Web Programming:
Intermediate
(Integrative-Generative AI Edition)

Contents:

Author: Student Price Book Center

คำนำ

ในโลกของการพัฒนาเว็บแอปพลิเคชันยุคปัจจุบัน นักพัฒนาไม่เพียงต้องเข้าใจพื้นฐานของภาษา PHP เท่านั้น แต่ยังต้องสามารถประยุกต์ใช้ความรู้เหล่านั้นให้เกิดผลลัพธ์ที่มีประสิทธิภาพ ปลอดภัย และสามารถดูแลต่อยอดได้ในระยะยาว หนังสือ *PHP Web Programming: Intermediate* เล่มนี้จึงถูกออกแบบมาเพื่อปูทางสำหรับผู้เรียนที่มีพื้นฐานแล้ว และต้องการต่อยอดสู่การเป็นนักพัฒนาเว็บที่มีความเชี่ยวชาญในระดับกลาง โดยเน้นทั้งแนวคิดการออกแบบโปรแกรม เทคนิคการเขียนโค้ดที่มีประสิทธิภาพ และการวางโครงสร้างโค้ดให้รองรับการพัฒนาในระยะยาว

เนื้อหาในหนังสือเล่มนี้เริ่มต้นที่ **บทที่ 5: การจัดการ Array และ String ขั้นสูง (Advanced Array and Advanced String)** ซึ่งเป็นการต่อยอดจากระดับพื้นฐาน โดยลงลึกในการใช้ *indexed array* และ *associative array* การจัดการข้อมูลในรูปแบบต่าง ๆ ด้วยฟังก์ชันที่ PHP มีให้ และการประมวลผลข้อความด้วยฟังก์ชันเช่น substr, strpos, preg_match พร้อมทั้งการใช้ foreach อย่างลึกซึ้งเพื่อจัดการข้อมูลเชิงโครงสร้างอย่างมีประสิทธิภาพ เนื้อหาส่วนนี้ไม่เพียงแสดงให้เห็นถึงความสามารถของ PHP ในการจัดการข้อมูล แต่ยังแฝงแนวคิดการออกแบบให้สามารถจัดการข้อมูลขนาดใหญ่หรือซับซ้อนในโลกจริงได้อีกด้วย

จากนั้นใน **บทที่ 6: การจัดการฟังก์ชันและการเขียนโปรแกรมแบบโมดูลาร์ (Function Management and Modular Programming)** ผู้อ่านจะได้เรียนรู้การออกแบบฟังก์ชันอย่างมีระบบ ตั้งแต่ฟังก์ชันที่รับพารามิเตอร์ ฟังก์ชันแบบไม่จำกัดจำนวนพารามิเตอร์ (Variadic) จนถึงฟังก์ชันแบบเรียกตัวเอง (Recursive) ที่เหมาะสำหรับโจทย์ที่มีรูปแบบการซ้ำซ้อน เช่น การทำงานกับข้อมูล Tree หรือการคำนวณแบบลำดับชั้น พร้อมทั้งเรียนรู้หลักการแยกไฟล์และจัดโครงสร้างโปรแกรมแบบโมดูลาร์ที่สามารถนำกลับมาใช้ซ้ำ แก้ไขง่าย และขยายต่อในระบบขนาดใหญ่ได้อย่างเป็นระบบ

หัวใจอีกประการหนึ่งของการพัฒนาเว็บคือการจัดการสถานะผู้ใช้ ซึ่งถูกอธิบายใน **บทที่ 7: การจัดการ Session และ Cookies (Session and Cookies)** โดยอธิบายตั้งแต่การสร้าง Session ใน PHP การใช้งาน Cookies เพื่อเก็บข้อมูลฝั่ง Client และที่สำคัญคือ แนวทางการรักษาความปลอดภัยพื้นฐาน เช่น การป้องกัน Session Fixation และ Cookie Hijacking เพื่อให้ผู้อ่านสามารถออกแบบระบบที่ไม่เพียงทำงานได้ แต่ยังปลอดภัยจากการถูกโจมตีในโลกแห่งความเป็นจริง

