  
    echo $this->name."<br>";  
    echo $this->age."<br>";  
    echo $this->color;  
}
```

# Ví dụ đối tượng

```
public function Gan()  
{  
    $this->name="keke";  
}  
public function Gan1($t)  
{  
    $this->name=$t;  
}  
public function __destruct()  
{  
    echo "Destruct duoc goi";  
}  
}
```

# Ví dụ đối tượng

```
// $m=new ConMeo ("Mimi",1,"do");  
$m=new ConMeo ();  
$m->Show ();
```

?>

# Thừa kế trong PHP

- Là hình thức định nghĩa lớp mới kế thừa từ một đã có sẵn.

Lớp cơ bản, cơ sở



Lớp dẫn xuất



# Thừa kế trong PHP \_ Khai báo

```
class Cơ sở  
{  
  
}  
class Dẫn xuất extends Cơ sở  
{  
  
}
```



# Thừa kế

- Lớp dẫn xuất thừa kế lại những phương thức và thuộc tính từ lớp cơ sở.
- Thừa kế trong PHP là thừa kế **public**

## Ví dụ - Lớp cơ sở

```
<?php
    class Hình{
        private $mau;
        public function GanMau ($m) {
            $this->mau=$m;
        }
        public function LayMau () {
            return $this->mau;
        }
        public function XuatMau () {
            echo "Màu:". $this->mau;
        }
    }
}??>
```

## Ví dụ - lớp dẫn xuất

```
<?php
    include "hinh.php";
    class HìnhTron extends Hình{
    private $R;
    public function DienTich()
    {
        return $this->R*pi();
    }
    public function GanR($m,$r)
    {
        $this->GanMau($m);
        $this->R=$r;
    }
}
```

## Ví dụ \_ Lớp dẫn xuất

```
public function Xuat() {  
  
    $this->XuatMau();  
    $dt=$this->DienTich();  
  
    $dt=number_format($dt,2);  
    echo "<br>Diện tích:". $dt;  
  
}  
  
?>
```

## Ví dụ - Tạo đối tượng và gọi hàm

```
<?php  
    include "hinhtron.php";  
    $ht=new HìnhTron();  
    $ht->GanR("Vàng",2);  
    $ht->Xuat();  
?>
```

**Màu: Vàng**  
**Diện tích: 6.28**

## Ví dụ - truy xuất thành phần qua tên lớp

```
<?php
    include "hinhtron.php";
    HìnhTron::GanR("Vàng", 2);
    HìnhTron::Xuat();
?>
```

**Màu: Vàng**  
**Diện tích: 6.28**

# Hàm chồng (overloading)

- Trong PHP **không** cho phép khai báo hàm chồng trong 1 lớp

```
public function Tong($n)
{
    $this->Tu=$this->Tu*$n->Mau+$this->Mau*$n->Tu;
    $this->Mau=$this->Mau*$n->Tu;
}
public function Tong()
{
    $this->Tu++;
    $this->Mau++;
}
```

**Chương trình báo lỗi**



# Tính override

- Trong lớp dẫn xuất định nghĩa lại một hay nhiều phương thức đã có trong lớp cơ sở
- Tính đa hình trong lập trình hướng đối tượng trên PHP thể hiện qua tính **override**.

## Ví dụ - tính đa hình

```
<?php
class Hình{
    public function Ve(){

        echo "Ve hình";

    }
}
class HìnhVuong extends Hình{

    public function Ve(){

        echo "Ve hình vuông";

    }
}
```

## Ví dụ - Tính đa hình

```
class HìnhTron extends Hình{  
    public function Ve() {  
  
        echo "Ve hình tron";  
  
    }  
}  
  
$a=new Hình();  
$a->Ve();  
$b=new HìnhTron();  
$b->Ve();?>
```

# clone và parent

- **clone**: tạo ra sao chép một đối tượng cho một đối tượng

```
$m=new HìnhVuong();
```

```
$m->Show();
```

```
$n=clone $m;
```

```
$n->Ve();
```

# clone và parent

- **parent**: Dùng để gọi phương thức của lớp cha trong trường hợp định nghĩa lại hàm
- Ví dụ:

```
<?php
class A
{
    private $name;
    public function __construct($name)
    {
        $this->name=$name;
    }
    public function view()
    {
        echo "name:".$this->name;
    }
}
```

## Ví dụ (tt)

