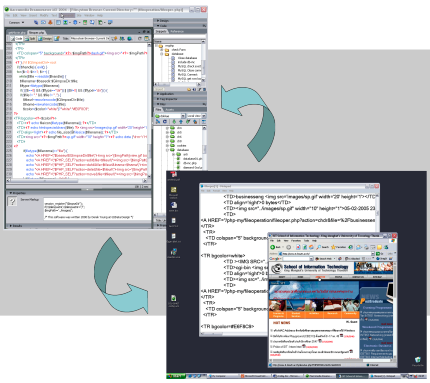


# บทที่ 2

## หลักการสำคัญของภาษา PHP

ภาษาการเขียนโปรแกรม (Programming Language) มีหลักการที่สำคัญของภาษาโดยทั่วไป เช่น การเขียนคำสั่งของโปรแกรม หมายเหตุ การกำหนดชื่อ การใช้ตัวแปร ชนิดของข้อมูล การกำหนดข้อมูล นิพจน์ (expression) คำสั่งประเภทต่างๆ ได้แก่ คำสั่งกำหนดค่าให้ตัวแปร คำสั่งตัดสินใจเลือกทำงาน คำสั่งวนรอบ คำสั่งกระโดดข้ามการทำงาน การเรียกใช้ฟังชัน การสร้างฟังก์ชันใหม่ เป็นต้น นอกจากนี้เครื่องมือของการสร้างโปรแกรมในภาษาต่างๆ ยังมีการเตรียมชุดของฟังก์ชันสำเร็จรูปที่สนับสนุนการทำงานเฉพาะรวมไว้ในไลบรารี่ เช่น การแสดงผลบน Windows ทั้งการแสดงผลแบบกราฟฟิกและการรับข้อมูลเข้าจากทั้ง keyboard และ mouse การทำงานกับเครือข่าย ซึ่งจะมีชุดของฟังก์ชันเฉพาะที่เรียกว่า Application Program Interface – API. เช่น Win32API เป็นชุดฟังก์ชันที่สนับสนุนการทำงานบน Microsoft window เป็นต้น

โปรแกรมในแต่ละภาษาจะมีรูปแบบคำสั่ง (syntax) ที่แตกต่างกัน อาจมีโครงสร้างภาษาไม่เหมือนกัน เช่น ภาษาที่เป็นโครงสร้าง (structured programming language) ภาษาเชิงวัตถุ (object-oriented programming) และยังมีลักษณะการสร้างโปรแกรมที่ทำงานได้หรือการแปลโปรแกรมที่แตกต่างกันเป็นแบบ compile และ interpret ดังนั้นการศึกษาการเขียนโปรแกรมในภาษาและในสภาวะแวดล้อมต่างๆ จึงต้องศึกษาใน 3 ด้านใหญ่ๆ คือ 1. หลักสำคัญของภาษานั้นๆ รูปแบบของคำสั่งและการทำงานของคำสั่ง 2. การใช้ฟังก์ชันเฉพาะที่สนับสนุนสภาวะแวดล้อม เช่น การเขียนเว็บโปรแกรม การเขียนโปรแกรมที่ทำงานบนอุปกรณ์เคลื่อนที่ การเขียนโปรแกรมที่ทำงานแยกเดี่ยว ฯลฯ และ 3. การแปลให้โปรแกรมนั้นทำงานได้ และอาจมีส่วนที่เกี่ยวข้องอีก ได้แก่ การใช้เครื่องมือสำหรับช่วยในการพัฒนาโปรแกรม (Programming Development Tools) หรือชุดเครื่องมือสภาพแวดล้อมสำหรับการพัฒนา (Integrated Development Environment – IDE.)

ในบทที่ 2 นี้จะกล่าวถึงหลักการสำคัญของภาษา PHP อย่างรวบรัดเนื่องจากผู้เรียนจะต้องผ่านการเรียนรู้หลักการด้านการเขียนโปรแกรมคอมพิวเตอร์มาบ้างแล้ว และจะยกตัวอย่างประกอบกับการกล่าวถึงคำสั่งต่างๆ

หมายเหตุ ในการศึกษาจากตำรานี้ผู้ศึกษาควรจะเรียนรู้คำสั่งในภาษา HTML มาก่อนแล้ว เนื่องจากในตำรานี้จะไม่ได้กล่าวถึงรายละเอียดทั้งหมดของคำสั่งหรือ tag ในภาษา HTML เพียงแต่จะกล่าวถึงเฉพาะ tag บาง tag เพื่อใช้ในการสร้างโปรแกรมเว็บเท่านั้น และในตัวอย่างของ script ที่จะใช้อธิบายบางตัวอย่างจะปรากฏหมายเลขบรรทัดนำหน้าคำสั่ง หมายเลขบรรทัดดังกล่าวแสดงเพื่อการอ้างอิงถึงในการอธิบายเท่านั้น ในการเขียน script จริงจะไม่มีการพิมพ์หมายเลขบรรทัดนำหน้า

# 1 การเขียน *PHP Script* และการแสดงผลเบื้องต้น

ภาษา PHP เป็นภาษาในการสร้างเว็บโปรแกรมที่เป็นภาษา script สามารถเขียนโปรแกรมเป็นส่วนย่อยรวมกับคำสั่ง HTML ในไฟล์หรือเพจเดียวกันได้ แต่เมื่อเพจนั้นถูกเรียกใช้งานจากผู้ใช้จะเกิดการประมวลผลบนแม่ข่าย web server ซึ่งจะมีการดำเนินการต่างๆ กับทรัพยากรบนแม่ข่าย เช่น เรียกใช้ไฟล์ข้อมูล ใช้หรือบันทึกข้อมูลในฐานข้อมูล เรียกใช้บริการจากแม่ข่ายๆ อื่นๆ เกิดขึ้นบนตัว web server เอง และจะส่งผลไปยัง web browser (อาจเรียกว่า client) จากคำสั่งส่งผลออก (output statement) ซึ่งผู้ใช้จะเห็นผลลัพธ์เฉพาะที่มีการส่งผลลัพธ์ออกจากคำสั่ง PHP หรือคำสั่งใน mode HTML เท่านั้น ส่วนตัวคำสั่งในภาษา PHP จะไม่ไปปรากฏที่เครื่องของผู้ใช้ และ code ที่ส่งกลับไปหาเครื่องผู้ใช้จะถือว่าเป็น source code ของ browser ในรูปแบบที่ browser รู้จัก เช่น HTML, CSS, JavaScript ฯลฯ

การเขียนโปรแกรมเว็บด้วยภาษา PHP มีลักษณะเป็นคำสั่ง Script ที่สามารถแทรกหรือฝังตัวอยู่กับภาษา HTML ซึ่งเป็นภาษาของมาตรฐานเว็บปกติได้แต่จะต้องกำหนดให้ไฟล์นั้นมีชนิด (file extension) เป็น .php การแทรก PHP Script มีรูปแบบระบุส่วนที่เป็น PHP script ได้ 4 วิธีได้แก่

**<u>รูปแบบที่ 1</u>** การใช้ script tag ซึ่งเป็น tag มาตรฐานสำหรับการแทรกโปรแกรมในเว็บ รูปแบบคือ

```
<script language="PHP">
   php_instructions
</script>
```

เช่น

Listing 2.1-1a ตัวอย่างการใช้ <script> tag

```
<html>
  <head>
  <title><script language="PHP">echo "Hello world"</script></title>
 </head>
<body>
<script language="PHP">
   echo "<h1>First PHP Example</h1>\n";
   echo "Welcome to PHP programming . ";
   echo "Today : ",date("r"), " (server time) ";
   echo "<p align=center><a href=index.html>Home</a>\n<hr>";
   echo "<div align=\"right\">PHP. example by Sumet Angkasirikul";
</script>
</body>
</html>
```

**<u>รูปแบบที่ 2</u>** ซึ่งเป็นรูปแบบ PHP.tag คือการใช้ <?php … ?>

```
<?php
   php_instructions
?>
```

เช่น

Listing 2.1-1b ตัวอย่างการใช้ <?php ?> tag

```
<html>
  <head>
  <title><?php echo "Hello world" ?></title>
  </head>
<body>
<?php
   echo "<h1>First PHP Example</h1>\n";
   echo "Welcome to PHP programming . ";
   echo "Today : ",date("r"), " (server time) ";
   echo "<p align=center><a href=index.html>Home</a>\n<hr>";
   echo "<div align=\"right\">PHP. example by Sumet Angkasirikul";
?>
</body>
</html>
```

**<u>รูปแบบที่ 3</u>** การใช้รูปแบบ PHP.tag แบบสั้น คือ <? ?>

```
<?
  php_instructions
?>
```

เป็นรูปแบบสั้นของรูปแบบที่ 2 เช่น

Listing 2.1-1c ตัวอย่างการใช้ <? ?> tag

```
<html>
  <head>
  <title><? echo "Hello world" ?></title>
  </head>
<body>
<?
   echo "<h1>First PHP Example</h1>\n";
   echo "Welcome to PHP programming . ";
   echo "Today : ",date("r"), " (server time) ";
   echo "<p align=center><a href=index.html>Home</a>\n<hr>";
   echo "<div align=\"right\">PHP. example by Sumet Angkasirikul";
?>
</body>
</html>
```

**<u>รูปแบบที่ 4</u>** เรียกว่า expression tag

```
<?= php_expression ?>
```

เป็นคำสั่งที่แสดงผลลัพธ์จาก expression ส่งกลับมายัง browser นิยมใช้กับการส่งผลแทรกใน HTML code

Listing 2.1-1d ตัวอย่างการใช้ <?= ?> tag

```
<html>
<head>
<title><?= "Hello World" ?></title>
</head>

<body>
<h1><?= "First PHP Example" ?></h1>
Welcome to PHP programming .
Today : <?= date("r")?> (server time)
<p align=center><a href=<?= "index.html" ?>>Home</a><hr>
<div align="right">PHP. example by Sumet Angkasirikul</div>
</body>
</html>
```

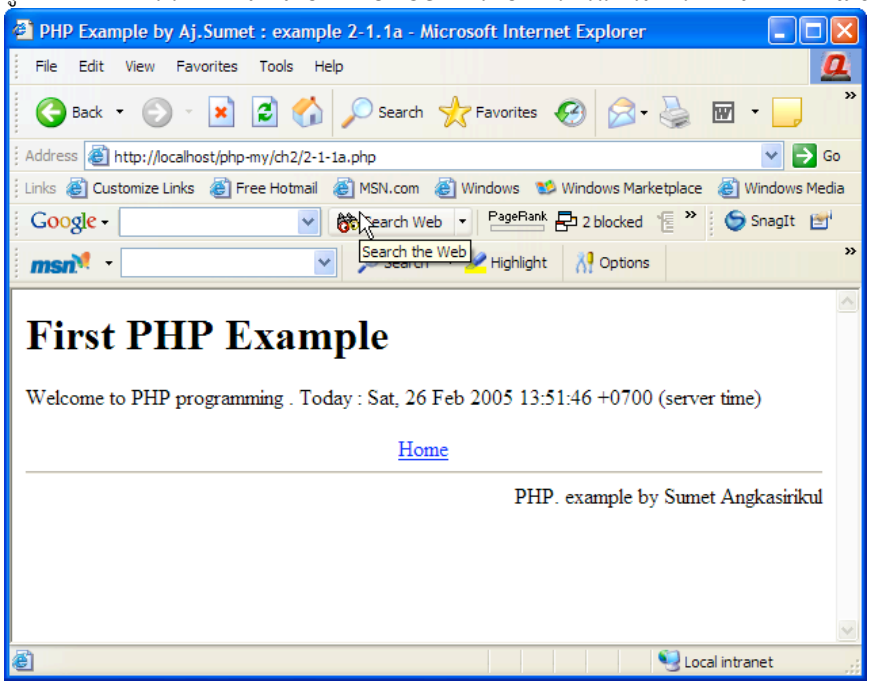
ผลลัพธ์จากการเรียกใช้งานตัวอย่างทั้งสี่ข้างต้น จะให้ผลลัพธ์ไปยัง browser ซึ่งทดสอบได้โดยการเรียกเพจนี้ให้ทำงาน แล้วใช้คำสั่ง View/Source (Internet Explorer) ดู code ที่ได้รับ จะปรากฏดัง Listing 2.1-1e ซึ่งจะไม่มีคำสั่ง PHP ปรากฏ และผลลัพธ์ที่แสดงบนหน้าจอ Browser จะปรากฏดังภาพที่ 2.1-1

Listing 2.1-1e ผลลัพธ์ output code ที่ได้จากตัวอย่าง 2.1-1a-d

```
<html>
<head>
<title>Hello World</title>
</head>

<body>
<h1>First PHP Example</h1>
Welcome to PHP programming .
Today : Sat, 26 Feb 2005 13:51:46 +0700 (server time)
<p align=center><a href=index.html>Home</a><hr>
<div align="right">PHP. example by Sumet Angkasirikul</div>
</body>
</html>
```

รูปที่ 2.1-1 ผลลัพธ์บนหน้าจอ Browser ที่ได้จากโปรแกรมตัวอย่าง 2.1-1 a-d



### 1 1) ตัวคั่นคำสั่งและการใช้ white space

คำสั่งแต่ละคำสั่งในภาษา PHP จะถือ white space (ได้แก่ เว้นวรรค Tab การขึ้นบรรทัดใหม่) เป็นเพียงตัวคั่นระหว่างแต่ละส่วนประกอบ (token) ของคำสั่งเท่านั้น หมายความว่า เราสามารถเขียนคำสั่งแต่ละคำสั่งแยกเป็นหลายบรรทัดได้ และในทางกลับกันในโปรแกรมแต่ละบรรทัดก็อาจจะมีคำสั่งอยู่มากกว่าหนึ่งคำสั่งได้เช่นเดียวกัน โดยมีตัวคั่นระหว่างคำสั่ง (instruction separator) คือ เครื่องหมาย semi-colon ; ตัวอย่างเช่น

Listing 2.1-2 ตัวอย่างการใช้ white space

```
<?
$userName = "Arnold Palmer" ;  $userTitle = "The Premier";
$printedTitle =   $userName . " : " .
                  $userTitle ;
echo $printedTitle ?>
```

จากตัวอย่างข้างต้น หลังจากเปิดเครื่องหมาย php tag <? แล้ว คำสั่ง $userName = "Arnold Palmer" ; และคำสั่ง $userTitle = "The Premier"; เป็น 2 คำสั่งที่เขียนอยู่ในบรรทัดเดียวกัน ส่วนคำสั่งในบรรทัดถัดไป ได้แก่ $printedTitle = $userName . " : " . และ $userTitle ; ที่อยู่ในบรรทัดถัดไปเป็นคำสั่งเดียวกันที่แยกออกไปอยู่ใน 2 บรรทัด ส่วนคำสั่งในบรรทัดสุดท้ายได้แก่ echo $printedTitle เป็นคำสั่งสุดท้ายของส่วนนี้ก่อนจะถูกปิดส่วน script (script fragment) อาจไม่มีเครื่องหมาย ; ปิดท้ายก็ได้ เนื่องจากเครื่องหมายปิด tag PHP ?> ก็ถือเป็นตัวปิดท้ายคำสั่งด้วยเช่นกัน (แต่หากจะใส่ ; เช่นเดียวกับคำสั่งในส่วนอื่นๆ ก็ไม่ผิดเช่นกัน)

### 1 2) การแสดงผลเบื้องต้น

สำหรับโปรแกรมโดยทั่วไปไม่ว่าจะเป็นโปรแกรมที่เขียนด้วยภาษาใด คำสั่งในการแสดงผล (output statement) มาตรฐานของภาษานั้นๆ มักจะส่งผลออกแสดงทางอุปกรณ์แสดงผลมาตรฐาน standard output device ซึ่งมักจะเป็นจอภาพ เช่น จอภาพของเครื่องพีซีหรือเป็นจอเทอร์มินัลของเครื่องเมนเฟรม แต่ในกรณีการเขียนโปรแกรมเว็บที่ทำงานฝั่งแม่ข่าย server-side web program การแสดงผลมาตรฐานจะไม่ได้แสดงออกทางจอภาพของเครื่องแม่ข่าย แต่จะส่งผลกลับไปหาผู้ใช้ที่อยู่บนเครื่องลูกข่ายโดยแสดงผลลัพธ์กลับไปยังเว็บบราวเซอร์ของลูกข่าย

ในหัวข้อนี้จะกล่าวถึงการแสดงผลเพียงเบื้องต้นโดยการใช้คำสั่ง echo และการใช้ expression tag <?= expr ?> ที่ได้กล่าวถึงมาแล้ว ก่อนที่จะกล่าวถึงวิธีการแสดงผลกลับไปยังเว็บบราวเซอร์อย่างละเอียดอีกครั้งในบทที่ 4

คำสั่ง echo เป็นคำสั่งพิเศษที่เรียกว่า language construct ใช้เพื่อส่งผลกลับไปยังเว็บบราวเซอร์ที่ผู้ใช้เรียกเพจนี้มา ลักษณะการใช้คำสั่งนี้จะต่างจากการใช้ฟังก์ชัน (หัวข้อ 2.4.3) โดยการระบุข้อมูลหรือนิพจน์ expression ที่จะส่งผลลัพธ์ ไม่จำเป็นต้องเขียนอยู่ภายในวงเล็บ และนิพจน์ที่จะแสดงผลจะเป็นนิพจน์ที่มีชนิดข้อมูล string หรือเป็นนิพจน์ข้อความ (หากเป็นนิพจน์ชนิดอื่นๆ ภาษา php จะแปลงให้กลายเป็นข้อความ)

รูปแบบทั่วไปของคำสั่ง echo ได้แก่

**echo** string *exp1*, string *exp2*, ...

ตัวอย่างของการใช้คำสั่ง echo ตัวอย่างที่ 2.1-3

Listing 2.1-3 ตัวอย่างการใช้คำสั่ง echo

```
<Html>
<head>
   <title>Example 2.1.1a Echo</title>
</head>
<body>
<?
$luckyNo = 9;
echo "Hello world.";
echo "This is echo example.\n";
echo "Lucky no. is ";
echo $luckyNo;
echo "<br>";
echo "Time now : " , date("r") , ".<br>\n" ;
echo "See you, Bye."
?>
</body>
```

ผลลัพธ์ที่แสดงบนเว็บบราวเซอร์จะได้เป็น

```
Hello world.This is echo example. Lucky no. is 9
Time now : Mon, 8 Nov 2004 23:28:23 +0700.
See you, Bye.
```

และหากใช้คำสั่ง view source ของ browser ดูจะพบว่า source code ที่เป็นภาษา html จะได้ดังนี้

```
<Html>
<head>
  <title>Example 2.1.1a Echo</title>
</head>
<body>
Hello world.This is echo example.
Lucky no. is 9<br>Time now : Mon,  8 Nov 2004 23:32:04 +0700.<br>
See you, Bye.</body>
```

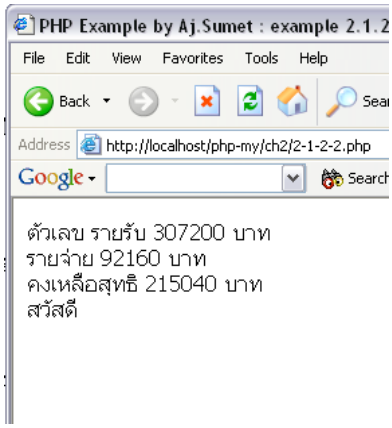
ตัวอย่างที่ 2.1-4 การแสดงผลโดยนิพจน์ข้อความและตัวเลข

Listing 2.1-4 ตัวอย่างการแสดงผลจากนิพจน์ตัวเลขและข้อความ

```
<?
$รายรับ = 25600 * 12;  // ดูเรื่องตัวแปรในหัวข้อ 2.1.3
$รายจ่าย = $รายรับ * 0.30; // รายจ่ายเป็น 30% ของรายรับ
echo "รายรับ $รายรับ บาท<br>\n";
echo "รายจ่าย $รายจ่าย บาท<br>\n";
echo "คงเหลือสุทธิ " , ($รายรับ - $รายจ่าย) , " บาท<br>\n";
echo "สวัสดี";
?>
```

ผลลัพธ์จากการทำงานแสดงบนเว็บบราวเซอร์แสดงดังรูปที่ 2.1-2

รูปที่ 2.1-2 แสดงผลลัพธ์จากการทำงานของตัวอย่างใน Listing 2.1-4




(ก) ผลที่ปรากฏบนหน้าจอ browser

(ข) ผลที่เป็น HTML code (เมื่อใช้คำสั่ง view source ที่ browser)

```
<Html>
<head>
  <meta http-equiv="Content-Type" content="text/html;
charset=tis-620">
  <title>PHP Example by Aj.Sumet : example 2.1.2-2</title>
</head>
<body>
ตัวเลข
รายรับ 307200 บาท<br>
รายจ่าย 92160 บาท<br>
คงเหลือสุทธิ 215040 บาท<br>
สวัสดี</body>
</html>
```

## *2* หมายเหตุ *comment*

การเขียนหมายเหตุในภาษา PHP นำรูปแบบมาตรฐานมาจากภาษา C และ Shell script ของระบบปฏิบัติการ UNIX มี 3 รูปแบบ ได้แก่

- หมายเหตุหลายบรรทัดแบบภาษา C /* comment */ สามารถเขียนข้อความ comment ต่อเนื่องกันหลายบรรทัดโดยใช้ตัวเปิด /* และตัวปิด */
- หมายเหตุบรรทัดเดียวแบบภาษา C, // comment เขียน comment หลังเครื่องหมาย // จนกระทั่งขึ้นบรรทัดใหม่ก็จะปิด comment โดยไม่ต้องมีเครื่องหมายปิด ซึ่งอาจจะเปิดเครื่องหมาย comment ต่อท้ายคำสั่งบางคำสั่งก็ได้
- หมายเหตุแบบ UNIX shell script, # comment เป็น comment แบบบรรทัดเดียว แต่ใช้เครื่องหมาย # แทน //

Comment หรือการเขียนหมายเหตุ เป็นข้อความที่ใช้เขียนอธิบายหรือเพื่อเตือนความจำให้ผู้พัฒนาโปรแกรมเองอ่าน โดยไม่มีผลต่อการทำงานของโปรแกรม ตัวอย่างการใช้หมายเหตุแสดงดังตัวอย่างต่อไปนี้

Listing 2.2-1 ตัวอย่าง การใช้ comment ในโปรแกรม php

```
1 <? /* Example 2.1.2 a : Comment example.
2 By  : Sumet Angkasirikul
3 Date : 7/11/2004
4 */
5 <? # this is 1st php code fragment
6  $titleMsg="Example 2.1.2 a : comment";
7  $hello = "Hello world";
8 ?>
9 <HTML>
10 <head>
11   <title><? echo $titleMsg; // send out title message from var.?>
12   </title>
13 </head>
14 <body>
15   <h1><?= $hello; // Just send out hello world to browser ?></h1>
16 </body>
17 </html>
```

จากตัวอย่างในส่วนของหมายเหตุในบรรทัดที่ 1-4 และ 5 จะไม่เกิดการทำงาน เช่นเดียวกับ PHP code ส่วนที่อยู่หลังเครื่องหมาย // ในบรรทัดที่ 11 และ 15 เว้นแต่เมื่อมีเครื่องหมายปิด tag PHP เช่นในบรรทัดที่ 15 เมื่อปิด ?> แล้วจะกลับสู่ HTML mode ซึ่ง </h1> ยังคงถูกส่งออกไปยัง browser

# 3 ชื่อ-ข้อมูล และตัวแปร

การใช้ตัวแปร (variable) เป็นหลักสำคัญประการหนึ่งในการเขียนโปรแกรมคอมพิวเตอร์ ซึ่งตัวแปรเป็นการสร้างพื้นที่หน่วยความจำที่ใช้เก็บข้อมูลในระหว่างการทำงานของโปรแกรม เราสามารถใช้ตัวแปรเพื่อเก็บค่าข้อมูลอาจเป็นข้อมูลที่มาจากการกำหนดค่าคงที่ อาจเป็นข้อมูลที่ได้จากการคำนวณหรือประมวลผล และสามารถนำตัวแปรนั้นไปเป็นข้อมูลสำหรับการประมวลผลอื่นๆ และส่งผลออกมาแสดงยังจอภาพ หรือส่งข้อมูลในตัวแปรนั้นเข้าไปเก็บในไฟล์หรือฐานข้อมูลได้