บทที่ 8: การเชื่อมต่อและจัดการฐานข้อมูล MySQL (Database Connection and MySQL DBMS) ถือเป็นบทที่มีความสำคัญอย่างยิ่งในการก้าวสู่การเขียนโปรแกรมเว็บระดับจริงจัง โดยผู้อ่านจะได้เรียนรู้การเชื่อมต่อฐานข้อมูลด้วย MySQLi และ PDO ซึ่งเป็นวิธีการมาตรฐาน พร้อมทั้งฝึกเขียนคำสั่ง SQL เบื้องต้นใน PHP ไม่ว่าจะเป็น SELECT, INSERT, UPDATE หรือ DELETE นอกจากนี้ยังอธิบายเทคนิคสำคัญในการใช้ *prepared statements* เพื่อป้องกัน SQL Injection และแนะนำแนวทางการจัดการข้อผิดพลาดจากฐานข้อมูล เพื่อให้แอปพลิเคชันทำงานได้อย่างมีเสถียรภาพ

หนังสือเล่มนี้ไม่เพียงเน้นแค่โค้ดหรือไวยากรณ์ แต่ยังแฝงแนวทางคิดเชิงระบบ (system thinking) และการออกแบบเชิงโครงสร้าง (structured design) ซึ่งเป็นพื้นฐานที่สำคัญของการพัฒนาเว็บในระดับองค์กร หรือการทำงานร่วมกับทีมพัฒนาในโลกของจริง ผ่านตัวอย่างโค้ดที่ครอบคลุม

กรณีศึกษาในสถานการณ์จริง พร้อมอธิบายอย่างเป็นขั้นเป็นตอนเพื่อให้ผู้อ่านสามารถทดลองและต่อยอดได้ด้วยตนเอง

ท้ายที่สุดนี้ ผู้เขียนหวังว่า *PHP Web Programming: Intermediate* จะเป็นอีกหนึ่งเครื่องมือสำคัญสำหรับนักพัฒนา PHP ที่กำลังมองหาหนทางในการยกระดับตนเอง จากผู้ที่เข้าใจพื้นฐานไปสู่การเป็นนักพัฒนาที่สามารถวางโครงสร้างระบบ จัดการข้อมูล และเขียนโปรแกรมที่มีคุณภาพ พร้อมรับมือกับโจทย์การพัฒนาเว็บแอปพลิเคชันที่แท้จริงในอนาคต.

ด้วยรักและปรารถนาดี

ศูนย์หนังสือราคานักเรียน

สารบัญ

หน้า

บทที่ 5

การจัดการ Array และ String ขั้นสูง

(Advanced Array and Advanced String)

เนื้อหา

- การจัดการ Array และ String ขั้นสูง
- การจัดการ Array และ String ขั้นสูง พร้อมอธิบายแนวคิด วิธีใช้งาน
- Array แบบ indexed และ associative array
- ฟังก์ชันจัดการ Array ใน PHP
- การทำงานกับ String ใน PHP
- การวนลูป Array และใช้ foreach ใน PHP แบบลึกซึ้ง
- ตัวอย่างโปรแกรม PHP แบบบูรณาการ

บทนำ

ในยุคที่ข้อมูลมีบทบาทสำคัญต่อการพัฒนาเว็บแอปพลิเคชัน การจัดการข้อมูลในรูปแบบโครงสร้างที่ยืดหยุ่นและมีประสิทธิภาพจึงเป็นสิ่งจำเป็นอย่างยิ่ง ภาษา PHP ซึ่งเป็นหนึ่งในภาษายอดนิยมสำหรับการพัฒนาเว็บ มอบเครื่องมืออันทรงพลังในการจัดการกับข้อมูลประเภท *Array* และ *String* ซึ่งเป็นพื้นฐานของกระบวนการประมวลผลข้อมูลที่หลากหลายในโลกของการเขียนโปรแกรม

