บทที่ 7
ข้อความ ตัวเลข วันและเวลา

ในบทนี้จะกล่าวถึงการใช้ฟังก์ชันของ PHP เพื่อจัดการข้อมูลใน 3 รูปแบบ คือ ข้อความ ตัวเลข วันและเวลา โดยในบทที่ผ่านๆ มานั้น ได้ใช้งานข้อมูลแบบข้อความ และตัวเลข กันมาบ้าง แต่ก็เป็นเพียงการใช้ในแบบพื้นฐานทั่วไป ข้อมูลทั้งสองอย่างนี้ยังมีรายละเอียดปลีกย่อยที่จำเป็นต้องรู้อีกมากมาย ส่วนข้อมูลประเภทวันและเวลานั้น ก็จัดว่าเป็นข้อมูลพิเศษอีกอย่างหนึ่ง ที่ไม่สามารถใช้ตัวดำเนินการใดๆ ไปจัดการโดยตรงได้ ต้องอาศัยฟังก์ชันเฉพาะสำหรับจัดการ

7.1 ฟังก์ชันที่ใช้จัดการข้อความ

ข้อความ (String) คือ ชุดของตัวอักษร หรือการนำอักษรแต่ละตัวมาวางเรียงต่อกันเพื่อสื่อความหมาย เช่นคำว่า “String” เกิดจากอักษร S + t + r + i + n + g เรียงต่อกัน เป็นต้น ทั้งนี้อักษรที่วางเรียงต่อกันเป็นข้อความ จะมีลำดับของการจัดเก็บเหมือนกับอาร์เรย์ของข้อความ ดังนั้นการอ้างถึงอักษรย่อยๆ แต่ละตัวภายในข้อความ จึงใช้เลขลำดับเป็นตัวกำหนดเหมือนกับการอ้างอิงสมาชิกของอาร์เรย์ เช่น

ตัวอย่างที่ 7.1 แสดงตัวอย่างข้อความ

```
$str = "Thailand" ;
echo $str[0] . $str[1] .$str[2] .$str[3] ;          //Thai
```

สำหรับฟังก์ชันที่ใช้จัดการข้อความมีจำนวนมาก แต่ที่ใช้งานกันบ่อยๆ มีอยู่ไม่มากนัก ดังนั้นในที่นี้จะกล่าวถึงเฉพาะฟังก์ชันที่น่าสนใจ และเพื่อให้ศึกษาได้ง่ายขึ้น จะแบ่งออกเป็นกลุ่มหัวข้อดังต่อไปนี้

7.1.1 ฟังก์ชันที่ใช้จัดการรหัสแอสกี

ฟังก์ชันที่ใช้จัดการรหัสแอสกี (ASCII) ที่น่าสนใจมีดังนี้

ตารางที่ 7.1 ฟังก์ชันที่ใช้จัดการรหัสแอสกี

ชื่อฟังก์ชัน	หน้าที่
ord (อักษร)	หาค่ารหัสแอสกีของอักษรที่ระบุ
chr (ค่าแอสกี)	แปลงค่ารหัสแอสกีที่ระบุไปเป็นตัวอักษร

ตัวอย่างที่ 7.2 การใช้ฟังก์ชัน ord ( ) และ chr ( )

```
<body>
<table border =  "1"  width = "100%" cellspacing = "0">
<caption>ตารางค่า ASCII ของ A – Z</caption>
<?php
        $a = ord ('A') ;
        $z = ord ('Z') ;
        for ($i = $a; $i <= $z; $i += 5) {
                echo "<tr align = center>";
                for ($j = $i ; $j < ($i+5) ; $j++) {
                        $ch = chr ($j) ;
                        echo "<td bgcolor = #ddd> $ch </td><td> $j </td>";
                }
                echo "</tr>" ;
        }
?>
</table>
</body>
```

// ผลลัพธ์ ดังนี้

ตารางค่า ASCII ของ A – Z									
A	65	B	66	C	67	D	68	E	69
F	70	G	71	H	72	I	73	J	74
K	75	L	76	M	77	N	78	O	79
P	80	Q	81	R	82	S	83	T	84
U	85	V	86	W	87	X	88	Y	89
Z	90	[	91	\	92	]	93	^	94

7.1.2 ฟังก์ชันที่ใช้สำหรับหาขนาดของข้อความ

ฟังก์ชันที่ใช้สำหรับหาขนาดข้อความที่น่าสนใจมีรายละเอียด ดังนี้

1) ฟังก์ชัน strlen ( ) เป็นฟังก์ชันที่ใช้สำหรับหาความยาวของข้อความ หรือนับจำนวนข้อความ โดยที่ช่องว่าง 1 ช่อง ก็จะนับเป็นอักษร 1 ตัวด้วย และในกรณีภาษาไทย และวรรณยุกต์ต่างๆ จะนับเป็นอักษร 1 ตัวเช่นกัน

รูปแบบ

```
int strlen ( string $string )
```

เมื่อ $string หมายถึง ข้อความที่ต้องการนับจำนวน

ตัวอย่างที่ 7.3 การใช้ฟังก์ชัน strlen ( ) สำหรับหาความยาวของข้อความ

```
1   <?php
2           $str = "abcdef";
3           echo strlen ($str);      // ผลลัพธ์ คือ 6
4           $str = "ab cd";
```

```
5               echo strlen ($str);       // ผลลัพธ์ คือ 7
6       ?>
```

จากตัวอย่างที่ 7.3 การใช้ฟังก์ชัน strlen ( ) สำหรับหาความยาวของข้อความ อธิบายดังนี้

บรรทัดที่ 1 เริ่มต้นสคริปต์ PHP

บรรทัดที่ 2 กำหนดค่าให้กับตัวแปร $str มีค่าเท่ากับ "abcdef"

บรรทัดที่ 3 แสดงผลการเรียกใช้ฟังก์ชัน strlen ( ) เพื่อนับตัวอักษรในตัวแปร $str ผลลัพธ์ที่แสดง คือ 6

บรรทัดที่ 4 กำหนดค่าให้กับตัวแปร $str มีค่าเท่ากับ "abc def" (เพิ่มเว้นวรรค)

บรรทัดที่ 5 แสดงผลการเรียกใช้ฟังก์ชัน strlen ( ) เพื่อนับตัวอักษรในตัวแปร $str ผลลัพธ์ที่แสดง คือ 7

บรรทัดที่ 6 สิ้นสุดสคริปต์ PHP

2) ฟังก์ชัน str_word_count ( ) เป็นฟังก์ชันที่ใช้สำหรับนับจำนวนคำ โดยใช้อักษรที่ไม่ใช่ตัวอักษร a-z เป็นตัวคัดแยก (รวมถึงตัวเลขด้วย) ยกเว้นเครื่องหมาย ' และ -

รูปแบบ

```
mixed str_word_count ( string $string [, int $format = 0 [, string $charlist ]] )
```

เมื่อ $string หมายถึง ข้อความที่ต้องการนับจำนวน

$format หมายถึง กำหนดรูปแบบการนับ มีรายละเอียดดังนี้

0 คือ นับจำนวนคำ (โดยปริยายจะเป็น 0)

1 คือ ส่งค่ากลับเป็นอาร์เรย์ของคำ (ค่าอินเด็กซ์เรียงตามลำดับ)

2 คือ ส่งค่ากลับเป็นอาร์เรย์ของคำ (ค่าอินเด็กซ์มาจากการนับจำนวนตัวอักษรต่อกันไปเรื่อยๆ)

$charlist หมายถึง กำหนดอักษรพิเศษอื่นๆ ที่ต้องการนับเป็นคำ

ตัวอย่างที่ 7.4 การใช้ฟังก์ชัน str_word_count ( ) สำหรับนับจำนวนคำ

```
1       <?php
2               $str = "Hello fri3nd, today! is good";
3               echo (str_word_count ($str));
4               print_r (str_word_count ($str, 1));
5               print_r (str_word_count ($str, 2));
6       ?>
```

จากตัวอย่างที่ 7.4 การใช้ฟังก์ชัน str_word_count ( ) สำหรับนับจำนวนคำ อธิบายดังนี้

บรรทัดที่ 1 เริ่มต้นสคริปต์ PHP

บรรทัดที่ 2 กำหนดค่าให้กับตัวแปร $str มีค่าเท่ากับ "Hello fri3nd, today! is good"

บรรทัดที่ 3 แสดงผลการเรียกใช้ฟังก์ชัน str_word_count ( ) เพื่อนับจำนวนคำในตัวแปร $str ผลลัพธ์ที่แสดง คือ 6

บรรทัดที่ 4 แสดงผลการเรียกใช้ฟังก์ชัน str_word_count ( ) โดยระบุพารามิเตอร์ 1 เพื่อส่งค่ากลับเป็นอาร์เรย์ของคำ ผลลัพธ์ที่แสดง คือ Array ([0] => Hello [1] => fri [2] => nd [3] => today [4] => is [5] => good)

บรรทัดที่ 5 แสดงผลการเรียกใช้ฟังก์ชัน str_word_count ( ) โดยระบุพารามิเตอร์ 2 เพื่อส่งค่ากลับเป็นอาร์เรย์ของคำ ผลลัพธ์ที่แสดง คือ Array ([0] => Hello [6] => fri [10] => nd [14] => today [21] => is [24] => good)

บรรทัดที่ 6 สิ้นสุดสคริปต์ PHP

7.1.3 ฟังก์ชันที่ใช้สำหรับเปลี่ยนรูปแบบของตัวพิมพ์

ฟังก์ชันที่ใช้สำหรับเปลี่ยนรูปแบบของตัวพิมพ์มีรายละเอียด ดังนี้

1) ฟังก์ชัน strtolower ( ) เป็นฟังก์ชันที่ใช้สำหรับเปลี่ยนข้อความที่ระบุให้เป็นตัวพิมพ์เล็กทุกตัว