```
class B extends A
{
    public function __construct($name)
    {
        parent::__construct($name);
    }

    public function view()
    {
        parent::view();
    }
}
```

## Ví dụ (tt)

```
$c=new B("Nguyễn Văn Tú");  
$c->view();
```

?>

# Bài tập

- Cài đặt lớp hình chữ nhật:
  - Khởi tạo
  - Diện tích
  - Chu vi
- Cài đặt lớp hình hộp chữ nhật:
  - Khởi tạo
  - Diện tích
  - Thể tích

(ghi chú: phương thức tính diện tích của 2 lớp đặt tên giống nhau )
- Tạo đối tượng thuộc lớp hình hộp chữ nhật, gọi phương thức thể tích, diện tích



# Bài tập

- Triển khai lớp hình tròn:
  - Với các phương thức: Khởi tạo có đối số, diện tích hình tròn, chu vi hình tròn
- Cài đặt lớp hình trụ tròn: Khởi tạo có đối số, Diện tích hình trụ tròn, thể tích hình trụ tròn.
- Tạo đối tượng và gọi phương thức diện tích hình trụ tròn, thể tích hình trụ tròn

# Lớp trừu tượng (abstract)

- Lớp **abstract** dùng để định nghĩa các phương thức mà các phương thức này sẽ được định nghĩa lại ở lớp dẫn xuất.
- Các phương thức của lớp **abstract** phải được khai báo **abstract** và có phạm vi truy xuất **public** hoặc **protected**.
- Có thể khai báo thuộc tính cho lớp **abstract**
- Không thể tạo ra một đối tượng thuộc lớp **abstract**

## Ví dụ - lớp abstract

```
<?php
    abstract class Hinh{
        private $mau;
        public function GanMau ($m) {
            $this->mau=$m;
        }
        public function LayMau () {
            return $this->mau;
        }
    }
```

## Ví dụ - lớp abstract

```
public function Xuat() {  
    echo "Màu:". $this->mau;  
}  
abstract public function DienTich();  
abstract public function ChuVi();  
}
```

## Ví dụ - lớp abstract

```
class HìnhTron extends Hình{
    private $R;
    public function Xuất() {
        parent::Xuất();
        echo "<br>Diện tích:".number_format($this->DienTich(), 2);
        echo "<br>Chu vi:".number_format($this->ChuVi(), 2);
    }
}
```

## Ví dụ - lớp abstract

```
public function DienTich() {  
    return pi()*$this->R*$this->R;    }  
public function ChuVi() {  
    return 2*pi()*$this->R;  
}  
public function GanR($r) {  
    $this->R=$r;  
}  
}
```

## Ví dụ - lớp abstract

```
class HìnhVuong extends Hình
{
    private $Canh;
    public function Xuat() {
        echo "<br>";
        parent::Xuat();
        echo "<br>Diện tích:".number_format($this->DienTich(), 2);
        echo "<br>Chu vi:".number_format($this->ChuVi(), 2);
    }
}
```

## Ví dụ - lớp abstract

```
public function DienTich () {  
    return $this->Canh*$this->Canh;  
}  
  
public function ChuVi () {  
    return $this->Canh*4;  
}  
  
public function GanCanh ($c) {  
    $this->Canh=$c;  
}  
}
```



## Ví dụ - lớp abstract

```
$ht=new HìnhTron();  
$ht->GanR(5);  
$ht->GanMau("Xanh");  
echo "Thông tin hình tròn<br>";  
$ht->Xuat();  
$hv=new HìnhVuong();  
$hv->GanMau("Đỏ");  
$hv->GanCanh(4);  
echo "<br>Thông tin hình vuông";  
$hv->Xuat(); ?>
```

# Interface

- **Interface** không phải là 1 lớp. Nó được mô tả như là 1 bản thiết kế cho các **class** có chung cách thức hoạt động
- Không thể định nghĩa các thuộc tính, khởi tạo đối tượng mà chỉ khai báo các phương thức
- Chỉ khai báo mà không có phần thân hàm

# Interface

- Không có khái niệm phạm vi của phương thức, tất cả đều là **public**.
- Lớp con kế thừa từ interface sẽ phải **override** tất cả các phương thức trong đó.
- Một lớp có thể kế thừa từ nhiều **interface** khác nhau bằng từ khóa **implements**

# Interface – ví dụ