บทที่ 5 ของหนังสือเล่มนี้มุ่งเน้นไปที่การจัดการ *Array* และ *String* ในระดับที่ลึกซึ้งขึ้น โดยเริ่มต้นจากการทำความเข้าใจโครงสร้างของ *Array* ทั้งในรูปแบบ **Indexed Array** และ **Associative Array** ซึ่งช่วยให้โปรแกรมสามารถจัดเก็บและเรียกใช้ข้อมูลในลักษณะลำดับหรือระบุด้วยชื่อคีย์ได้อย่างยืดหยุ่น เหมาะสำหรับการจัดเก็บข้อมูลประเภทต่าง ๆ ที่มีความสัมพันธ์กัน เช่น รายการสินค้า รายชื่อนักเรียน หรือผลการประมวลผลจากแบบฟอร์มผู้ใช้

เพื่อให้สามารถจัดการกับ Array ได้อย่างมีประสิทธิภาพ ผู้อ่านจะได้เรียนรู้การใช้งานฟังก์ชันพื้นฐานในการจัดการ Array เช่น array_push, array_pop, array_merge, sort และฟังก์ชันที่เกี่ยวข้องอื่น ๆ ซึ่งสามารถใช้จัดเรียง เพิ่ม ลบ หรือรวมข้อมูลได้อย่างง่ายดาย ฟังก์ชันเหล่านี้ไม่เพียงเพิ่มความสะดวกในการจัดการข้อมูล แต่ยังช่วยลดโค้ดซ้ำซ้อน และเพิ่มความยืดหยุ่นในการพัฒนาโปรแกรมเชิงโครงสร้าง

นอกจาก Array แล้ว อีกหนึ่งโครงสร้างข้อมูลที่มีความสำคัญไม่แพ้กันคือ *String* หรือข้อความ ซึ่งปรากฏอยู่ในแทบทุกส่วนของเว็บแอปพลิเคชัน บทนี้จะอธิบายฟังก์ชันสำคัญในการจัดการกับข้อความ เช่น substr สำหรับตัดคำ, strlen สำหรับนับความยาว, strpos สำหรับค้นหาตำแหน่งคำ และ

การใช้ preg_match เพื่อจับข้อความด้วย **Regular Expressions** ซึ่งเป็นเทคนิคสำคัญในการวิเคราะห์และตรวจสอบรูปแบบข้อมูล เช่น การตรวจสอบอีเมล รหัสไปรษณีย์ หรือหมายเลขโทรศัพท์

ในส่วนของการประมวลผล Array บทนี้ยังนำเสนอการใช้งาน **foreach loop** ในระดับที่ลึกซึ้งมากขึ้น โดยเน้นการวนซ้ำเพื่อเข้าถึงค่าและคีย์ในแต่ละตำแหน่งของ Array รวมถึงตัวอย่างการใช้งานกับ Array ซ้อนกัน (*Nested Arrays*) ซึ่งพบได้บ่อยในการจัดการข้อมูลเชิงโครงสร้าง เช่น JSON หรือข้อมูลจากฐานข้อมูลแบบหลายมิติ

จุดเด่นของบทนี้อยู่ที่การเชื่อมโยงระหว่างทฤษฎีและการปฏิบัติ โดยมีตัวอย่างโค้ดที่แสดงการใช้งานจริงในสถานการณ์หลากหลาย ทั้งในรูปแบบโค้ดง่าย ๆ และกรณีศึกษาที่ซับซ้อนยิ่งขึ้น เพื่อให้ผู้อ่านสามารถฝึกฝนและนำไปประยุกต์ใช้ในการพัฒนาโปรแกรมได้อย่างมีประสิทธิภาพ