รูปแบบ

```
string strtolower ( string $str )
```

เมื่อ $str หมายถึง ข้อความที่ต้องการเปลี่ยนรูปแบบ

ตัวอย่างที่ 7.5 ฟังก์ชัน strtolower ( ) สำหรับเปลี่ยนข้อความที่ระบุให้เป็นตัวพิมพ์เล็กทุกตัว

```
1  <?php
2      $str = "Mary Had A Little Lamb and She LOVED It So";
3      $str = strtolower($str);
4      echo $str;
5  ?>
```

ผลลัพธ์

```
mary had a little lamb and she loved it so
```

2) ฟังก์ชัน strtoupper ( ) เป็นฟังก์ชันที่ใช้สำหรับเปลี่ยนข้อความที่ระบุให้เป็นตัวพิมพ์ใหญ่ทุกตัว

รูปแบบ

```
string strtoupper ( string $str )
```

เมื่อ $str หมายถึง ข้อความที่ต้องการเปลี่ยนรูปแบบ

ตัวอย่างที่ 7.6 ฟังก์ชัน strtoupper ( ) ใช้สำหรับเปลี่ยนข้อความที่ระบุให้เป็นตัวพิมพ์ใหญ่ทุกตัว

```
1   <?php
2       $str = "Mary Had A Little Lamb and She LOVED It So";
3       $str = strtoupper ($str);
4       echo $str;
5   ?>
```

ผลลัพธ์

```
MARY HAD A LITTLE LAMB AND SHE LOVED IT SO
```

3) ฟังก์ชัน ucfirst ( ) เป็นฟังก์ชันที่ใช้สำหรับเปลี่ยนข้อความที่ระบุ เฉพาะตัวอักษรตัวแรกเท่านั้นเป็นตัวพิมพ์ใหญ่ ส่วนข้อความที่เหลือเหมือนเดิม

รูปแบบ

```
string ucfirst ( string $str )
```

เมื่อ $str หมายถึง ข้อความที่ต้องการเปลี่ยนรูปแบบ

ตัวอย่างที่ 7.7 การใช้ฟังก์ชัน ucfirst ( )

```
1   <?php
2       $foo = 'welcome to thailand';
3       echo ucfirst ($foo);            // ผลลัพธ์ คือ Welcome to thailand
4       $bar = 'WELCOME TO THAILAND';
5       echo ucfirst ($bar);            // ผลลัพธ์ เหมือนเดิม คือ WELCOME TO THAILAND
6       echo ucfirst (strtolower ($bar)); // ผลลัพธ์ คือ Welcome to thailand
7   ?>
```

4) ฟังก์ชัน ucwords ( ) เป็นฟังก์ชันที่ใช้สำหรับเปลี่ยนข้อความที่ระบุ เฉพาะตัวอักษรตัวแรกของทุกคำเป็นตัวพิมพ์ใหญ่ ส่วนข้อความที่เหลือเหมือนเดิม

รูปแบบ

```
string ucwords ( string $str )
```

เมื่อ $str หมายถึง ข้อความที่ต้องการเปลี่ยนรูปแบบ

ตัวอย่างที่ 7.8 การใช้ฟังก์ชัน ucwords ( )

```
<?php
        $foo = 'welcome to thailand';
        echo ucwords ($foo);      //ผลลัพธ์ คือ Welcome To Thailand
        $bar = 'WELCOME TO THAILAND';
        echo ucwords ($bar);      // ผลลัพธ์ เหมือนเดิม คือ WELCOME TO THAILAND
        echo ucwords (strtolower ($bar));        // ผลลัพธ์ คือ Welcome To Thailand
?>
```

7.1.4 ฟังก์ชันที่ใช้สำหรับแยกและรวมข้อความ

ฟังก์ชันที่ใช้สำหรับแยกและรวมข้อความที่น่าสนใจมีรายละเอียด ดังนี้

1) ฟังก์ชัน explode ( ) ใช้สำหรับคัดแยกข้อความออกเป็นข้อความย่อยๆ ด้วยสัญลักษณ์ที่กำหนดโดยผลลัพธ์ที่ได้จะอยู่ในรูปแบบของอาร์เรย์ของข้อความย่อยที่ถูกแยกออกมา

รูปแบบ

```
array explode ( string $delimiter , string $string  )
```

เมื่อ $delimiter หมายถึง สัญลักษณ์ที่ใช้แยก

$string หมายถึง ข้อความที่ต้องการแยก

ตัวอย่างที่ 7.9 การใช้ฟังก์ชัน explode ( ) คัดแยกข้อความออกเป็นข้อความย่อยๆ

```
<?php
        // Example 1
        $pizza  = "piece1 piece2 piece3 piece4 piece5 piece6";
        $pieces = explode (" ", $pizza);
        echo $pieces [0];          // ผลลัพธ์ คือ piece1
        echo $pieces [1];          // ผลลัพธ์ คือ piece2
        // Example 2
        $data = "foo:*:1023:1000::/home/foo:/bin/sh";
        list ($user, $pass, $uid, $gid, $gecos, $home, $shell) = explode (":", $data);
        echo $user;                // ผลลัพธ์ คือ foo
        echo $pass;                // ผลลัพธ์ คือ *
?>
```

2) ฟังก์ชัน implode ( ) และ join ( ) ทั้ง 2 ฟังก์ชันจะทำงานตรงข้ามกับฟังก์ชัน explode ( ) คือ เป็นฟังก์ชันรวมอาร์เรย์เข้าด้วยกันเป็นข้อความ โดยคั่นแต่ละข้อความด้วยสัญลักษณ์ที่ระบุ รูปแบบทั้ง 2 ฟังก์ชันเหมือนกัน ดังนั้นจะแนะนำเฉพาะฟังก์ชัน implode ( )

รูปแบบ

```
string implode ( string $glue , array $pieces )
```

เมื่อ $glue หมายถึง สัญลักษณ์ระบุเพิ่มเข้าไประหว่างข้อความ (หากไม่ต้องการคั่นด้วยสัญลักษณ์ก็สามารถระบุเป็นค่าว่างหรือเว้นวรรคได้ แต่หากจะแยกใหม่ในภายหลังจะทำได้ยุ่งยากหากไม่มีสัญลักษณ์คั่นในแต่ละข้อความ)

$pieces หมายถึง อาร์เรย์ที่ต้องการนำมารวมกันเป็นข้อความ

ตัวอย่างที่ 7.10 การใช้ฟังก์ชัน implode ( ) และ join ( )

```
1   <?php
2       $array = array ("lastname", "email", "phone");
3       $comma_separated = implode (", ", $array);
4       echo $comma_separated;
5   ?>
```

ผลลัพธ์

```
lastname, email, phone
```

7.1.5 ฟังก์ชันที่ใช้จัดการข้อความย่อย

ฟังก์ชันที่ใช้จัดการข้อความย่อยที่น่าสนใจมีรายละเอียด ดังนี้

1) ฟังก์ชัน strstr ( ) เป็นฟังก์ชันที่ใช้สำหรับค้นหาและเลือกเอาเฉพาะข้อความที่ค้นพบเป็นต้นไป

รูปแบบ

```
string strstr ( string $haystack , mixed $needle [, bool $before_needle = false ] )
```

เมื่อ $haystack หมายถึง ข้อความต้นฉบับ

$needle หมายถึง ระบุข้อความที่ต้องการ

$before_needle หมายถึง โดยปริยายจะมีค่าเป็น false คือ เอาเฉพาะข้อความที่ระบุตั้งแต่คำที่พบ $needle เป็นต้นไป แต่ถ้าระบุเป็น true จะเอาเฉพาะข้อความทั้งหมดในด้านซ้ายก่อนข้อความที่ระบุ

ตัวอย่างที่ 7.11 การใช้ฟังก์ชัน strstr ( )

```
<?php
        $email  = 'name@example.com';
        $domain = strstr ($email, '@');
        echo $domain;          // ผลลัพธ์ คือ @example.com
        $user = strstr ($email, '@', true);
        echo $user;            // ผลลัพธ์ คือ name
?>
```

2) ฟังก์ชัน substr ( ) เป็นฟังก์ชันที่ใช้สำหรับคัดเอาข้อความย่อย โดยกำหนดเอาตำแหน่งเริ่มต้น และความยาวที่ต้องการ

รูปแบบ

```
string substr ( string $string , int $start [, int $length ] )
```

เมื่อ $string หมายถึง ข้อความต้นฉบับ

$start หมายถึง ตำแหน่งเริ่มต้น เช่น ข้อความ 'abcdef' ตำแหน่งของตัวอักษรจะเริ่มต้นที่ 0 คือ 'a' ตำแหน่งของตัวอักษรตำแหน่งที่ 2 คือ 'c' และตำแหน่งอื่นๆ ตามลำดับ

$length หมายถึง ความยาว

ตัวอย่างที่ 7.12 การใช้ฟังก์ชัน substr ( )

```
<?php
    $rest = substr ("abcdef", 0, -1);    // ผลลัพธ์ คือ "abcde"
    $rest = substr ("abcdef", 2, -1);    // ผลลัพธ์ คือ "cde"
    $rest = substr ("abcdef", 4, -4);    // ผลลัพธ์ คือ false ไม่สามารถคัดเอาข้อความได้
    $rest = substr ("abcdef", -3, -1);   // ผลลัพธ์ คือ "de"
?>
```

3) ฟังก์ชัน substr_count ( ) เป็นฟังก์ชันที่ใช้สำหรับนับจำนวนคำที่มีอยู่ในข้อความ โดยรูปแบบของการค้นหาคำที่ตรงกัน (case-sensitive)

รูปแบบ

```
int substr_count (string $haystack , string $needle [, int $offset = 0 [, int $length ]])
```