```
interface Move {  
    function run(); }  
class Dog implements Move {  
    public function run ()  
    { echo "Con chó chạy bằng 4 chân";  
    }  
}  
class Car implements Move {  
    public function run ()  
    {  
        echo "Xe hơi chạy bằng 4 bánh";  
    }  
}
```

# So sánh Abstract - interface

- **Không** thể khởi tạo đối tượng **Abstract Class**
- **Bất kỳ** lớp nào có chứa ít nhất 1 phương thức trừu tượng thì chắc chắn nó phải là **Abstract Class**. 1 **Abstract Class** có thể chứa các phương thức trừu tượng hoặc không trừu tượng.
- Phương thức **abstract** của **Abstract Class** không có thân hàm
- Các phương thức **abstract** phải được định nghĩa lại ở lớp dẫn xuất
- Không hỗ trợ đa thừa kế

# So sánh Abstract - interface

- **Interface** được định nghĩa để cung cấp các tên hàm chung để có thể triển khai.
- **Interface** được xem như là bộ khung của lớp dẫn xuất
- **Interface** cũng không thể khởi tạo
- Các phương thức trong **Interface** mặc định là các phương thức trừu tượng

# So sánh Abstract - interface

- Các phương thức trong Interface phải có phạm vi truy xuất public và không có thân hàm
- Interface có thể được **extends** với nhau
- 1 lớp thể **implements** nhiều Interface





# 5. Bài tập





# Bài tập

- **Bài 1:** Cài đặt lớp mảng một chiều các số nguyên với các phương thức
  - ✧ Khởi tạo số phần tử và giá trị cho từng phần tử của mảng
  - ✧ Phương thức tính tổng các phần tử trong mảng
  - ✧ Phương thức xuất mảng ra trang web
  - ✧ Phương thức tìm phần tử lớn nhất
  - ✧ Phương thức tìm phần tử nhỏ nhất
  - ✧ Phương thức tìm một phần tử có hay không trong mảng
  - ✧ Phương thức xóa một phần tử trong mảng
  - ✧ Phương thức sắp xếp mảng tăng giảm
  - ✧ Phương thức kiểm tra mảng có đối xứng hay không
  - ✧ Phương thức đảo một mảng
  - ✧ Phương thức đếm số phần tử có giá trị bằng x trong mảng

# Bài tập (tt)

- **Bài 2:** Cài đặt lớp sinh viên, biết rằng thông tin của các sinh viên gồm: Mã số sinh viên, tên sinh viên, điểm trung bình. Và các phương thức sau:
  - Phương thức khởi tạo
  - Gán mã sinh viên, tên sinh viên, điểm trung bình
  - Lấy mã sinh viên, tên sinh viên, điểm trung bình

# Bài tập (tt)

- **Bài 3:** Cài đặt lớp danh sách sinh viên để quản lý danh sách các sinh viên trong lớp, với các chức năng sau:
  - Xuất lớp học ra trang web
  - Thêm một sinh viên vào lớp
  - Xóa một sinh viên theo mã số sinh viên
  - Tìm một sinh viên theo tên
  - Cho biết điểm trung bình cao nhất trong lớp học là bao nhiêu
  - Sắp xếp danh sách sinh viên tăng theo điểm trung bình

# Bài tập (tt)

- **Bài 4:** Cài đặt lớp phân số với các phương thức
  - Khởi tạo một phân số
  - Xuất phân số ra trang web
  - Cộng, trừ, nhân, chia hai phân số
  - Đơn giản một phân số
  - So sánh hai phân số

## Bài tập (tt)

- **Bài 5:** Cài đặt lớp ngày tháng năm với các phương thức cần thiết để in ra thứ của một ngày bất kỳ.

## Bài tập (tt)

**Bài 6:** Một chiếc xe máy chạy 100km tốn 2 lít xăng, cứ chở thêm 10kg hàng xe tốn thêm 0.1lit xăng.

Một chiếc xe tải chạy 100km tốn 20lit xăng, cứ chở thêm 1000kg hàng xe tốn thêm 1lit xăng.

Dùng kế thừa xây dựng lớp XeMay và XeTai cho phép:

- Chất một lượng hàng lên xe.
- Bỏ bớt một lượng hàng xuống xe.
- Đổ một lượng xăng vào xe.
- Cho xe chạy một đoạn đường.
- Kiểm tra xem xe đã hết xăng chưa.

Cho biết lượng xăng còn trong xe.



## Cảm ơn đã theo dõi

Hy vọng cùng nhau đi đến thành công.