ในการใช้ตัวแปรจะต้องกำหนดชื่อให้กับตัวแปรแต่ละตัวเพื่อใช้อ้างถึงในการเรียกใช้งานตัวแปรนั้นๆ ข้อมูลที่เก็บอยู่ในตัวแปรจะมีการระบุชนิดของข้อมูลที่มีได้แตกต่างกัน ภาษาโปรแกรมแต่ละภาษาจะมีความแตกต่างในการกำหนดชนิดข้อมูลต่างๆ แต่ก็มีชนิดข้อมูลที่เป็นหลัก ได้แก่ ข้อมูลตัวเลข แบ่งเป็นตัวเลขที่มีเฉพาะจำนวนเต็ม (integer) และตัวเลขที่มีจุดทศนิยม (floating point) ข้อมูลอักขระ (character) และข้อความ (string) ข้อมูลตรรกที่เป็นจริงหรือเท็จ ฯลฯ และนอกจากนั้นยังมีข้อมูลที่เป็นโครงสร้างซับซ้อนได้แก่ Array, Object, Resoures

หัวข้อต่อไปนี้จะอธิบายถึงเรื่องที่เกี่ยวข้องกับตัวแปรและข้อมูลได้แก่ การตั้งชื่อ การตั้งตัวแปร ชนิดข้อมูลที่จะเก็บในตัวแปร

### 3 1) ชื่อ identifier

การกำหนดชื่อสำหรับตัวแปรหรือฟังก์ชั่น ในแต่ละภาษาจะมีวิธีกำหนดชื่อที่ใช้ตั้งตัวแปรหรือฟังก์ชั่นและคำสั่งต่างๆ สำหรับภาษา PHP มีกฎเกณฑ์การกำหนดชื่อดังนี้

- รูปแบบทั่วไปของการตั้งชื่อได้แก่ [a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*
- ขึ้นต้นด้วยตัวอักษรภาษาอังกฤษตัวใหญ่หรือตัวเล็ก หรือขีดล่าง (underscore _) หรืออักขระตั้งแต่หมายเลข 127-255 (x7F – xFF) หมายความว่าสามารถใช้ชื่อตัวแปรหรือฟังก์ชั่นเป็นรหัสภาษาไทยได้ (ภาษา PHP ยังใช้รหัสแทนอักขระต่างๆ แบบ 8 บิตเป็น ASCII หรือ รหัสประจำท้องถิ่นที่ไม่ใช่ Unicode)
- ตัวต่อไปอาจเป็นตัวอักษร ตัวเลข ขีดล่าง และอักขระหมายเลข 127-255
- ไม่สามารถขั้นด้วยเว้นวรรคได้
- ตัวอักษรภาษาอังกฤษตัวใหญ่หรือตัวเล็กถือว่าเป็นคนละตัวกัน (Case-sensitive) ดังนั้นชื่อ userName และชื่อ UserName หรือ USERNAME จะถือว่าเป็นคนละชื่อกัน
- ไม่มีข้อจำกัดด้านความยาวของชื่อ
- ต้องไม่ซ้ำกับ Reserved word ของภาษา PHP

ตัวอย่างชื่อที่ถูกต้อง

- Score , userAccount , address01 , วันที่ป้อน , last_modified_date , validAccount

ตัวอย่างชื่อที่ไม่ถูกต้อง

- 2004version (ไม่สามารถใช้ตัวเลขนำหน้า)
- user name (ไม่สามารถใช้เว้นวรรคคั่นในชื่อ)
- pass-code (ไม่สามารถใช้ – ได้ หากต้องการคั่นระหว่างคำให้ใช้ _ )
- 1024-200 (ไม่สามารถใช้ตัวเลขนำหน้าและมี – คั่นในชื่อ)
- <secret name> (ไม่สามารถใช้อักขระอื่นๆ เช่น < หรือ > ในชื่อได้)

### 3 2) ชื่อตัวแปร

ในภาษา PHP การใช้ชื่อตัวแปรจะนำหน้าชื่อด้วยเครื่องหมาย dollar sign $ ตามด้วยชื่อ (identifier) ตามหลักการตั้งชื่อที่ถูกต้องที่ได้อธิบายในหัวข้อ 2.3.1

ตัวอย่างชื่อตัวแปรได้แก่

- $userName
- $_lastLogin
- $รหัสผ่าน
- $nameMD5

ข้อแนะนำและธรรมเนียมสำหรับการตั้งชื่อโดยทั่วไป มีข้อแนะนำดังนี้

- ควรตั้งชื่อให้สื่อกับความหมายของข้อมูลที่จะจัดเก็บ เช่น $titleName เพื่อเก็บข้อความที่เป็น title , $userID เพื่อเก็บชื่อประจำตัวของผู้ใช้ , $หมายเลข_ภงด01 เป็นต้น
- ไม่ควรตั้งชื่อยาวจนเกินไป (ปกติไม่ควรยาวกว่า 32 อักขระ) แม้ว่าภาษา PHP จะไม่กำหนดความยาวสุดของชื่อแต่การตั้งชื่อยาวเกินไปจะทำให้ source code ใหญ่ขึ้นและมีโอกาสเสี่ยงต่อการสะกดผิดหรือสะกดไม่เหมือนกันได้
- เนื่องจากไม่สามารถใช้เครื่องหมายเว้นวรรคในชื่อได้ ดังนั้นหากจะตั้งชื่อหลายคำในภาษาอังกฤษ มีวิธีที่นิยมกัน ได้แก่
  - การใช้ขีดล่างแทนเว้นวรรค เช่น $total_tax , $doc_password เป็นต้น
  - ใช้อักษรตัวใหญ่ขึ้นต้นคำ เช่น $totalTax , $docPassword เป็นต้น
- ปัจจุบันภาษา php ยังไม่สนับสนุนรหัสอักขระแบบ Unicode ในการตั้งชื่อจะต้องเป็นรหัส 8 บิต ดังนั้นการตั้งชื่อตัวแปรภาษาไทยจะต้องระวังการบันทึก source code ต้องเป็นรหัส ASCII หรือใช้ภาษาไทย 8 บิต (เช่น TIS-620)
- ธรรมเนียมทั่วไปของภาษาเขียนโปรแกรมแบบ Object (Object-Oriented Programming) มักจะกำหนดการตั้งชื่อ Object ให้อักษรตัวแรกเป็นตัวเล็ก เช่น $currentMember หากเป็นการกำหนดชื่อ class จะให้อักษรตัวแรกของชื่อเป็นอักษรตัวใหญ่ เช่น $Member เป็นต้น

### 3 3) <u>ชนิดข้อมูล</u>

การใช้ข้อมูลไม่ว่าจะเป็นข้อมูลที่เก็บลงในตัวแปรหรือข้อมูลที่ได้จากการคำนวณหรือประมวลผลในนิพจน์ expression จะมีชนิดของข้อมูลที่แตกต่างกันได้หลายชนิด เช่น เป็นข้อมูลตัวเลขที่นำไปใช้คำนวณทางคณิตศาสตร์ได้ เป็นข้อมูลข้อความ ฯลฯ เป็นต้น ในภาษา PHP มีชนิดข้อมูล 8 ชนิด เป็นข้อมูลแบบพื้นฐาน 4 ชนิด และเป็นชนิดข้อมูลที่เป็นโครงสร้างซับซ้อน 4 ชนิด ได้แก่

- Boolean
- Integer
- Float/double
- String
- Array
- Object
- Resource
- Null

ชนิดข้อมูล Boolean เป็นข้อมูลแบบตรรกะที่มีข้อมูลเป็นจริงหรือเท็จ อาจใช้ประโยชน์เพื่อเก็บหรือแสดงสถานะต่างๆ ที่เป็นแบบเปิดหรือปิด, ใช่หรือไม่ใช่ ในทำนองเช่นนี้ สามารถนำข้อมูลชนิด Boolean ไปใช้ในคำสั่งที่เกี่ยวข้องกับการตัดสินใจเลือกการทำงาน หรือคำสั่งทำงานวนรอบเพื่อใช้กำหนดเป็นเงื่อนไข

ชนิดข้อมูล Integer คือข้อมูลที่เป็นตัวเลขจำนวนเต็มไม่มีจุดทศนิยม ได้ทั้งค่าบวกและค่าลบ ใช้เก็บข้อมูลที่เป็นเลขจำนวนเต็ม เช่น วันที่ เดือนที่ เลขที่ พศ./คศ. จำนวนชิ้นของสิ่งของ จำนวนบุคคล เป็นต้น สามารถนำข้อมูลตัวเลขจำนวนเต็มไปคำนวณโดยใช้เครื่องหมายทางคณิตศาสตร์ได้ เช่น $A = $B * 7 * (5+16)

ชนิดข้อมูล Float/double คือข้อมูลที่เป็นตัวเลขมีจุดทศนิยมทั้งค่าบวกและค่าลบ สามารถใช้แทนค่าตัวเลขที่มีจำนวนมากๆ (หากเป็นข้อมูลตัวเลขจำนวนเต็มจะสามารถเก็บข้อมูลได้สูงสุดประมาณ +/- 2 พันล้าน) หรือตัวเลขเศษส่วนที่มีความละเอียดของหลักเลข ตัวอย่างการใช้ข้อมูล Float เช่น เงิน ที่มีเศษสตางค์ น้ำหนัก ระยะทาง ฯลฯ ข้อมูลตัวเลขมีจุดทศนิยม สามารถนำไปคำนวณทางคณิตศาสตร์ได้

ชนิดข้อมูล String คือข้อมูลที่เป็นข้อความที่มีทั้งข้อความภาษาอังกฤษ เครื่องหมายพิเศษ หรือข้อความในภาษาไทย ซึ่งภาษา PHP ยังเก็บข้อมูลแบบ 8 บิต โดยใช้ ASCII code ที่มีส่วนขยายคือมีรหัสสำหรับอักขระพิเศษ หรือภาษาไทยด้วย

ชนิดข้อมูล Array เป็นการเก็บข้อมูลที่เป็นโครงสร้างแบบ ตารางเมตริกซ์ ภายใต้ชื่อตัวแปรชื่อเดียวสามารถเก็บค่าข้อมูลหลายข้อมูลได้ โดยการกำกับด้วยดัชนี (index) ซึ่ง Array ในภาษา PHP มีลักษณะเหมือนภาษา C คือเป็น Array ที่มี index เริ่มต้นที่หมายเลข 0 (Zero-base array ) และยังสามารถสร้าง index ที่ไม่ใช่ตัวเลข คือสามารถใช้ข้อความใดๆ เป็น index ก็ได้ เรียกว่า Associative Array

ชนิดข้อมูล Object คือการสร้างข้อมูลเชิงวัตถุที่สนับสนุนหลักการของการโปรแกรมเชิงวัตถุ information hiding (encapsulation)

ชนิดข้อมูล Resource ใช้กับการทำงานกับคำสั่งที่เกี่ยวข้อกับ hardware resources ต่าง เช่น ไฟล์ข้อมูล ฐานข้อมูล ข้อมูลที่สื่อสารในเครือข่าย ฯลฯ เป็นต้น

ชนิดข้อมูล Null เป็นข้อมูลพิเศษที่แสดงว่ายังไม่มีการกำหนดข้อมูลใดๆ ให้แก่ตัวแปรหรือไม่ได้รับข้อมูลใดๆ กลับคืนมา

### 3 4) <u>ค่าข้อมูลแต่ละชนิด (Literal)</u>

- ข้อมูลชนิด Boolean กำหนดค่าได้ 2 ค่าได้แก่ true หรือ false

- ข้อมูลชนิด Integer สามารถกำหนดค่าให้เป็นเลขจำนวนเต็มที่เขียนอยู่ในรูปเลขฐานสิบ (decimal) ฐานแปด (octal) และฐานสิบหก (hexadecimal)
  - เลขฐานสิบ ทั้งค่าบวกหรือค่าลบ เช่น 2547, -50000, 0 เป็นต้น การเขียนค่าตัวเลขในภาษาโปรแกรมจะต้องไม่มีเครื่องหมายคั่นระหว่างหลักเลขคือ , เช่น 2,000,254 เป็นตัวเลขที่ไม่ถูกต้อง
  - เลขฐานแปด เขียนนำหน้าตัวเลขด้วยเลข 0 ตามด้วยตัวเลขที่แต่ละหลักอยู่ระหว่าง 0-7 เช่น 0357, -0164 เป็นต้น
  - เลขฐานสิบหก เขียนนำหน้าด้วย 0x ตามด้วยเลขฐานสิบหก (0-9,A-F) เช่น 0xA408 , -0x4E7D เป็นต้น
- ข้อมูลชนิด Float/double สามารถเขียนในรูปแบบตัวเลขทศนิยมธรรมดาหรือลักษณะแบบวิทยาศาสตร์
  - ตัวเลขจุดทศนิยม เช่น 129.0958, 0.001479, -31254.50 การเขียนเลขจุดทศนิยมไม่สามารถใช้ comma คั่นหลักเลขได้เช่นเดียวกับเลขจำนวนเต็ม
  - เขียนเลขในรูปแบบวิทยาศาสตร์ (Scientific notation) ใช้กับกรณีที่ตัวเลขมีขนาดใหญ่มากๆ หรือน้อยกว่า 1 มากๆ โดยเขียนในรูปแบบ +/-9.9E+/-99 ซึ่ง E (exponent) จะหมายถึง $x10^n$ เช่น 36.1275E+6 หมายถึงค่า $36.1275x10^6$ หรือ 36127500
    8.4148E-3 หมายถึง $8.4148\ x10^{-3}$ หรือ 0.0084148 เป็นต้น
- String การเขียนค่าข้อมูลข้อความ สามารถเขียนข้อความในเครื่องหมายคำพูด 2 แบบ คือ single quote และ double quote
  - การเขียนข้อความใน single quote ' เช่น 'Enter your name : ' หากต้องการให้มีอักขระ single quote อยู่ในข้อความด้วยจะใช้ \' เช่น 'Arnold\'s hand phone' หมายถึงข้อความ Arnold's hand phone และหากต้องการใช้เครื่องหมาย \ ในข้อความก็ต้องพิมพ์ \ 2 ครั้ง เช่น 'users\\arnold\ \mybio.txt' หมายถึงข้อความ user\arnold\mybio.txt เป็นต้น
    <u>หมายเหตุ</u> เครื่องหมาย \ ที่ใช้นำหน้าอักขระพิเศษใน string เรียกว่า escape character
  - การเขียนข้อความใน double quote " เช่น "Enter password "
    การใช้ double quote มีความพิเศษกว่า single quote ได้แก่
    - มี escape character ที่เป็นคำสั่งพิเศษหรือเป็น character พิเศษมากขึ้น ได้แก่

      \n หมายถึงการขึ้นบรรทัดใหม่ (line feed)
      \r หมายถึงรหัส carriage return
      \t หมายถึงรหัส horizontal tab
      \\ หมายถึงอักขระ \
      \" หมายถึงอักขระ "
      \$ หมายถึงอักขระ $

\[0-7]{1,3}  หมายถึงอักขระตามรหัสที่ระบุเป็นเลขฐาน 8
\x[0-9A-Fa-f]{1,2} หมายถึงอักขระตามรหัสที่ระบุเป็นเลขฐาน 16

- สามารถแทรกข้อมูลจากตัวแปรลงในข้อความได้โดยใช้ชื่อตัวแปรอยู่ภายในข้อความ เช่น

```
$user1 = "Arnold";
$estYr = 1958;
$hello = "Member $user1, Welcome"; //result: Member Arnold, Welcome
$estMsg = "Our organization was established in $estYr ."
```

จากตัวอย่างตัวแปร $hello จะมีข้อความ Member Arnold, Welcome ซึ่งคำว่า Arnold มาจากค่าในตัวแปร $user1 และในตัวแปร $estMsg จะเก็บข้อความ Our organization was established in 1958 . เป็นต้น

<u>หมายเหตุ</u> หากเป็นตัวแปรที่ซับซ้อนเช่น Array หรือ Object หรือการแสดงข้อความจากตัวแปรติดกับข้อความปกติจะต้องครอบชื่อตัวแปรด้วยวงเล็บปีกกา { } ตัวอย่างเช่น

```
 "Item no. {$item[2]} : quantity {$order->quan} : status OK"
```

### 3 5) <u>ค่าสั่งสร้าง-กำหนดค่าตัวแปร</u>

ในภาษา PHP เป็นภาษา script ที่ไม่เข้มงวดเรื่องชนิดของตัวแปร และการสร้างตัวแปร (variable declaration) ต่างจากภาษาโปรแกรมหลายภาษาเช่น C, C++, Java ที่เข้มงวดเรื่องชนิดและการสร้างตัวแปร สำหรับภาษา PHP ตัวแปรที่อยู่ในโปรแกรมจะถูกสร้างขึ้นโดยอัตโนมัติเมื่อมีการกำหนดข้อมูลให้ตัวแปรเหล่านั้นเป็นครั้งแรกโดยไม่ต้องมีคำสั่งสร้างตัวแปร และจะถูกกำหนดชนิดของข้อมูลให้โดยอัตโนมัติตามข้อมูลที่เก็บลงในตัวแปร แต่หากยังไม่มีการกำหนดข้อมูลให้กับตัวแปรจะเก็บข้อมูลชนิดพิเศษที่เรียกว่า Null คือไม่มีข้อมูลอยู่ หรือตัวแปรนั้นยังไม่ถูกสร้างขึ้น

ชนิดของข้อมูลที่เก็บในตัวแปรนั้นสามารถแปรเปลี่ยนไปได้ตลอดเวลา ขึ้นกับข้อมูลหรือ expression ที่กำหนดเข้าไปให้ตัวแปร โดยไม่เกิดข้อผิดพลาดขึ้นในโปรแกรม เช่น

```
$A = 10;     // variable A is an integer
$B = 8;      // variable B is an integer
$A = $A . "-" . $B; // คำสั่งนี้ทำให้ A เป็น string variable
```

การกำหนดค่าข้อมูลให้แก่ตัวแปรสามารถใช้คำสั่ง variable assignment statement ซึ่งมีรูปแบบ $variable = expression; เช่น

```
$A = 100;  $B = $A * 7 /100;
```

ตัวอย่างการใช้ตัวแปร

รายการที่ 2.3-1 ตัวอย่างการใช้ตัวแปรที่ไม่ต้องประกาศ

```
<? /* Example 2.1.3 a : variable usage. */
$a = 3200;
$b = 7;
$c = $a * $b / 100;
?>
<html>
<head><title> Example 2.1.3 a : variable usage.</title></head>
<body>
   <p><u>Example 2.1.3</u>
   <p><?= "$a * $b % = $c" ?>
   <p><?= "The value of variable D is now $d" // $d is not declared?>
</body>
</html>
```

### 3 6) <u>ขอบเขตและอายุของตัวแปร (scope and lifetime)</u>

โดยปกติแล้วตัวแปรในภาษาต่างๆ จะมีขอบเขตการมองเห็นจากคำสั่งต่างในโปรแกรมแบ่งเป็น 2 แบบได้แก่ Global scope / Local scope

ตัวแปรที่สร้างขึ้นภายนอก function จะมี Scope เป็น global scope ที่สามารถใช้ตัวแปรนั้นได้ตลอดทั้งโปรแกรม เว้นแต่ภายใน function หากมีการอ้างถึงชื่อตัวแปรใดๆ ภายใน function

declaration จะถือว่าตัวแปรที่อยู่ภายในแต่ละ function เป็นตัวแปรเฉพาะของ function นั้นเรียกว่า Local scope แม้ว่าชื่อตัวแปรจะเหมือนกันกับตัวแปร global ที่มีอยู่ภายนอกก็ตาม

หากต้องการใช้ตัวแปร global ภายใน function declaration จะต้องใช้ keyword global กำหนดตัวแปรที่ต้องการนั้นภายใน function เช่น

รายการที่ 2.3-2 ตัวอย่างการประกาศตัวแปร Global

```
$a = 1; $b = 2;
function Sum()
{
    global $a, $b;
    $b = $a + $b;
}
Sum();
echo $b;
```

ตัวแปรที่อยู่ใน Local scope จะมีลักษณะเป็นตัวแปร dynamic คือตัวแปรนั้นจะถูกสร้างขึ้นเมื่อฟังก์ชันถูกเรียกใช้ และเมื่อออกจากฟังก์ชัน ตัวแปร local จะถูกยกเลิก หากเข้ามาในฟังก์ชันใหม่ตัวแปรนั้นก็จะถูกสร้างขึ้นใหม่ไม่สามารถเก็บข้อมูลค่าเดิมไว้ได้ หากต้องการให้ตัวแปร local ของฟังก์ชันตัวใดยังคงเก็บค่าตัวแปรนั้นไว้ไม่ให้ถูกยกเลิกจะต้องกำหนดให้ตัวแปรนั้นเป็นตัวแปรแบบ static ทำได้โดยการใช้ keyword static นำหน้าชื่อตัวแปรภายใน function declaration เช่น

รายการที่ 2.3-3 ตัวอย่างการประกาศตัวแปร Static

```
<?php
function Test()
{
    static $a = 0;
    echo $a;
    $a++;
}
?>
```

ข้อควรระวัง เนื่องจากภาษา PHP เป็นภาษาที่สามารถใช้ตัวแปรโดยไม่ต้องมีการประกาศ (Declare) ก่อน ซึ่งแม้ว่าจะทำให้เกิดความสะดวกในการเขียนโปรแกรม แต่ก็สามารถก่อให้เกิดความสับสนในการตรวจสอบเมื่อเกิดข้อผิดพลาดขึ้น โดยเฉพาะการใช้ตัวแปร หากต้องการใช้ตัวแปร global แต่ไม่มีการใช้คำสั่ง global ในฟังก์ชัน เมื่อมีการเรียกใช้ตัวแปรที่ชื่อซ้ำกับตัวแปร global จะเกิดตัวแปรที่มีชื่อเดียวกันแต่เป็นตัวแปรภายในฟังก์ชันที่ถือเป็นคนละตัวกับตัวแปรภายนอก เกิดขึ้นโดยอัตโนมัติ แต่จะมีค่าเริ่มต้นเป็น null จึงควรระวังการใช้ตัวแปร global ภายในฟังก์ชันต่างๆ ต้องไม่ลืมในการประกาศใช้ตัวแปร global ด้วย

การใช้ตัวแปร global จะถูกกล่าวถึงอีกครั้งในหัวข้อ 2.7.4 ภายใต้หัวข้อ User defined function

### 3 7) การแปลงระหว่างชนิดข้อมูล (type casting)

แม้ว่าภาษา php จะไม่เข้มงวดเรื่องชนิดข้อมูล และมีการแปลงชนิดข้อมูลให้โดยอัตโนมัติตามสภาพแวดล้อมในนิพจน์ แต่บางครั้งอาจมีความจำเป็นต้องบังคับชนิดข้อมูลภายในนิพจน์เพื่อไม่ให้เกิดความกำกวมของการดำเนินการภายในนิพจน์ การบังคับชนิดข้อมูลทำได้ 2 วิธีดังนี้

- ใช้ casting operator (*type*) *expression* เช่น $quan = (int) $mquan;
  ค่าที่ใช้กำหนด *type* ได้แก่ boolean หรือ bool, int หรือ integer, float / double หรือ real, string, array, object

- ฟังก์ชันมีรูปแบบคือ **settype** (*expression*, *type* ) เช่น
  $quan = settype($mquan,"integer"); ค่าที่ใช้กำหนด type สามารถใช้เช่นเดียวกับชื่อชนิดที่ใช้ใน casting operator

ตัวอย่างต่อไปนี้เป็นการหารข้อมูลจากตัวแปร $a ที่แปลงเป็นตัวเลขจำนวนเต็มด้วยข้อมูลจากตัวแปร $c คูณกับ $d ปัดเศษทิ้ง

```
$c = (int) $a / (int) ($c * $d);
```

### 3 8) การแปลงชนิดข้อมูลให้อัตโนมัติ type juggling

นอกจากภาษา PHP เป็นภาษาที่ไม่เข้มงวดเรื่องชนิดข้อมูลแล้ว มันยังจะพยายามแปลงข้อมูลใน expression ให้เหมาะกับการทำงานให้เองอีกด้วย ซึ่งการแปลงข้อมูลจากชนิดหนึ่งไปยังอีกชนิดหนึ่งโดยอัตโนมัติมีกฎเกณฑ์ดังนี้

- การแปลงให้เป็น Boolean

  จากข้อมูลชนิดอื่นใดจะแปลงเป็นค่า false ในกรณีดังนี้

  - integer ที่มีค่า 0
  - float ที่มีค่า 0.0
  - string ที่มีข้อคาม empty string "" และ string "0"
  - array ที่ไม่มี elements
  - object ที่ไม่มี member variables, NULL

