Functions and Arrays in PHP

Function Basics

Function Structure

```
function functionName($parameter1, $parameter2) {
    // Function body
    return $value; // Optional return statement
}
```

Simple Function Examples

```
// Function with no parameters
function sayHello() {
    echo "Hello World!";
}

// Function with parameters
function greetUser($name) {
    echo "Hello, $name!";
}

// Function with return value
function addNumbers($a, $b) {
    return $a + $b;
}

// Function with default parameter
function createGreeting($name, $greeting = "Hello") {
    return "$greeting, $name!";
}
```

Calling Functions

```
sayHello();                              // Output: Hello World!
greetUser("Alice");                      // Output: Hello, Alice!
$sum = addNumbers(5, 3);                // $sum = 8
$message = createGreeting("Bob");        // $message = "Hello, Bob!"
$custom = createGreeting("Carol", "Hi"); // $custom = "Hi, Carol!"
```

Array Basics

Creating Arrays

```
// Indexed arrays
$fruits = array("apple", "banana", "orange");
$numbers = [1, 2, 3, 4, 5];  // PHP 5.4+ syntax

// Associative arrays (key-value pairs)
```

```php
$person = array(
    "name" => "John",
    "age" => 25,
    "city" => "New York"
);

$student = [
    "id" => 123,
    "name" => "Alice",
    "grade" => "A"
];
```

Accessing Arrays

```php
$fruits = ["apple", "banana", "orange"];

// Access by index
$first = $fruits[0];      // "apple"
$second = $fruits[1];     // "banana"

// Modify elements
$fruits[1] = "blueberry"; // ["apple", "blueberry", "orange"]

// Add elements
$fruits[] = "grape";      // Adds to end
array_push($fruits, "kiwi"); // Also adds to end

// Associative array access
$person = ["name" => "John", "age" => 25];
$name = $person["name"];  // "John"
$person["email"] = "john@example.com"; // Add new key
```

Array Functions

```php
$numbers = [1, 2, 3, 4, 5];

// Array information
$length = count($numbers);        // 5
$size = sizeof($numbers);         // 5 (same as count)

// Array manipulation
array_push($numbers, 6);          // Add to end: [1,2,3,4,5,6]
$last = array_pop($numbers);      // Remove from end, returns 6
array_unshift($numbers, 0);       // Add to beginning: [0,1,2,3,4,5]
$first = array_shift($numbers);   // Remove from beginning, returns 0

// Search functions
$position = array_search(3, $numbers);  // Returns index of 3
$exists = in_array(4, $numbers);        // Returns true if 4 exists
```

Array Loops

```php
$fruits = ["apple", "banana", "orange"];
```

```php
// Foreach loop (recommended)
foreach ($fruits as $fruit) {
    echo $fruit . "<br>";
}

// Foreach with index
foreach ($fruits as $index => $fruit) {
    echo "$index: $fruit<br>";
}

// Associative array loop
$person = ["name" => "John", "age" => 25, "city" => "Boston"];
foreach ($person as $key => $value) {
    echo "$key: $value<br>";
}

// Traditional for loop
for ($i = 0; $i < count($fruits); $i++) {
    echo $fruits[$i] . "<br>";
}
```

Functions with Arrays

Array Processing Functions

```php
// Function to find maximum in array
function findMax($numbers) {
    $max = $numbers[0];
    foreach ($numbers as $number) {
        if ($number > $max) {
            $max = $number;
        }
    }
    return $max;
}

// Function to calculate array sum
function calculateSum($numbers) {
    $sum = 0;
    foreach ($numbers as $number) {
        $sum += $number;
    }
    return $sum;
    // Alternative: return array_sum($numbers);
}

// Function to filter even numbers
function getEvenNumbers($numbers) {
    $evens = [];
    foreach ($numbers as $number) {
        if ($number % 2 == 0) {
            $evens[] = $number;
        }
    }
    return $evens;
```

```
}
```

String Array Functions

```
// Function to find longest string
function findLongest($strings) {
    $longest = $strings[0];
    foreach ($strings as $string) {
        if (strlen($string) > strlen($longest)) {
            $longest = $string;
        }
    }
    return $longest;
}

// Function to capitalize all strings
function capitalizeAll($strings) {
    $capitalized = [];
    foreach ($strings as $string) {
        $capitalized[] = ucfirst($string);
    }
    return $capitalized;
}
```

Built-in Array Functions

```
$numbers = [3, 1, 4, 1, 5, 9, 2];

// Sorting
sort($numbers);           // [1, 1, 2, 3, 4, 5, 9]
rsort($numbers);          // [9, 5, 4, 3, 2, 1, 1]

// Mathematical functions
$sum = array_sum($numbers);       // Sum of all elements
$product = array_product($numbers); // Product of all elements

// Array transformation
$doubled = array_map(function($x) { return $x * 2; }, $numbers);
$filtered = array_filter($numbers, function($x) { return $x > 3; });

// Array merging
$arr1 = [1, 2, 3];
$arr2 = [4, 5, 6];
$merged = array_merge($arr1, $arr2); // [1, 2, 3, 4, 5, 6]
```

Practical Examples

Example 1: Student Grade System

```
function calculateGrade($scores) {
    $total = array_sum($scores);
    $average = $total / count($scores);
```

```
    if ($average >= 90) return 'A';
    elseif ($average >= 80) return 'B';
    elseif ($average >= 70) return 'C';
    elseif ($average >= 60) return 'D';
    else return 'F';
}

$studentScores = [85, 92, 78, 96, 88];
$grade = calculateGrade($studentScores);
echo "Grade: $grade";
```

Example 2: Shopping Cart

```
function addToCart(&$cart, $item, $price) {
    $cart[] = ["item" => $item, "price" => $price];
}

function calculateTotal($cart) {
    $total = 0;
    foreach ($cart as $product) {
        $total += $product["price"];
    }
    return $total;
}

$cart = [];
addToCart($cart, "Apple", 1.50);
addToCart($cart, "Bread", 2.99);
$total = calculateTotal($cart);
echo "Total: $" . number_format($total, 2);
```