เมื่อ $haystack หมายถึง ข้อความที่ต้องการค้นหาเพื่อนับจำนวน

$needle หมายถึง คำที่ใช้สำหรับการค้นหา

$offset หมายถึง ตำแหน่งของตัวอักษรที่จะเริ่มนับ

$length หมายถึง ความยาว

ตัวอย่างที่ 7.13 การใช้ฟังก์ชัน substr_count ( )

```
<?php
    $text = 'This is a test';
    echo strlen ($text);                    // ผลลัพธ์ ของจำนวนตัวอักษร คือ 14
    echo substr_count ($text, 'is');        // ผลลัพธ์ ของ 'is' คือ 2
    echo substr_count ($text, 'is', 3);     // การนับจะเริ่มต้นที่ 'is a test' ผลลัพธ์ คือ 1
    // ไม่พบคำที่ต้องการนับเพราะตำแหน่งนับ คือ 's i' ผลลัพธ์ คือ 0
    echo substr_count ($text, 'is', 3, 3);
    // ไม่สามารถนับได้และแสดงข้อความเตือน เพราะ 5+10 > 14 (ยาวกว่าข้อความ)
    echo substr_count($text, 'is', 5, 10);
?>
```

7.1.6 ฟังก์ชันค้นหาข้อความ

ฟังก์ชันค้นหาข้อความ มีดังนี้

1) ฟังก์ชัน strpos ( ) เป็นฟังก์ชันที่ใช้สำหรับค้นหาตำแหน่งของข้อความย่อย ค่าที่คืนกลับมาจะเป็นตำแหน่งที่ฟังก์ชันค้นพบข้อความย่อยเป็นครั้งแรก แต่หากไม่พบจะคืนค่า null กลับมา (ตัวพิมพ์เล็กและพิมพ์ใหญ่ถือว่าเป็นคนละตัวกัน: case-sensitive)

รูปแบบ

```
int strpos ( string $haystack , mixed $needle [, int $offset = 0 ] )
```

เมื่อ $haystack หมายถึง ข้อความที่ต้องการค้นหา

$needle หมายถึง คำที่ใช้สำหรับการค้นหา

$offset หมายถึง ตำแหน่งของตัวอักษรที่จะเริ่มค้นหา

ตัวอย่างที่ 7.14 การใช้ฟังก์ชัน strpos ( )

```
<?php
    $findme = "ya";
    $mystring = "parinya";
    $pos = strpos ($mystring, $findme);
    echo "Finding is $findme in $mystring ";
    echo "at $pos";
?>
```

2) ฟังก์ชัน stripos ( ) เป็นฟังก์ชันที่ใช้สำหรับค้นหาตำแหน่งของข้อความย่อย เหมือนฟังก์ชัน strpos ( ) ค่าที่คืนกลับมาจะเป็นตำแหน่งที่ฟังก์ชันค้นพบย่อยเป็นครั้งแรก แต่หากไม่พบจะคืนค่า null กลับมา (ตัวพิมพ์เล็กและพิมพ์ใหญ่ถือว่าเป็นตัวเดียวกัน)

รูปแบบ

```
int stripos ( string $haystack , string $needle [, int $offset = 0 ] )
```

เมื่อ $haystack หมายถึง ข้อความที่ต้องการค้นหา
$needle หมายถึง คำที่ใช้สำหรับการค้นหา
$offset หมายถึง ตำแหน่งของตัวอักษรที่จะเริ่มค้นหา

ตัวอย่างที่ 7.15 การใช้ฟังก์ชัน stripos ( )

```
1   <?php
2       $findme = "ya";
3       $mystring = "parinya";
4       $pos = stripos ($mystring, $findme);
5       echo "Finding is $findme in $mystring ";
6       echo "at $pos";
7   ?>
```

ผลลัพธ์ ของตัวอย่างที่ 7.14 และ 7.15 ดังนี้

```
Finding is ya in parinya at 5
```

7.1.7 ฟังก์ชันแทนที่ข้อความ

ฟังก์ชันแทนที่ข้อความ มีรายละเอียด ดังนี้

1) ฟังก์ชัน str_replace ( ) เป็นฟังก์ชันที่ใช้สำหรับค้นหา และแทนที่ข้อความย่อยด้วยข้อความใหม่ที่ต้องการ หากข้อความที่ต้องการค้นหามีมากกว่า 1 ครั้ง ก็จะถูกแทนที่ทั้งหมด (ตัวพิมพ์เล็กและพิมพ์ใหญ่ถือว่าเป็นตัวเดียวกัน)

รูปแบบ

```
mixed str_replace ( mixed $search , mixed $replace , mixed $subject [, int &$count ] )
```

เมื่อ $search หมายถึง ข้อความที่ต้องการค้นหา
$replace หมายถึง ข้อความที่ต้องการแทนที่
$subject หมายถึง ข้อความ หรืออาร์เรย์ ที่ใช้สำหรับการค้นหาและแทนที่ (ไม่ต้องระบุก็ได้)
$count หมายถึง จำนวนครั้งที่ต้องการให้แทนที่

ตัวอย่างที่ 7.16 การใช้ฟังก์ชัน str_replace ( )

```
1   <?php
2       $vowels = array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U");
3       echo str_replace ($vowels, "", "Hello World of PHP");
4       $phrase = "You should eat fruits, vegetables, and fiber every day.";
5       $healthy = array ("fruits", "vegetables", "fiber");
6       $yummy = array ("pizza", "beer", "ice cream");
7       echo str_replace ($healthy, $yummy, $phrase);
8   ?>
```

ผลลัพธ์

```
You should eat pizza, beer, and ice cream every day
```

จากตัวอย่างที่ 7.16 การใช้ฟังก์ชัน str_replace ( ) สำหรับค้นหาและแทนที่ข้อความย่อยเดิม ด้วยข้อความย่อยใหม่ อธิบายเพิ่มเติม ดังนี้

บรรทัดที่ 1 เริ่มต้นสคริปต์ PHP

บรรทัดที่ 2 กำหนดค่าให้กับตัวแปร $vowels มีค่าเท่ากับ array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U")

บรรทัดที่ 3 แสดงผลการเรียกใช้ฟังก์ชัน str_replace ( ) ค้นหาและแทนที่ข้อความย่อยเดิม ด้วยข้อความย่อยใหม่ จากตัวอย่างกำหนดเป็นค่าว่าง ดังนั้นผลลัพธ์ที่จะแสดงผล คือ Hll Wrld f PHP

บรรทัดที่ 4 กำหนดค่าให้กับตัวแปร $phrase มีค่าเท่ากับข้อความ "You should eat fruits, vegetables, and fiber every day."

บรรทัดที่ 5 กำหนดค่าให้กับตัวแปร $healthy มีค่าเท่ากับ array ("fruits", "vegetables", "fiber")

บรรทัดที่ 6 กำหนดค่าให้กับตัวแปร $yummy มีค่าเท่ากับ array ("pizza", "beer", "ice cream")

บรรทัดที่ 7 แสดงผลการเรียกใช้ฟังก์ชัน str_replace ( ) ค้นหาและแทนที่ข้อความย่อยเดิม ด้วยข้อความย่อยใหม่ ผลลัพธ์ที่จะแสดงผล คือ You should eat pizza, beer, and ice cream every day.

บรรทัดที่ 8 สิ้นสุดสคริปต์ PHP

2) ฟังก์ชัน str_ireplace ( ) เป็นฟังก์ชันที่ใช้สำหรับค้นหาและแทนที่ข้อความย่อยเดิมด้วยข้อความใหม่ที่ต้องการ หากข้อความย่อยที่ต้องการค้นหามีมากกว่า 1 ครั้ง เหมือนกับฟังก์ชัน str_replace ( ) เพียงแต่จะไม่สนใจความแตกต่างของตัวพิมพ์ (ignore case)

รูปแบบ

```
mixed str_ireplace ( mixed $search , mixed $replace , mixed $subject [, int &$count ] )
```

เมื่อ $search หมายถึง ข้อความที่ต้องการค้นหา

$replace หมายถึง ข้อความที่ต้องการแทนที่

$subject หมายถึง ข้อความ หรืออาร์เรย์ ที่ใช้สำหรับการค้นหาและแทนที่ (ไม่ต้องระบุก็ได้)

$count หมายถึง จำนวนครั้งที่ต้องการให้แทนที่

3) ฟังก์ชัน strtr ( ) เป็นฟังก์ชันที่ใช้สำหรับการแปลตัวอักษรหรือแทนที่ข้อความ และสามารถกำหนดข้อความย่อยในการแทนที่ได้มากกว่า 1 คำ โดยอาร์เรย์ของข้อความที่จะใช้แทนที่จะต้องกำหนดในรูปแบบของคีย์และค่าอาร์เรย์ โดยใช้คีย์อาร์เรย์เป็นสิ่งที่ต้องการค้นหา และค่าอาร์เรย์เป็นสิ่งที่จะนำไปแทนที่ รูปแบบดังนี้

รูปแบบ

```
string strtr ( string $str , array $replace_pairs )
```

เมื่อ $str หมายถึง ข้อความที่ต้องการแปล

$from หมายถึง ข้อความที่ต้องการค้นหา

$to หมายถึง ข้อความที่ต้องการแทนที่

$replace_pairs หมายถึง พารามิเตอร์ ใช้ระบุข้อความที่ต้องการค้นหาและแทนที่ในกรณีของอาร์เรย์ array ('from' => 'to', ...) ในในรูปแบบของคีย์และค่าอาร์เรย์

ตัวอย่างที่ 7.17 การใช้งานฟังก์ชัน strtr ( )

```
1  <?php
2      $trans = array ("hello" => "hi", "hi" => "hello");
3      echo strtr ("hi all, I said hello", $trans);
4  ?>
```

ผลลัพธ์

```
hello all, I said hi
```

7.1.8 ฟังก์ชันที่ใช้สำหรับตัดช่องว่างและเติมข้อความ

1) ฟังชันที่ใช้สำหรับตัดช่องว่าง ประกอบไปด้วย 1) ltrim ( ) 2) rtrim ( ) และ 3) trim ( ) โดยทั้ง 3 ฟังก์ชันมีรูปแบบการใช้งานเหมือนกัน มีรายละเอียดดังนี้

ตารางที่ 7.2 ฟังก์ชันในการตัดช่องว่าง

ชื่อฟังก์ชัน	หน้าที่
ltrim ( )	ตัดช่องว่างที่อยู่ด้านซ้ายของข้อความออกทั้งหมด
rtrim ( )	ตัดช่องว่างที่อยู่ด้านขวาของข้อความออกทั้งหมด
trim ( )	ตัดช่องว่างทั้งด้านซ้าย และด้านขวาของข้อความออกทั้งหมด