  นอกจากนั้นจะแปลงเป็น true

- การแปลงเป็น Integer

  จากข้อมูลชนิดต่างๆ สามารถแปลงเป็น integer ได้ดังนี้

  - จาก Boolean : true=1 , false=0
  - จาก Float ให้ตัวเลขจำนวนเต็มปัดเศษทิ้ง เช่น 3.9 → 3
  - จาก String หากข้อความมีตัวเลขขึ้นต้นจะแปลงตัวเลขนั้นจนหมดชุดที่เป็นตัวเลข หากอักขระที่ขึ้นต้นไม่ใช่ตัวเลขจะให้ 0 หากเป็น empty string จะให้ 0 ตัวอย่าง เช่น
    - "550.99US$" → 550
    - "91/35 Central Street" → 91
    - "-841,92:Somchai " → -841
    - "$492, $309" → 0
    - "R2D2, C3P0" → 0

- การแปลงเป็น Floating point

  จากข้อมูลชนิดต่างๆ สามารถแปลงเป็น floating point ได้ดังนี้

  - จาก Boolean : true=1.0, false=0.0
  - จาก integer จะได้ตัวเลขที่มาจากจำนวนเต็มมีเศษเป็น .0 เช่น 398 → 398.0
  - จาก string หากข้อความขึ้นต้นด้วยข้อความที่เป็น floating point number literal จะแปลงให้เป็น floating point ตาม literal นั้น (รวมถึงรูปแบบ scientific notation ด้วย) หากในตอนต้นไม่เป็น floating literal หรือเป็น empty string จะได้ 0.0 ตัวอย่างเช่น
    - "550.99US$" → 550.99
    - "91/35 Central Street" → 91.0
    - "-841e2:Somchai " → -841e2 หรือ -84100
    - "$492, $309" → 0.0
    - "R2D2, C3P0" → 0.0

- การแปลงเป็น Array จากข้อมูลชนิดอื่นๆ

  - จาก Boolean, integer, float, string, resource ให้เป็น array จะได้ array 1 element ที่มี index เป็น [0] เช่น $a = (array) $b; หาก $b มีข้อมูล 300 จะได้ $a ( [0]=>300 )
  - จาก object จะได้ associative array ที่มี index เป็น string ตามชื่อ variable ของ object เช่น $a = (array) $b; หาก $b เป็น object ที่มี member variable 3 ตัวได้แก่ $m1=50, $m2=100 และ $m3=150 จะได้ $a ('m1'=>50,'m2'=>100,'m3'=>150) ดูเรื่อง array เพิ่มเติมในหัวข้อ 2.8

# 4 EXPRESSION

Expression หรือ นิพจน์ เป็นการเขียนการคำนวณหรือประมวลผลให้ผลลัพธ์เป็นข้อมูลประเภทใดๆ สามารถนำผลไปกำหนดค่าให้แก่ตัวแปร ส่งผลไปเป็น argument ของฟังก์ชั่น ใช้เป็นเงื่อนไขต่างๆ หรือในคำสั่งอื่นใดที่สามารถกำหนดข้อมูลในลักษณะ expression ได้

นิพจน์หรือ expression ประกอบด้วยส่วนประกอบ ได้แก่ operand คือข้อมูลที่นำมาดำเนินการและ operator คือเครื่องหมายของการดำเนินการ

รูปแบบของ expression ได้แก่

- operand มีเฉพาะการกำหนดข้อมูล เช่น $X = 30; ค่า 30 เป็น operand ชนิด literal (ชนิดของ operand ในหัวข้อถัดไป) หรือ if ( isInt ($amount) ) คำสั่ง if มี expression ที่เป็นฟังก์ชั่น isInt ( ) เป็นฟังก์ชั่นเดียว และใน isInt( ) ก็มีการส่ง argument เป็น expression ที่มีเพียง operand เดียวคือตัวแปร $amount เป็นต้น
- operand operator operand (ในกรณี operator ต้องการ 2 operand หรือ binary operator) เช่น $A * $B ตัวแปร $A และ $B เป็น operand ส่วน * เป็น operator
- operand operator หรือ operator operand กรณี operator ต้องการ 1 operand เรียกว่า unary operator เช่น - $credit เป็น expression ที่มี – เป็น operator สร้างค่าที่เป็นลบของค่าจากตัวแปร $credit (ไม่ใช่การลบค่าหนึ่งจากอีกค่าหนึ่ง) มี $credit เป็น operand หรือ $count++ หรือ --$i เป็นต้น

## 4 1) Operand

Operand คือข้อมูลที่จะนำมากระทำการคำนวณหรือประมวลผล ประกอบด้วย

- ตัวแปร (variable) ตามรายละเอียดที่กล่าวมาแล้วในหัวข้อ 2.3 เช่น $totalAmount
- ค่าข้อมูล (literal) ตามรายละเอียดที่กล่าวมาแล้วในหัวข้อ 2.3.4 เช่น 49.50, "Bht."
- ชื่อแทนค่าคงที่ constant ตามรายละเอียดที่จะกล่าวถึงในหัวข้อ 2.4.5 เช่น VATrate
- ฟังก์ชั่น function คือการดำเนินการคำนวณหรือประมวลผลที่มีความซับซ้อนกระทำกับข้อมูลที่ระบุ (หัวข้อ 2.4.3)
- Expression แต่ละ operand อาจเป็น expression ย่อยได้ เช่น $amount – $saleAmount * 0.80 ตัวอย่าง expression นี้ operator – (ลบ) มี operand คือ ตัวแปร $amount และ expression $saleAmount * 0.80 เป็นต้น

## 4 2) Operator

Operator เครื่องหมายคำนวณ/ดำเนินการประมวลผล แบ่งออกเป็นกลุ่มต่างๆ มีรายละเอียดดังนี้

- ( ) การจัดกลุ่ม grouping การเขียนนิพจน์คำนวณหรือประมวลผลบางครั้งจะต้องมีการจัดลำดับการคำนวณให้ถูกต้อง ซึ่งอาจต่างจากลำดับการคำนวณตามลำดับความสำคัญของเครื่องหมาย (หัวข้อ 2.4.3) จะต้องใช้ ( ) ครอบ expression ย่อยบางกลุ่ม เพื่อแสดงว่าต้องการให้มีการคำนวณในส่วนที่อยู่ภายในวงเล็บให้เสร็จก่อน เช่น $c = $a * ($b + 3 * $x) เป็น expression ที่จะทำการคำนวณ 3*$x ก่อน จากนั้นจะ + กับตัวแปร $b แล้วจึง * กับ $a หากไม่มีการใช้วงเล็บจัดกลุ่ม $c=$a * $b + 3 * $x การคำนวณจะดำเนินการตามความสำคัญของเครื่องหมาย คือ นำผลที่ได้จาก $a * $b กับ 3 * $x มาบวกกันซึ่งจะให้ผลลัพธ์ที่แตกต่างกัน
- เครื่องหมายคำนวณทางคณิตศาสตร์ Mathematic operators ได้แก่
    - + การบวกเลข ตัวอย่างเช่น 1000+500 ได้ผลเป็น 1500, 800+1438.75 ได้ผลเป็น 2238.75 หรือ $a + 500 จะได้ผลจากข้อมูลที่อยู่ในตัวแปร $a บวกด้วยตัวเลขจำนวนเต็ม 500 เป็นต้น
    - - การลบเลข เช่น $a - $b จะได้ผลเป็นการนำข้อมูลจากตัวแปร $a ลบด้วยข้อมูลจากตัวแปร $b ให้ผลลัพธ์เป็นตัวเลขที่ลบได้
    - * การคูณเลข เช่น $prc * 1.07 จะได้ผลเป็นข้อมูลตัวเลขที่เป็นข้อมูลจากตัวแปร $prc คูณด้วยข้อมูลตัวเลข 1.07 เป็นต้น
    - / การหารเลข เช่น $sum / $num

- % การหาเศษของการหารจากเลขจำนวนเต็ม (modulus) เช่น 15 % 4 เหลือเศษ 3 หรือ 36 % 3 เหลือเศษ 0 เป็นต้น

หมายเหตุ หากนำข้อมูล string มาใช้เป็น operand สำหรับเครื่องหมายคำนวณทางคณิตศาสตร์ PHP จะทำการแปลงให้ string กลายเป็นตัวเลขแล้วจึงดำเนินการตามเครื่องหมายคำนวณ เช่น "-84US$" * "0.07VAT." จะได้ผลเช่นเดียวกับ -84 * 0.07 หรือ "$299.50" + 50 จะได้ 50 เนื่องจาก "$299.50" อักขระตัวแรกเป็น $ ไม่ใช่ตัวเลข PHP จะหยุดการแปลงและได้ผลเป็น 0 เมื่อนำไปบวกกับ 50 จะได้ผลลัพธ์ 50 เป็นต้น

- เครื่องหมายกำหนดค่าให้ตัวแปร assignment operators ได้แก่
  - = กำหนดข้อมูลให้ตัวแปรด้านซ้ายเท่ากับนิพจน์ด้านขวา $var = expr
    ตัวอย่างเช่น $a = $b * 7 / $c;
  - += เพิ่ม(บวก)ค่าในตัวแปรด้านซ้ายขึ้นด้วยจำนวนในนิพจน์ด้านขวา $var += expr
    เทียบได้กับ $var = $var + expr
    ตัวอย่างเช่น $a += 5; $b += $c เป็นต้น
  - -= ลด(ลบ)ค่าในตัวแปรด้านซ้ายลงด้วยจำนวนในนิพจน์ด้านขวา $var -= expr
    เทียบได้กับ $var = $var - expr
    ตัวอย่างเช่น $a -= 5; $b -= $c เป็นต้น
  - *= คูณค่าในตัวแปรด้านซ้ายด้วยจำนวนในนิพจน์ด้านขวา $var *= expr
    เทียบได้กับ $var = $var * expr
    ตัวอย่างเช่น $a *= 1.2; $b *= $c เป็นต้น
  - /= หารค่าในตัวแปรด้านซ้ายด้วยจำนวนในนิพจน์ด้านขวา $var /= expr
    เทียบได้กับ $var = $var / expr
    ตัวอย่างเช่น $a /= 1.2; $b /= $c เป็นต้น
  - %= นำเศษจากการหารจำนวนเต็มค่าในตัวแปรด้านซ้ายด้วยจำนวนในนิพจน์ด้านขวาเก็บลงในตัวแปรด้านซ้าย $var %= expr
    เทียบได้กับ $var = $var % expr
    ตัวอย่างเช่น $a %= 1.2; เป็นต้น
  - .= ต่อข้อความในตัวแปร string ด้านซ้ายด้วยนิพจน์ด้านขวา $var .= expr
    เทียบได้กับ $var = $var . expr
    ตัวอย่างเช่น $hello .= $userName;
- เครื่องหมายดำเนินการทางลอจิกแบบหลักเลขฐานสอง bitwise operators เป็นเครื่องหมายที่จะดำเนินการทางตรรกศาสตร์กับข้อมูลแต่ละบิตของนิพจน์ด้านซ้ายและขวา ประกอบด้วย 6 operators ได้แก่
  - & การ AND แบบบิตต่อบิต เช่น 3754 & 7129 หากคิดเทียบเป็นเลขฐานสองจะได้ 0111010101010 & 1101111011001 ได้ผลลัพธ์ 0101010001000
  - | การ OR แบบบิตต่อบิต เช่น 3754 | 7129 หากคิดเทียบเป็นเลขฐานสองจะได้ 0111010101010 & 1101111011001 ได้ผลลัพธ์ 1111111111011
  - ^ การ EXCLUSICE-OR แบบบิตต่อบิต เช่น 3754 ^ 7129 หากคิดเทียบเป็นเลขฐานสองจะได้ 0111010101010 & 1101111011001 ได้ผลลัพธ์ 1010101110011
  - ~ การ not (inverse) นิพจน์ทางขวา (เป็น operator ที่ต้องการ operand เดียว) เช่น ~1010101110011 ได้รับผลลัพธ์ 0101010001100
  - << การเลื่อนบิตไปทางซ้าย (shift left) ทำการเลื่อนบิตเลขฐานสองของนิพจน์ด้านซ้ายมือของเครื่องหมายไปทางซ้ายเป็นจำนวนครั้งเท่ากับจำนวนนิพจน์ทางขวามือและเลื่อน 0 เข้าทางบิตขวาสุด (การเลื่อนบิตเลขฐานสองไปทางซ้าย 1 ครั้งเท่ากับเป็นการคูณเลขฐานสิบด้วย 2
    ตัวอย่างเช่น นิพจน์ 18 << 3 หรือเป็นเลขฐานสอง 10010 << 3 ได้ผลลัพธ์ 10010000 หรือเป็นเลขฐานสิบ 144 ( คือ 18*2*2*2) เป็นต้น
  - >> การเลื่อนบิตไปทางขวา (shift right) ทำการเลื่อนบิตเลขฐานสองของนิพจน์ด้านซ้ายมือของเครื่องหมายไปทางขวาเป็นจำนวนครั้งเท่ากับจำนวนนิพจน์ทางขวามือและเลื่อน 0 เข้าทางบิตซ้ายสุด (การเลื่อนบิตเลขฐานสองไปทางขวา 1

ครั้งเท่ากับเป็นการหารเลขฐานสิบด้วย 2
ตัวอย่างเช่น นิพจน์ 1800 >> 3 หรือเป็นเลขฐานสอง 11100001000 >> 3 ได้ผลลัพธ์ 11100001 หรือเป็นเลขฐานสิบ 225 ( คือ 1800/2/2/2) เป็นต้น

- เครื่องหมายเปรียบเทียบข้อมูล comparision operators ทำการเปรียบเทียบข้อมูลระหว่างนิพจน์ทางซ้ายกับทางขวา ให้ผลลัพธ์เป็น true หรือ false ตามเครื่องหมายประกอบด้วย
  - == เปรียบเทียบข้อมูลจากนิพจน์ทางซ้ายและทางขวามีค่าเท่ากันหรือไม่ หากเท่ากันจะให้ผลลัพธ์เป็น true มิฉะนั้นจะได้ false
  - != เปรียบเทียบข้อมูลจากนิพจน์ทางซ้ายและทางขวามีค่าไม่เท่ากันหรือไม่ หากไม่เท่ากันจะให้ผลลัพธ์เป็น true แต่หากเท่ากันจะให้ผลเป็น false
  - >= เปรียบเทียบ หากข้อมูลจากนิพจน์ทางซ้ายมีค่ามากกว่าหรือเท่ากันกับนิพจน์ทางขวาจะให้ผลลัพธ์เป็น true มิฉะนั้นจะให้ผลเป็น false
  - <= เปรียบเทียบ หากข้อมูลจากนิพจน์ทางซ้ายมีค่าน้อยกว่าหรือเท่ากันกับนิพจน์ทางขวาจะให้ผลลัพธ์เป็น true มิฉะนั้นจะให้ผลเป็น false
  - > เปรียบเทียบ หากข้อมูลจากนิพจน์ทางซ้ายมีค่ามากกว่านิพจน์ทางขวาจะให้ผลลัพธ์เป็น true มิฉะนั้นจะให้ผลเป็น false
  - < เปรียบเทียบ หากข้อมูลจากนิพจน์ทางซ้ายมีค่าน้อยกว่านิพจน์ทางขวาจะให้ผลลัพธ์เป็น true มิฉะนั้นจะให้ผลเป็น false
  - === เปรียบเทียบข้อมูลจากนิพจน์ทางซ้ายและทางขวามีค่าเท่ากันและเป็นข้อมูลชนิดเดียวกัน จะให้ผลลัพธ์เป็น true มิฉะนั้นจะได้ false
  - !== เปรียบเทียบข้อมูลจากนิพจน์ทางซ้ายและทางขวามีค่าไม่เท่ากันหรือเป็นข้อมูลต่างชนิดกัน จะให้ผลลัพธ์เป็น true มิฉะนั้นจะได้ false
- เครื่องหมายให้ผลลัพธ์ตามเงื่อนไข conditional operator
  รูปแบบ <cond>?<expr1>:<expr> เครื่องหมายนี้จะทำการตรวจสอบ expression <cond> หากเป็น true จะให้ผลเป็น <expr1> หากเป็น false จะให้ผลเป็น <expr2> ตัวอย่างเช่น $max = ($a1>$a2) ? $a1 : $ a2; // หากตัวแปร $a1 มีข้อมูลที่มีค่ามากกว่าในตัวแปร $a2 ตัวแปร $max จะได้รับค่าเป็นข้อมูลจาก $a1 แต่หากไม่ใช่ ตัวแปร $max จะได้รับค่าข้อมูลจากตัวแปร $a2 หรือ
  $ค่าลดหย่อนสูงสุด = 30000;
  $ค่าลดหย่อน = $รายรับ * 0.30;
  $ค่าลดหย่อนจริง = $ค่าลดหย่อน<$ค่าลดหย่อนสูงสุด ? $ค่าลดหย่อน : $ค่าลดหย่อนสูงสุด เป็นต้น
- เครื่องหมายควบคุมการแสดงข้อผิดพลาด error control operator @ หากใช้เครื่องหมายนี้นำหน้าฟังก์ชั่นใด หากฟังก์ชั่นนั้นเกิดข้อผิดพลาดที่ไม่ร้ายแรง โดยปกติ PHP จะส่งข้อความคำเตือนกลับไปยัง browser แต่หากใช้เครื่องหมาย @ นำหน้าแม้ว่าเกิดข้อผิดพลาดก็จะไม่ปรากฏข้อความแสดงข้อผิดพลาดขึ้น แต่จะต้องเขียนคำสั่งตรวจสอบผลลัพธ์จากฟังก์ชั่นนั้นว่าดำเนินการสำเร็จหรือเกิดข้อผิดพลาด
- เครื่องหมายเพิ่ม-ลดค่าข้อมูลในตัวแปร incrementing/decrementing operators
  เครื่องหมาย ++, -- เป็นเครื่องหมายที่ทำการเพิ่มค่าข้อมูลในตัวแปรขึ้น 1 หรือลดค่าข้อมูลในตัวแปรลง 1 มีการใช้เครื่องหมายนี้ได้ 2 วิธีได้แก่
  - Pre-increment/decrement เพิ่ม/ลด ค่าในตัวแปรก่อนแล้วให้ผลลัพธ์เป็นค่าจากตัวแปรที่เพิ่มหรือลดค่าแล้ว ใช้เครื่องหมาย increment/decrement นำหน้าตัวแปร
  - Post-increment/decrement ให้ผลลัพธ์เป็นข้อมูลจากตัวแปรเดิมแล้วจึงเพิ่มหรือลดค่าในตัวแปรนั้น ใช้เครื่องหมาย increment/decrement ตามหลังตัวแปร
  - ตัวอย่างเช่น

```
$a = 10; $b = 15; $c = 20; $d = 25;
$w = ++$a;      // $w = 11, $a = 11
$x = $b++;      // $x = 15, $b = 16
$y = --$c;      // $y = 19, $c = 19
$z = $d--;      // $z = 25, $d = 24
```

- เครื่องหมายดำเนินการทางตรรก logical operators ใช้เพื่อเชื่อมนิพจน์ Boolean ที่ให้ผลเป็น true หรือ false ได้แก่
  - and หรือ && เชื่อมด้วยการดำเนินการแบบและ คือนิพจน์ทั้งด้านซ้ายและขวาของ operator จะต้องได้ผลเป็น true ทั้งคู่ ผลลัพธ์จาก operator จึงจะได้เป็น true
  - or หรือ || เชื่อมด้วยการดำเนินการแบบหรือ คือนิพจน์ทางด้านซ้ายหรือขวาของ operator ด้านใดด้านหนึ่งหรือทั้งสองด้านเป็น true ผลลัพธ์จาก operator จะได้เป็น true
  - xor คือการเชื่อมที่จะให้ผลลัพธ์เป็น true ก็ต่อเมื่อนิพจน์ด้านซ้ายหรือด้านขวาตัวใดตัวหนึ่งเพียงตัวเดียวเท่านั้นที่เป็น true
  - ! นิเสธ เป็น operator ที่มี operand เดียว กลับผลลัพธ์เป็นตรงกันข้าม หาก operand เป็น false จะให้ผลเป็น true หาก operand เป็น true จะให้ผลลัพธ์เป็น false

หมายเหตุ ข้อควรระวังในการใช้ operator and, or และ &&, || จะมีลำดับความสำคัญของ operator ต่างกัน เช่น && จะมีลำดับความสำคัญมากกว่าเครื่องหมาย and และ || จะมีลำดับความสำคัญสูงกว่า or

- เครื่องหมายการต่อข้อความ string concatenate operator ในภาษา php ใช้เครื่องหมายจุด . เป็นเครื่องหมายสำหรับการต่อนิพจน์ข้อความ ตัวอย่างเช่น
  ```
  $myName = "Arnold";
  $greeting = "Hello " . $myName ;
  ```
  เป็นต้น การต่อข้อความนิพจน์ที่เป็น operand อาจจะเป็นข้อมูลชนิดอื่นๆ ที่ไม่ใช่ string หากเป็น operator นี้ php จะแปลงข้อมูลให้เป็นข้อความ string โดยอัตโนมัติ เช่น
  ```
  $hr = 10; $mn = 20;
  $time = $hr . ":" . $mn; // ผลลัพธ์จะได้ 10:20
  ```

### 4 3) ลำดับการดำเนินการด้วยเครื่องหมายต่างๆ

ในกรณีที่ภายในนิพจน์ประกอบไปด้วยชุดเครื่องหมาย operator หลายตัวอยู่ภายในนิพจน์เดียวกัน ลำดับการดำเนินการของเครื่องหมายจะมีลำดับการทำงานโดยดูจาก 2 ประการได้แก่ ลำดับความสำคัญของเครื่องหมาย และลำดับการดำเนินการสำหรับเครื่องหมายที่สำคัญเท่ากัน (precedence and asscociativity) ซึ่งเครื่องหมายที่มีลำดับความสำคัญสูงกว่าจะดำเนินการก่อน เครื่องหมายที่มีความสำคัญต่ำกว่า แต่หากมีเครื่องหมายที่มีความสำคัญเท่ากันอยู่ในนิพจน์เดียวกัน เครื่องหมายที่ถูกดำเนินการก่อนจะเป็นไปตามลำดับ associativity ซึ่งจะเป็นการดำเนินการด้านซ้ายก่อน (จากซ้ายไปขวา) หรือดำเนินการด้านขวาก่อน (จากขวาไปซ้าย)

ในตารางรูปที่ 2.4-1 แสดงลำดับความสำคัญและลำดับการดำเนินการเมื่อมีความสำคัญเท่ากันโดยเรียงจากเครื่องหมายที่มีความสำคัญสูงสุด (New) ไปยังเครื่องหมายที่มีความสำคัญต่ำสุด (or)

รูปที่ 2.4-1 ตารางแสดงความสำคัญและลำดับการดำเนินการของ operator ต่างๆ

| **Associativity** | **Operators** |
|---|---|
| ไม่มี | New |
| ขวาไปซ้าย | [ |
| ขวาไปซ้าย | ! ~ ++ -- (int) (float) (string) (array) (object) @ |
| ซ้ายไปขวา | * / % |
| ซ้ายไปขวา | + - . |
| ซ้ายไปขวา | << >> |
| ไม่มี | < <= > >= |
| ไม่มี | == != === !== |
| ซ้ายไปขวา | & |
| ซ้ายไปขวา | ^ |
| ซ้ายไปขวา | \| |

| Associativity | Operators |
|---|---|
| ซ้ายไปขวา | && |
| ซ้ายไปขวา | \|\| |
| ซ้ายไปขวา | ? : |
| ขวาไปซ้าย | = += -= *= /= .= %= &= \|= ^= <<= >>= |
| ซ้ายไปขวา | And |
| ซ้ายไปขวา | Xor |
| ซ้ายไปขวา | Or |

ตัวอย่างเช่น

$x = $b + $a * 1.07 - $n * $q / $j;

นิพจน์นี้จะทำการคำนวณตามลำดับ ได้แก่

(1) $a * 1.07
(2) $n * $q
(3) (2) / $j
(4) $b + (1)
(5) (4) – (3)
(6) $x = (5)

หากเขียนเป็นนิพจน์ทางคณิตศาสตร์จะได้เป็น

$$x = b + 1.07a - \frac{n \times q}{j}$$

จากลำดับความสำคัญที่ทำให้เกิดลำดับของการคำนวณหรือประมวลผลที่ขึ้นกับเครื่องหมายที่ประกอบอยู่ภายในนิพจน์ ดังนั้นการเขียนนิพจน์เพื่อทำการคำนวณจึงต้องระวังเพื่อให้ได้ผลตามที่ต้องการ ซึ่งในกรณีที่ต้องการจัดลำดับของการคำนวณใหม่เพื่อให้สามารถสร้างนิพจน์ตามที่ต้องการได้สามารถใช้ ( ) ครอบกลุ่มของนิพจน์ย่อยเพื่อทำให้ในกลุ่มนั้นเกิดการคำนวณให้เสร็จสิ้นก่อนแม้ว่าจะมีเครื่องหมายที่มีความสำคัญสูงกว่าอยู่นอกกลุ่มก็ตาม ตัวอย่างเช่น หากต้องการเขียนนิพจน์

$$\frac{n}{m} + \frac{k}{3 + j}(6 + x)$$

หากเขียนนิพจน์ตามลำดับคือ $n / $m + $k / 3 + $j * 6 + $x ผลที่ได้จะเป็น

$$\frac{n}{m} + \frac{k}{3} + 6j + x$$

ซึ่งไม่ตรงกับผลที่ต้องการ จึงต้องใช้ ( ) เพื่อจัดกลุ่มการคำนวณเป็น

```
$n / $m + $k / (3 + $j) * (6 + $x)
```