โดยสรุป บทที่ 5 นี้จะช่วยเสริมสร้างความเข้าใจในการจัดการข้อมูลประเภท Array และ String ซึ่งเป็นรากฐานสำคัญของโปรแกรม PHP ทุกประเภท ทั้งในระดับเริ่มต้นและระดับประยุกต์ การเรียนรู้ในบทนี้จึงเปรียบเสมือนการวางรากฐานที่มั่นคงก่อนต่อยอดไปสู่การจัดการข้อมูลที่ซับซ้อนมากขึ้นในบทถัด ๆ ไป

ผู้เขียนหวังว่าผู้อ่านจะได้ทั้งความรู้ ความเข้าใจ และแรงบันดาลใจจากบทเรียนในบทนี้ เพื่อนำไปพัฒนาทักษะการเขียนโปรแกรม PHP ให้สามารถจัดการข้อมูลได้อย่างยืดหยุ่น มีประสิทธิภาพ และตอบโจทย์การพัฒนาเว็บแอปพลิเคชันในโลกปัจจุบันได้อย่างแท้จริง.

การจัดการ Array และ String ขั้นสูง

1. Array แบบ indexed และ associative array

Indexed array คืออาเรย์ที่ใช้ดัชนีเป็นตัวเลข (0,1,2,...)

```
$fruits = ["apple", "banana", "cherry"];
echo $fruits[1]; // banana
```

Associative array คืออาเรย์ที่ใช้คีย์เป็น string หรือแบบกำหนดเอง

```
$person = [
    "name" => "John",
    "age" => 30,
    "city" => "Bangkok"
];
echo $person["name"]; // John
```

---

2. ฟังก์ชันจัดการ Array สำคัญ

ฟังก์ชัน	คำอธิบาย	ตัวอย่าง
array_push()	เพิ่มสมาชิกที่ท้าย array	array_push($arr, "new");

ฟังก์ชัน	คำอธิบาย	ตัวอย่าง
array_pop()	เอาสมาชิกตัวสุดท้ายออก	$last = array_pop($arr);
array_merge()	รวม array หลายตัวเป็นตัวเดียว	$merged = array_merge($arr1, $arr2);
sort()	เรียงลำดับ array แบบ indexed	sort($arr);
asort()	เรียงลำดับ array แบบ associative ตามค่า	asort($assocArr);
ksort()	เรียงลำดับ associative array ตามคีย์	ksort($assocArr);
count()	นับจำนวนสมาชิกใน array	$len = count($arr);

---

3. การทำงานกับ String

- substr($string, $start, $length) — ตัดเอาช่วง substring
- strlen($string) — หาความยาว string
- strpos($haystack, $needle) — หาตำแหน่งแรกของข้อความใน string
- preg_match($pattern, $subject) — ตรวจสอบ pattern ด้วย regular expression

ตัวอย่าง:

```
$str = "Hello World";
echo substr($str, 6, 5); // World
echo strlen($str); // 11
echo strpos($str, "o"); // 4
if (preg_match("/World/", $str)) {
    echo "เจอคำว่า World!";
}
```

---

4. การวนลูป array และการใช้ foreach แบบลึกซึ้ง

- การวนลูปด้วย foreach แบบง่าย

```
foreach ($fruits as $fruit) {
    echo $fruit . "<br>";
}
```

- การใช้คีย์และค่าพร้อมกัน

```
foreach ($person as $key => $value) {
    echo "$key: $value<br>";
}
```

- การวนลูปลึกกับ array ซ้อน (multidimensional)

```
$users = [
```

```
    ["name" => "John", "age" => 30],
    ["name" => "Jane", "age" => 25]
];
foreach ($users as $user) {
    foreach ($user as $key => $val) {
        echo "$key: $val; ";
    }
    echo "<br>";
}
```

การจัดการ Array และ String ขั้นสูง พร้อมอธิบายแนวคิด วิธีใช้งาน

---

1. Array แบบ Indexed กับ Associative Array

Indexed Array

- อาเรย์แบบลำดับที่ใช้เลขจำนวนเต็มเริ่มจาก 0 เป็นดัชนี
- เหมาะกับข้อมูลที่เป็นลิสต์เรียงลำดับ เช่น รายชื่อสินค้า
- สามารถเพิ่มสมาชิกใหม่ได้ด้วยฟังก์ชัน array_push() หรือกำหนดด้วยดัชนีโดยตรง
- ดัชนีต้องเป็นจำนวนเต็มและเรียงตามลำดับ (0,1,2,...)