ตัวอย่างที่ 7.18 การประยุกต์ใช้ฟังก์ชันที่ใช้สำหรับตัดช่องว่าง

```
1    <?php
2        $str = "      PHP      ";
3        echo strlen ($str)."<br>";          // ผลลัพธ์ของการนับ คือ 19
4        echo strlen (ltrim ($str))."<br>" ; // ผลลัพธ์ของการนับ คือ 11
5        echo strlen (rtrim ($str))."<br>" ; // ผลลัพธ์ของการนับ คือ 11
6        echo strlen (trim ($str)) ;         // ผลลัพธ์ของการนับ คือ 3
7    ?>
```

จากตัวอย่างจะใช้วิธีการทดสอบโดยการนับจำนวนเนื่องจากช่องว่าง (whitespace) จะไม่สามารถมองเห็นได้ แต่จะส่งผลเมื่อใช้สำหรับการค้นหาหรือเปรียบเทียบข้อความ เพราะช่องว่างเทียบได้กับอักษร 1 ตัว

2) ฟังก์ชัน str_pad ( ) เป็นฟังก์ชันที่ใช้สำหรับขยายความยาวของข้อความหรือเติมอักษรหรือข้อความลงในข้อความเดิม

รูปแบบ

```
string str_pad ( string $input , int $pad_length [, string $pad_string = " " [, int $pad_type = STR_PAD_RIGHT ]] )
```

เมื่อ $input หมายถึง ข้อความนำเข้า

$pad_length หมายถึง ความยาวที่ต้องการขยายเพิ่มเติม (ถ้าค่าตัวเลขน้อยกว่าหรือเท่ากับจำนวนความกว้างของข้อความใน $input จะไม่มีการขยาย)

$pad_string หมายถึง ตัวอักษรหรือข้อความอื่นๆ ที่จะเติมลงไป (ไม่ระบุก็ได้)

$pad_type หมายถึง ด้านที่ต้องการขยายเพิ่มเติม โดยปริยายจะเติมด้านขวา มีรายละเอียด ดังนี้ STR_PAD_RIGHT (ขยายเพิ่มด้านขวา), STR_PAD_LEFT (ขยายเพิ่มด้านซ้าย) หรือ STR_PAD_BOTH (ขยายทั้งซ้ายและขวาเท่าๆ กัน)

ตัวอย่างที่ 7.19 การใช้ฟังก์ชัน str_pad ( )

```
1   <?php
2       $input = "Alien";
3       echo str_pad ($input, 10);                      // ผลลัพธ์ คือ "Alien     "
4       echo str_pad ($input, 10, "-=", STR_PAD_LEFT);  // ผลลัพธ์ คือ "-=-=-Alien"
5       echo str_pad ($input, 10, "_", STR_PAD_BOTH);   // ผลลัพธ์ คือ "__Alien___"
6       echo str_pad ($input, 6 , "___");               // ผลลัพธ์ คือ "Alien_"
7   ?>
```

7.1.9 ฟังก์ชันที่ใช้จัดการอักษรพิเศษของ HTML

ฟังก์ชันที่ใช้จัดการอักษรพิเศษของ HTML มีรายละเอียด ดังนี้

1) ฟังก์ชัน nl2br ( ) เป็นฟังก์ชันสำหรับเปลี่ยนอักษรพิเศษ เช่น \r\n, \n\r, \n และ \r เป็นคำสั่งที่ใช้สำหรับขึ้นบรรทัดใหม่ ให้เปลี่ยนเป็นแท็ก
 ของภาษา HTML มีความจำเป็นในกรณีที่ต้องการให้ผู้ใช้ป้อนข้อมูลผ่านทางจอภาพ เช่น ป้อนข้อมูลผ่านกล่องข้อความ (Text Field หรือ Text Box) หลายๆ บรรทัด หรือโดยปกติแล้วเมื่อกดปุ่ม Enter เพื่อขึ้นบรรทัดใหม่ ตำแหน่งที่กดปุ่ม Enter ก็จะถูกบันทึกด้วยสัญลักษณ์ \n หรือ new line เพื่อเป็นสัญลักษณ์บ่งชี้ว่าเป็นตำแหน่งที่จะต้องขึ้นบรรทัดใหม่ แม้จะมองไม่เห็นสัญลักษณ์นี้ก็ตาม แต่ในเอกสาร HTML สัญลักษณ์ \n จะไม่ทำให้ขึ้นบรรทัดใหม่ เพราะในเอกสาร HTML จะขึ้นบรรทัดใหม่ด้วยสัญลักษณ์
 เท่านั้น ดังนั้นฟังก์ชัน nl2br ( ) จึงใช้ในการเปลี่ยนจากสัญลักษณ์ \n ให้เป็น

รูปแบบ

```
string nl2br ( string $string [, bool $is_xhtml = true ] )
```

เมื่อ $string หมายถึง ข้อความ

$is_xhtml หมายถึง จะใช้รูปแบบ xhtml หรือไม่ (ค่าโดยปริยายจะกำหนดเป็น true)

ตัวอย่างที่ 7.20 การใช้ฟังก์ชัน nl2br ( )

```
1   <?php
2       echo nl2br ("สวัสดีครับ\nwww.sru.ac.th");
3   ?>
```

ผลลัพธ์

```
สวัสดีครับ
www.sru.ac.th
```

2) ฟังก์ชัน wordwrap ( ) เป็นฟังก์ชันสำหรับแบ่งข้อความตามจำนวนตัวอักษรที่กำหนด โดยจะไม่ตัดคำหรือฉีกข้อความ ใช้แยกคำจากการเว้นวรรค ภายหลังจากการแบ่งข้อความแล้วจะดำเนินการอย่างไรก็แล้วแต่กำหนด

รูปแบบ

```
string wordwrap (string $str [, int $width = 75 [, string $break = "\n" [, bool $cut = false ]]])
```

เมื่อ $str หมายถึง ตัวแปรที่ใช้เก็บค่าข้อความ

$width หมายถึง จำนวนตัวอักษรที่ต้องการ (โดยปริยาย คือ 75)

$break หมายถึง คำสั่งที่ต้องการให้ดำเนินการ (โดยปริยาย คือ \n เพื่อขึ้นบรรทัดใหม่)

$cut หมายถึง ต้องการตัดคำหรือไม่ หากกำหนดเป็น true จะตัดคำที่มีความยาวเกินกว่าที่ $width กำหนด

ตัวอย่างที่ 7.21 การใช้ฟังก์ชัน wordwrap ( )

```
1  <?php
2      $text = "The quick brown fox jumped over the lazy dog.";
3      $newtext = wordwrap ($text, 20, "<br>\n");
4      echo $newtext;
5  ?>
```

ผลลัพธ์

```
The quick brown fox
jumped over the lazy
dog.
```

7.2 ฟังก์ชันที่ใช้จัดการจำนวนและตัวเลข

ฟังก์ชันที่ใช้จัดการจำนวนและตัวเลข ในภาษา PHP มีฟังก์ชันที่ใช้จัดการข้อมูลชนิดนี้อยู่หลายฟังก์ชัน แต่สำหรับในบทที่ จะแนะนำเฉพาะฟังก์ชันที่นิยมใช้งาน มีรายละเอียด ดังนี้

7.2.1 ฟังก์ชันประมาณค่าจำนวนและตัวเลข

ฟังก์ชันประมาณค่าจำนวนและตัวเลข มีดังนี้

ตารางที่ 7.3 ฟังก์ชันประมาณค่า

ceil (จำนวน)	ใช้ในการปัดเศษขึ้นไปเป็นเลขจำนวนเต็มตัวถัดไปถ้าเศษมีค่ามากกว่า 0 เช่น	
	echo ceil (10.01);	//ผลลัพธ์ คือ 11
	echo ceil (10.50);	//ผลลัพธ์ คือ 11

ตารางที่ 7.3 (ต่อ)

floor (จำนวน)	ใช้ในการตัดเศษทิ้ง ไม่ว่าเศษนั้นจะมีค่าเท่าใดก็ตาม เช่น
	echo floor (10.01); //ผลลัพธ์ คือ 10
	echo floor (10.50); //ผลลัพธ์ คือ 10

round (จำนวน, [, ทศนิยม])	ใช้ในการประมาณค่าเป็นจำนวนเต็มที่ใกล้เคียง หากเศษน้อยกว่า 0.5 จะตัดเศษทิ้ง แต่หากเศษมีค่าตั้งแต่ 0.5 จะปัดขึ้นไปเป็นจำนวนเต็มถัดไป เช่น
	echo round (10.49); //ผลลัพธ์ คือ 10
	echo round (10.50); //ผลลัพธ์ คือ 11
	ส่วนทศนิยมเป็นการกำหนดว่าจะให้มีทศนิยมกี่ตำแหน่ง

intval (จำนวน)	เลือกเอาเฉพาะจำนวนเต็มของจำนวนที่ระบุ หากมีทศนิยมจะตัดทิ้งเช่น
	echo intval (10.01); //ผลลัพธ์ คือ 10
	echo intval (10.99); //ผลลัพธ์ คือ 10
	echo intval (-1.23); //ผลลัพธ์ คือ -1

floatval (จำนวน)	เลือกเอาจำนวนที่ระบุในแบบ float ฟังก์ชันนี้น่าจะมีประโยชน์ในกรณีที่จำนวนนั้นวางอยู่ข้างหน้าข้อความ เช่น echo floatval (“12.34 MB”); //ผลลัพธ์ คือ 12.34