### 4 4) Functions

ฟังก์ชันเป็นส่วนประกอบอีกชนิดหนึ่งของนิพจน์ เป็นการประมวลผลซึ่งอาจมีการดำเนินการภายในหลายขั้นตอนหรือมีการคำนวณที่ซับซ้อน หรืออาจประกอบด้วยคำสั่งภายในหลายคำสั่งดำเนินการเป็นชุดต่อเนื่องกันเป็นลำดับ ฟังก์ชันแต่ละฟังก์ชันจะมีชื่อที่ใช้เรียกอ้างอิงถึง ซึ่งเป็นชื่อที่ตรงกับกฎเกณฑ์การตั้งชื่อ (Identifier) มาตรฐานของภาษา PHP และอาจมีการส่งข้อมูลเข้าไปคำนวณหรือประมวลผลในฟังก์ชันนั้น เรียกว่า argument หรือ parameter ฟังก์ชันบางฟังก์ชันจะให้ผลลัพธ์กลับมาเพื่อใช้ในนิพจน์ แต่บางฟังก์ชันทำงานในลักษณะของคำสั่งที่ไม่มีการส่งค่าข้อมูลกลับมา ฟังก์ชันมี 2 ประเภทใหญ่ได้แก่

- ฟังก์ชันมาตรฐานที่มีอยู่ในภาษา PHP (PHP. standard functions)
- ฟังก์ชันที่ผู้เขียนโปรแกรมสามารถสร้างขึ้นใหม่ (user defined functions)

ไม่ว่าจะเป็นฟังก์ชันมาตรฐานหรือฟังก์ชันที่สร้างขึ้นเอง การเรียกใช้ฟังก์ชันจะมีวิธีเหมือนกัน ได้แก่

```
<functionName> ( [<parameter1>[,<parameter2>[,…]]])
```

โดยที่

| | |
|---|---|
| <functionName> | คือชื่อฟังก์ชัน |
| ( ) | เป็นวงเล็บที่แสดงว่าชื่อนี้เป็นชื่อฟังก์ชันและอาจส่งข้อมูล parameter ไปให้ฟังก์ชัน |
| [<parameter>] | คือ นิพจน์ที่จะส่งไปเป็นข้อมูลให้แก่ฟังก์ชัน (ในลักษณะส่งค่าไปให้ หรือ pass by reference) อาจจะไม่มี parameter หรือมี parameter หนึ่งหรือหลายตัวก็ได้ หากมีหลายตัวคั่นแต่ละตัวด้วย comma |

ตัวอย่างการเรียกใช้ฟังก์ชัน

```
$c = $a * sin( $x ); // sin is trigonometric sine function
echo "Password : " . str_repeat("x",$n);
closeFiles ( );
```

รายชื่อฟังก์ชันมาตรฐานของภาษา PHP แบ่งเป็นกลุ่มต่างๆ แสดงและอธิบายรายละเอียดบางส่วนในบทที่ 3 ส่วนการสร้างฟังก์ชันใหม่และการเรียกใช้ แสดงรายละเอียดอยู่ในหัวข้อ 2.7

### 4 5) <u>การสร้างชื่อค่าคงที่ (constant)</u>

ในการเขียนโปรแกรมหากต้องมีการใช้ค่าคงที่บางค่าในโปรแกรมเพื่อวัตถุประสงค์ใด เช่น ค่าลดหย่อน ส่วนลด อัตราภาษีมูลค่าเพิ่ม ฯลฯ อาจใช้ชื่อที่กำหนดแทนค่าคงที่เหล่านั้น เพื่อให้สามารถสื่อความหมายแทนค่าคงที่ต่างๆ ในโปรแกรม ทำให้ผู้เขียนโปรแกรมสามารถเข้าใจวัตถุประสงค์ของ expression เมื่อกลับมาอ่านโปรแกรมได้

ชื่อแทนค่าคงที่สามารถสร้างขึ้นได้โดยการใช้ฟังก์ชัน define ( ) ซึ่งมีรูปแบบคือ

**define** ('identifier', value )

ตัวอย่างเช่น define ('VATrate',0.07);

value ที่กำหนดให้ชื่อ constant จะต้องเป็น literal ที่มีชนิด boolean, integer, floating point หรือ string เท่านั้น

เมื่อกำหนดชื่อนี้แล้วสามารถเรียกใช้ชื่อได้โดยไม่ต้องมีเครื่องหมาย $ เช่น $VATamount = $totalPrice * VATrate; เป็นต้น

หากมีการเรียกใช้ชื่อ constant ที่ยังไม่ได้มีการกำหนดโดยฟังก์ชัน define( ) ในภาษา PHP จะสร้างชื่อนั้นขึ้นโดยอัตโนมัติให้แทนข้อมูลตามชื่อนั้นเอง เช่น หากใช้ชื่อ VATrate โดยไม่มีการกำหนดชื่อนี้มาก่อน จะถูกแทนด้วยข้อมูล "VATrate" เป็นต้น ซึ่งแตกต่างจากตัวแปรที่จะเป็น Null หากไม่มีการกำหนดข้อมูลให้แก่ตัวแปรนั้นมาก่อน

ข้อควรระวังในการเขียนโปรแกรมโดยเฉพาะผู้ที่คุ้นเคยกับการเขียนโปรแกรมในภาษาอื่น เช่น C, Java มาก่อน คือในการใช้ชื่อตัวแปรหากลืมใส่ $ นำหน้า PHP จะถือว่าเป็นชื่อ constant หากไม่มีการสร้าง constant นั้นมาก่อนจะได้ string ตามชื่อที่ระบุ

# 5 คำสั่งเลือกทำงานตามเงื่อนไข *Condition Statements*

เมื่อกล่าวถึงข้อมูล ตัวแปรและนิพจน์แล้วในหัวข้อนี้จะกล่าวถึงคำสั่งที่เกี่ยวข้องกับการเลือกเงื่อนไขในการทำงาน และการใช้ตรรกนิพจน์ logical expression ที่มาจากการเปรียบเทียบและเครื่องหมายดำเนินการงานตรรก

คำสั่งประเภทเลือกเงื่อนไขในการทำงานถือเป็นการประมวลผลประเภทหนึ่ง โดยการตรวจสอบเงื่อนไขที่ให้ผลเป็นข้อมูล Boolean ที่เป็น true หรือ false ซึ่งจะทำงานชุดคำสั่งชุดหนึ่งหากเงื่อนไขได้เป็น true หรือทำงานชุดคำสั่งอีกชุดหนึ่งหากได้รับผลของเงื่อนไขเป็น false โดยการใช้คำสั่ง if หรืออาจจะเลือกเงื่อนไขกลุ่มคำสั่งโดยตรวจสอบค่าข้อมูลว่าตรงกับกรณีใดก็จะไปทำชุดคำสั่งของกรณีนั้น โดยใช้คำสั่ง switch

การเลือกทำงานที่ใช้คำสั่ง if รวมไปถึงเงื่อนไขสำหรับคำสั่งการทำงานวนรอบที่อยู่ในหัวข้อถัดไป ล้วนแต่อาศัยการตรวจสอบเงื่อนไขที่เป็น จริง หรือ เท็จ โดยใช้ตรรกนิพจน์ (Boolean expression) ซึ่งอาจประกอบไปด้วย การเปรียบเทียบค่าด้วยการใช้ comparison operators การเชื่อมเงื่อนไขต่างๆ ด้วยเครื่องหมายดำเนินการทางตรรก logical operator อาจมีการใช้ฟังก์ชันที่ให้ผลเป็น Boolean และตัวแปร Boolean ด้วย

## 5 1) <u>boolean expression</u>

Boolean expression ตรรกนิพจน์ คือนิพจน์ประเภทที่ให้ผลออกมาเป็นข้อมูล Boolean ซึ่งเป็น true หรือ false อาจประกอบด้วย operator, operand, function ที่ให้ผล Boolean ประกอบด้วย

- เครื่องหมายเปรียบเทียบค่า ได้แก่ == , != , > , >= , < , <= , === , !== ซึ่งได้กล่าวถึงมาแล้วครั้งหนึ่งในหัวข้อ 2.4.2 ตัวอย่างของการใช้เครื่องหมายเปรียบเทียบค่า เช่น
  if ($a > $b) { … }
  while ($x <= $y+5) (นิพจน์นี้เครื่องหมาย + มีความสำคัญกว่า <= จึงดำเนินการก่อน)
  $c= ($a < maxList)?$d: maxList; (สามารถเปรียบเทียบกับ constant)
  หากมีการเปรียบเทียบค่าที่อยู่ในช่วง เช่นต้องการตรวจสอบว่าค่าตัวแปร $a มีข้อมูลที่มีค่าอยู่ระหว่าง 0 ถึง 30 หรือไม่ ในภาษาโปรแกรมจะเปรียบเทียบในลักษณะ 0 <= $a <=30 ไม่ได้ ต้องแบ่งเป็น 2 นิพจน์เชื่อมด้วยเครื่องหมายดำเนินการทางตรรก
- เครื่องหมายดำเนินการทางตรรก logical operator ได้แก่ &&, ||, !, ^, and, xor, or (หัวข้อ 2.4.2) ใช้ดำเนินการกับ operand ที่เป็น Boolean หรือใช้เชื่อมระหว่างการเปรียบเทียบหลายชุด เช่นการเปรียบเทียบค่าตัวแปร a ว่าอยู่ระหว่าง 0 ถึง 30 หรือไม่ จะต้องเขียน expression ($a >= 0) && ($a <= 30)

<u>short-circuit evaluation</u>

การคำนวณหาผลลัพธ์จากนิพจน์ของภาษา PHP มีลักษณะเป็น short-circuited คือหากได้รับผลลัพธ์ที่แน่นอนแล้วจะหยุดการคำนวณในส่วนที่เหลือ เช่น ในนิพจน์ ($a >= 0) && ($a++ <= 30) หากตัวแปร $a มีข้อมูล -3 การประมวลผลจะดำเนินการเพียงส่วน ($a >= 0) ซึ่งให้ผลเป็น false นิพจน์นี้สามารถสรุปได้ว่าผลลัพธ์จะเป็น false แล้วเนื่องจาก false นำไป and กับค่าใดๆ ก็จะได้ false จึงไม่จำเป็นต้องรู้ค่าที่เหลือ ดังนั้นนิพจน์ย่อย ($a++ <= 30) จึงไม่ถูกดำเนินการ และตัวแปร $a ก็จะไม่ถูกเพิ่มค่าด้วย

<u>การตรวจสอบค่าชนิดอื่นในลักษณะ Boolean</u>

ในการเขียน Boolean expression บางครั้งต้องใช้ข้อมูลที่ไม่ใช่ Boolean มาใช้เป็นเงื่อนไขแบบ Boolean เช่น expression ที่ให้ผลลัพธ์เป็น interger, null หรือ resource ต่างๆ เราสามารถใช้เทคนิคของ type juggling คือการแปลงชนิดให้โดยอัตโนมัติ (type juggling) มาใช้ประโยชน์ในการตรวจสอบเงื่อนไขได้ ซึ่งใช้วิธีการแปลงจากข้อมูลชนิดอื่นมาเป็น Boolean ดังที่ได้กล่าวถึงในหัวข้อ 2.3.8 ในส่วนการแปลงให้เป็น Boolean

## 5 2) <u>การจัดกลุ่มคำสั่ง Block statement</u>

คำสั่งควบคุมการทำงานไม่ว่าจะเป็นกลุ่มคำสั่งเลือกการทำงาน หรือกลุ่มคำสั่งทำงานวนรอบรวมไปถึงการสร้างฟังก์ชันใหม่ คำสั่งทำงานที่อยู่ภายในการเลือกหรือการวนรอบจะมีได้เพียงคำสั่งเดียว หากต้องการให้มีคำสั่งได้หลายคำสั่ง จะต้องรวมจัดเป็นกลุ่มคำสั่งหรือ Block statment โดยใช้เครื่องหมาย { } ครอบชุดของคำสั่งที่ต้องการสร้างเป็น block อีกกรณีหนึ่งที่ต้องใช้ block statement กับคำสั่งควบคุมการทำงานก็คือเมื่อมีการใช้ HTML mode ภายในคำสั่งควบคุมการทำงานไม่ว่าจะมีคำสั่ง HTML ยาวเท่าใดก็ตาม

```
{statement1; statement2; ... ; }

{ ?>
HTML tags
?> }
```

ตัวอย่าง เช่น

```
$lineCount = 0;
while ($dat = fgets($fp,1024)) {
   echo $lineCount, "  ", $dat, "<br>";
   $lineCount++;
}

if ($mailAvail) { ?>
<p>You have mail.</p>
?> }
```

**5 3) คำสั่ง if**

คำสั่ง if, if else เป็นคำสั่งที่เลือกทำงานตามเงื่อนไขหากเป็น true จะทำงานในชุดคำสั่งที่อยู่หลัง if แต่หากเงื่อนไขเป็น false อาจไม่ต้องทำคำสั่งใดเป็นพิเศษ หรืออาจทำงานในอีกชุดคำสั่งอีกชุดหนึ่ง หรืออาจมีการตรวจสอบหลายเงื่อนไข โดยใช้ if elseif จะทำงานในเงื่อนไขที่ตรงกับที่กำหนด ซึ่งมีการใช้คำสั่ง if 3 รูปแบบได้แก่

- รูปแบบที่ 1 คือจะดำเนินการคำสั่งหรือชุดคำสั่งเมื่อเงื่อนไขเป็น true หากเป็น false ไม่ต้องทำชุดคำสั่งพิเศษใดๆ

```
if (cond) statement;
```

ตัวอย่างเช่น

```
if ($expense > MAXEXPENSE) $expense = MAXEXPENSE;
if ($usr != "") {
  $usrAvail = true;
  $response = "Hello $usr.";
  }
```

ตัวอย่าง

```
<?
$DOW = date('D'); // get day of week short string 'Mon'..'Sun'
if ($DOW=='Sat' || $DOW=='Sun')  echo "Happy weekend";
?>
```

- รูปแบบที่ 2 หากผลลัพธ์ของเงื่อนไขเป็น true จะทำชุดคำสั่งชุดหนึ่งแต่หากเป็น false จะทำชุดคำสั่งอีกชุดหนึ่ง

```
if (cond) statement1; else statement2;
```

ตัวอย่างเช่น

```
<?
if (isset($uname)) $res = $uname; else $res = "undefined user";

if ($fp=@fopen($fname,'r')) {
   $dat = $fgetcsv($fp,200,":");
   echo "<a href=\"{$dat[1]}\">{$dat[0]}</a>\n";
   $lineCount++;
}
else
   echo "<p style="warning">Cannot open data file</p>";
?>
```

- รูปแบบที่ 3 มีการตรวจสอบเงื่อนไขหลายชุด จะทำงานชุดคำสั่งของเงื่อนไขที่เป็น true แรกสุดเท่านั้น โดยการใช้ if elseif ... else

```
if (cond)
  statement1;
elseif (cond)
```

```
   statement2;
elseif (cond)
   statement3
…
else
   statement;
```

สามารถใช้ block statement เพื่อให้เขียนหลายคำสั่งในแต่ละเงื่อนไขได้เช่นเดียวกับรูปแบบอื่นๆ

### 5 4) คำสั่ง switch case

Switch เป็นอีกคำสั่งหนึ่งที่ใช้เพื่อเลือกการทำงานตามเงื่อนไขของข้อมูล โดยการตรวจสอบผลลัพธ์ที่ได้จาก expression ว่ามีค่าเป็นตรงกับกรณีใด ก็จะทำชุดคำสั่งของกรณีนั้นไปจนจบคำสั่ง switch

- รูปแบบทั่วไป

```
switch (expr) {
   case const1: statements;
   case const2: statements;
   …
   default: statements;
}
```

เนื่องจากคำสั่งที่จะถูกดำเนินการเมื่อตรงกับกรณีใดกรณีหนึ่งแล้วจะทำตั้งแต่คำสั่งของกรณีนั้นไปจนจบถึง default รวมไปถึงคำสั่งที่อยู่ใน case ถัดลงไปด้วย ดังนั้นหากต้องการให้ชุดคำสั่งที่ถูกดำเนินการมีเฉพาะคำสั่งใน case นั้น จะต้องปิดท้ายชุดคำสั่งด้วยคำสั่ง break เพื่อให้ออกจากกลุ่มคำสั่ง switch ที่เหลือ

```
switch (expr) {
   case const1: statements; break;
   case const2: statements; break;
   …
   default: statements;
}
```

ตัวอย่าง เช่น

```
<?
$choice = $_POST['uChoice'];
switch ($choice) {
   case '1' :
      doFunction1 ( );
      $result = getSomeResult ( );
      break;
   case '2','3' :
      doFunction2 ($choice);
      break;
   case '4' :
      if ($name=='') echo ('Operation Failed. User name not found');
   default:
      echo ('Reenter command');
      include ('askUser.php');
      exit();
}
?>
```

ในตัวอย่างข้างต้นคำสั่ง switch จะตรวจสอบค่า expression ซึ่งเป็นตัวแปร $choice ว่าตรงกรณีใด หากเป็นค่า '1' จะทำคำสั่งใน case '1' คือ doFunction1 ( ); และ $result=getSomeResult ( ); และออกจากคำสั่ง switch (เนื่องจากการใช้คำสั่ง break) หากเป็นค่า 2 หรือ 3 จะทำคำสั่ง doFunction2() แล้วออกจากคำสั่ง switch หากเป็นค่า 4 จะทำคำสั่ง if ( ) และต่อเนื่องไปถึงคำสั่งใน default ด้วย หากเป็นค่าอื่นๆ นอกเหนือจากกรณีข้างต้น จะแสดงผล echo ( ) และเรียกฟังก์ชัน include ( ) จากนั้นจะจบโปรแกรมโดย exit ( );

# *6 คำสั่งทำงานวนรอบ Loop/Repetitive Statements*

คำสั่งทำงานวนรอบคือคำสั่งให้วนทำงานในชุดคำสั่ง และเมื่อทำงานครบแต่ละรอบจะมีการตรวจสอบเงื่อนไข Boolean expression เพื่อตัดสินใจว่าจะยังคงวนทำงานรอบต่อไปอีกหรือหลุดจากการวนรอบ ซึ่งในภาษา PHP มีคำสั่งทำงานวนรอบ 3 คำสั่งเหมือนกับภาษา C หรือ Java ได้แก่ while, do while และ for และยังเพิ่มคำสั่งวนรอบที่ดำเนินการกับตัวแปร Array โดยเฉพาะอีกคำสั่งหนึ่งคือ คำสั่ง foreach

### 6 1) <u>while statements</u>

มีรูปแบบของคำสั่งได้แก่

```
while (cond) statement;
```

หรือ

```
while (cond) {
   statements
}
```

คำสั่ง while จะทำการตรวจสอบเงื่อนไข cond ซึ่งเป็น Boolean expression ก่อนทำงานในชุดคำสั่งวนรอบ หรือที่เรียกว่าทำงานใน loop หาก cond เป็น true จะเข้าทำงานภายใน loop และเมื่อทำงานครบรอบแล้ว ก็จะกลับมาตรวจสอบ cond อีกครั้งหากยังเป็น true อยู่ก็จะวนทำงานใน loop ในรอบต่อไป ซึ่งจะวนจนกว่าการตรวจสอบเงื่อนไข cond จะได้ผลเป็น false จึงจะข้ามการทำงานของ loop นั้น ไปยังคำสั่งต่อจาก while ซึ่งการตรวจสอบเงื่อนไขก่อนเข้าทำงานใน loop นี้มีโอกาสที่คำสั่งใน loop ไม่ถูกเข้าไปทำงานตั้งแต่รอบแรกหากเงื่อนไขเป็น false มาก่อน

นอกจากการตรวจสอบ cond และการจบการทำงานจะต้องทำงานจนจบคำสั่งต่างๆ ใน loop แล้ว ยังอาจใช้คำสั่งกระโดดข้าม ซึ่งมี 2 คำสั่งได้แก่ continue และ break ซึ่งมีรายละเอียดคำสั่งในหัวข้อ 2.5.6

การใช้คำสั่งวนรอบ while และ do while ในหัวข้อถัดไป จะต้องมีคำสั่งที่ทำการเปลี่ยนแปลงข้อมูลที่นำมาใช้เป็นเงื่อนไขภายใน loop หรือจะต้องมีคำสั่งทำให้เกิดเงื่อนไขสำหรับการออกจาก loop ได้ มิเช่นนั้น การทำงานใน loop จะเกิดแบบไม่จบสิ้นคือวนทำงานภายใน loop โดยไม่มีการออก หากเป็นโปรแกรมที่ทำงานแบบทั่วไปจะดูเหมือนโปรแกรมทำงานค้าง แต่ในกรณีของ web application โดยเฉพาะการใช้ภาษา script บน web server มักจะมีการตั้งระยะเวลาทำงาน เพื่อไม่ให้ program ใดๆ ใช้ทรัพยากรของระบบมากเกินไป เช่น เวลาในการทำงานของ script PHP ที่ถูกตั้งเป็นค่า default จะให้เวลา 30 วินาที หากโปรแกรมใดมีการทำงานนานกว่านี้ จะถูกตัดการทำงานและแสดงผล error กลับไปยัง browser

ตัวอย่างเช่น

```
<?
$credit = 1000;
while ($credit > 0) {
   $interest = ($debt - $credit) * INTrate;
   echo ($interest, ",");
}
?>
```

ตัวอย่างด้านบนนี้ใช้ตัวแปร $credit เป็นเงื่อนไข แต่ภายใน loop ไม่มีคำสั่งใดที่ทำให้ค่าของ $credit เปลี่ยนแปลงดังนั้นหากมีการเข้าทำงานใน while loop แล้วจะเกิดการทำงานไม่หยุดภายใน loop เนื่องจากไม่มีการเปลี่ยนแปลง $credit ทำให้เงื่อนไข $credit > 0 เป็น true ตลอดเวลา ในการเขียนโปรแกรมจึงต้องระวังการออกแบบโปรแกรมที่จะให้ while ให้มีการใช้ค่าที่จะตรวจสอบเงื่อนไขและการปรับปรุงค่าตัวแปรที่ใช้เป็นเงื่อนไขอย่างรอบคอบ

### 6 2) <u>do while statements</u>

```
do
   statement
while (cond);
```

หรือ

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

do while เป็นคำสั่งวนรอบเช่นเดียวกับ while จะมีการตรวจสอบ cond ซึ่งเป็น Boolean expression หากยังมีค่าเป็น true ก็ยังคงทำงานวนรอบอีก 1 รอบ จนกว่าเงื่อนไข cond จะได้เป็น false จึงจะหลุดจากการทำงานของ do while แต่สิ่งที่ต่างกันก็คือ do while จะตรวจสอบเงื่อนไขหลังจากที่ทำงานใน loop เสร็จแล้ว ดังนั้นการทำงานจะต้องเข้าไปทำงานคำสั่งต่างๆ ใน loop 1 รอบก่อนจะตรวจสอบเงื่อนไข (ต่างกับ while ที่ตรวจสอบก่อนจะเข้าใน loop)

คำสั่งใน do while สามารถใช้คำสั่ง continue และ break เพื่อข้ามการทำงานได้เช่นเดียวกันกับคำสั่ง while

### 6 3) for statements

```
for (expr1; expr2; expr3) statement
```

หรือ

```
for (expr1; expr2; expr3) {
   statements
}
```

คำสั่ง for เป็นคำสั่งวนรอบที่มีการระบุเงื่อนไข การกำหนดค่าตั้งต้นของตัวแปรที่ใช้เป็นเงื่อนไข และการปรับปรุงค่าตัวแปรที่ใช้เป็นเงื่อนไขเมื่อทำงานแต่ละรอบ ซึ่งหากตัวแปรที่ใช้เป็นเงื่อนไขมีการเปลี่ยนแปลงค่าด้วย expression ที่แน่นอน การใช้คำสั่ง for จะทำให้เกิดข้อผิดพลาดได้น้อยกว่าคำสั่ง while เนื่องจากกำหนดคำสั่งปรับปรุงค่าตัวแปรเงื่อนไขไว้ในคำสั่ง for เลย