ตัวอย่าง

```
$fruits = ["apple", "banana", "cherry"];
echo $fruits[0];  // apple
array_push($fruits, "date");
print_r($fruits);
```

Associative Array

- ใช้คีย์ที่เป็น string หรือจำนวนเต็มแบบไม่เรียงลำดับ
- เหมาะกับข้อมูลที่ต้องการอ้างอิงโดยชื่อ เช่น ข้อมูลผู้ใช้
- สามารถเพิ่มสมาชิกได้ด้วยการกำหนด key-value คู่ใหม่
- คีย์สามารถเป็น string หรือตัวเลขได้ แต่ไม่จำเป็นต้องเรียงลำดับ

ตัวอย่าง

```
$person = [
    "name" => "Alice",
    "age" => 28,
    "city" => "Bangkok"
];
```

```
echo $person["name"];  // Alice
$person["email"] = "alice@example.com";
print_r($person);
```

---

2. ฟังก์ชันจัดการ Array สำคัญและการใช้งานเชิงลึก

array_push() และ array_pop()

- array_push() เพิ่มสมาชิกใหม่ที่ท้ายอาเรย์
- array_pop() ดึงสมาชิกตัวสุดท้ายออกจากอาเรย์และคืนค่านั้น

```
$arr = [1, 2, 3];
array_push($arr, 4, 5);
print_r($arr);  // [1,2,3,4,5]
$last = array_pop($arr);
echo $last;     // 5
print_r($arr);  // [1,2,3,4]
```

array_merge()

- รวมอาเรย์หลายตัวเข้าด้วยกัน
- ถ้าเป็น indexed array จะต่อท้ายกัน, ถ้าเป็น associative array จะรวม key-value

```
$a = ["red", "green"];
$b = ["blue"];
$c = array_merge($a, $b);
print_r($c);  // [ "red", "green", "blue" ]

$assoc1 = ["a" => 1, "b" => 2];
$assoc2 = ["b" => 3, "c" => 4];
$mergedAssoc = array_merge($assoc1, $assoc2);
print_r($mergedAssoc); // ["a"=>1, "b"=>3, "c"=>4] - key 'b' ถูกแทนที่
```

การเรียงลำดับ

- sort() — เรียง indexed array ตามค่า
- asort() — เรียง associative array ตามค่า (คีย์คงที่)
- ksort() — เรียง associative array ตามคีย์

```
$arr = [3, 1, 4];
sort($arr);  // [1,3,4]

$assoc = ["b" => 2, "a" => 1];
```

```
asort($assoc);  // ["a"=>1, "b"=>2]
ksort($assoc);  // ["a"=>1, "b"=>2]
```

count()

- คืนค่าจำนวนสมาชิกในอาเรย์

```
$arr = [1, 2, 3];
echo count($arr);  // 3
```

---

3. การทำงานกับ String ขั้นสูง

substr()

- ตัดเอาส่วนย่อยของ string
- สามารถใช้เลขติดลบเพื่อเริ่มตัดจากท้าย string ได้

```
$str = "Hello World";
echo substr($str, 6, 5);  // World
echo substr($str, -5);    // World
```

strlen()

- คืนค่าความยาว string (จำนวนอักขระ)
- ใน PHP แบบพื้นฐาน นับเป็น bytes (ควรใช้ mb_strlen() สำหรับ multibyte)

```
echo strlen("สวัสดี");  // 12 (เพราะเป็น UTF-8 multibyte)
echo mb_strlen("สวัสดี"); // 6 (นับจริงตามอักขระ)
```

strpos()