7.2.2 ฟังก์ชันเปรียบเทียบจำนวน

ฟังก์ชันเปรียบเทียบจำนวน มีรายละเอียด ดังนี้

ตารางที่ 7.4 ฟังก์ชันเปรียบเทียบจำนวน

min (n1, n2, ...n) หรือ min (อาร์เรย์)	หาค่าที่น้อยที่สุดของช่วงที่กำหนด	
	$m1 = min (6, 7, 3, 8, 9);	//ผลลัพธ์ คือ $m1 = 3
	$m2 = min (array (10, 10.5, 3.4, 5.2, 20));	//ผลลัพธ์ คือ $m2 = 3.4
max (n1, n2, ...n) หรือ max (อาร์เรย์)	หาค่าที่มากที่สุดของช่วงที่กำหนด	
	$m1 = max (6, 7, 3, 8, 9);	//ผลลัพธ์ คือ $m1 = 9
	$m2 = max (array (10, 10.5, 3.4, 5.2, 20));	//ผลลัพธ์ คือ $m2 = 20

7.2.3 ฟังก์ชันการตรวจสอบ และจัดรูปแบบตัวเลข

ฟังก์ชันที่น่าสนใจเกี่ยวกับการตรวจสอบ และจัดรูปแบบตัวเลข มีดังนี้

ตารางที่ 7.5 ฟังก์ชันการตรวจสอบ และจัดรูปแบบตัวเลข

number_format (จำนวน)	ใช้ในการจัดรูปแบบตัวเลข เช่น จาก 1234 เป็น 1,234 เป็นต้น เช่น echo number_format (1234567); // 1,234,567

ตารางที่ 7.5 (ต่อ)

is_numeric (ข้อมูล)	ตรวจสอบว่าข้อมูลที่ระบุนั้นเป็นตัวเลขหรือไม่ โดยจะคืนค่า true ถ้าเป็นข้อมูลตัวเลขที่ไม่มีอักษรอื่นๆปะปนอยู่ด้วย ทั้งนี้ตัวเลขที่เขียนในแบบข้อความ เช่น “123” ก็ถือว่าเป็นตัวเลขด้วยเพราะสามารถนำไปใช้ คำนวณได้ เช่น
	$a = is_numeric(123); //ผลลัพธ์ คือ $a = ture
	$b = is_numeric(“1.23”); //ผลลัพธ์ คือ $b = ture
	$c = is_numeric(“123abc”); //ผลลัพธ์ คือ $c = false

is_int (ข้อมูล)	ตรวจสอบว่าเป็นจำนวนเต็มหรือไม่ หากใช่จะคืนค่า true
is_float (ข้อมูล)	ตรวจสอบว่าเป็นจำนวนจริงจริงหรือไม่ หากใช่จะคืนค่า true

7.2.4 ฟังก์ชันอื่นๆ ที่เกี่ยวข้องกับจำนวนและตัวเลข

ฟังก์ชันอื่นๆ ที่เกี่ยวข้องกับจำนวนและตัวเลข มีรายละเอียด ดังนี้

ตารางที่ 7.6 ฟังก์ชันอื่นๆ ที่เกี่ยวข้องกับตัวเลข

abs (จำนวน)	ใช้สำหรับหาค่า absolute เช่น
	$x = abs(10); //ผลลัพธ์ คือ $x = 10
	$y = abs (-10); //ผลลัพธ์ คือ $y = 10

rand ( ) หรือ rand (ค่าน้อยที่สุด, ค่ามากที่สุด)	ใช้ในการสร้างเลขสุ่ม โดยค่าที่ได้จะไม่แน่นอนในแต่ละครั้ง โดยปกติแล้วเลขสุ่มที่ได้จะเป็นเลขจำนวนเต็มที่มีค่าระหว่าง 0-32,768 แต่อย่างไรก็ตาม PHP อนุญาตให้สามารถกำหนดช่วงตัวเลขผลลัพธ์ที่ต้องการได้ว่าให้อยู่ระหว่างค่าใดถึงค่าใด เช่น $r1 = rand ( ); $r2 = rand (10, 20); //เลขสุ่มที่ได้จะมีค่าระหว่าง 10-20
pow (เลขฐาน, เลขชี้กำลัง)	ใช้ในการค่าเลขยกกำลัง เช่น echo pow (10, 2); //100
sqrt (ตัวเลข)	หาค่ารากที่สองของจำนวนที่ระบุ แต่ต้องไม่ใช่จำนวนที่ติดลบ เช่น $a = sqrt (100); //$a =10

7.3 ฟังก์ชันที่ใช้จัดการวันและเวลา

ข้อมูลเกี่ยวกับวันและเวลา นับว่าเป็นข้อมูลที่สำคัญอีกอย่างหนึ่งของการเขียนโปรแกรมแบบสคริปต์ การใช้งานส่วนใหญ่จะทำผ่านฟังก์ชันต่างๆ ที่เกี่ยวข้อง รูปแบบของวันที่และเวลาสามารถแบ่งเป็น 2 ประเภท คือ 1) Local Mean Time (LMT) คือ เวลาเฉลี่ยของแต่ละท้องถิ่น และ 2) Greenwich Mean Time (GMT) คือ เวลามาตรฐานสากล เป็นเวลาที่เทียบกับเมืองกรีนิช ประเทศอังกฤษ โดยหน่วยเวลาอย่างเป็นทางการ เรียกว่า UTC ฟังก์ชันที่ใช้จัดการวันและเวลา มีดังนี้

7.3.1 ฟังก์ชัน date_default_timezone_set ( )

ฟังก์ชัน date_default_ timezone_set ( ) ใช้สำหรับกำหนดรูปแบบ Timezone ที่ต้องการใช้ และจะแสดงวันและเวลาตาม Timezone ที่กำหนด รูปแบบการใช้ฟังก์ชัน มีดังนี้

รูปแบบ

```
bool date_default_timezone_set ( string $timezone_identifier )
```

เมื่อ $timezone_identifier หมายถึง เขตเวลา (Timezone) สำหรับเวลาในประเทศไทย ต้องระบุค่าเป็น "Asia/Bangkok" เป็นเวลา LMT สำหรับเวลา GMT สามารถระบุค่าเป็น GMT หรือ UTC

7.3.2 ฟังก์ชัน time ( ) เป็นฟังก์ชันที่ใช้แสดงวันที่และเวลาปัจจุบันในรูปแบบของ Unix Timestamp จะแสดงเวลาเป็นหน่วยวินาที โดยเริ่มนับค่าตั้งแต่วันที่ 1 มกราคม ค.ศ.1970 เวลา 00:00:00 น. ตามเวลา GMT จนถึงวันที่และเวลาปัจจุบัน ยกตัวอย่างเช่น วันที่ 26 พฤศจิกายน 2012 เวลา 13:47:00 มีค่า Unix Timestamp คือ "1353912424"

รูปแบบ

```
int time ( void )
```

ตัวอย่างที่ 7.22 การใช้ฟังก์ชัน time ( )

```
1   <?php
2       $nextWeek = time ( ) + (7 * 24 * 60 * 60);
3       echo "Now: ". date ("Y-m-d") ."<br>";
4       echo "Next Week: ". date ("Y-m-d", $nextWeek) ."<br>";
5       echo "Next Week: ". date ("Y-m-d", strtotime("+1 week")) ."<br>";
6   ?>
```

ผลลัพธ์

```
Now: 2012-11-26
Next Week: 2012-12-03
Next Week: 2012-12-03
```

จากตัวอย่างที่ 7.22 การใช้ฟังก์ชัน การใช้ฟังก์ชัน time ( ) สำหรับคำนวณหาวัน อธิบายดังนี้

บรรทัดที่ 1 เริ่มต้นสคริปต์ PHP

บรรทัดที่ 2 กำหนดค่าให้กับตัวแปร $nextWeek มีค่าเท่ากับ time ( ) บวกด้วยเวลาใน 1 สัปดาห์ คือ 7 วัน คูณด้วย 24 ชั่วโมง คูณด้วย 60 นาที คูณด้วย 60 วินาที

บรรทัดที่ 3 แสดงผลวันที่ปัจจุบัน โดยใช้ฟังก์ชัน date ( )

บรรทัดที่ 4 แสดงผลวันที่ในสัปดาห์ถัดไป โดยใช้ฟังก์ชัน date ( )

บรรทัดที่ 5 แสดงผลวันที่ในสัปดาห์ถัดไป โดยใช้ฟังก์ชัน date ( ) ร่วมกับตัวแปรฟังก์ชันแปลงข้อความเป็นเวลา (มีค่าเท่ากัน เหมือนในบรรทัดที่ 4)

บรรทัดที่ 6 สิ้นสุดสคริปต์ PHP

7.3.3 ฟังก์ชัน strtotime ( ) เป็นฟังก์ชันที่ใช้สำหรับแปลงข้อความให้เป็นข้อมูลวันเวลาในรูปแบบ timestamp แต่ข้อความนั้นต้องเขียนในรูปแบบที่สามารถแปลงได้ ถ้าแปลงไม่ได้จะคืนค่า false

รูปแบบ

```
int strtotime ( string $time [, int $now = time() ] )
```

เมื่อ $time หมายถึง วันและเวลาในรูปแบบข้อความ

$now หมายถึง วันและเวลาปัจจุบันที่เรียกใช้ฟังก์ชัน

ตัวอย่างที่ 7.23 การใช้ฟังก์ชัน strtotime ( ) สำหรับแปลงข้อความให้เป็นข้อมูลวันและเวลา

```
1   <?php
2       echo strtotime ("now"), "<br>";
3       echo strtotime ("10 September 2012"), "<br>";
4       echo strtotime ("+1 day"), "<br>";
5       echo strtotime ("+1 week"), "<br>";
6   ?>
```

ผลลัพธ์

```
1432783796
1347206400
1432870196
1433388596
```

จากตัวอย่างที่ 7.23 การใช้ฟังก์ชัน strtotime ( ) สำหรับแปลงข้อความให้เป็นข้อมูลวันและเวลา ในรูปแบบ timestamp นั้น ค่าผลลัพธ์ที่ได้ จะยากแก่การทำความเข้าใจ ดังนั้นหากต้องการเปลี่ยนรูปแบบ timestamp เป็นวันที่จะต้องใช้ร่วมกับฟังก์ชัน date ( ) ตัวอย่างดังนี้