จากรูปแบบของ for ประกอบด้วย 3 expression ได้แก่

- expr1 สำหรับการกำหนดค่าเริ่มต้นให้แก่ตัวแปรที่ใช้เป็นเงื่อนไข เช่น for ($i=0; ..)
- expr2 เป็น boolean expression สำหรับตรวจสอบ cond ว่าจะยังคงให้ทำงานใน loop หรือไม่
- expr3 สำหรับปรับปรุงค่าตัวแปรที่กำหนดเงื่อนไขหลังจากการทำงานจบแต่ละรอบ

ตัวอย่างเช่น

for ($i=0; i<10; i++) { statements } // i=0,1,2,…,9

for ($k=$thisYear; $k>$firstYear; $k=getNextYear($k)) { … }

### 6 4) Foreach statement

```
foreach (array_expr as $value) statement

foreach (array_expr as $value) {
   statements
}

foreach (array_expr as $key => $value) statement

foreach (array_expr as $key => $value) {
   statements
}
```

คำสั่ง foreach เป็นคำสั่งวนรอบเพื่อเข้าไปยัง element ต่างๆ ในตัวแปร array ไม่ว่า array นั้นจะเป็น index แบบตัวเลขหรือใช้ key (Associative array) ก็ตาม ในการวนรอบจะทำการดึงข้อมูล value จาก element ของ array เรียงตามลำดับใส่ลงในตัวแปร ซึ่งสามารถนำตัวแปรนั้นมาใช้ประโยชน์ภายใน loop ได้ และอาจดึงค่า index ของ element เข้ามาใส่ในอีกตัวแปรหนึ่งก็ได้เช่นกัน

การใช้คำสั่งและตัวอย่างของการใช้ foreach กับ array แสดงอีกครั้งในหัวข้อ 2.6.2

# *7* คำสั่งควบคุมที่เกี่ยวข้องกับการทำงานใน *block*

- คำสั่ง **break**

เป็นคำสั่งให้ออกจากการทำงานใน block ซึ่งใช้ร่วมกับคำสั่งควบคุมต่างๆ ได้แก่ switch case, while, do while, for, foreach วัตถุประสงค์และตัวอย่างของการใช้ break ในคำสั่ง switch case ได้กล่าวถึงแล้วในหัวข้อ 2.5.4 ส่วนการใช้กับคำสั่งวนรอบมักจะใช้โดยการตรวจสอบเงื่อนไขพิเศษระหว่างการทำงานภายใน loop และออกจาก loop หากเกิดเงื่อนไขบางประการเป็นจริงขึ้น โดยการใช้ break ร่วมกับ if เช่น

```
<?
for ($i=0; $i<maxSize; $i++) {
   …
   if ($uArray[$i]==-999) break;
   …
}
?>
```

- คำสั่ง **continue**

เป็นคำสั่งที่ใช้กับคำสั่ง loop ต่างๆ ให้ข้ามการทำงานในส่วนที่เหลือของ block ใน loop ไปทำการตรวจสอบเงื่อนไขของ loop ในรอบใหม่ เช่น

```
<?
while ($row = getNextChoice( )) {
   if ($row=="dummy") continue;
   statements
}
?>
```

หากตัวแปร $row ที่ได้รับจาก getNextChoice() มีข้อความ "dummy" จะข้ามการทำงานคำสั่งที่เหลือใน while แล้วทำคำสั่ง $row=getNextChoice() และตรวจสอบผลลัพธ์อีกครั้ง ว่าค่า $row ที่ได้นั้นเทียบเท่ากับ true หรือ false

- ฟังก์ชัน **exit** (string str); และ **die** (string str);

ทั้งสองฟังก์ชันจะสั่งให้แสดงข้อความจาก str แล้วจบการทำงานของเว็บโปรแกรมดังนั้น code ในส่วนเหลือทั้ง php และ html จะไม่ถูกทำงาน มักจะใช้ในกรณีที่เกิดข้อผิดพลาด (error) ที่ทำให้โปรแกรมไม่สามารถทำงานต่อได้ เมื่อตรวจสอบพบข้อผิดพลาดนั้นจะใช้คำสั่ง die ส่งข้อความแสดงเหตุของข้อผิดพลาดก่อนจะจบการทำงาน

เนื่องจาก die ( ) เป็น function จึงสามารถนำไปใช้ใน expression ได้ ในการเขียนโปรแกรมนิยมที่จะใช้ function นี้ร่วมใน expression ที่เกี่ยวข้องกับการใช้ resource เช่น การเปิดไฟล์ การติดต่อฐานข้อมูล ซึ่งฟังก์ชันเหล่านั้นโดยส่วนใหญ่จะส่งผลเป็นตัวชี้ไปยัง resource หากสามารถดำเนินการได้สำเร็จ เช่น function เปิดไฟล์จะให้ผลลัพธ์เป็นตัวชี้ไปยัง file buffer หากเปิดไฟล์นั้นได้สำเร็จ แต่หากไม่สามารถเปิดไฟล์ได้จะให้ค่า Null ซึ่งเทียบได้กับค่า false ของ boolean ซึ่งสามารถนำไปเขียนร่วมกับ die( ) ให้ทำงานในกรณีที่ไม่สามารถดำเนินการใช้ resource ได้ ตัวอย่างเช่น

```
$filePt = fopen ("a.txt","r") || die ("Cannot open file! Please
contact admin.");
```

ใน expression นี้คือการเปิดไฟล์ด้วย function fopen ( ) หากดำเนินการสำเร็จจะได้รับ file pointer กลับไปให้ตัวแปร $filePt และถือว่าเป็นค่า true จึงทำให้ expression หยุดการทำงาน เนื่องจากเชื่อมด้วย Boolean operator || (or) ซึ่งหากมี operand ด้านซ้ายเป็น true แล้ว operand ด้านขวาจะไม่ต้องดำเนินการต่อเนื่องจากทราบผลแล้วว่าจะได้ true แต่หากไม่สามารถเปิดไฟล์ได้จะได้ผลลัพธ์เป็น Null หรือ false ดังนั้น operand ด้านขวาของ || จะถูกดำเนินการคือทำงานในฟังก์ชัน die( ) คือการส่งข้อความและหยุดทำงาน

# 8 ตัวแปร *Arrays*

ตัวแปร Array เป็นตัวแปรโครงสร้างที่สามารถมี element ภายในหลาย element ภายใต้ชื่อตัวแปรเดียวกัน จะอ้างถึง element ต่างๆ โดยใช้ index ซึ่งในภาษา PHP เป็น zero-based index array คือเป็น array ที่เริ่มต้น index ที่ 0

Array ในภาษา PHP แตกต่างจากภาษามาตรฐานเช่น C, C++, Java หรือภาษาอื่นๆ ที่โดยทั่วไป เนื่องจากตัวแปร array ในภาษาต่างๆ เป็นตัวแปรที่จะเก็บ element หลายๆ element ที่มีข้อมูลชนิดเดียวกัน แต่ในภาษา PHP ตัวแปร array แต่ละ element สามารถเก็บข้อมูลชนิดใดๆ ก็ได้โดยไม่จำเป็นต้องเหมือนกัน และไม่จำเป็นต้องประกาศจำนวน element ใน array สามารถเปลี่ยนแปลงเพิ่มจำนวน element ได้ในระหว่างโปรแกรมทำงาน

ในภาษา PHP ถือว่า array เป็น map ที่เก็บค่าข้อมูลสัมพันธ์กับคีย์ต่างๆ (map values to keys) ซึ่ง key ก็คือ index และการเก็บค่าข้อมูลมีการยึดถือลำดับของการเก็บ (ordered map) คือสามารถอ้างอิงตำแหน่งของข้อมูลแต่ละตำแหน่งด้วย index ได้

## 8 1) Array มิติเดียว (single dimensional array)

- การจองชื่อตัวแปร array (array declaration)

การสร้างตัวแปรให้เป็น array มีหลายรูปแบบได้แก่

```
$varname = array();
$varname = array(element_no);
$varname = array(value,value,value,…)
$varname = array(key=>value, key=>value, … )
```

ตัวอย่างเช่น

```
$student = array(30);
$payment = array ("Cash","Cheque","Credit Card","Bank Transfer");
$list = array( );
$ticketFee = array ("child" => 5, "local" => 10, "tourist" => 100);
```

ในตัวอย่างแรกเป็นการจองตัวแปร $student เป็น array ที่มี 30 elements โดยยังไม่มีข้อมูลเก็บอยู่ (แต่ละ element จะมีค่าเป็น Null) ตัวอย่างที่สองกำหนดตัวแปร $payment โดยให้มีข้อมูล 4 elements มีค่าตามที่ระบุ ตัวอย่างที่ 3 สร้างตัวแปร $list เป็น array ที่ยังไม่กำหนดจำนวน element ยังไม่มี element ใดๆ จะเพิ่ม element ได้โดยการกำหนดค่าให้ตัวแปรให้แก่ element และตัวอย่างที่ 4 กำหนดข้อมูลและค่า index ให้แก่แต่ละ element โดยใช้ key เป็นข้อความ

- การอ้างถึงข้อมูลแต่ละ element ใน array

การอ้างถึงข้อมูลแต่ละ element ทำได้โดยระบุ index ใน [ ] ต่อท้ายชื่อตัวแปร array เช่น

```
$student [20], $payment [0], $list[7], $ticketFee["local"]
```

รูปแบบทั่วไปคือ

```
$arrayname [<index expr>]
```

การเก็บข้อมูลลงใน element ของตัวแปร array ทำได้โดยคำสั่ง variable assignment ที่ระบุ index ที่ต้องการใน [ ] ซึ่งสามารถใช้ expression เพื่อระบุ index ได้ เช่น

```
$student[10] = "Somchai";
$list[$j] = $student [$j];
```

นอกจากนี้ยังสามารถระบุ index ถัดไปได้อัตโนมัติ เช่น

```
$user = array( );
$user[ ] = "Admin"; // user[0]
$user[ ] = "Somchai"; // user[1]
$user[ ] = "Hanuman"; // user[2]
$user[ ] = "Tossakan"; // user[3]
```

เป็นการกำหนดค่าให้แก่ element ที่ 0 ถึง 3 ของ array $user การอ้างถึง element ต่อไปโดยอัตโนมัติได้เนื่องจากภายใน array แต่ละตัวจะมี array pointer อยู่การอ้างถึง array โดยไม่ระบุ index จะใช้ pointer นี้เป็น index และจะเพิ่มค่าเองทุกครั้งที่ถูกอ้างถึง หากต้องการให้ pointer ชี้ index เริ่มต้นใหม่ให้ใช้ฟังก์ชัน reset($array) เช่น reset($user); เป็นต้น

- เครื่องหมายดำเนินการ operator ที่ใช้กับ array

- + , $a + $b เครื่องหมาย union คือการนำ array operand ด้านขวามาต่อท้าย operand ด้านซ้ายยกเว้น element ที่มี index ซ้ำกันจะไม่นำมาทับ
- เครื่องหมาย comparison operator == , != หรือ <> ตรวจสอบว่า array ที่เป็น operand มี element เหมือนกันหรือไม่
- เครื่องหมาย comparison operator === , !== ตรวจสอบว่า element ใน array $a และ $b มี element เหมือนกันและวางลำดับของ element เหมือนกันหรือไม่

• ฟังก์ชันที่เกี่ยวข้องกับ Array

- **unset**( ) สามารถใช้ยกเลิกการกำหนดค่าในบาง element ของ array หากกำหนดชื่อ element ใน parameter เช่น unset($student[5]) จะทำให้ $student[5] ไม่มีข้อมูลเก็บอยู่หรือเป็น Null, หรืออาจใช้ยกเลิกตัวแปร array ทั้งตัว หากส่งชื่อตัวแปร array เป็น parameter เช่น unset($student)
- **array_values**($array) จะทำการจัดตำแหน่ง index ของ element ใหม่ให้เรียงกันเฉพาะ element ที่มีข้อมูล เช่น

```
<?php
$a = array(1 => 'one', 2 => 'two', 3 => 'three');
unset($a[2]);
/* will produce an array that would have been defined as
   $a = array(1 => 'one', 3 => 'three');
   and NOT
   $a = array(1 => 'one', 2 =>'three');
*/

$b = array_values($a);
// Now $b is array(0 => 'one', 1 =>'three')
?>
```

- print_r($array) เป็นฟังก์ชันแสดงข้อมูลเกี่ยวกับตัวแปร หากส่งตัวแปร array ไปเป็น parameter จะแสดง index (key) และข้อมูลแต่ละ element ของตัวแปรออกไปยังบราวเซอร์ มักใช้เพื่อดูผลลัพธ์ตัวแปร array ในระหว่างตรวจสอบโปรแกรม
- reset($array) ทำการ reset ตัวแปร pointer ที่ใช้ชี้ index ของตัวแปรอัตโนมัติให้เริ่มต้นชี้ที่ element แรกสุด
- ฟังก์ชันอื่นๆ ที่เกี่ยวข้องกับ array ในภาษา PHP มีเป็นจำนวนมากหลายสิบฟังก์ชัน แสดงรายการในหัวข้อ 3.2 ในบทที่ 3

• การใช้ตัวแปร array ใน string literal

หากต้องการใช้ element ของ array ใน string literal ต้องครอบด้วย block { } เช่น

```
echo  "Name : {$user[$i]}. ";
```

### 8 2) <u>การใช้ foreach กับ array</u>

คำสั่ง foreach เป็นคำสั่งวนรอบเพื่อเข้าถึงข้อมูลใน element ต่างๆ ใน Array โดยจะนำข้อมูลจาก element ของ array ใส่ลงในตัวแปรที่กำหนดในคำสั่ง foreach และยังนำค่า index/key ของ array ไม่ว่าจะเป็น enumerate หรือ associative ใส่ลงในตัวแปรได้ด้วย คำสั่ง foreach เป็นคำสั่งที่ต้องใช้ร่วมกับตัวแปร array เท่านั้น

รูปแบบคำสั่ง foreach มี 2 รูปแบบ คือ การเข้าถึงค่าใน element ของ array เพียงอย่างเดียว และแบบที่ให้ค่า index ของ array ด้วย มีรูปแบบดังนี้

```
foreach (array_expression as $value) statement
foreach (array_expression as $key => $value) statement
```

ตัวอย่าง

```
<?
```

```
$dow = array('sun'=>'Sunday' , 'mon'=>'Monday', 'tue'=>'Tuesday',
'wed'=>'Wednesday', 'thu'=>'Thursday', 'fri'=>'Friday', 'sat'=>
'Saturday');

foreach ($dow as $dayOfWeek) {
   $msg = $dayOfWeek . "<br>\n";
   echo $msg;
}

foreach ($dow as $day =>$dayOfWeek) {
   $msg = "$day - $dayOfWeek <br>\n";
   echo "<p>",$msg;
}
?>
```

ตัวอย่างนี้ foreach ชุดแรกจะนำค่า value ใน array ได้แก่ 'Sunday' 'Monday'... ออกมาแสดงผล โดยแต่ละรอบตัวแปร $dayOfWeek จะได้รับข้อมูลที่เก็บใน element ของตัวแปร array $dow รอบละ 1 element ส่วนตัวอย่าง foreach ชุดหลังใน foreach loop จะได้รับตัวแปร 2 ตัวคือ $day มีข้อมูลจากค่า key ของ element และ $dayOfWeek มีค่าข้อมูลที่เก็บใน element นั้น

```
$ticketFee = array ("child" => 5, "local" => 10, "tourist" => 100);
foreach ($ticketFee as $fee) {
   $msg = $fee . "\n";
   echo $msg;
}
foreach ($ticketFee as $grp => $fee) {
   $oTicketFee[$grp] = $fee;
   $ticketFee[$grp] += 5;
}
```

ดังนั้นโดยสรุปแล้ว foreach ทำให้เราสามารถใช้ element แต่ละ element ใน array ด้วยการเข้าถึงทีละ 1 element เรียงกับตามลำดับโดยไม่จำเป็นต้องทราบว่าใน array นั้นมี index หรือ key อย่างไร

### 8 3) Array หลายมิติ (Multidimensional array)

ตัวแปร Array สามารถเก็บ element ที่เป็นข้อมูลชนิดใดๆ รวมทั้ง array ได้ เราจึงสามารถสร้าง array ที่เก็บ array อยู่ภายใน ซึ่งเรียกว่าเป็น multidimensional array เช่น

```
<?php
$fruits = array ( "fruits"  => array ( "a" => "orange", "b" =>
"banana", "c" => "apple"  ),
          "numbers" => array ( 1, 2, 3, 4, 5, 6 ),
          "holes"   => array ( "first", 5 => "second", "third")
                );

// Some examples to address values in the array above
echo $fruits["holes"][5];    // prints "second"
echo $fruits["fruits"]["a"]; // prints "orange"
unset($fruits["holes"][0]);  // remove "first"

// Create a new multi-dimensional array
$juices["apple"]["green"] = "good";
?>
```

การกำหนด multidimensional array ซึ่งหลักการโดยหลักคือการเก็บ array ใน array จึงมีหลายวิธีที่จะสร้างตัวแปร multidimensional array ตัวอย่าง เช่น

```
<?
$basket = array ( );
$basket[] = array ('prodID'=>'230019', 'quan'=>12, 'prc'=4800);
$basket[] = array ('prodID'=>'230507', 'quan'=>20, 'prc'=18000);
$basket[] = array ('prodID'=>'230321', 'quan'=>1, 'prc'=300);
$basket[] = array ('prodID'=>'230523', 'quan'=>5, 'prc'=2500);
```

```
$myFriend = array (
   "Julia" => array('name'=>'Julia Robert', 'email'=>'julia@abc.com'),
   "Tom" => array('name'=>'Tom Hang', 'email'=>'hang@1234.com'));

$newFriend = array('name'=>'Richard Gere', 'email'=>'gere@xyz.com');
$myFriend['richard'] = $newFriend;
?>
```

ในตัวอย่างแรกสร้างตัวแปร $basket เป็น array 4 elements (index 0-3) ซึ่งแต่ละ element เก็บ array ภายในที่มี 3 elements ได้แก่ prodID, quan และ prc

- การอ้างถึง element ใน multidimensional array

การอ้างถึง array element ที่เป็น multidimensional array สามารถอ้างได้โดยรูปแบบ $array[m][n] เช่นจากตัวอย่างที่ผ่านมาตัวแปร $basket สามารถอ้างถึง element ต่างๆ ได้ดังตัวอย่าง

- $item = $basket[0] ตัวแปร $item จะได้ array ที่ประกอบด้วย 3 element มีข้อมูลได้แก่ $item['prodID'] มีข้อมูล '230019', $item['quan'] มีข้อมูล 12 และ $item['prc'] มีข้อมูล 4800
- totalPrc= $basket[0]['prc'] + $basket[1]['prc'] + $basket[2]['prc'];
- $basket[4][prodID] = '990102';

ตัวอย่างการกำหนดข้อมูลให้ element ต่างๆ ของ array หลายมิติ

```
<?
$student = array( );
$student['24131001']['name'] = 'Arnold';
$student['24131002']['name'] = 'Anny';
$student['24131003']['name'] = 'Barny';
$student['24131004']['name'] = 'Conan';

$student['24131001']['birth'] = '01/02/63';
$student['24131002']['birth'] = '26/08/63';
$student['24131003']['birth'] = '09/12/62';
$student['24131004']['birth'] = '14/05/63';

$student['24131001']['prog'] = 'ELE';
$student['24131002']['prog'] = 'ELE';
$student['24131003']['prog'] = 'ELE';
$student['24131004']['prog'] = 'ELE';

echo "<pre>\n";
print_r ($student);
echo "</pre>\n";
?>
```

จากตัวอย่างนี้จะเกิดตัวแปร array ชื่อ $student 4 elements แต่ละ element เป็น array ที่มี 3 elements ซึ่งเมื่อใช้คำสั่ง print_r แสดง array นั้นออกมาจะปรากฏผลบน browser ดังนี้

```
Array
 (
    [24131001] => Array
        (
            [name] => Arnold
            [birth] => 01/02/63
            [prog] => ELE
        )

    [24131002] => Array
        (
            [name] => Anny
            [birth] => 26/08/63
            [prog] => ELE
        )
```

```
    [24131003] => Array
        (
            [name] => Barny
            [birth] => 09/12/62
            [prog] => ELE
        )

    [24131004] => Array
        (
            [name] => Conan
            [birth] => 14/05/63
            [prog] => ELE
        )

)
```

การสร้าง array สามารถใส่ข้อมูลชนิด array ที่ซ้อนกันได้ไม่จำกัดจำนวนชั้น และในแต่ละ element ก็ไม่จำเป็นต้องเก็บข้อมูลชนิดเดียวกันด้วย เช่น

```
$userConfig = array (
  "bgColor" => "blue",
  "textColor" => "brown",
  "history" => array (
    19=> array ("date"=>"02/15/2004", "url"=>"abc.com"),
    20=> array ("date"=>"03/12/2004", "url"=>"xyz.com"),
    21=> array ("date"=>"04/12/2004", "oper"=>"chpass")
    ),
  "favorite" => array (
    "folder1"=>"entertain",
    "folder2"=>"development",
    array ("desc"=>"my website", "url"=>"http://www.aa.com"),
    array ("desc"=>"most happy", "url"=>"http://www.happy.com")
    )
  );
```

ในตัวอย่างข้างต้น ตัวแปร $userConfig เป็น array ที่มีโครงสร้างข้อมูลภายที่เป็น array ที่มี element แตกต่างกัน หากใช้คำสั่ง print_r ( ) ร่วมกับ html tag <pre> จะได้ผลแสดงดังรายการด้านล่าง

```
<pre><? print_r( $userConfig); ?></pre>
```

```
Array
(
    [bgColor] => blue
    [textColor] => brown
    [history] => Array
        (
            [19] => Array
                (
                    [date] => 02/15/2004
                    [url] => abc.com
                )

            [20] => Array
                (
                    [date] => 03/12/2004
                    [url] => xyz.com
                )

            [21] => Array
                (
                    [date] => 04/12/2004
                    [oper] => chpass
                )
```

```
        )

    [favorite] => Array
        (
            [folder1] => entertain
            [folder2] => development
            [0] => Array
                (
                    [desc] => my website
                    [url] => http://www.aa.com
                )

            [1] => Array
                (
                    [desc] => most happy
                    [url] => http://www.happy.com
                )

        )

)
```

**2. <u>การใช้ foreach กับ multidimensional array</u>**

foreach เป็นคำสั่งวนรอบที่เข้าถึง array เพียง 1 ลำดับชั้นเท่านั้น เช่น ตัวอย่างต่อไปนี้ใช้ array $student ที่กำหนดในตัวอย่างของหัวข้อ 2.6.3 ที่ผ่านมา

```
<?
foreach ($student AS $id => $record) {

}
?>
```

ข้อมูลที่ได้ในตัวแปร $record เป็นตัวแปร array ที่อยู่ลำดับข้างในมี 3 elements คือ $record['name'], $record['birth'] และ $record['prog']

หากต้องการใช้ foreach เพื่อเข้าถึง element ใน multidimensional array จะต้องใช้ foreach ซ้อน foreach เพื่อเข้าถึง element แต่ละลำดับชั้น เช่น

```
<?
foreach ($student AS $id => $record) {
   foreach ($record AS $field => $data) {
      echo ("$field of student id. $id is $data<br>");
      }
   echo "<hr>\n";
}
?>
```

ผลจากส่วนโปรแกรมในตัวอย่างนี้จะแสดงผลบนจอ browser ดังนี้

name of student id. 24131001 is Arnold
birth of student id. 24131001 is 01/02/63
prog of student id. 24131001 is ELE

---

name of student id. 24131002 is Anny
birth of student id. 24131002 is 26/08/63
prog of student id. 24131002 is ELE

---

name of student id. 24131003 is Barny
birth of student id. 24131003 is 09/12/62
prog of student id. 24131003 is ELE

---

name of student id. 24131004 is Conan
birth of student id. 24131004 is 14/05/63
prog of student id. 24131004 is ELE

## *2.1 User defined function*

การรวมกลุ่มคำสั่งเพื่อทำงานอย่างใดอย่างหนึ่ง เป็นความสามารถในการแบ่งโปรแกรมย่อยของภาษาโปรแกรมต่างๆ ทำให้ผู้เขียนโปรแกรมสามารถรวมกลุ่มคำสั่งที่ต้องใช้แบบเดียวกันซ้ำกันหลายๆ ครั้ง หรือแบ่งส่วนโปรแกรมเพื่อให้ง่ายต่อการคิดการออกแบบและเขียนโปรแกรม แบ่งออกเป็นกลุ่มคำสั่งต่างๆ เรียกว่าเทคนิค divide and conquer ซึ่งสามารถตั้งชื่อโปรแกรมย่อยแต่ละกลุ่มและเรียกใช้ด้วยชื่อของโปรแกรมย่อยที่ตั้งให้ได้ ภาษา PHP มีการแบ่งโปรแกรมย่อยเรียกว่าฟังก์ชัน เช่นเดียวกันกับภาษา C/C++ และ Java แต่มีรูปแบบของการสร้างที่ต่างกัน