- หาตำแหน่งของข้อความแรกที่เจอใน string
- คืนตำแหน่งเริ่มต้น (0-based) หรือ false ถ้าไม่เจอ

```
echo strpos("Hello World", "World");  // 6
```

preg_match()

- ใช้ตรวจสอบ pattern ด้วย regular expression
- คืน 1 หากตรง, 0 หากไม่ตรง

```
if (preg_match("/^[a-zA-Z]+$/", "Hello")) {
    echo "ตรงตาม pattern";
}
```

---

4. การวนลูป Array และใช้ foreach แบบลึกซึ้ง

การวนลูปแบบพื้นฐาน

```
$fruits = ["apple", "banana", "cherry"];
foreach ($fruits as $fruit) {
```

```
    echo $fruit . "<br>";
}
```

การใช้คีย์และค่า

```
$person = ["name" => "John", "age" => 30];
foreach ($person as $key => $value) {
    echo "$key: $value<br>";
}
```

การวนลูปอาเรย์แบบหลายมิติ (Multidimensional Array)

```
$users = [
    ["name" => "Alice", "age" => 28],
    ["name" => "Bob", "age" => 32]
];

foreach ($users as $user) {
    foreach ($user as $key => $val) {
        echo "$key = $val; ";
    }
    echo "<br>";
}
```

เคล็ดลับและ Best Practices

- ใช้ array_key_exists() เพื่อตรวจสอบว่าคีย์มีในอาเรย์หรือไม่
- ใช้ mb_* functions (เช่น mb_strlen) เมื่อทำงานกับ string ที่มี multibyte character (เช่น ภาษาไทย)
- หลีกเลี่ยงการใช้ for กับ associative array ใช้ foreach แทนเพื่อความปลอดภัยและอ่านง่าย
- ใช้ explode() และ implode() เพื่อแปลงระหว่าง string กับ array

---

Array แบบ indexed และ associative array

---

1. Array แบบ Indexed

- เป็นอาเรย์ที่ใช้ดัชนีเป็นตัวเลขจำนวนเต็ม (เริ่มจาก 0 ขึ้นไป)
- เหมาะกับข้อมูลที่เรียงตามลำดับ เช่น รายการสินค้า, รายชื่อ ฯลฯ
- สามารถประกาศได้ทั้งแบบกำหนดค่าเลย หรือเพิ่มสมาชิกทีหลัง

ตัวอย่าง

```
// ประกาศ array แบบ indexed พร้อมค่า
$fruits = ["apple", "banana", "cherry"];

// เข้าถึงสมาชิกโดยใช้ดัชนี (index)
echo $fruits[0];  // apple
echo $fruits[2];  // cherry

// เพิ่มสมาชิกใหม่ด้วย array_push หรือ กำหนดดัชนีใหม่
array_push($fruits, "date");
$fruits[] = "elderberry";

print_r($fruits);
/*
ผลลัพธ์:
Array
(
    [0] => apple
    [1] => banana
    [2] => cherry
    [3] => date
    [4] => elderberry
)
*/
```

- ดัชนีใน Indexed Array จะเรียงลำดับจาก 0 ขึ้นไปโดยอัตโนมัติ (ถ้าไม่ระบุเอง)
- สามารถกำหนดดัชนีเองได้แต่ไม่ควรข้ามลำดับมากเกินไป เพราะจะมีช่องว่าง

```
$arr = [];
$arr[0] = "one";
$arr[2] = "three";  // ดัชนีข้าม 1 ไปเลย
print_r($arr);
/*
ผลลัพธ์:
Array
(
    [0] => one
```

```
    [2] => three
)
*/
```

---

2. Associative Array

- เป็นอาเรย์ที่ใช้คีย์เป็น string หรือจำนวนเต็มแบบไม่เรียงลำดับ
- เหมาะกับข้อมูลที่ต้องการอ้างอิงแบบมีชื่อ เช่น รายละเอียดผู้ใช้, config ฯลฯ
- สามารถเพิ่มสมาชิกใหม่ด้วยการระบุคีย์และค่าได้เลย

