1. **Write PHP scripts that demonstrate fundamentals PHP**
    1. **Printing "Hello, World!" on the screen:**

```
<?php
  echo "Hello, World!";
?>
```

    2. **Defining and using variables:**

```
<?php
  $name = "John";
  $age = 25;
  echo "My name is " . $name . " and I am " . $age . " years old.";
?>
```

    3. **Performing arithmetic operations:**

```
<?php
  $num1 = 10;
  $num2 = 5;
  echo "Addition: " . ($num1 + $num2) . "<br>";
  echo "Subtraction: " . ($num1 - $num2) . "<br>";
  echo "Multiplication: " . ($num1 * $num2) . "<br>";
  echo "Division: " . ($num1 / $num2) . "<br>";
?>
```

    4. **Using conditional statements:**

```
<?php
  $num = 10;
  if ($num > 0) {
    echo "The number is positive.";
  } else if ($num < 0) {
    echo "The number is negative.";
  } else {
    echo "The number is zero.";
  }
?>
```

    5. **Using loops:**

```
<?php
  // while loop
  $num = 1;
  while ($num <= 5) {
    echo $num . "<br>";
    $num++;
  }

  // for loop
  for ($i = 1; $i <= 5; $i++) {
    echo $i . "<br>";
  }

  // foreach loop
  $colors = array("red", "green", "blue");
  foreach ($colors as $color) {
    echo $color . "<br>";
  }
```

```
?>
```

6. Defining and calling functions:

```
<?php
   function square($num) {
      return $num * $num;
   }

   $result = square(5);
   echo "The square of 5 is " . $result;
?>
```

2. Write PHP script that will display grade based on criteria given below using the marks obtained in Examination.

```
<?php
$marks = 85; // replace with the actual marks obtained

if ($marks >= 90) {
  echo "Grade A+";
} elseif ($marks >= 80) {
  echo "Grade A";
} elseif ($marks >= 70) {
  echo "Grade B+";
} elseif ($marks >= 60) {
  echo "Grade B";
} elseif ($marks >= 50) {
  echo "Grade C+";
} elseif ($marks >= 40) {
  echo "Grade C";
} else {
  echo "Fail";
}
?>
```

3. Write a PHP script to demonstrate different String functions.

```
<?php
$string = "The quick brown fox jumps over the lazy dog.";

// Length of the string
echo "Length of the string: " . strlen($string) . "<br>";

// Convert string to uppercase
echo "Uppercase: " . strtoupper($string) . "<br>";

// Convert string to lowercase
echo "Lowercase: " . strtolower($string) . "<br>";

// Replace a substring
echo "Replace 'fox' with 'cat': " . str_replace("fox", "cat", $string) . "<br>";

// Substring
echo "Substring from index 4 to 15: " . substr($string, 4, 11) . "<br>";
```

```
// Split a string into an array
echo "Split string into an array: ";
print_r(explode(" ", $string));

// Join an array into a string
$array = array("The", "quick", "brown", "fox", "jumps", "over", "the", "lazy", "dog.");
echo "<br>Join array into a string: " . implode(" ", $array);
?>
```

4. Write a PHP script to Demonstrate OOPS Concept in PHP.

```
<?php

// Define a class named 'Person'
class Person {

  // Define the properties of the class
  public $name;
  public $age;

  // Define a constructor method for the class
  public function __construct($name, $age) {
    $this->name = $name;
    $this->age = $age;
  }

  // Define a method to display the person's name and age
  public function displayInfo() {
    echo "Name: " . $this->name . "<br>";
    echo "Age: " . $this->age . "<br>";
  }
}

// Define a class named 'Student' that extends the 'Person' class
class Student extends Person {

  // Define additional properties of the class
  public $rollNo;
  public $marks;

  // Define a constructor method for the class
  public function __construct($name, $age, $rollNo, $marks) {
    parent::__construct($name, $age);
    $this->rollNo = $rollNo;
    $this->marks = $marks;
  }

  // Define a method to display the student's information
  public function displayStudentInfo() {
    echo "Name: " . $this->name . "<br>";
    echo "Age: " . $this->age . "<br>";
    echo "Roll Number: " . $this->rollNo . "<br>";
```

```
    echo "Marks: " . $this->marks . "<br>";
  }
}

// Create an instance of the 'Person' class
$person = new Person("John Doe", 30);

// Call the 'displayInfo()' method of the 'Person' class
$person->displayInfo();

// Create an instance of the 'Student' class
$student = new Student("Jane Smith", 20, "A123", 85);

// Call the 'displayInfo()' method of the 'Person' class from the 'Student' class
$student->displayInfo();

// Call the 'displayStudentInfo()' method of the 'Student' class
$student->displayStudentInfo();

?>
```

5. **Write a PHP script to demonstrate Form Data Handling using Get and Post methods.**

