# PHP Coding Challenges - Chapter 2

## Challenge 1: Basic Functions

Create utility functions for common operations:

```
<?php
// TODO: Create function 'multiply' that takes two numbers and returns their
product

// TODO: Create function 'isEven' that takes a number and returns true if
even

// TODO: Create function 'getGreater' that takes two numbers and returns the
larger one

// Test your functions
echo multiply(6, 7) . "<br>";        // Should output: 42
echo (isEven(8) ? "true" : "false") . "<br>";  // Should output: true
echo getGreater(15, 12) . "<br>";    // Should output: 15
?>
```

## Challenge 2: Array Operations

Create functions that work with arrays:

```
<?php
// TODO: Create function 'printArray' that outputs all elements in an array

// TODO: Create function 'findSum' that returns sum of all numbers in array

// TODO: Create function 'countElements' that returns the number of elements

// Test your functions
$numbers = [5, 10, 15, 20];

printArray($numbers);                  // Should output each number
echo "Sum: " . findSum($numbers) . "<br>";      // Should output: Sum: 50
echo "Count: " . countElements($numbers) . "<br>"; // Should output: Count: 4
?>
```

## Challenge 3: String Array Processing

Work with arrays of strings:

```
<?php
// TODO: Create function 'joinStrings' that joins array elements with a
separator

// TODO: Create function 'findLongestWord' that returns the longest string
// TODO: Create function 'containsWord' that checks if array contains
specific word

// Test your functions
$words = ["php", "programming", "web", "development"];

echo joinStrings($words, " - ") . "<br>";       // Should output: php -
programming - web - development
echo "Longest: " . findLongestWord($words) . "<br>"; // Should output:
Longest: programming
echo (containsWord($words, "web") ? "Found" : "Not found") . "<br>"; //
Should output: Found
?>
```

## Challenge 4: User Profile System

Create a user management system:

```
<?php
// TODO: Create function 'createUser' that takes name, email, age and returns
user array

// TODO: Create function 'displayUser' that shows user information nicely

// TODO: Create function 'isAdult' that checks if user is 18 or older

// TODO: Create function 'updateAge' that updates user's age

// Test your functions
$user = createUser("Alice Johnson", "alice@email.com", 25);
displayUser($user);
echo (isAdult($user) ? "Adult" : "Minor") . "<br>";

$user = updateAge($user, 26);
echo "Updated age: " . $user['age'] . "<br>";
?>
```

## Solutions

### Challenge 1 Solutions:

```
<?php
function multiply($a, $b) {
    return $a * $b;
}

function isEven($number) {
    return $number % 2 == 0;
}

function getGreater($a, $b) {
    return ($a > $b) ? $a : $b;
}
?>
```

## Challenge 2 Solutions:

```
<?php
function printArray($arr) {
    foreach ($arr as $element) {
        echo $element . "<br>";
    }
}

function findSum($arr) {
    $sum = 0;
    foreach ($arr as $number) {
        $sum += $number;
    }
    return $sum;
    // Alternative: return array_sum($arr);
}

function countElements($arr) {
    return count($arr);
}
?>
```

## Challenge 3 Solutions:

```
<?php
function joinStrings($arr, $separator) {
    return implode($separator, $arr);
}

function findLongestWord($arr) {
    $longest = $arr[0];
    foreach ($arr as $word) {
        if (strlen($word) > strlen($longest)) {
            $longest = $word;
        }
    }
    return $longest;
}

function containsWord($arr, $target) {
    return in_array($target, $arr);
}
?>
```

## Challenge 4 Solutions:

```
<?php
function createUser($name, $email, $age) {
    return [
        'name' => $name,
```

```php
        'email' => $email,
        'age' => $age
    ];
}

function displayUser($user) {
    echo "Name: " . $user['name'] . "<br>";
    echo "Email: " . $user['email'] . "<br>";
    echo "Age: " . $user['age'] . "<br>";
    echo "---<br>";
}

function isAdult($user) {
    return $user['age'] >= 18;
}

function updateAge($user, $newAge) {
    $user['age'] = $newAge;
    return $user;
}
?>
```

## Bonus Challenge: Simple Web Form Processor

```php
<?php
// Bonus: Create a function that processes form data
function processContactForm($formData) {
    $name = trim($formData['name']);
    $email = trim($formData['email']);
    $message = trim($formData['message']);

    // Basic validation
    $errors = [];

    if (empty($name)) {
        $errors[] = "Name is required";
    }

    if (empty($email) || !filter_var($email, FILTER_VALIDATE_EMAIL)) {
        $errors[] = "Valid email is required";
    }

    if (empty($message)) {
        $errors[] = "Message is required";
    }

    return [
        'success' => empty($errors),
        'errors' => $errors,
        'data' => [
            'name' => $name,
            'email' => $email,
            'message' => $message
        ]
    ];
}

// Example usage
$formData = [
    'name' => 'John Doe',
    'email' => 'john@example.com',
    'message' => 'Hello, this is a test message!'
];

$result = processContactForm($formData);

if ($result['success']) {
    echo "Form submitted successfully!<br>";
    echo "Thank you, " . $result['data']['name'] . "!<br>";
} else {
    echo "Errors found:<br>";
    foreach ($result['errors'] as $error) {
        echo "- $error<br>";
    }
}
?>
```

# Web Development Patterns

## Pattern 1: Database-like Array Operations

```php
<?php
// Simulate database operations with arrays
$users = [
    ['id' => 1, 'name' => 'Alice', 'role' => 'admin'],
    ['id' => 2, 'name' => 'Bob', 'role' => 'user'],
    ['id' => 3, 'name' => 'Charlie', 'role' => 'user']
];

function findUserById($users, $id) {
    foreach ($users as $user) {
        if ($user['id'] == $id) {
            return $user;
        }
    }
    return null;
}

function getUsersByRole($users, $role) {
    $filtered = [];
    foreach ($users as $user) {
        if ($user['role'] == $role) {
            $filtered[] = $user;
        }
    }
    return $filtered;
}

// Usage
$admin = findUserById($users, 1);
```

```
$regularUsers = getUsersByRole($users, 'user');
?>
```

## Pattern 2: HTML Generation Functions

```
<?php
function createLink($url, $text, $class = '') {
    $classAttr = $class ? " class=\"$class\"" : '';
    return "<a href=\"$url\"$classAttr>$text</a>";
}

function createList($items, $type = 'ul') {
    $html = "<$type>";
    foreach ($items as $item) {
        $html .= "<li>$item</li>";
    }
    $html .= "</$type>";
    return $html;
}

// Usage
$navItems = ['Home', 'About', 'Contact'];
$navigation = createList($navItems);
$homeLink = createLink('/', 'Home', 'nav-link');

echo $navigation;
echo $homeLink;
?>
```