ตัวอย่าง

```
// ประกาศ associative array
$person = [
    "name" => "John",
    "age" => 30,
    "city" => "Bangkok"
];

// เข้าถึงสมาชิกโดยใช้คีย์
echo $person["name"];  // John
echo $person["age"];   // 30

// เพิ่มสมาชิกใหม่
$person["email"] = "john@example.com";

print_r($person);
/*
ผลลัพธ์:
Array
(
    [name] => John
    [age] => 30
    [city] => Bangkok
    [email] => john@example.com
)
*/
```

- คีย์สามารถเป็น string หรือจำนวนเต็มก็ได้
- ไม่มีลำดับที่ตายตัว แต่สามารถใช้ฟังก์ชันเช่น ksort() เพื่อเรียงลำดับตามคีย์ได้
- ใช้กับข้อมูลที่ต้องการอ้างอิงแบบ key-value

เปรียบเทียบสั้น ๆ

จุดเด่น	Indexed Array	Associative Array
ดัชนี (Key)	ตัวเลขลำดับ (0,1,2...)	String หรือจำนวนเต็มแบบกำหนดเอง
เหมาะกับ	ข้อมูลที่มีลำดับชัดเจน	ข้อมูลแบบ key-value
ตัวอย่างการเข้าถึง	$arr[0], $arr[1]	$arr['key'], $arr['name']
การเพิ่มสมาชิก	$arr[] = value หรือ array_push()	$arr['newKey'] = value

---

นี่คือตัวอย่างโปรแกรมแบบเต็มไฟล์ จำนวน 5 โปรแกรม และตัวอย่างแนวประยุกต์ 5 โปรแกรมเกี่ยวกับ **Array แบบ indexed และ associative array** พร้อมโครงสร้าง คำอธิบายโค้ด และผลการรัน

ตัวอย่างโปรแกรมพื้นฐาน 5 โปรแกรม

โครงสร้างโปรเจกต์

```
array_basic_examples/
├── indexed_array.php
├── associative_array.php
├── array_push_pop.php
├── array_merge_sort.php
└── multidimensional_array.php
```

---

1. indexed_array.php

```php
<?php
// ตัวอย่าง Indexed Array พื้นฐาน
$colors = ["red", "green", "blue"];

// เข้าถึงสมาชิก
echo "สีที่ 1: " . $colors[0] . "<br>";  // red
echo "สีที่ 3: " . $colors[2] . "<br>";  // blue

// เพิ่มสมาชิกใหม่
```

```
$colors[] = "yellow";

echo "อาเรย์สีทั้งหมด:<br>";
print_r($colors);
?>
```

ผลการรัน

```
สีที่ 1: red
สีที่ 3: blue
อาเรย์สีทั้งหมด:
Array
(
    [0] => red
    [1] => green
    [2] => blue
    [3] => yellow
)
```

---

2. associative_array.php

```
<?php
// ตัวอย่าง Associative Array
$user = [
    "username" => "john123",
    "email" => "john@example.com",
    "age" => 25
];

// เข้าถึงสมาชิก
echo "ชื่อผู้ใช้: " . $user["username"] . "<br>";
echo "อีเมล: " . $user["email"] . "<br>";

// เพิ่มสมาชิกใหม่
$user["country"] = "Thailand";

echo "ข้อมูลผู้ใช้ทั้งหมด:<br>";
```

```
print_r($user);
?>
```

ผลการรัน

```
ชื่อผู้ใช้: john123
อีเมล: john@example.com
ข้อมูลผู้ใช้ทั้งหมด:
Array
(
    [username] => john123
    [email] => john@example.com
    [age] => 25
    [country] => Thailand
)
```