ตัวอย่างที่ 7.24 การใช้ฟังก์ชัน strtotime ( ) ร่วมกับฟังก์ชัน date ( )

```
1   <?php
2       echo date("Y-m-d", strtotime ("now")), "<br>";
3       echo date("Y-m-d", strtotime ("10 September 2012")), "<br>";
4       echo date("Y-m-d", strtotime ("+1 day")), "<br>";
5       echo date("Y-m-d", strtotime ("+1 week")), "<br>";
6   ?>
```

ผลลัพธ์

```
2012-05-28
2012-09-10
2012-05-29
2012-06-04
```

จากตัวอย่างที่ 7.24 การใช้ฟังก์ชัน strtotime ( ) ร่วมกับฟังก์ชัน date ( ) เพื่อแสดงผลในรูปแบบวันที่ที่ง่ายแก่การทำความเข้าใจ

7.3.4 ฟังก์ชัน date ( ) เป็นฟังก์ชันที่ใช้สำหรับแสดงวันที่และเวลา โดยสามารถกำหนดรูปแบบของผลลัพธ์ของฟังก์ชันได้ เช่น ต้องการเฉพาะวันที่ หรือต้องการเฉพาะเวลา รูปแบบคำสั่งมีดังนี้

รูปแบบ

```
string date ( string $format [, int $timestamp = time() ] )
```

เมื่อ $format หมายถึง รูปแบบวันและเวลา

$timestamp หมายถึง พารามิเตอร์เสริมชนิดตัวเลขจำนวนเต็ม ของเวลาปัจจุบันบนระบบ Unix ถ้าไม่มีการกำหนดค่าใดๆ ค่าโดยปริยาย คือ ค่าของฟังก์ชัน time ( ) หรือวันและเวลาปัจจุบัน

อักษรต่างๆ ที่ใช้กำหนดรูปแบบวันและเวลาสำหรับฟังก์ชัน date ( ) ส่งผ่านพารามิเตอร์ $format สามารถแยกออกเป็นกลุ่มได้ดังนี้

1) อักษรที่ใช้แทน วันและสัปดาห์

ตารางที่ 7.7 อักษรที่ใช้แทนความหมายของวันและสัปดาห์ สำหรับฟังก์ชัน date ( )

อักษร	ความหมาย
d	ใช้แทนวันที่ของเดือน โดยเป็นเลข 2 หลัก เช่น 01, 02, 31
j	ใช้แทนวันที่ของเดือน แบบไม่มี 0 นำหน้าเลขหลักเดียว เช่น 1, 2, 30
D	ใช้แทนวันในรอบสัปดาห์แบบย่อ เช่น Sun, Mon
l	ตัวแอลพิมพ์เล็กใช้แทนวันในรอบสัปดาห์แบบชื่อเต็ม เช่น Sunday, Monday
S	ตัวS(พิมพ์ใหญ่) เป็นตัวอักษรย่อที่ใช้บ่งบอกลำดับที่ของวันในรอบเดือน เช่น st, nd, rd, th โดยปกติจะใช้ต่อท้ายตัว j หรือ d (ใช้แทนวันที่) เช่น jS ลักษณะผลลัพธ์คือ 1st
w	ใช้แทนลำดับของวันในรอบสัปดาห์จาก 0-6
z	แทนลำดับของวันในรอบปีจาก 0-365
W	ลำดับของสัปดาห์ในรอบปี

2) อักษรที่ใช้แทนความหมายเดือน สำหรับฟังก์ชัน date ( )

ตารางที่ 7.8 อักษรที่ใช้แทนความหมายของเดือน สำหรับฟังก์ชัน date ( )

อักษร	ความหมาย
F	ชื่อเดือนแบบเต็ม เช่น January, March
M	ลำดับที่ของเดือนในรอบปีโดยมี 0 นำหน้าเลขหลักเดียว เช่น 01, 02, 12
M	ชื่อของเดือนแบบย่อ เฉพาะ 3 ตัวแรก เช่น Jan, Feb
n	ลำดับที่ของเดือนแบบไม่มี 0 นำหน้าเลขหลักเดียว เช่น 1, 2, 12
t	จำนวนวันของเดือนนั้นๆ โดยมีค่าระหว่าง 28- 31

3) อักษรที่ใช้แทนความหมายของปี สำหรับฟังก์ชัน date ( )

ตารางที่ 7.9 อักษรที่ใช้แทนความหมายของปี สำหรับฟังก์ชัน date ( )

อักษร	ความหมาย
L	ตรวจสอบว่าเป็นปี Leap Year หรือไม่(เดือนกุมภาพันธ์มี 29 วัน) ถ้าใช่จะคืนค่า 1 ถ้าไม่ใช่จะคืนค่า 0
Y	ใช้แทนปี ค.ศ. แบบเลข 4 หลัก เช่น 1975
y	ใช้แทนปี ค.ศ. แบบเลข 2 หลัก เช่น 75

4) อักษรที่ใช้แทนความหมายของเวลา สำหรับฟังก์ชัน date ( )

ตารางที่ 7.10 อักษรที่ใช้แทนความหมายของเวลา สำหรับฟังก์ชัน date ( )

อักษร	ความหมาย
a	ใช้แทนค่า Ante Meridiem และ Post Meridiem แบบพิมพ์เล็ก ได้แก่ am หรือ pm
A	ใช้แทนค่า Ante Meridiem และ Post Meridiem แบบพิมพ์ใหญ่ ได้แก่ AM หรือ PM
g	ใช้แทนค่าชั่วโมงของวันแบบ 12 ชั่วโมง โดยไม่มี 0 นำหน้าเลขหลักเดียวระหว่าง 0-12
G	ใช้แทนค่าชั่วโมงของวันแบบ 24 ชั่วโมง โดยไม่มี 0 นำหน้าเลขหลักเดียวระหว่าง 0-23
h	ใช้แทนค่าชั่วโมงของวันแบบ 12 ชั่วโมง โดยไม่มี 0 นำหน้าเลขหลักเดียวระหว่าง 01-12
H	ใช้แทนค่าชั่วโมงของวันแบบ 24 ชั่วโมง โดยไม่มี 0 นำหน้าเลขหลักเดียวระหว่าง 00-23
i	ค่านาทีแบบมี 0 นำหน้าเลขหลักเดียวจาก 00-59
s	ค่าวินาทีแบบมี 0 นำหน้าเลขหลักเดียวจาก 00-59
O	ค่าความแตกต่างเมื่อเทียบกับเวลา Greenwich(GMT) เช่น +0700
r	เวลาตามมาตรฐานของ RFC 2822 เช่น Sun, 1 Mar 2009 15:03:57 +0700

ตัวอย่างที่ 7.25 การแสดงวันที่และเวลาในรูปแบบต่างๆ ด้วยฟังก์ชัน date ( ) แบบที่ 1

```
1    <?php
2        date_default_timezone_set('Asia/Bangkok');
```

```
3               echo "การแสดงวันที่และเวลาในรูปแบบต่างๆ ด้วยฟังก์ชัน date ( ) <br>";
4               echo date ("r") . "<br>";
5               echo date ('l jS \of F Y h:i:s A') . "<br>";
6               echo "July 1, 2012 is on a " . date ("l", mktime(0, 0, 0, 7, 1, 2012)) . "<br>";
7       ?>
```

ผลลัพธ์แบบที่ 1

```
การแสดงวันที่และเวลาในรูปแบบต่างๆ ด้วยฟังก์ชัน date ( )
Mon, 26 Nov 2012 16:36:34 +0700
Monday 26th of November 2012 04:36:34 PM
July 1, 2012 is on a Sunday
```

ตัวอย่างที่ 7.26 การแสดงวันและเวลาในรูปแบบต่างๆ ด้วยฟังก์ชัน date ( ) แบบที่ 2

```
1       <?php
2           date_default_timezone_set('Asia/Bangkok');
3           $birth = strtotime ("12/10/1978");
4           echo date ("ข้าพเจ้าเกิดเมื่อ  j-m-Y <br>", $birth);
5           $day = array("อาทิตย์", "จันทร์", "อังคาร", "พุธ", "พฤหัสบดี", "ศุกร์", "เสาร์");
6           $month =array("มกราคม", "กุมภาพันธ์", "มีนาคม", "เมษายน", "พฤษภาคม", "มิถุนายน",
"กรกฎาคม", "สิงหาคม" , "กันยายน" , "ตุลาคม" , "พฤศจิกายน" , "ธันวาคม");
7           $d = date ("w");
8           $day = $day [$d];
9           $date = date ("j");
10          $m = date ("m")-1;                  // เพราะลำดับเดือนจะมีค่าระหว่าง 1-12
11          $month = $months [$m];
12          $year = date ('Y') + 543;
13          echo "วันนี้ตรงกับวัน $day  วันที่ $date เดือน $month  ปี $year  ";
14          echo date("ขณะนี้เวลา H:i:s");
15      ?>
```

ผลลัพธ์แบบที่ 2

```
ข้าพเจ้าเกิดเมื่อ 10-12-1978
วันนี้ตรงกับวัน จันทร์ วันที่ 26 เดือน ปี 2555 ขณะนี้เวลา 16:43:48
```

7.3.5 ฟังก์ชัน getdate ( ) เป็นฟังก์ชันที่ใช้สำหรับแสดงวันที่และเวลาเช่นเดียวกับฟังก์ชัน date ( ) แต่ไม่สามารถกำหนดรูปแบบเองได้ และฟังก์ชันจะคืนค่าเป็นอาร์เรย์แบบคีย์และค่าอาร์เรย์ รูปแบบมีดังนี้

รูปแบบ

```
array getdate ([ int $timestamp = time() ] )
```

เมื่อ $timestamp หมายถึง พารามิเตอร์ชนิดตัวเลขจำนวนเต็ม ของเวลาปัจจุบันบนระบบ Unix ถ้าไม่มีการกำหนดค่าใดๆ ค่าโดยปริยาย คือ ค่าของฟังก์ชัน time ( ) หรือวันและเวลาปัจจุบัน

อาร์เรย์ที่ได้จากฟังก์ชัน getdate ( ) ส่งคืนค่า ประกอบด้วยสมาชิกของอาร์เรย์ ดังต่อไปนี้