HTML form

```
<!DOCTYPE html>
<html>
<head>
	<title>Form Data Handling</title>
</head>
<body>
	<form method="GET" action="handle_form_data.php">
		<label for="name">Name:</label>
		<input type="text" name="name" id="name">
		<br><br>
		<label for="email">Email:</label>
		<input type="email" name="email" id="email">
		<br><br>
		<input type="submit" value="Submit (GET)">
	</form>
	<br>
	<form method="POST" action="handle_form_data.php">
		<label for="username">Username:</label>
		<input type="text" name="username" id="username">
		<br><br>
		<label for="password">Password:</label>
		<input type="password" name="password" id="password">
		<br><br>
		<input type="submit" value="Submit (POST)">
	</form>
</body>
</html>
```

PHP script (handle_form_data.php):

```
<!DOCTYPE html>
<html>
<head>
	<title>Form Data Handling</title>
</head>
<body>
	<h2>Form data submitted via GET method:</h2>
	<?php
	if (isset($_GET['name'])) {
		echo "Name: " . $_GET['name'] . "<br>";
	}
	if (isset($_GET['email'])) {
		echo "Email: " . $_GET['email'] . "<br>";
	}
	?>

	<br><br>

	<h2>Form data submitted via POST method:</h2>
	<?php
	if (isset($_POST['username'])) {
		echo "Username: " . $_POST['username'] . "<br>";
	}
	if (isset($_POST['password'])) {
		echo "Password: " . $_POST['password'] . "<br>";
	}
	?>
</body>
</html>
```

6. **Design a database in MYSQL. Create table in database. Store, Update, Delete and Retrieve data from the table. Display the data from the table.**

1. **Create a database**

```
CREATE DATABASE library;
```

2. **Create tables in the database**

```
CREATE TABLE books (
  book_id INT(11) NOT NULL AUTO_INCREMENT,
  book_title VARCHAR(255) NOT NULL,
  author VARCHAR(255) NOT NULL,
  publisher VARCHAR(255) NOT NULL,
  category VARCHAR(255) NOT NULL,
  PRIMARY KEY (book_id)
);

CREATE TABLE users (
  user_id INT(11) NOT NULL AUTO_INCREMENT,
  first_name VARCHAR(255) NOT NULL,
  last_name VARCHAR(255) NOT NULL,
  email VARCHAR(255) NOT NULL,
```

```
  phone_number VARCHAR(20) NOT NULL,
  address VARCHAR(255) NOT NULL,
  PRIMARY KEY (user_id)
);

CREATE TABLE borrowed_books (
  borrow_id INT(11) NOT NULL AUTO_INCREMENT,
  user_id INT(11) NOT NULL,
  book_id INT(11) NOT NULL,
  borrow_date DATE NOT NULL,
  return_date DATE NOT NULL,
  PRIMARY KEY (borrow_id),
  FOREIGN KEY (user_id) REFERENCES users(user_id),
  FOREIGN KEY (book_id) REFERENCES books(book_id)
);
```

3. **Define the fields in the tables**
    - The books table has the following fields:

book_id: an auto-incremented integer that serves as the primary key
book_title: the title of the book
author: the name of the book's author
publisher: the name of the book's publisher
category: the category of the book (e.g. science fiction, romance, etc.)

- The users table has the following fields:

user_id: an auto-incremented integer that serves as the primary key
first_name: the user's first name
last_name: the user's last name
email: the user's email address
phone_number: the user's phone number
address: the user's address

- The borrowed_books table has the following fields:

borrow_id: an auto-incremented integer that serves as the primary key
user_id: the ID of the user who borrowed the book
book_id: the ID of the borrowed book
borrow_date: the date the book was borrowed
return_date: the date the book is due to be returned

4. **Establish relationships between tables if necessary**

Insert data into the books table:

```
INSERT INTO books (book_title, author, publisher, category)
VALUES ('The Great Gatsby', 'F. Scott Fitzgerald', 'Charles Scribner\'s Sons', 'Classics');
```

Update data in the **users** table:

```
UPDATE users
SET phone_number = '123-456-7890'
WHERE user_id = 1;
```

Delete data from the borrowed_books table:

```
DELETE FROM borrowed_books
WHERE book_id = 1;
```

Retrieve data from the books table:

```
SELECT *
FROM books;
```

7. Write a PHP script to store, retrieve and delete cookies on your local machine.

```
<?php

// Set a cookie
setcookie("username", "John Doe", time() + (86400 * 30), "/");

// Retrieve a cookie
if(isset($_COOKIE["username"])) {
  echo "Welcome " . $_COOKIE["username"] . "!<br>";
} else {
  echo "No cookie found.<br>";
}

// Delete a cookie
setcookie("username", "", time() - 3600, "/");

?>
```

8. Write a PHP script to store, retrieve and delete data using session variables.

```
<?php
session_start();

// Set session variables
$_SESSION["username"] = "JohnDoe";
$_SESSION["email"] = "johndoe@example.com";

// Retrieve session variables
$username = $_SESSION["username"];
$email = $_SESSION["email"];

echo "Username: " . $username . "<br>";
echo "Email: " . $email . "<br>";

// Delete session variables
unset($_SESSION["username"]);
unset($_SESSION["email"]);
?>
```