---

3. array_push_pop.php

```
<?php
// ใช้ array_push และ array_pop กับ Indexed Array
$stack = ["apple", "banana"];

array_push($stack, "cherry", "date");
print_r($stack);

$lastItem = array_pop($stack);
echo "<br>ลบออก: $lastItem<br>";

print_r($stack);
?>
```

ผลการรัน

```
Array
(
    [0] => apple
    [1] => banana
    [2] => cherry
    [3] => date
```

```
)
ลบออก: date
Array
(
    [0] => apple
    [1] => banana
    [2] => cherry
)
```

---

4. array_merge_sort.php

```
<?php
// รวมและเรียงลำดับ Indexed Array
$arr1 = [5, 2, 9];
$arr2 = [1, 7, 3];

$merged = array_merge($arr1, $arr2);
sort($merged);

print_r($merged);
?>
```

ผลการรัน

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
    [3] => 5
    [4] => 7
    [5] => 9
)
```

---

5. multidimensional_array.php

```
<?php
// ตัวอย่าง Associative Array หลายมิติ
```

```
$users = [
    ["name" => "Alice", "age" => 28],
    ["name" => "Bob", "age" => 32]
];

// แสดงชื่อและอายุผู้ใช้
foreach ($users as $user) {
    echo "ชื่อ: " . $user["name"] . ", อายุ: " . $user["age"] . "<br>";
}
?>
```

ผลการรัน

ชื่อ: Alice, อายุ: 28

ชื่อ: Bob, อายุ: 32

ตัวอย่างโปรแกรมแนวประยุกต์ 5 โปรแกรม

โครงสร้างโปรเจกต์

```
array_advanced_examples/
├── user_management.php
├── product_catalog.php
├── count_values.php
├── sort_associative.php
└── search_in_array.php
```

---

1. user_management.php

```
<?php
// ตัวอย่างการจัดการข้อมูลผู้ใช้ด้วย Associative Array
$users = [
    ["id" => 1, "name" => "John", "role" => "admin"],
    ["id" => 2, "name" => "Jane", "role" => "user"],
    ["id" => 3, "name" => "Mark", "role" => "editor"],
];

// หาผู้ใช้ที่เป็น admin
foreach ($users as $user) {
```

```
    if ($user["role"] === "admin") {
        echo "Admin: " . $user["name"] . "<br>";
    }
}
?>
```

---

2. product_catalog.php

```
<?php
// ตัวอย่างสร้างแคตาล็อกสินค้าแบบ Indexed และ Associative
$products = [
    ["id" => 101, "name" => "T-Shirt", "price" => 299],
    ["id" => 102, "name" => "Jeans", "price" => 799],
    ["id" => 103, "name" => "Sneakers", "price" => 1599],
];

// แสดงรายการสินค้า
foreach ($products as $product) {
    echo "สินค้า: " . $product["name"] . " ราคา: " . $product["price"] . " บาท<br>";
}
?>
```

---

3. count_values.php

```
<?php
// นับจำนวนค่าว่าซ้ำใน Indexed Array เท่าไรบ้าง
$colors = ["red", "blue", "red", "green", "blue", "blue"];
$countColors = array_count_values($colors);

print_r($countColors);
?>
```

ผลการรัน

```
Array
(
    [red] => 2
    [blue] => 3
```

```
    [green] => 1
)
```

---

4. sort_associative.php

```php
<?php
// เรียง associative array ตามคีย์และค่าด้วย ksort() และ asort()
$ages = [
    "John" => 30,
    "Jane" => 25,
    "Mark" => 35,
];

echo "เรียงตามคีย์:<br>";
ksort($ages);
print_r($ages);

echo "<br>เรียงตามค่า:<br>";
asort($ages);
print_r($ages);
?>
```

---

5. search_in_array.php

```php
<?php
// ค้นหาค่าใน Indexed Array
$fruits = ["apple", "banana", "cherry", "date"];

$search = "banana";
if (in_array($search, $fruits)) {
    echo "$search พบในอาเรย์";
} else {
    echo "$search ไม่พบในอาเรย์";
}
?>
```