การสร้างฟังก์ชันในภาษา PHP มีรูปแบบดังนี้

### 2.1 1) Function declaration

```
function funcIdentifier ([$argument1[,$argument2[,…]]])  {
   statements
   [return <expr>;]
}
```

โดยที่

- funcIdenifier คือชื่อฟังก์ชันที่ต้องการตั้ง มีวิธีการตั้งชื่อเช่นเดียวกับ identifier อื่นๆ ซึ่งได้กล่าวถึงในหัวข้อ 2.3.1
- $arguments คือตัวแปรที่จะรับการส่งข้อมูลมาให้ฟังก์ชัน เพื่อใช้เป็นข้อมูลสำหรับทำการประมวลผลในฟังก์ชันนั้น การส่ง argument มาให้ function ในภาษา PHP เป็นการ pass by value คือการส่งค่าข้อมูลมาให้ ตัวแปรใน function การเรียกใช้ฟังก์ชันสามารถส่ง argument เป็น expression ได้ และการเปลี่ยนแปลงค่าในตัวแปร argument จากคำสั่งใน function declaration จะไม่มีผลต่อตัวแปรที่ระบุในการเรียกใช้ฟังก์ชัน
  ตัวอย่างการกำหนด argument เช่น
  function getRecordsFromFile ($filename, $mode) { ... }
  function calcPtax ($netincome) { ... }
- หากเป็น function ที่ต้องการให้ผลลัพธ์คืนเพื่อนำไปใช้เป็นส่วนหนึ่งใน expression การคืนผลลัพธ์จะใช้คำสั่ง return <expr> เพื่อส่งค่ากลับ

ตัวอย่างเช่น

```
function getBannerTag ( ) {
   $bNo = mt_rand(1,5);
   $Src= "images/banner/banner$bNo.gif";
   $href = "ads/ads$bNo.php";
   $tag = "<a href='$href'><img src='$Src'></a>";
   return $tag;
}
```

ตัวอย่างนี้สร้างฟังก์ชันชื่อ getBannerTag( ) ไม่ต้องการการส่ง argument ใดๆ จากการเรียกใช้ฟังก์ชัน และมีการส่งผลลัพธ์กลับจากฟังก์ชันด้วยคำสั่ง return โดยให้ผลลัพธ์จากตัวแปร $tag;

ตัวอย่าง

```
function deduct ($actual,$limit,$mode='abs',$inc=0) {
   if ($mode=='per') $limit = $inc * $limit/100;
   return min($actual, $limit);
}
```

ตัวอย่างนี้เป็นการสร้างฟังก์ชันชื่อ deduct โดยีการส่ง parameter 4 ตัว ซึ่ง 2 ตัวหลังเป็น optional คือคำสั่งที่เรียกใช้ฟังก์ชันนี้จะส่ง argument $mode และ $inc หรือไม่ก็ได้ เนื่องจากมีการตั้งค่า default argument ซึ่งในภาษา PHP สามารถกำหนด default argument values ได้ เช่นเดียวกับภาษา C ดังตัวอย่างข้างต้น หากไม่มีการส่ง argument นั้นจะใช้ค่า default ที่กำหนดใน function declration ในตัวอย่างมีการส่งผลลัพธ์กลับด้วย

หมายเหตุ ในแต่ละ function declaration ไม่จำเป็นต้องมี return เพียงคำสั่งเดียวอาจมีหลายคำสั่งก็ได้ เช่น อาจมีหลายเงื่อนไขในการคำนวณหาผลลัพธ์ ซึ่งคำสั่ง return นอกจากจะแสดงการส่งค่าผลลัพธ์กลับแล้วยังถือเป็นการจบการทำงานของฟังก์ชันนั้นด้วย

ตัวอย่าง

```
function calcSome ($x1,$x2) {
  if ($x1<100) return $x2 /100;
  else return $x2;
}
```

หากต้องการ pass by reference ทำได้โดยระบุเครื่องหมาย & นำหน้าชื่อ argument ใน function declaration และในคำสั่งที่เรียกใช้ต้องส่ง argument ที่ pass by reference เป็นชื่อตัวแปรเท่านั้น

```
function funcIndentifier ( &$argument ) {
  $argument = <expr> ;
  ...
  statements;
}
```

ตัวอย่างเช่น

```
<?
function nextIncomingTime(&$last)
{
    $last+= mt_rand(2,10);
}

$visitTime = 0;
echo "Before $visitTime, ";
nextIncomingTime($visitTime);
echo "After $visitTime.";
?>
```

หมายเหตุ หาก function declaration nextIncomingTime กำหนด argument ($last) เป็น pass by value ค่าของตัวแปร $visitTime ทั้งก่อนและหลังการเรียกใช้ function nextIncomingTime จะไม่เปลี่ยนแปลง

### 2.1 2) การเรียกใช้ฟังก์ชัน (function call)

การเรียกใช้ฟังก์ชันทั้งในลักษณะคำสั่ง หรือเป็นส่วนหนึ่งใน expression จะเรียกใช้โดยชื่อฟังก์ชันที่ต้องการตามด้วยวงเล็บ อาจมีการส่ง argument ภายในวงเล็บหรือไม่ก็ได้ แล้วแต่ข้อกำหนดของฟังก์ชันนั้นๆ ตัวอย่าง

```
<?
$pageTitle = "Example 2.7.2 Function Call";
$copyright = "copyright&copy;2005 by sumet angkasirikul";
function showHeader ($title ) {
  echo "<html>\n\t<head>\n\t\t<title>$title</title>\n\t</head>\n";
  echo "<body>\n";
}
function showFooter () {
  global $copyright;
  echo "<hr>\n<center>$copyright</center>\n</body>\n</html>";
}
function getArg ( ) {
  $arg = $_GET['name'];
  if ($arg!= "") return $arg;
  echo "<p>Missing argument.</p>\n";
  showFooter();
  exit (-1);
}
// Main program
showHeader ($pageTitle);
$greeting = "<p>Hello ". getArg(). ".</p>\n";
echo $greeting;
showFooter();
?>
```

นอกจากการเรียกใช้โดยอ้างถึงชื่อ function โดยตรงแล้วในภาษา PHP ยังสามารถใช้ค่าในตัวแปรเป็นชื่อเพื่อเรียกฟังก์ชันได้ด้วย เช่น

```
<?
$validateMethod = gettype($arg)=='integer'?
'intValidate':'floatValidate';
if ($validateMethod($arg)) {
}
```

ในตัวอย่างนี้หากตัวแปร $arg มีชนิดเป็น integer จะกำหนดค่าตัวแปร $validateMethod เป็น intValidate แต่หากเป็นค่าอื่นจะกำหนดให้ตัวแปร$validateMethod เป็น floatValidate และทำการเรียกใช้ฟังก์ชันในคำสั่ง if ซึ่งเป็นฟังก์ชันที่กำหนดชื่อโดยตัวแปร $validateMethod นั่นหมายถึง ฟังก์ชันที่ถูกเรียกใช้คือ function intValidate( ) หรือ floatValidate( ) ตัวแปรที่ใช้อ้างถึงชื่อฟังก์ชัน อาจเป็น string ธรรมดาหรือเป็น array element ก็ได้ ดังเช่นตัวอย่างต่อไปนี้

```
<?
function intValidate($i) {
echo "int val called with $i<br>\n";
}
function floatValidate($f) {
echo "float val called with $f<br>\n";
}
function strValidate($s) {
echo "str val called with $s<br>\n";
}
$funcName = array('intValidate','floatValidate','strValidate');
$funcName[0](50);
$funcName[1](100.5);
$funcName[2]('Hi');
?>
```

### 2.1 3) ตำแหน่งในโปรแกรมของการสร้างฟังก์ชัน

ตำแหน่ง code ของการสร้างฟังก์ชันอาจอยู่ในส่วนใดๆ ของเพจก็ได้ เช่นในส่วนบนสุดก่อนหน้า code html อื่นๆ ในส่วน head หรือในส่วน body ก็ได้ มีข้อระวังบางประการสำหรับการใช้ PHP version 3 คือ code ของการสร้าง function ต้องปรากฏก่อนคำสั่งเรียกใช้ฟังก์ชัน แต่ใน PHP version 4 จะทำงานได้โดยไม่มีข้อจำกัดนี้ อย่างไรก็ตามในภาษา PHP ยังมีความสามารถสร้าง function ภายในคำสั่ง conditional statement if เช่น

```
if ($specialReq) {
   $m = true;
   function specialCalc ($int, $cap) { … }
   }
```

ในตัวอย่างนี้ function speciaCalc จะถูกสร้างขึ้นหากเงื่อนไขของ if คือตัวแปร $specialReq เป็น true และจะปรากฏฟังก์ชันนี้หลังจากคำสั่ง if ถูกเรียกให้ทำงานแล้ว แต่หากเงื่อนไขเป็น false จะไม่ปรากฏ function specialCalc นี้ขึ้น รวมทั้งการเรียกใช้ฟังก์ชัน specialCalc ก่อนหน้าคำสั่ง if ที่มี function declaration นี้จะทำงานก็จะเกิด error เนื่องจากยังไม่มี function specialCalc เกิดขึ้น

นอกจากนี้ function declaration อาจจะยังสร้างภายใน function อื่นๆ ได้ด้วย โดยไม่ได้ถือว่าเป็น local function แต่การเรียกใช้จะต้องให้ฟังก์ชันที่มีการสร้างฟังก์ชันนั้นถูกเรียกทำงานก่อน เช่น

```
<?
getAnotherTag();
getSomeTag();
$a=10; $b=20; $c=$a+$b; echo $c;
function getSomeTag(){
   function getAnotherTag() {
     echo "O O O ";}
   echo"Bye...";}
?>
```

ในตัวอย่างนี้ function getAnotherTag ( ) ถูกสร้างขึ้นภายในฟังก์ชัน getSomeTag( ) การเรียกใช้ getAnotherTag( ); ก่อนที่ getSomeTag( ) จะถูกเรียกใช้ดังในตัวอย่างจะทำให้เกิด error เนื่องจากยังไม่ปรากฏฟังก์ชัน getAnotherTag แต่หากมีการเรียกใช้ getAnotherTag( ) หลังคำสั่งเรียกใช้ getSomeTag( ) จะสามารถทำงานได้ตามปกติ

หมายเหตุ ตัวอย่างข้างต้นนี้ เป็นตัวอย่างที่ code ของการสร้าง function (ทั้ง getSomeTag และ getAnotherTag) ปรากฏหลังคำสั่งที่เรียกใช้ ซึ่งทำงานได้เป็นปกติสำหรับ PHP version 4 แต่จะเกิด error หากใช้ PHP version 3

### 2.1 4) Scope และ Lifetime ของตัวแปรในฟังก์ชัน

การสร้างฟังก์ชัน user defined function จะแยกตัวแปรที่ใช้ภายในฟังก์ชันออกจากตัวแปรที่กำหนดอยู่ภายนอก ซึ่งตัวแปรที่ใช้ภายในฟังก์ชันเรียกว่า local variable ส่วนตัวแปรที่สร้างขึ้นนอกฟังก์ชันเรียกว่า global variable เมื่อมีการเรียกใช้ตัวแปรชื่อใดๆ ก็ตามในการกำหนดฟังก์ชัน ตัวแปรเหล่านั้นจะถูกสร้างขึ้นใหม่ แม้ว่าจะมีตัวแปรชื่อเดียวกันกับภายนอกแต่ก็ถือว่าเป็นคนละตัวแปรกัน ไม่มีการถ่ายทอดค่าถึงกัน

ตัวอย่างเช่น

```
<?
$greetingMsg = "Hello world";

function hello( ) {
echo '1-',$greetingMsg ;
$greetingMsg = "Dear friend";
}
hello( );
echo '2-',$greetingMsg;
?>
```

จะพบว่าข้อความที่ echo ภายใน function hello จะไม่มีข้อความใดแสดงออกมาเนื่องจากตัวแปร $greetingMsg ใน function hello ( ) ถือว่าเป็นคนละตัวกับที่กำหนดไว้ด้านนอกฟังก์ชัน และการกำหนดค่าใหม่ให้ตัวแปร $greetingMsg ใน function hello ( ) ก็ไม่มีผลต่อค่าของตัวแปร global ด้วย

หากต้องการใช้ตัวแปรที่กำหนดหรือสร้างขึ้นภายนอกฟังก์ชัน คือตัวแปร global ภายใน function จะต้องประกาศการใช้ตัวแปร global ด้วยคำสั่ง global ดังรูปแบบต่อไปนี้

```
global varname1, varname2, … ;
```

เช่นจากตัวอย่างด้านบนนี้หากต้องการให้ตัวแปร $greetingMsg ใน function เป็นตัวแปรเดียวกับ global ที่กำหนดไว้ภายนอก ทำได้ดังนี้

```
<?
$greetingMsg = "Hello world";

function hello( ) {
  global $greetingMsg;
  echo '1-',$greetingMsg ;
  $greetingMsg = "Dear friend";
}
hello( );
echo '2-',$greetingMsg;
?>
```

นอกจากนี้ตัวแปร local ที่สร้างขึ้นใน function declaration ยังมีลักษณะเป็นตัวแปร dynamic คือจะถูกสร้างขึ้นเมื่อเข้าสู่ฟังก์ชัน และเมื่อฟังก์ชันทำงานจบก็จะยกเลิกตัวแปรนั้น ดังนั้นเมื่อมีการเรียกเข้าฟังก์ชันเดิมอีก ตัวแปร local ทุกตัวก็จะถูกสร้างขึ้นใหม่โดยไม่มีการเก็บค่าเดิมล่าสุดไว้ หากเกิดกรณีที่เราต้องการให้ตัวแปร local ใน function สามารถเก็บข้อมูลไว้ใช้เมื่อถูกเรียกอีก ทำให้มีข้อมูลที่ต่อเนื่อง ก็สามารถทำได้โดยการประกาศให้ตัวแปรเป็น static คือจะถูกสร้างขึ้นและกำหนดค่าเริ่มต้นเมื่อเข้าฟังก์ชันนั้นครั้งแรกครั้งเดียว แต่เมื่อฟังก์ชันทำงานจบตัวแปร static นั้นก็จะไม่ถูกยกเลิกยังคงเก็บข้อมูลล่าสุดอยู่

รูปแบบของการกำหนดตัวแปร static ใน function declaration คือ

```
static varname = intial_literal;
```

เช่น

```
static $rnd = 0;
static $warnMsg = "Message: ";
```

หมายเหตุ การกำหนดค่าเริ่มต้นให้แก่ตัวแปร static ต้องกำหนดด้วยค่าข้อมูลคงที่ (literal) เท่านั้นไม่สามารถใช้ expression ได้

ตัวอย่างการใช้งานตัวแปร static เช่น

```
<?
function output ($msg) {
global $title;
static $header = "<html><head><title>$title</title></head><body>";
  echo $header . $msg;
  $header = "";
?>
```

หมายเหตุ การเก็บข้อมูลตัวแปร static จะเก็บจนกระทั่ง page นี้ทำงานเสร็จเท่านั้น ตัวแปรทุกตัวของเพจจะถูกยกเลิกทั้งหมด หากมีการเรียกเพจเดิมใหม่อีกครั้งตัวแปรทุกตัวก็จะเริ่มต้นใหม่

## 2.2 การใช้ไฟล์จากภายนอก

การสร้างรหัสโปรแกรมเพื่อสร้างงานใดงานหนึ่ง อาจมีการใช้ code ที่ซ้ำๆ กันในหลายๆ เพจ เช่นอาจจะมีการกำหนดค่าตัวแปรที่เหมือนๆ กัน อาจมีการใช้ฟังก์ชันที่สร้างขึ้นเอง (user defined function) ร่วมกัน ในภาษาโปรแกรมที่เป็นแบบ compile จะมีวิธีการสร้างส่วนที่ต้องใช้ร่วมกันเหล่านี้ในรูป object library หรือ dynamic linked library แต่ในภาษา PHP ซึ่งเป็นภาษา script ทำงานแบบ interpret การใช้ส่วนโปรแกรมร่วมจากหลายๆ เพจ ทำโดยการสร้างส่วนคำสั่งที่ต้องใช้ร่วมกันแยกเป็นไฟล์ต่างหาก เช่น แยกคำสั่งที่กำหนดค่าตัวแปรหรือชื่อ constant ร่วม ส่วนสร้างฟังก์ชันที่ต้องใช้ในหลายๆ page ไว้ในไฟล์ที่เรียกว่า include file และในเพจต่างๆ ที่ต้องการใช้ก็จะทำการใช้คำสั่งเพื่อเรียกไฟล์ที่แยกไว้เข้ามารวม

Include file ที่ใช้ในภาษา PHP ก็เป็นไฟล์ที่เป็น script เช่นเดียวกับไฟล์ของเพจต่างๆ ที่ไม่ต้องมีการ compile ใดๆ ก่อน และยังทำการ include ได้ทั้งไฟล์ประเภท html ล้วน, หรือมี code PHP ประกอบ โดยถือว่า code ที่อยู่ใน include file นั้นเริ่มต้นอยู่ใน mode HTML ดังนั้นหากจะมี code PHP ก็ต้องเปิดปิดด้วย tag ของ PHP ด้วย และหากเป็นไฟล์ที่มีโปรแกรม PHP ควรใช้ file extension เป็น .php ด้วย

ประโยชน์จากการใช้ include file มีหลายประการอาทิเช่น

- เพื่อรวมส่วนของการกำหนด function ที่ต้องใช้ในหลายๆ เพจ
- เพื่อสร้างและกำหนดค่าตัวแปรที่ต้องใช้เหมือนกันในเพจต่างๆ ใน application เดียวกัน
- เพื่อกำหนด code HTML ที่ต้องใช้ร่วมกันในหลายเพจ เช่น header, footer
- แยก code ของเพจออกเป็นส่วนย่อยๆ ที่จะถูก load เข้ามาทำงานเฉพาะเมื่อถูกเรียกใช้ตามเงื่อนไขเท่านั้น
- ใช้ในลักษณะเหมือนการ forward ไปยังเพจอื่นๆ เมื่อเกิดเงื่อนไข เช่น เกิดข้อผิดพลาดแล้วให้ไปทำงานหรือแสดงผลตามที่กำหนดใน include file (เพื่อกำหนดให้เป็นรูปแบบเดียวกัน) แต่การแสดงผลออกนั้นผู้ใช้ยังเห็นอยู่ภายใต้ชื่อของเพจที่เรียกคำสั่ง include หรือ เมื่อเรียกใช้เพจแล้วไม่มีการส่ง parameter จาก form หรือ query string จะให้กลับไปแสดง form สำหรับป้อนข้อมูล แต่หากมีการส่ง parameter มาจะนำไปประมวลผลทันที ฯลฯ เป็นต้น
- สามารถใช้ในรูปคล้ายกับการเรียก function ได้ แต่จะแตกต่างในด้านการใช้ประโยชน์จากตัวแปรต่างๆ

ตัวอย่างการใช้ include file ปรากฏอยู่ในบทต่างๆ ตั้งแต่บทที่ 4 เป็นต้นไป

### 2.2 1) คำสั่งที่เกี่ยวข้องกับการใช้ include file

ในภาษา PHP มีหลายคำสั่งที่เกี่ยวกับการใช้ include file ได้แก่ include ( ), require( ), require_once( ), include_once( ),

ฟังก์ชัน include ( ) และ require ( ) จะทำการดึงไฟล์ที่กำหนดเป็น include file เข้ามาในตำแหน่งที่ถูกเรียก เข้าสู่ mode HTML แล้วเริ่มทำงานในส่วนหลัก (main) ของ script หากเป็น code HTML ก็จะส่งผลนั้นกลับไป browser หากทำการเปิด PHP script ก็จะทำในส่วน main program หากเป็นส่วน function declaration ก็จะสร้างให้เกิด function นั้นขึ้น การกำหนดตัวแปรขึ้นในส่วน main program ก็จะกลายเป็นตัวแปร global แม้ว่า script ใน include file นั้นทำงานจบแล้วตัวแปร global ก็ยังคงอยู่

ทั้ง include และ require สามารถใช้เป็น expression ได้ โดยสามารถใส่คำสั่ง return เพื่อคืนค่าจาก include file และใช้เป็นส่วนหนึ่งใน expression เช่น

```
$aNum = include('calcInc.php') * $scale;
```

เนื่องจาก include( ) และ require( ) เป็นคำสั่งพิเศษ (special language constructs) ที่จะดึงเอาคำสั่งที่อยู่ภายในไฟล์ที่อาจมีหลายคำสั่งเข้ามารวมกับไฟล์หลัก ดังนั้นการเรียกใช้ include หรือ require ในคำสั่ง conditional block จะต้องเขียนอยู่ใน statement block { } ดังเช่น

ตัวอย่างการใช้ include() ใน conditional blocks

```
if ($mode=='A') { include("methodA.php"); }
else { include("methodB.php"); }
```

ความแตกต่างกันระหว่างฟังก์ชัน include( ) กับ require ( ) ก็คือหากทำการ include ไม่สำเร็จ การใช้ include ( ) จะเพียงเกิดข้อความเตือน (warning) แต่จะยังคงทำงานส่วนอื่นๆ ต่อไป แต่หากเป็น require ( ) จะเกิดข้อความแจ้งข้อผิดพลาดแล้วโปรแกรมจะหยุดทำงาน

ตัวอย่างการใช้ include ( ) เบื้องต้น

vars.php
```
<?php
$color = 'green';
$fruit = 'apple';
?>
```

test.php
```
<?php
echo "A $color $fruit"; // A
include 'vars.php';
echo "A $color $fruit"; // A green apple
?>
```

include ( ) สามารถใช้ในลักษณะฟังก์ชันที่ให้ค่าผลลัพธ์คืนกลับไปยังผู้เรียก เช่นเดียวกับการใช้ฟังก์ชันทั่วๆ ไป ใน include file สามารถใช้คำสั่ง return เพื่อคืนค่ากลับไปได้ แต่คำสั่ง return จะต้องอยู่ใน php fragment หลัก (ไม่ได้อยู่ในการกำหนด function) หากไม่มีการใช้คำสั่ง return ฟังก์ชัน include( ) จะให้ผลลัพธ์เป็น 1 หรือ 0 แสดงว่าสามารถ include ได้สำเร็จหรือไม่

ตัวอย่าง การใช้ include ( ) กับ return ()

return.php
```
<?php
$var = 'PHP';
return $var;
?>
```

noreturn.php
```
<?php
$var = 'PHP';
?>
```

testreturns.php
```
<?php
$foo = include 'return.php';
echo $foo; // prints 'PHP'
$bar = include 'noreturn.php';
echo $bar; // prints 1
?>
```

การ include เพื่อเรียก include file เดียวกันหลายครั้งภายใน page โดยใช้ include( ) หรือ require( ) หากใน include file นั้นมี function declaration จะเกิด function redefine error คือการสร้างฟังก์ชันซ้ำฟังก์ชันที่มีอยู่แล้ว หรือมีการกำหนดค่าตัวแปรเป็นค่าเริ่มต้นใหม่ ซึ่งในโปรแกรมอาจไม่ต้องการให้เกิดเหตุเช่นนั้น

คำสั่ง require_once() และ include_once() เป็นคำสั่งที่มีการทำงานเหมือน require( ) และ include( ) แต่จะแตกต่างกันที่หากมีการ include file นั้นมาแล้วในเพจปัจจุบัน จะไม่มีการเรียกเข้ามาทำงานใหม่อีกเมื่อถูกเรียกซ้ำ ก็คือจะมีการ include เข้ามาเพียงครั้งเดียวซึ่งจะแก้กรณีที่อาจเกิดการ include เข้าหลายครั้งโดยไม่ให้เกิด error ได้