ตารางที่ 7.11 คีย์และค่าของอาร์เรย์ที่ได้จากการใช้ฟังก์ชัน getdate ( )

คีย์	คำอธิบาย	ลักษณะค่าที่ได้
seconds	อ่านค่า “วินาที”	0-59
minutes	อ่านค่า “นาที”	0 to 59
hours	อ่านค่า “ชั่วโมง”	0 to 23
mday	อ่านค่า “วันที่”	1 to 31
wday	อ่านค่าลำดับของวันในรอบสัปดาห์	0 (อาทิตย์)-6 (เสาร์)
mon	อ่านค่าลำดับของเดือนในรอบปี	1-12
Year	อ่านค่า “ปี” แบบเลข 4 หลัก	เช่น 2010
yday	อ่านค่าลำดับของวันในรอบปี	0-365
weekday	อ่านค่า “ชื่อวัน” ของสัปดาห์	Sunday-Saturday
month	อ่านค่า “ชื่อเดือน” ของปี	January-December

ตัวอย่างที่ 7.27 การใช้ฟังก์ชัน getdate ( )

```
<?php
        $today = getdate ( );   print_r ($today);
?>
```

ผลลัพธ์

```
Array ( [seconds] => 30 [minutes] => 55 [hours] => 16 [mday] => 26 [wday] => 1 [mon] => 11 [year] => 2012 [yday] => 330 [weekday] => Monday [month] => November [0] => 1353923730 )
```

7.3.6 ฟังก์ชัน mktime ( ) เป็นฟังก์ชันที่ใช้สำหรับหาค่า Unix Timestamp ของวันที่และเวลา เช่นเดียวกับฟังก์ชัน time ( ) แต่ฟังก์ชัน time ( ) หาค่า Unix Timestamp ของวันที่และเวลาปัจจุบันเท่านั้น ส่วนฟังก์ชัน mktime ( ) สามารถกำหนดวันและเวลาที่ต้องการหาค่า Unix Timestamp ได้ รูปแบบคำสั่งมีดังนี้

รูปแบบ

```
int mktime ([ int $hour = date("H") [, int $minute = date("i") [, int $second = date("s") [, int $month = date("n") [, int $day = date("j") [, int $year = date("Y") [, int $is_dst = -1 ]]]]]] )
```

เมื่อ $hour หมายถึง ชั่วโมง

$minute หมายถึง นาที

$second หมายถึง วินาที

$month หมายถึง เดือน

$day หมายถึง วัน

$year หมายถึง ปี

$is_dst หมายถึง การกำหนดรูปแบบวันและเวลาตามฤดูกาล (กำหนดพารามิเตอร์เป็น 1) ไม่กำหนดเวลา (กำหนดพารามิเตอร์เป็น 0) หรือหากไม่ทราบสามารถกำหนดเป็น -1 หรือไม่กำหนด จะมีค่าปริยายเท่ากับ -1

ตัวอย่างที่ 7.28 การใช้ฟังก์ชัน mktime ( )

```
1  <?php
2      date_default_timezone_set ('Asia/Bangkok');
3      echo date("M-d-Y", mktime(0, 0, 0, 12, 32, 2012)) . "<br>";
4      echo date("M-d-Y", mktime(0, 0, 0, 13, 1, 2012)). "<br>";
5      echo date("M-d-Y", mktime(0, 0, 0, 1, 1, 2013)). "<br>";
6      echo date("M-d-Y", mktime(0, 0, 0, 1, 1, 13));
7  ?>
```

ผลลัพธ์

```
Jan-01-2013
Jan-01-2013
Jan-01-2013
Jan-01-2013
```

7.3.7 ฟังก์ชัน checkdate ( ) เป็นฟังก์ชันที่ใช้สำหรับตรวจสอบความถูกต้องของวันที่ เช่น วันที่ที่ถูกต้องจะต้องไม่เกิน 31 วัน หรือเดือนจะต้องเป็นตัวเลขที่อยู่ระหว่าง 1 ถึง 12 หากฟังก์ชันตรวจสอบแล้วพบว่าวันที่ไม่ถูกต้อง จะคืนค่าเป็นเท็จ (False) จากถูกต้องจะคืนค่าเป็นจริง (True) รูปแบบคำสั่งมีดังนี้

รูปแบบ

```
bool checkdate ( int $month , int $day , int $year )
```

เมื่อ $month หมายถึง จำนวนเดือนใน 1 ปี มีค่าอยู่ระหว่าง 1 - 12 เดือน

$day หมายถึง จำนวนวันของแต่ละเดือน

$year หมายถึง ปี มีค่าอยู่ระหว่าง 1 - 32767

ตัวอย่างที่ 7.29 การใช้ฟังก์ชัน checkdate ( ) สำหรับตรวจสอบความถูกต้องของวันที่

```
1   <?php
2       var_dump (checkdate (12, 31, 2012)); // เป็นจริงเพราะเดือนธันวาคมมี 31 วัน ในปี 2012
3       var_dump (checkdate (2, 29, 2013)); // เป็นเท็จเพราะเดือนกุมภาพันธ์ในปี 2013 มี 28 วัน
4   ?>
```

จากตัวอย่างที่ 7.29 การใช้ฟังก์ชัน checkdate ( ) สำหรับตรวจสอบความถูกต้องของวันที่ ในบรรทัดที่ 2 จะได้ผลเป็นจริง เพราะเดือนธันวาคม ในปี ค.ศ.2012 มี 31 วัน สำหรับผลของบรรทัดที่ 3 จะได้ผลเป็นเท็จ เพราะเดือนกุมภาพันธ์ ในปี ค.ศ.2012 มี 28 วัน

7.3.8 การจัดการกับวันที่และเวลาแบบ Greenwich Mean Time

1) ฟังก์ชัน gmdate ( ) เป็นฟังก์ชันที่ใช้แสดงวันที่และเวลาเหมือนกับฟังก์ชัน date ( ) แต่จะคืนค่าเป็นเวลา GMT โดยสามารถกำหนดรูปแบบวันที่และเวลาได้เหมือนกับฟังก์ชัน date ( ) มีรูปแบบการใช้งานฟังก์ชัน ดังนี้

รูปแบบ

```
string gmdate ( string $format [, int $timestamp = time() ] )
```

เมื่อ $format หมายถึง รูปแบบวันและเวลา

$timestamp หมายถึง พารามิเตอร์เสริมชนิดตัวเลขจำนวนเต็ม ของเวลาปัจจุบันบนระบบ Unix ถ้าไม่มีการกำหนดค่าใดๆ ค่าโดยปริยาย คือ ค่าของฟังก์ชัน time ( ) หรือวันและเวลาปัจจุบัน

ตัวอย่างที่ 7.30 การใช้ฟังก์ชัน gmdate ( )

```
1   <?php
2       echo date("M d Y H:i:s", mktime(0, 0, 0, 1, 1, 2012)) . "<br>";
3       echo gmdate("M d Y H:i:s", mktime(0, 0, 0, 1, 1, 2012));
4   ?>
```

ผลลัพธ์

```
Jan 01 2012 00:00:00
Dec 31 2011 17:00:00
```

2) ฟังก์ชัน gmmktime ( ) เป็นฟังก์ชันที่ใช้แสดงวันที่และเวลาเหมือนกับฟังก์ชัน mktime ( ) แต่จะคืนค่าเป็นเวลา GMT มีรูปแบบ ดังนี้

รูปแบบ

```
int gmmktime ([ int $hour = gmdate("H") [, int $minute = gmdate("i") [, int $second = gmdate("s") [, int $month = gmdate("n") [, int $day = gmdate("j") [, int $year = gmdate("Y") [, int $is_dst = -1 ]]]]]] )
```

ตัวอย่างที่ 7.31 การใช้งานฟังก์ชัน gmmktim ( )

```
<?php
        echo "July 1, 2012 is on a " . date ("l", gmmktime (0, 0, 0, 7, 1, 2012));
?>      // ผลลัพธ์ คือ July 1, 2012 is on a Sunday
```

7.3.9 การประยุกต์ใช้งานฟังก์ชันที่เกี่ยวข้องกับวันที่และเวลา

1) การประยุกต์ใช้ฟังก์ชันวันที่และเวลา เพื่อสร้างปฏิทินออนไลน์ เป็นการประยุกต์ใช้ฟังก์ชันวันที่และเวลาที่ได้กล่าวถึงไว้ ตัวอย่างดังนี้

ตัวอย่างที่ 7.32 การประยุกต์ใช้ฟังก์ชันเพื่อทำปฏิทินออนไลน์ส่วนของโครงสร้าง CSS tags เพื่อควบคุมการแสดงผลข้อมูลในรูปแบบตาราง

```
<style type="text/css">
div.holder{
        position:relative;
        font-family:tahoma, "Microsoft Sans Serif", Vanessa;
        border:2px solid #999;
        float:left;
        font-size:12px;
        width:175px;
        padding:5px;
}
div.month {
        position:relative;
        display:block;
        height:18px;
        width:100%;
        float:left;
        left:0;
        top:0;
```

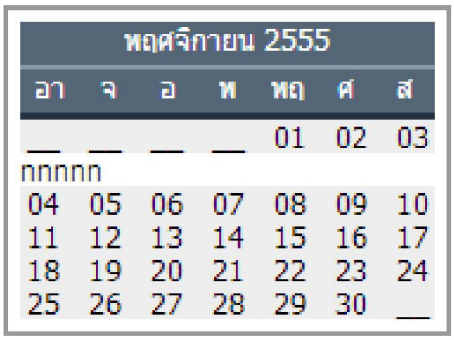
```
        background:#567;
        color:#fff;
        border-bottom:2px solid #89a;
        text-align:center;
}
div.wkday {
        position:relative;
        clear:both;
        float:left;
        height:18px;
        display:block;
        width:100%;
        background:#567;
        color:#fff;
        border-bottom:3px solid #234;
}
div.wkday span {
        display:block;
        float:left;
        width:25px;
        text-align:center;
}
div.box_day {
        position:relative;
        width:100%;
        clear:both;
        float:left;
        background:#EEEEEE;
        color:#000;
        word-spacing:8px;
        text-indent:3px;
        }
```

```
</style>
```

ตัวอย่างที่ 7.33 การประยุกต์ใช้ฟังก์ชันเพื่อทำปฏิทินออนไลน์ส่วนของ PHP สคริปต์ เพื่อประมวลผลและแสดงผลร่วมกับ CSS tags

```
<body>
<?php
        $thai_month_arr=array ("0"=>"", "1"=>"มกราคม", "2"=>"กุมภาพันธ์", "3"=>"มีนาคม",
                                "4"=>"เมษายน", "5"=>"พฤษภาคม", "6"=>"มิถุนายน",
                                "7"=>"กรกฎาคม", "8"=>"สิงหาคม", "9"=>"กันยายน",
                                "10"=>"ตุลาคม", "11"=>"พฤศจิกายน", "12"=>"ธันวาคม");
        $now_month = date ("Y-m-01");
        $mk_time = strtotime ($now_month);
        $day_no = date ("t",$mk_time);
        $wan_no = date ("w",$mk_time);
        $box_day = $day_no+$wan_no;
        $rows_week = ceil ($box_day/7);
        $total_box = $rows_week * 7;
        function get_day ($no_day,$wan_no,$day_no) {
                $wan_tee = $no_day-$wan_no;
                i f($wan_tee<=0) {
                        $wan_tee="__";
                        return $wan_tee;
                } else {
                        if ($wan_tee <= $day_no) {
                                return str_pad ($wan_tee,2,"0",STR_PAD_LEFT);
                        } else {
                                return "__";
                        }
                }
        }
?>
<div class="holder">
```

```
<div class="month">
<?=$thai_month_arr [intval (date ("m"))] ?> <?=date ("Y")+543 ?>
</div>
<div class="wkday">
        <span>อา</span>
        <span>จ</span>
        <span>อ</span>
        <span>พ</span>
        <span>พฤ</span>
        <span>ศ</span>
        <span>ส</span>
</div>
<?php
        for ($i = 1; $i <= $total_box; $i++) {
?>
<?php
        if ($i%7 == 1) echo "<div class='box_day'>";
?>
<?=get_day ($i,$wan_no,$day_no) ?>
<?php
        if ($i%7 == 0 || $i == $total_box ) {
                echo "</div>n";
        }
?>
<?php } ?>
</div>
<br style="clear:both;" />
</body>
```



ภาพที่ 7.1 แสดงตัวอย่างผลลัพธ์การสร้างปฏิทินออนไลน์

2) การประยุกต์ใช้ฟังก์ชันวันที่และเวลา เพื่อคำนวณหาอายุ มีตัวอย่าง ดังนี้

ตัวอย่างที่ 7.34 การประยุกต์โปรแกรมคำนวณอายุ กรณีศึกษาแบบที่ 1

```
1    <?php
2        function AgeCalculator ($year,$month,$day) {
3            $birthday_convert = mktime (1, 1, 1, $month, $day, $year);
4            return (floor ((time ( ) - $birthday_convert) / 31556926));
5        }
6        echo AgeCalculator (1978, 1, 2);          // ผลลัพธ์จะได้ 35
7    ?>
```

จากตัวอย่างที่ 7.34 เป็นตัวอย่างการประยุกต์ใช้ฟังก์ชันวันที่และเวลาเพื่อคำนวณหาอายุ อธิบายดังนี้

บรรทัดที่ 6 แสดงผลค่าการเรียกใช้ฟังก์ชัน AgeCalculator ( ) โดยพารามิเตอร์ที่ส่งประกอบด้วย ปี ค.ศ.เกิด เดือนเกิด และวันที่เกิด ตัวอย่างค่าที่ส่งเข้าไปประกอบด้วย ปี ค.ศ.1978 เดือนมกราคม วันที่ 2 เรียกใช้ฟังก์ชันด้วย AgeCalculator (1978, 1, 2)

บรรทัดที่ 2 เมื่อฟังก์ชัน AgeCalculator ( ) ถูกเรียกใช้ ฟังก์ชัน AgeCalculator ( ) กำหนดรับพารามิเตอร์ 3 ค่า ประกอบด้วย $year, $month และ $day โดยค่าถูกส่งมาจากบรรทัดที่ 6 ดังนั้นหมายความว่า กำหนดค่า $year มีค่าเท่ากับ 1978 กำหนดค่า $month มีค่าเท่ากับ 1 และ $day มีค่าเท่ากับ 2 ตามลำดับ

บรรทัดที่ 3 กำหนดให้ตัวแปร $birthday_convert มีค่าเท่ากับผลของฟังก์ชัน mktime ( ) เพื่อแปลงวัน เดือน ปี ค.ศ. เกิด เป็นรูปแบบของ timestamp จะมีค่าเท่ากับ 252525661

บรรทัดที่ 4 ส่งผลการประมวลผลกลับไปยังส่วนที่เรียกใช้ คือ บรรทัดที่ 6 การประมวลผลนิพจน์ ประกอบด้วยค่าดังนี้ time ( ) คือ (วันและเวลาปัจจุบัน – วัน เดือน ปี เกิด) ทั้งหมดหารด้วย 31556926 (เมื่อ 1 ปี มีค่าเท่ากับ 31556926 วินาที) ผลหารจะออกมาอยู่ในรูปเลขทศนิยม ดังนั้นจึงใช้ฟังก์ชัน floor ( ) เพื่อตัดทศนิยม

หมายเหตุ ถ้าในกรณีจะเขียนฟังก์ชันรับค่าเป็นปี พ.ศ. จะต้องลบออกด้วย 543 เพื่อแปลงเป็น ปี ค.ศ.ก่อนเข้าสู่นิพจน์

7.3.10 การเทียบเวลาวินาที

การเทียบเวลาวินาทีเพื่อความสะดวกในการพัฒนาโปรแกรมที่เกี่ยวข้องกับเวลา มีรายละเอียด ดังนี้ (วิกิพีเดีย สารานุกรมเสรี. 2556)

1 มิลลิวินาที เท่ากับ 0.001 วินาที

1 วินาที เท่ากับ 1 วินาที

1 นาที เท่ากับ 60 วินาที

1 ชั่วโมง เท่ากับ 3600 วินาที

1 วัน เท่ากับ 86400 วินาที

1 สัปดาห์ เท่ากับ 604800 วินาที

1 เดือน เท่ากับ 2629743.83 วินาที

1 ปี เท่ากับ 31556926 วินาที

1 ปีปกติสุรทิน เท่ากับ 31536000 วินาที (ปีปกติที่มี 365 วัน)

1 ปีอธิกสุรทิน เท่ากับ 31622400 วินาที (เป็นปีที่มีการเพิ่มหนึ่งวัน ผลมาจากเดือนกุมภาพันธ์ในปีอธิกสุรทินมี 29 วัน แทนที่จะมี 28 วันตามปกติ ดังนั้น ปีดังกล่าวจึงมี 366 วัน แทนที่จะมี 365 วันตามปกติ)

สรุป

กระบวนการทำงานของภาษา PHP ตั้งแต่นำข้อมูลเข้า ประมวล และแสดงผลลัพธ์ ชนิดของข้อมูลที่จำเป็นต้องข้องเกี่ยวอยู่บ่อยครั้ง นั่นก็คือ ข้อมูลชนิดข้อความ ตัวเลข วันและเวลา ลักษณะข้อมูลดังกล่าวถือเป็นพื้นฐานสำคัญ ดังนั้นภาษา PHP จึงมีฟังก์ชันการทำงานที่สนับสนุนการประมวลผล ข้อมูลดังกล่าวอยู่เป็นจำนวนมาก ผู้พัฒนาสามารถประยุกต์ใช้ฟังก์ชันที่มีอยู่ในรูปแบบต่างๆ เพื่อประมวลผลให้ได้ผลลัพธ์ตามต้องการ เช่น การนับจำนวนตัวอักษร การนับคำ การค้นหาข้อความ การเปลี่ยนรูปแบบตัวพิมพ์ การแยกและเชื่อมต่อข้อความ การจัดการรูปแบบการแสดงผลตัวเลข การแสดงผลวันที่และเวลา การคำนวณหาวันเกิด และปฏิทินออนไลน์ เป็นต้น

คำถามท้ายบท

1. จงอธิบายฟังก์ชันที่ใช้สำหรับหาขนาดของข้อความ พร้อมยกตัวอย่างประกอบ
2. จงอธิบายหลักการทำงานของสคริปต์ด้านล่างตั้งแต่บรรทัดที่ 1 ถึงบรรทัดสุดท้าย และบอกผลลัพธ์ที่ได้จากสคริปต์

```
1    <?php
2        $variable = "Computer Science of Suratthani Rajabhat University";
3        $pieces = explode (" ", $variable);
4        echo $pieces [0];
5        echo $pieces [3];
6    ?>
```

3. จงอธิบายหลักการทำงานของสคริปต์ด้านล่างตั้งแต่บรรทัดที่ 1 ถึงบรรทัดสุดท้าย และบอกผลลัพธ์ที่ได้จากสคริปต์

```
1    <?php
2        $array = array("Color"=>"yellow", "red", "pink", "black", "blue");
3        $array = ucwords( implode (" ", $array));
4        echo $array;
5    ?>
```

4. จงอธิบายหลักการทำงานของสคริปต์ด้านล่างตั้งแต่บรรทัดที่ 1 ถึงบรรทัดสุดท้าย และบอกผลลัพธ์ที่ได้จากสคริปต์

```
1    <?php
2        function digit ($bit) {
3            for ($loop = 0, $number = 0; $loop <= $bit; $loop++){
4                $number = $number + pow (2, $loop);
5            }
6            return $number;
7        }
8        echo digit(7);
9    ?>
```

5. จงอธิบาย หน้าที่ของฟังก์ชัน strtotime ( ) รูปแบบการใช้งาน และยกตัวอย่างการประยุกต์ใช้งาน ไม่น้อยกว่า 1 ตัวอย่าง