ตัวอย่างเช่น

file_a.php
```
<?
include "file_b.php";
…
include "file_c.php";
…
?>
```

file_b.php
```
<?
```

```
$ufilename = "user.dat";
$count = 0;
function updateCounter ( ) {
…
}
function initialCounter( ) {
…
}
?>
```

file_c.php

```
<?
include_once "file_b.php";
…
?>
```

ในตัวอย่างข้างต้น file_a.php มีคำสั่ง include เพื่อเรียก file_b.php เข้ามาทำงานซึ่งจะมีการสร้างตัวแปรและฟังก์ชันเกิดขึ้น หลังจากนั้นมีการเรียก file_c.php เข้ามาทำงานซึ่งใน file_c.php อาจเป็น file ที่สามารถทำงานด้วยตัวเองได้และต้องการ function และตัวแปรที่สร้างใน file_b.php จึงมีคำสั่ง include "file_b.php" อยู่ภายใน แต่เมื่อ file_c.php ถูกเรียกใช้จาก file_a.php ตามตัวอย่างข้างต้น หากใช้คำสั่ง include หรือ require file_b.php ใน file_c.php จะทำให้เกิด error เนื่องจากมีการเรียกใช้ file_b.php จาก file_a.php มาแล้วทำให้ฟังก์ชัน updateCounter ( ) และ initialCounter ( ) ที่อยู่ใน file_b.php ถูกสร้างขึ้นแล้ว เมื่อถูก file_c.php เรียกใช้อีกครั้งคำสั่ง function declaration จะทำให้เกิด error เนื่องจากสร้างฟังก์ชันซ้ำกับที่มีอยู่เดิม จึงต้องใช้ include_once ( ) หรือ require_once ( ) ใน file_c.php (อาจใช้ใน file_a.php ด้วยก็ได้)

ตามปกติแล้วการกำหนดชื่อไฟล์ใน include( ) หรือ require( ) ไฟล์นั้นจะถูกค้นหาจากตำแหน่ง current directory (ใน directory เดียวกันกับเพจที่เรียกใช้ include( ) อยู่) หากต้องการระบุตำแหน่ง directory อื่นๆ ก็ทำได้โดยใช้ relative path เช่น "../inc-file/db-inc.php" หรือ absolute path ซึ่งเป็น path ที่เทียบกับ root ของ file system ในเครื่องแม่ข่าย เช่น "/usr/sumet/httpdocs/include/db-inc.php" หรือ "C:\Apache\httpdocs\include\db-inc.php" เป็นต้น (ดูเรื่องเกี่ยวกับ file system และการกำหนด path เพิ่มเติมในบทที่ 10)

### 2.2 2) การใช้ตัวแปรใน include file

เมื่อไฟล์ถูก include รวมเข้ามาจะได้รับการสืบทอดตัวแปรต่างๆ มาด้วย แล้วแต่ว่าจะ include ในตำแหน่งใด เช่นหาก include ใน main program (นอก function ต่างๆ ) code ที่ถูก include ก็จะมองเห็นตัวแปร global ทั้งหมดที่เกิดขึ้นแล้ว หากคำสั่ง include อยู่ใน function code ของ include ก็จะสามารถใช้ local variable ของ function นั้นรวมทั้งตัวแปรที่ประกาศเป็น global ของฟังก์ชันนั้นด้วย

ตัวอย่างเช่น

```
<?
$greetingMsg= "Hello world.";
include "my-inc.php";
?>
```

จากตัวอย่าง code ใน include file my-inc.php สามารถอ้างถึงตัวแปร $greetingMsg ได้ เว้นแต่ส่วน function declaration หากต้องการใช้ตัวแปรนี้ก็จะต้องประกาศคำสั่ง global เช่นเดียวกัน

ในกรณีที่มีการสร้างตัวแปรขึ้นใน include file ตัวแปรที่สร้างนั้นก็จะสามารถเรียกใช้และปรากฏในส่วนโปรแกรมนั้นด้วย เช่น หากมีการเรียกใช้คำสั่ง include ใน main program ตัวแปรที่สร้างใน include file (ส่วนที่อยู่นอก function declaration) ก็จะถือเป็น global variable ของ main program หรือเรียกใช้ include ใน function ตัวแปรที่สร้างใน main ของ include file ก็จะเป็น local variable ของฟังก์ชันนั้นด้วย

### 2.2 3) ตัวอย่างของการใช้ประโยชน์จาก include file

ตัวอย่างแรกเป็นตัวอย่างการสร้าง include file ที่ใช้กำหนดข้อมูลให้ตัวแปรที่ต้องการใช้ร่วมกันในหลายๆ page

Listing 2.10-1 my-var-inc.php

```
<?
$mySite = "Dokkae.com";
```

```
$myApp = "Member ";
$contact = "info@dokkae.com";
$copyright = "copyright &copy; 2000-2005";

$_MemberLogin = "loginName"; // logon user name cookie/session var
name
$_MemberPwd = "passwd"; // logon user password cookie/session var name
$_LastLoginName = "defLoginName"; // default user login name cookie
var name
$keepLoginPeriod = 30 * 24 * 60 * 60; // expire time for defLoginName
cookie

function pageFooter ( ) {
global $mysite, $contact, $copyright; ?>
<table>
  <tr>
    <td class="footertext" align="left"><?= "$mySite $copyright" ?></
td>
    <td class="footertext" align="right"><?= "<webmaster> $contact" ?
></td>
  </tr>
</table>
<? } ?>
```

ไฟล์ my-var-inc.php เป็นไฟล์ที่กำหนดตัวแปร และ function เพื่อใช้ใน page อื่นๆ ของ application เดียวกันทำให้ไม่ต้องมี script ที่กำหนดตัวแปรเหล่านี้ในทุกๆ page เพียงแต่ include ไฟล์นี้เข้าไป และเมื่อแก้ไขข้อมูลใน include file นี้ก็จะทำให้ผลในทุกๆ page เปลี่ยนแปลงตามไปด้วย

ใน page ที่ต้องการใช้ตัวแปรและฟังก์ชันที่กำหนดใน my-var-inc.php ก็จะเรียกใช้โดย

```
<?
include ("my-var0inc.php");
…
```

ตัวอย่างที่ 2 การใช้ include file เพื่อเลือก page ที่จะใช้ตามเงื่อนไข ในตัวอย่างนี้จะแสดง page พิเศษเฉพาะวันอาทิตย์ ไฟล์ index-workday.php (ในการใช้งานจริงอาจกำหนดให้เป็น index.php) จะทำการตรวจสอบเงื่อนไขหาก function date("D") ให้วันประจำสัปดาห์เป็น "Sun" แสดงว่าวันนี้ (เวลาที่ server) เป็นวันอาทิตย์ จะทำการ include นำเอาไฟล์ index-sunday.php เข้ามาทำงาน และจบการทำงาน ด้วยคำสั่ง exit ( ) ไม่แสดงผลในส่วนที่เหลือของ index-workdat.php แต่หากไม่ใช่วันอาทิตย์ ก็จะแสดงผลในส่วนของ HTML ตามปกติ

Listing 2.10-2 index-workday.php

```
<?
if (date('D')=='Sun') {
  include("index-sunday.php");
  exit( );
}
?>
<html>
<head>
<title>Workday</title>
</head>

<body>
<h1>… tag for work day page …</h1>
<p>Today : <?= date("r") ?></p>

</body>
</html>
```

Listing 2.10-3 index-Sunday.php

```
<html>
<head>
<title>Happy weekend</title>
</head>
```

```
<body>
<p>Today : <?= date("r") ?></p>
.. Sunday special page
</body>
</html>
```

ตัวอย่างที่ 3 เราสามารถ include ไฟล์ที่เป็น html ได้ เช่น ตัวอย่างไฟล์ index-Sunday2.php ใน Listing 2.10-4 มีคำสั่ง include ("hotNews.html") ซึ่งมี source code ดัง Listing 2.10-5 เมื่อทำการ execute จะได้รับผลลัพธ์ดัง Listing 2.10-6 และรูปผลที่แสดงบน browser ดังรูป 2.10-1

Listing 2.10-4 index-Sunday2.php

```
<html>
<head>
<title>Happy weekend</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body>
<p>Today : <?= date("r") ?></p>
<p>
<? include ("hotNews.html"); ?>
</p>
<p>.. Sunday special page
</p>
</body>
</html>
```

Listing 2.10-5 hotNews.html

```
<style type="text/css">
<!--
.style1 {
   font-family: Tahoma;
   color: #6699FF;
}
-->
</style>
<table>
<tr>
<td><h2 class="style1">ผู้โชคดีรายที่ 2 </h2>
  <p align="left"><img src="../images/pic-20050314-1308.jpg"
width="87" height="100" border="1" align="left">นายหวิน พัฒนศรีชัย อายุ 50
ปี อาชีพเก็บขยะผู้โชคดีได้รับเงิน 1 ล้านจากเครื่องดื่มชาเขียวยี่ห้อ โออิชิ หลังจากได้เก็บขวดเครื่อง
ดื่มจากถังขยะ แล้วเปิดดูใต้ฝาพบสัญลักษณ์หมายเลข 1 ซึ่งเป็นคนเก็บขยะรายที่สองที่โชคดี</p></
td>
</tr>
</table>
```

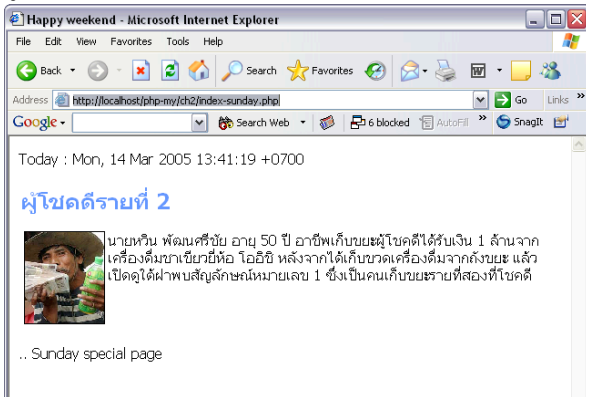
Listing 2.10-6 Source code ของผลลัพธ์ที่ปรากฏที่ browser

```
<html>
<head>
<title>Happy weekend</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body>
<p>Today : Mon, 14 Mar 2005 13:41:19 +0700</p>
<p>
<style type="text/css">
<!--
.style1 {
```

```
   font-family: Tahoma;
   color: #6699FF;
}
-->
</style>
<table>
<tr>
<td><h2 class="style1">ผู้โชคดีรายที่ 2 </h2>
  <p align="left"><img src="../images/pic-20050314-1308.jpg"
width="87" height="100" border="1" align="left">นายหวิน พัฒนศรีชัย อายุ 50
ปี อาชีพเก็บขยะผู้โชคดีได้รับเงิน 1 ล้านจากเครื่องดื่มชาเขียวยี่ห้อ โออิชิ หลังจากได้เก็บขวดเครื่อง
ดื่มจากถังขยะ แล้วเปิดดูใต้ฝาพบสัญลักษณ์หมายเลข 1 ซึ่งเป็นคนเก็บขยะรายที่สองที่โชคดี</p></
td>
</tr>
</table></p>
<p>.. Sunday special page
</p>
</body>
</html>
```

รูปที่ 2.10-1 ภาพผลลัพธ์ของ index-sunday2.php แสดงบนจอ browser




จากตัวอย่างเป็นการแสดงว่าเราสามารถใช้ include กับไฟล์ชนิดใดก็ได้ โดย PHP จะนำไฟล์นั้นเข้ามารวมและถือว่าข้อมูลในไฟล์นั้นเริ่มต้นอยู่ใน mode HTML เสมอ ซึ่งเราสามารถใส่คำสั่งของ PHP ได้โดยจะต้องเปิดและปิด PHP tag ครอบคำสั่งของ PHP ไว้

## 2.3 สรุปประจำบท

ในบทที่ 2 นี้ได้กล่าวถึงการเขียนเว็บโปรแกรมด้วยภาษา PHP ในลักษณะที่เป็น script ที่สามารถแทรกอยู่กับเอกสาร HTML ได้ โดยระบุชนิดของไฟล์เป็น .php หรือ .phtml มีวิธีการแทรกรหัส PHP ใน web page ได้ 4 วิธีได้แก่ (1) <script language="PHP">instructions</script> (2) <?php instructions ?> (3) <? instructions ?> และ (4) <?= expression ?>

การเขียนคำสั่งต่างๆ ในภาษา PHP ใช้ white space ได้แก่ space, enter (newline) และ tab เป็นเพียงตัวคั่นระหว่างส่วนประกอบภายในคำสั่ง และจะคั่นระหว่างคำสั่งด้วยเครื่องหมาย semicolon ; ดังนั้นการเขียนคำสั่งสามารถเขียนแยกหลายบรรทัดใน 1 คำสั่งหรือ เขียนหลายคำสั่งใน 1 บรรทัด หรือ 1 คำสั่ง 1 บรรทัดก็ได้

คำสั่งต่างๆ ที่อยู่ใน script PHP จะเกิดการทำงานใน web server โดยมี PHP script engine ทำหน้าที่แปลและสั่งให้ทำงาน (interprete) ผลลัพธ์ที่จะถูกส่งกลับไปยัง web browser จะส่งผลในส่วนที่ใช้คำสั่งแสดงผล (output) และการสลับไปยัง mode HTML เท่านั้น คำสั่งสำหรับการแสดงผลในภาษา PHP เบื้องต้นได้แก่คำสั่ง echo และคำสั่ง print แต่ก็ยังมีคำสั่งแสดงผลอื่นๆ ใน PHP อีกหลายคำสั่ง การแสดงผลกลับไปยัง browser จะต้องใช้ภาษาที่ web browser รู้จัก เช่นภาษา HTML, CSS, JavaScript ฯลฯ เป็นต้น

ในการเขียนโปรแกรมคอมพิวเตอร์ทุกภาษาจะมีส่วนประกอบสำหรับเขียนคำอธิบายที่ใช้เพื่อให้ผู้เขียน แก้ไข หรือศึกษาโปรแกรมได้อ่าน แต่จะไม่เกี่ยวข้องกับการทำงานของโปรแกรมเรียกว่า หมายเหตุ หรือ comment สำหรับภาษา PHP มีการใช้ comment รูปแบบที่นำมาจากภาษา C/C++ (และเหมือนกันกับภาษา Java) ได้แก่ /* multiple line comment */ , // one line comment และรูปแบบ comment ของ UNIX shell command ได้แก่ #

หลักการที่สำคัญของการประมวลผลคือการใช้ตัวแปรที่เปลี่ยนแปลงค่าได้เพื่อเก็บข้อมูลและนำไปประมวลผลในรูปแบบต่างๆ ให้ได้ผลลัพธ์ตามที่ต้องการ ตัวแปรในภาษา PHP มีการกำหนดชื่อตัวแปรโดยมีเครื่องหมาย $ นำหน้า ตามด้วยชื่อ (identifier) ที่ประกอบด้วย อักษรภาษาอังกฤษ _ ตัวเลข 0-9 และ อักขระ ASCII ที่มีรหัส 0x7F – 0xFF แต่อักขระตัวแรกที่ตามหลัง $ ไม่สามารถใช้ตัวเลขได้ ไม่จำกัดความยาวของชื่อ เป็นลักษณะ case sensitive คืออักษรภาษาอังกฤษตัวใหญ่กับตัวเล็กถือว่าต่างกัน และต้องไม่ซ้ำกับคำสงวน reserved word ของ PHP ที่ใช้กับชื่อคำสั่งหรือฟังก์ชันต่างๆ ที่มีการกำหนดอยู่แล้ว

ตัวแปรในภาษา PHP ไม่ต้องมีการประกาศ (variable declaration) ขึ้นมาก่อน และไม่ต้องกำหนดว่าตั้งแปรนั้นจะเก็บข้อมูลชนิดใด interpreter ของ PHP จะเปลี่ยนชนิดตัวแปรไปตามค่าข้อมูลที่ถูกกำหนดให้ และสามารถเปลี่ยนชนิดการเก็บข้อมูลให้ตัวแปรได้ใหม่ตลอดเวลา ตัวแปรจะเกิดขึ้นเองอัตโนมัติเมื่อมีการอ้างถึงครั้งแรก หากยังไม่มีการกำหนดข้อมูลให้ตัวแปรนั้นจะถือว่าข้อมูลที่อยู่ในตัวแปรนั้นเป็น Null ชนิดข้อมูลที่อยู่ในภาษา PHP มี 8 ชนิดได้แก่ Boolean, Integer, Float(double), String, Array, Object, Resource และ Null

ค่าข้อมูล (literal) ของข้อมูลแต่ละชนิดได้แก่

- ชนิด Boolean ข้อมูล true หรือ false
- ชนิด integer เขียนได้ทั้งในรูปเลขฐานสิบ (นำหน้าด้วยตัวเลขที่ไม่ใช่ 0) เลขฐานแปด (นำหน้าด้วยเลข 0) และเลขฐานสิบหก (นำหน้าด้วย 0x) ได้ทั้งจำนวนบวกและลบ
- ชนิด float หรือ double เขียนได้ทั้งเลขทศนิยม เช่น 1234.567 และเลข scientific notation 4.567e-12
- ชนิด string เขียนข้อความภายในเครื่องหมายคำพูดได้ทั้ง 'single quote' และ "double quote" แต่การเขียนใน double จะมี special character มากกว่าและสามารถแทรกข้อมูลจากตัวแปรลงในข้อความได้ด้วย

การกำหนดค่าข้อมูลให้แก่ตัวแปรใช้ operator เกี่ยวกับการกำหนดค่าได้แก่ $varname=expr; ซึ่งตัวแปรจะถูกกำหนดชนิดไปตามค่าผลลัพธ์ expression ที่ได้ ในกรณีที่มีการนำข้อมูลต่างชนิดมาดำเนินการคำนวณใร expression จะเกิดการแปลงข้อมูลให้เข้ากันโดยอัตโนมัติ (type juggling) หรือหากต้องการบังคับชนิดของข้อมูลให้แปลงไปเป็นชนิดที่ต้องการก็สามารถใช้วิธีการเลือกชนิดข้อมูล (type casting) ได้ 2 วิธีได้แก่ (1) (type) expr และ (2) ใช้ฟังก์ชัน settype ($varname,"type" )

การเขียน expression สำหรับทำการคำนวณหรือประมวลผลให้ได้ผลลัพธ์ที่ต้องการมีส่วนประกอบได้แก่ operand คือข้อมูลที่จะนำมาคำนวณ-ประมวลผล operator เครื่องหมายของการดำเนินการคำนวณ-ประมวลผล และยังอาจประกอบด้วยการเรียกใช้ฟังก์ชันที่รวมเอาการทำงานที่สามารถรับข้อมูลเข้าไปดำเนินการและได้ผลลัพธ์กลับออกมา

ในส่วนของ operand อาจจะเป็น ตัวแปร ค่าข้อมูล ชื่อแทนค่าคงที่ (constant) ฟังก์ชัน และอาจจะเป็น expression เอง (expression อาจประกอบด้วย expression ย่อยอยู่ภายใน) ส่วน operator จะมี operator ที่ดำเนินการด้านเลขคณิต ด้านตรรก (Logical operator) ด้านข้อความ ด้าน array และการกำหนดค่า

Operator ทางเลขคณิตได้แก่ +, -, *, /, %, ++, -- ด้านตรรกได้แก่ การเปรียบเทียบค่า ==, != , >, >=, <, <= , ===, !== ตัวเชื่อมทางตรรกได้แก่ AND, OR, &&, ||, ! การดำเนินการกับบิตของเลขฐานสองได้แก่ >>, <<, &, |, ~ ด้านข้อความ . (concate operator) ด้าน array + ด้านการกำหนดค่า =, +=, -=, *=, /=, %=, .= การเลือกผลลัพธ์ตามเงื่อนไข ? : และเครื่องหมายระงับการแสดงข้อความผิดพลาด @

ในบรรดาเครื่องหมายต่างๆ สามารถเขียนผสมกันอยู่ใน expression เดียวกันได้ ซึ่งในภาษา PHP มีการกำหนดลำดับความสำคัญของเครื่องหมายเมื่อมีการเขียนรวมอยู่ใน expression เดียวกัน หากเครื่องหมายใดมีความสำคัญมากกว่าจะดำเนินการก่อน (เรียกว่า operator precedence) และหากมีเครื่องหมายที่มีลำดับความสำคัญเท่ากันก็จะมีการกำหนดลำดับว่าจะคำนวณตัวใดก่อน ซ้ายไปขวาหรือขวาไปซ้าย (associativity)

คำสั่งประเภทต่างๆ ที่เขียนอยู่ใน script นอกจากการกำหนดค่าข้อมุลให้ตัวแปรแล้วยังมีคำสั่งเกี่ยวกับการควบคุมแบ่งเป็นประเภทตามหลักของการเขียนโปรแกรมคอมพิวเตอร์ ได้แก่

คำสั่งเลือกการทำงานตามเงื่อนไข (conditional statement) ประกอบด้วยคำสั่ง if ( ) else ซึ่งจะมีการตรวจสอบเงื่อนไขที่เป็น Boolean expression เมื่อเงื่อนไขเป็น true หรือ false จะเลือกทำงานในกลุ่มคำสั่งที่ต่างกัน และคำสั่ง switch case ที่จะเลือกทำงานตามกรณีที่เกิดขึ้นตรงกับข้อมูลที่ระบุ การใช้คำสั่งภายในคำสั่งเกี่ยวกับการควบคุม สามารถใช้คำสั่งได้เพียง 1 คำสั่ง เว้นแต่จะรวมกลุ่มคำสั่งเหล่านั้นให้กลายเป็น block statement โดยการครอบกลุ่มคำสั่งใน block ด้วยวงเล็บปีกกา { }

คำสั่งทำงานวนรอบ (repetitive statement) เป็นกลุ่มคำสั่งที่ทำให้เกิดการทำงานซ้ำๆ ภายในกลุ่มคำสั่งได้โดยตรวจสอบจากเงื่อนไขหากเงื่อนไขที่เป็น Boolean expression ยังเป็น true ก็จะวนทำงานกลุ่มคำสั่งภายในคำสั่งวนรอบ จนกว่าเมื่อทดสอบเงื่อนไขแล้วได้เป็น false จึงจะไปทำงานในคำสั่งถัดไป คำสั่งในกลุ่มนี้ของภาษา PHP ได้แก่ for ( ), while ( ), และ do while ( ) และยังมีคำสั่ง foreach ( ) ที่วนทำงานโดยใช้ตัวแปร array ซึ่งจะทำการวนเพื่อดึงข้อมูลจาก array มาใช้ทีละ element จนครบทุก element

คำสั่งควบคุมที่ใช้ร่วมกับการทำงานภายใน program block ได้แก่คำสั่งให้หลุดออกจากการทำงานใน block คือคำสั่ง break; คำสั่งให้วนกลับไปยังตอนต้นของการวนรอบเพื่อตรวจสอบเงื่อนไขคือคำสั่ง continue และฟังก์ชันที่เกี่ยวกับการทำงานของ script ทั้งหมดได้แก่ exit ( ) และ die ( )สั่งให้จบการทำงานจะไม่มีการทำงานในส่วนที่เหลือรวมทั้งการส่ง code HTML

การใช้ตัวแปรที่เก็บข้อมูลจำนวนมากได้หลาย element ในชื่อตัวแปรเดียวหรือข้อมูลในลักษณะ matrix คือตัวแปร Array ซึ่งในภาษา PHP นั้น ตัวแปร array จะเหมือน list ที่มี key เป็นตัวบอกตำแหน่งของการเก็บข้อมูลและมี value คือค่าข้อมูลที่เก็บใน list ณ ตำแหน่งที่กำกับด้วย key ค่า key หรือ index ในภาษาโปรแกรมทั่วไป สามารถใช้ได้ทั้ง enumerate คือเป็นตัวเลขที่เริ่มจาก 0 อาจจะมี index ที่ไม่ติดกันก็ได้ หรือใช้ associative array คือ index ที่เป็นข้อความหรือ key ซึ่งไม่จำกัดขนาดจำนวน element สามารถเพิ่มขึ้นได้เรื่อยๆ ขณะที่โปรแกรมทำงาน และแต่ละ element สามารถเก็บข้อมูลชนิดใดๆ ได้โดยไม่จำเป็นต้องเหมือนกันทุก element ซึ่งทำให้สามารถเก็บข้อมูล array ซ้อนใน element ของ array ได้ เป็นวิธีการเก็บข้อมูลแบบ array หลายมิติ multi-dimensional array

การสร้างตัวแปร array ทำได้โดยการระบุ $varname= array( array assignment) ส่วนการอ้างถึงข้อมูลใน element ของ array มีลักษณะเช่นเดียวกับภาษา C/C++ คือการใช้ [ ] เช่น $scoreAry[15] หรือ $movie["v13092"] หากเป็น multi-dimensional array ก็สามารถใช้ [ ][ ] เช่น basket[5]['k0924'] เป็นต้น หากอ้างถึงเฉพาะชื่อตัวแปรที่กำหนดเป็น เช่น $basket โดยไม่กำหนด index จะหมายถึงตัวแปร array ทุกๆ element

เนื่องจาก array ในภาษา PHP เป็นการเก็บข้อมูลแบบ list จึงได้มีการเตรียมฟังก์ชันเพื่อใช้กับข้อมูลใน array มากมายทั้งฟังก์ชันเกี่ยวกับการกำหนดข้อมูลใน array การค้นหาข้อมูล การปรับข้อมูลใน array ฯลฯ ซึ่งฟังก์ชันเหล่านั้นจะกล่าวถึงฟังก์ชันที่สำคัญในบทที่ 3 และสามารถใช้คำสั่ง foreach เพื่อสร้างคำสั่งวนรอบให้ดึงข้อมูลจาก element ของ array ออกมาประมวลผลทีละ element ได้ด้วย

การใช้ฟังก์ชันในภาษา PHP มีทั้งฟังก์ชันที่กำหนดในภาษา PHP แล้ว (predefined function) ซึ่งมีจำนวนฟังก์ชันแบ่งเป็นประเภทต่างๆ นับพันฟังก์ชัน และผู้เขียนโปรแกรมสามารถสร้างฟังก์ชันขึ้นใหม่ได้ (user-defined function) โดยรูปแบบของการสร้างฟังก์ชันในภาษา PHP ได้แก่ function

*funcname* (argument list) { statements } ซึ่งไม่มีการระบุชนิดของผลลัพธ์ที่จะได้จาก function และไม่ต้องระบุชนิดของ argument ที่ส่งมาให้แก่ฟังก์ชัน

ในการสร้างฟังก์ชันตัวแปรที่อยู่ภายในฟังก์ชันโดยปกติจะถือว่าเป็นคนละตัวกับตัวแปรที่อยู่ภายนอกหรืออยู่ฝในฟังก์ชันอื่น เรียกว่า local variable หากต้องการใช้ตัวแปรร่วมกับที่กำหนดไว้ภายนอกฟังก์ชัน (เรียกว่า global variable) จะต้องใช้คำสั่ง global เพื่อกำหนดชื่อตัวแปรที่จะใช้จากภายนอก และตัวแปรในฟังก์ชันจะมีคุณสมบัติเป็นตัวแปรแบบ dynamic คือจะถูกสร้างเมื่อฟังก์ชันนั้นถูกเรียกใช้และจะถูกยกเลิกเมื่อจบออกจากการทำงานของฟังก์ชัน ไม่สามารถเก็บรักษาค่าข้อมูลในตัวแปร local ไว้ได้ เว้นแต่จะกำหนดให้ตัวแปรเป็น static โดยใช้คำสั่ง static ระบุชื่อตัวแปรที่ต้องการ

การสร้าง web application ที่ประกอบด้วย page ต่างๆ หลายๆ page แต่ละ page มีหน้าที่ทำงานแตกต่างกันไป บ่อยครั้งจะมีการกำหนดข้อมูลที่เหมือนกัน อาจเป็นตัวแปร title หรือ header หรือการกำหนดค่าบางอย่างที่เหมือนกัน การใช้ include file เป็นวิธีที่ทำให้สามารถสร้างส่วนคำสั่งของ script ที่ต้องใช้ร่วมกันเหมือนกันในหลายๆ page ให้มารวมอยู่ในไฟล์แยกต่างหาก แล้วใน page ต้องการใช้คำสั่งเหล่านั้นจะเรียกคำสั่งที่แยกอยู่ในไฟล์อื่นเข้ามารวมได้โดยใช้คำสั่ง include ( ) หรือ require ( ) ซึ่งนอกจากจะเรียกไฟล์ที่มีคำสั่ง PHP เข้ามารวมแล้วยังอาจใช้เรียก page อื่นๆ ทั้ง page หรือไฟล์ที่มีเฉพาะ HTML หรือไฟล์ข้อความธรรมดา เข้ามารวมใน page เหมือนเป็นส่วนหนึ่ง และอาจจะเรียกเข้ามารวมภายในคำสั่งตรวจสอบเงื่อนไข คือเลือกไฟล์ที่จะดึงเข้ามาร่วมทำงานตามเงื่อนไขก็ได้ ซึ่งสามารถใช้ประโยชน์ได้อย่างมาก

หลักการที่กล่าวมาทั้งหมดในบทนี้ถือว่าเป็นพื้นฐานของการเขียนเว็บโปรแกรมในภาษา PHP ที่มีพื้นฐานหลักเช่นเดียวกับภาษาโปรแกรมโดยทั่วไป ประกอบการการใช้ฟังก์ชันต่างๆ ที่จะได้กล่าวถึงในบทที่ 3 จนถึงบทที่ 10 จะทำให้สามารถสร้าง web application ที่สมบูรณ์

ในบทต่อไปคือบทที่ 3 จะได้แนะนำฟังก์ชันที่ใช้งานทั่วไปในด้านต่างๆ ของคำสั่ง PHP เช่น ฟังก์ชันคำนวณทางเลขคณิต พีชคณิต การสุ่มตัวเลขการแปลงหน่วยเลข ฟังก์ชันที่เกี่ยวกับข้อมูล string, array ตัวแปรต่างๆ และฟังก์ชันเกี่ยวกับวันที่และเวลา

## *2.4* แบบฝึกหัดท้ายบท

1. การเขียน script ภาษา PHP รวมกับ HTML มี 4 วิธีได้แก่อะไรบ้าง
2. จาก code ต่อไปนี้จงทำให้ script นี้ทำงานได้โดยใช้วิธีใส่ tag PHP วิธีใดวิธีหนึ่ง

```
$title = "My Website : Welcome";
$heading = "Welcome to my world";
<HTML>
<HEAD><TITLE>echo $title;</TITLE></HEAD>
<BODY>
   <H1>echo $heading; </H1>
   <p>Hello, this is exercise 2</p>
</BODY>
</HTML>
```

3. จงเปลี่ยน code ในข้อ 2 จากการใช้ echo ให้เป็นการใช้ expression tag
4. จาก script ต่อไปนี้จงแปลงให้เป็นผลลัพธ์ที่จะจัดส่งไปยัง browser ทั้ง source code และผลลัพธ์ที่ปรากฏบนจอ browser

```
<html>
<head>
<script language="PHP">
echo "<TITLE>";
echo "PHP Chapter 2 exercise 4";
echo "</TITLE>";
echo "\n";
</script>
</head>
<body>
<script language="PHP">
echo "<H1>Exercise no.4 chapter 2\n<H1>";
echo "This is PHP code that generate simple HTML. source code then
send to browser. The browser will interpret the source as an HTML, so
HTML white space rule will applied to the result.";
</script>
</body>
</html>
```

5. แก้ไข script ในข้อ 4 เพื่อให้ผลลัพธ์ที่ส่งไปยัง browser ในรูปแบบของ source code เป็นดังนี้

```
<html>
<head>
<title>
   PHP Chapter 2 exercise 4
</title>
</head>
<body>
<H1>Exercise no.4 chapter 2<H1>
This is PHP code that generate simple HTML. source code then send to
browser. The browser will interpret the source as an HTML, so HTML
white space rule will applied to the result.
</body>
</html>
```

6. จงใส่ PHP comment เพื่ออธิบายโดยใช้ข้อความที่มีความยาว 2-3 บรรทัดแทรกลงใน script จากแบบฝึกหัดข้อ 4 ก่อนหน้าคำสั่ง echo "<title>"; ให้เป็นลักษณะ multiple line comment และ แทรก PHP comment ต่อท้ายคำสั่ง echo แต่ละคำสั่ง เขียนข้อความ comment สั้นๆ โดยใช้ one line comment และ แทรก comment ในภาษา HTML ก่อนหน้า <script language="PHP"> เพื่อบอกว่าส่วนต่อไปนี้เป็นส่วนที่สร้างโดย PHP script
7. script ต่อไปนี้ทำงานได้หรือไม่ หรือมีข้อผิดพลาดอย่างไร อธิบายและแก้ไขให้ถูกต้อง

```
1 <?PHP $title="PHP Chapter 2: exercises"; $heading
2 ="Exercise 7" ?>
```

```
3 <?PHP $thisYear = 2005; $myBirth= 1980;
4 $my myAge = $thisYear
5 -
6 myBirth
7 ;
8 ?>
9 <HTML>
10 <HEAD><?PHP echo "<TITLE>"; echo $title; echo "</TITLE>" ?>
11 </HEAD>
12 <BODY>
13 <?PHP echo <H1> $heading </H1> ?>
14 <?PHP "This exercise is white space usage in PHP programming."
15 "Many PHP instruction can wrote in any line or one instruction "
16 "can split to many line."?>
17 <?PHP echo "Each PHP instruction must end with semicolon."?>
18 <?PHP echo "This year is " . $thisYear .
19 ", My birth year is " . $myBirth .
20 ", Now I am "
21 . $myAge . " years old.";
22 ?>
23 </BODY>
24 </HTML>
```

8. จงตั้งชื่อตัวแปรที่ถูกต้องตามหลักภาษา PHP เพื่อใช้เก็บข้อมูลต่อไปนี้ (ให้ใช้ภาษาอังกฤษ)
   - E-mail address ของฝ่ายขาย
   - จำนวนสินค้าที่ลูกค้าได้เลือกไว้ในรถเข็น และราคารวมของสินค้านั้น
   - วัน/เดือน/ปี วันนี้ (เก็บเป็นข้อความในตัวแปรเดียว)
   - ตัวแปรเก็บข้อมูลที่สัมพันธ์กัน 4 ตัว หัวข้อข่าว หมายเลขรหัสของข่าว เนื้อข่าว ประเภทของข่าว
   - ดัชนีหุ้นประจำวันนี้

9. ใน script ต่อไปนี้การกำหนดชื่อตัวแปรแต่ละตัวถูกต้องหรือไม่เพราะเหตุใดอธิบาย (อธิบายทั้งที่ถูกและไม่ถูกต้อง) ตัวแปรใดที่ไม่ถูกต้องให้เปลี่ยนชื่อให้ถูกต้อง

```
<?
$myName = "John";
$post-code = "10110";
$Page= 1;
$pages= 10;
$_1stLine_ = "This is first line message";
$userpassword = "abc123";
$this year = 2005;
$target year = 2010;
$years to target = $Target Year - $ This Year;
?>
```

10. จากข้อตัวแปรที่ได้ตั้งชื่อในข้อ 8 เขียน script เพื่อกำหนดข้อมูลให้แก่ตัวแปรแต่ละตัว และแสดง page ดังต่อไปนี้โดยใช้ข้อมูลจากตัวแปรและใช้ expression tag

11. ใน script ต่อไปนี้จะเกิดผลใดขึ้นจงอธิบาย และจะแก้ไขให้ทำงานได้ถูกต้องอย่างไร

```
1 <?
2 $limitage = 18;
3 $userName = "John";
4 $this-year = 2005;
5 $userbirthdate = "1990-02-28";
6 $user-age = $thisyear - (int) $userbirthdate
7 ; echo "<b>Hello ", $username, "<b>\n";
8 echo "Your birth date is", userBirthDate, "<br>.";
9 if (user-age < limitage) {
10 echo "This page is prohibited for the people under", $limitAge,
".<br>\n";
11 echo "Now you are ", $userAge, years old.<br>\n";
```

```
12 exit ();
13 ?>
```

12. จงเขียน script เพื่อคำนวณดังต่อไปนี้โดยใช้ expression เดียวในการคำนวณและแสดงผลออกมา
    - ค่าไฟฟ้า = ราคาต่อหน่วยบวกค่าเอฟที คูณจำนวนหน่วย แล้วบวกด้วยค่ารักษามิเตอร์ จากนั้นบวกด้วยภาษีมูลค่าเพิ่ม (ภาษีมูลค่าเพิ่มคิดจากราคาทั้งหมดคูณอัตราภาษีมูลค่าเพิ่ม)
    - ค่าสินค้าแต่ละชิ้น = ราคาสินค้าต่อหน่วย คูณจำนวนหน่วย หักส่วนลด % ของสินค้าชนิดนั้น บวกค่าขนส่ง บวกด้วยภาษีมูลค่าเพิ่ม แล้วบวกค่า surcharge ของบัตรเครดิต (3% ของจำนวนเงินข้างต้น)
    - จำนวนหน่วยแตกต่างจากค่าเฉลี่ย = จำนวนหน่วยของเดือนนี้ ลบด้วยค่าเฉลี่ยของจำนวนหน่วยเดือนที่ 1 ถึง เดือนที่ 4 (ค่าเฉลี่ยคือผลลรวมของทุกเดือนหารด้วยจำนวนเดือน)
13. จากแบบฝึกหัดรายการแรกของข้อ 12 จงเขียนคำสั่งเพื่อกำหนดค่าให้ตัวแปรข้อความให้มีข้อมูลดังนี้ โดยใช้ตัวแปรใน double quote.
    - "จำนวนหน่วยที่ใช้ 999 หน่วย ราคาต่อหน่วย 9.99 บาท ค่าเอฟทีต่อหน่วย 9.99 บาท จำนวนเงินที่ต้องชำระทั้งสิ้นรวมภาษีมูลค่าเพิ่ม 9999.99 บาท"
14. จากแบบฝึกหัดรายการที่ 2 ของข้อ 12 จงนำข้อมูลมาแสดงผลในรูปของตาราง <TABLE> ขนาด 2 row โดย row แรกแสดงหัวตารางในแต่ละ column เป็นชื่อข้อมูล และ row ที่สองแสดงข้อมูลจากตัวแปรที่คำนวณได้

15. Expression เหล่านี้ให้ผลเป็นข้อมูลชนิดใดและได้รับผลลัพธ์เป็นค่าใด เมื่อ $x="5-2a"

%L. $x + "90/2 Suksawasd Rd."

%L. $x*5>=maxrate + 30

%L. 30 . "e5" * 0XABC

%L. 500 + "-$x) main" ? "1-Yes" : "0-No"

%L. 101001101 >> $x

(แนะนำ ให้ดูเรื่องชนิดของ operand ที่ operator นั้นต้องการ, การแปลงชนิด, ลำดับความสำคัญของเครื่องหมาย )

16. จาก expression ต่อไปนี้จงแสดงว่าจะเกิดการคำนวณหรือประมวลผลเป็นลำดับอย่างไรและได้รับผลลัพธ์อย่างไร

%L. 30 + 2 * 10 / 5 + 8

%L. x2-x1 / y2-y1

%L. $y1-$y2 > 5?$x1 + $j : $x2 - $j

%L. "20W50" + 100 / 2 * "3R" - "3E2J"

17. จงเขียนเว็บโปรแกรมให้กำหนดข้อมูลให้แก่ตัวแปร 4 ตัว ตัวแรกเก็บชื่อบุคคล (ให้สมมติชื่อเอง) ตัวที่สองและสามเก็บเวลาที่เข้างานเป็นหลัก ชม.และนาที ตัวแปรที่สี่และห้าเก็บเวลาเลิกงานเป็นหลักนาทีและวินาที โดยให้สมมติค่าให้แก่ตัวแปรทั้งห้าตัวแรก แล้วให้เขียนคำสั่งเพื่อคำนวณหาระยะเวลาในการทำงาน เป็นชั่วโมงและนาที จากนั้น นำตัวแปรทั้งหมดมาแสดงผลกลับไปยัง browser ในรูปแบบ

เวลาทำงานของ <ชื่อบุคคล>

- เวลาเข้างาน hh:mm น.
- เวลาเลิกงาน hh:mm น.
- เวลาทำงานทั้งสิ้น hh ชั่วโมง mm นาที

ข้อแนะนำ การคำนวณควรจะนำเวลาเข้างานมาแปลงให้เป็นจำนวนนาที และคำนวณเวลาทำงานจากหน่วยนาทีแล้วจึงแปลงกลับเป็น ชม.-นาทีเพื่อแสดงผล การแปลงจากจำนวนนาทีมาเป็น ชม.-นาที สามารถใช้ type casting (int) และ modulo operator % ช่วย

18. จงแสดงผลลัพธ์ HTML จากการทำงานของเว็บโปรแกรมต่อไปนี้

```
<?
$myBirth = "1966-12-15";
$toDay = "2004-12-16";
$myAge = $toDay - $myBirth;
echo "My birth date $myBirth and my age is $myAge :-)";
?>
```

19. จงแสดงผลลัพธ์จากการทำงานของเว็บโปรแกรมต่อไปนี้

```
<?
$thisMonth = "2-February";
$phase1Month = "4-April";
$finishedMonth = "8-August";
$monthUsed1 = $phase1Month - thisMonth;
$monthUsed2 = $finishMonth - $thisMonth;
echo "Job start from $thismonth, PhaseI finished on $Phase1month\n";
echo "Whole job finished month is $finishedMonth.\n";
echo "Time used for phaseI is $monthUsed1 monthes and $monthUsed2
monthes for whole job.";
?>
```

20. จงแสดงผลลัพธ์ HTML จากการทำงานของเว็บโปรแกรมต่อไปนี้

```
<?
  $arr = array( 2=>"Love", 4=>"Me", 0=>"Love", 3=>"My", 1=> "Dog");
  $today = "2004-12-17";
  print_r($arr);
```

```
   echo "<ol>";
   for ($i=1; $i<=strlen($today); $i+=2) {
      $j = substr ($today, $i, 1) + 0; ?>
      <li>+ <?= $i . "[$j]" ?> : <?= $arr[$j] ?> +</li>
   <? }
?>
</ol>
```

21. เขียน PHP program fragment เพื่อสร้าง html dropdown menu (<SELECT>) ที่ประกอบด้วยตัวเลือก 5 ตัวแต่ละตัวแสดงและส่งผลเป็น random password ซึ่งประกอบด้วยอักขระที่สุ่มมาจาก [A-Z,a-z,0-9,$,*,@,&,+] จำนวน 7-9 อักขระ โดยให้แยกส่วนของการสร้างข้อความสุ่มออกเป็นฟังก์ชัน 1 ฟังก์ชัน ให้มีการส่ง parameter คือจำนวนอักขระที่ต้องการ และแยกการสร้างข้อความตัวเลือก (<OPTION>) ออกเป็นอีก 1 ฟังก์ชัน

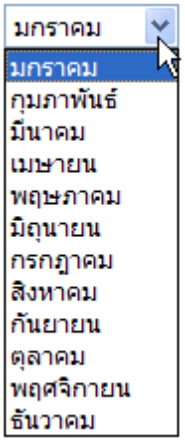
22. จงใช้คำสั่งวนรอบเขียน program fragment เพื่อสร้าง dropdown list (ใช้ HTML tag <select> และ <option>) เพื่อแสดงปี พศ. ย้อนหลังไป 5 ปี จนถึงปี พศ. ปัจจุบัน ซึ่งปี คศ. ปัจจุบันถูกกำหนดในตัวแปร thisYear และจะให้ผลลัพธ์จากการเลือกเป็นปี คศ. เช่น หากปี คศ. ปัจจุบันที่เก็บในตัวแปร thisYear คือ 2005 ผลลัพธ์จะสร้าง dropdown list ที่มีตัวเลือกเป็น 2544, 2545, 2546, 2547 และ 2548 หากผู้ใช้เลือกรายการ 2546 จะให้ผลลัพธ์จาก dropdown เป็น 2003 เป็นต้น

23. จงเขียน program fragment เพื่อแสดงข้อความ ไตรมาสที่ *9* โดย *9* คือการแสดงหมายเลข 1-4 โดยตรวจสอบจากตัวแปร thisMonth ซึ่งเก็บข้อความแสดงหมายเลขของเดือน ("01"-"12") เช่นหาก thisMonth มีข้อมูล "09" จะแสดงข้อความ ไตรมาสที่ 3 เป็นต้น

24. กำหนดตัวแปร array $monthTh มีจำนวน 12 elements โดยให้ key เป็น 1-12 มี value เป็นชื่อเดือน เช่น "มกราคม", "กุมภาพันธ์", ... , "ธันวาคม" จงเขียนส่วนของโปรแกรมเพื่อกำหนดตัวแปร $monthTh และส่วนของโปรแกรมเพื่อสร้าง dropdown list ที่มีตัวเลือกเดือนต่างๆ และให้ค่าตามเลขที่ของเดือนโดยใช้คำสั่ง foreach ให้กำหนดค่า value attribute ของ tag option ด้วยค่า key ของ array และข้อความที่จะแสดงใน list ด้วยค่าข้อมูลใน array HTML tag ของ dropdown list มีรูปแบบตัวอย่างดังนี้

```
<SELECT name="month">
<option value="1">มกราคม</option>
<option value="2">กุมภาพันธ์</option>
…
<option value="12">ธันวาคม</option>
</SELECT>
```

ภาพของ dropdown list ที่ต้องการ




25. จงใช้คำสั่ง while แทน foreach เพื่อสร้าง dropdown list ในข้อ 24

26. จงเขียนส่วนของโปรแกรมโดยใช้ for เพื่อสร้าง dropdown list ให้ผู้ใช้เลือกวันที่ 1-31

27. จากข้อ 24 จงแยกการกำหนดค่า array $monthTH ออกไปไว้ใน include file ชื่อ common.php และแก้ไขข้อ 24 ให้เรียกใช้ include file common.php

ข้อกำหนดต่อไปนี้ใช้ทำแบบฝึกหัดข้อ 28-32

- รถเข็นใส่สินค้า (cart) เป็น array ที่แต่ละ component เก็บข้อมูลเป็น array แบบ associative ที่มี 3 element ได้แก่ ['pid'] เก็บรหัสสินค้า, ['pdesc'] เก็บชื่อสินค้า, ['quan'] เก็บจำนวนสินค้า, ['unitPrc'] เก็บราคาต่อหน่วย
- คูปองลดราคา เก็บข้อมูลข้อความ "Y" หรือ "N"
- ราคารวม รวมราคาสินค้าที่ผู้ใช้ซื้อทั้งหมด (ราคาสินค้าแต่ละรายการ ได้จาก ราคาต่อหน่วย * จำนวนสินค้า)
- เปอร์เซนต์ส่วนลด เก็บส่วนลดเป็นเปอร์เซนต์ เช่น 10 หมายถึง 10%
- ส่วนลด มีกฎเกณฑ์การคิดคือ หากซื้อสินค้าเกินราคาที่กำหนดในตัวแปร Discount Limit และมีคูปองลดราคา (ตัวแปรเก็บคูปองลดราคามีข้อมูล "Y") จะคิดส่วนลดได้จากราคารวมเฉพาะส่วนที่เกินจาก discount limit * เปอร์เซนต์ส่วนลด (เช่น หาก discount limit เป็น 1000 บาท และราคารวมเป็นเงิน 3500 บาทแล้ว ส่วนที่จะนำไปคิดเป็นส่วนลดคือ 3500-1000 บาท) หากมีเศษให้ปัดเศษทิ้ง
- ราคาสุทธิ คิดได้จาก ราคารวมหักลดด้วยส่วนลด

28. เขียนส่วนของโปรแกรมเพื่อสร้างตัวแปร array รถเข็นใส่สินค้าพร้อมทั้งกำหนดค่าลงในรถเข็นจำนวน 4 รายการ ดังนี้
    - รายการที่ 1 รหัสสินค้า "10290" ชื่อสินค้า "บะหมี่เอเอเอ" จำนวน 25 ราคาต่อหน่วย 4.50
    - รายการที่ 2 รหัสสินค้า "48300" ชื่อสินค้า "กาแฟบีบวก" จำนวน 5 ราคาต่อหน่วย 49.50
    - รายการที่ 3 รหัสสินค้า "32011" ชื่อสินค้า "ปากกาดีดี" จำนวน 24 ราคาต่อหน่วย 8.25
    - รายการที่ 4 รหัสสินค้า "79992" ชื่อสินค้า "โคมไฟเอบีบี" จำนวน 10 ราคาต่อหน่วย 199
29. เขียนส่วนของโปรแกรมเพื่อสร้างฟังก์ชันคำนวณราคาสินค้าแต่ละรายการสร้างเป็น element ใหม่ชื่อ ['amount'] สำหรับสินค้าแต่ละรายการในรถเข็น และเขียนฟังก์ชันเพื่อคำนวณราคารวม
30. เขียนส่วนของโปรแกรมเพื่อสร้างการคำนวณส่วนลดตามกฎเกณฑ์ข้างต้นให้ใช้คำสั่ง if
31. จากข้อ 30 ให้ใช้ conditional operator ( ? : ) แทนการใช้คำสั่ง if
32. จากข้อ 28-31 นำมาสร้างเว็บโปรแกรมเพื่อกำหนดค่าตัวแปรต่างๆ และแสดงผลบนจอ browser ดังรูปต่อไปนี้ (กำหนด discount limit 1000 และ เปอร์เซนต์ส่วนลด 10%)

# Your Cart

| ลำดับที่ | (รหัส) รายการ | จำนวน | @ | รวม |
|---|---|---|---|---|
| 1 | (10290) บะหมี่เอเอเอ | 25 | 4.5 | 112.5 |
| 2 | (48300) กาแฟบีบวก | 5 | 49.5 | 247.5 |
| 3 | (32011) ปากกาดีดี | 24 | 8.25 | 198 |
| 4 | (79992) โคมไฟเอบีบี | 10 | 199 | 1990 |
| | | | รวมเป็นเงิน | 2548 |
| | | | ส่วนลด (มีคูปอง) | 154 |
| | | | ราคาสุทธิ | 2394 |