PHP arrays and conditionals

4 Define an array and different types of conditional statements in PHP.? Explain
about the types of arrays in PHP with examples .

Let’s go step by step 👇

🧩 1. Definition of an Array in PHP

An **array** in PHP is a **special variable** that can hold **multiple values** under a **single name**, and each value can be accessed using an **index** or a **key**.

Syntax:

```php
$arrayName = array(value1, value2, value3, ...);
```

Example:

```php
$fruits = array("Apple", "Banana", "Cherry");
echo $fruits[1];  // Output: Banana
```

⚙️ 2. Conditional Statements in PHP

Conditional statements are used to **make decisions** based on conditions (true/false).
PHP supports several types of conditional statements:

🔸 a) `if` Statement

Executes code if a condition is **true**.

```php
$age = 20;
if ($age >= 18) {
```

```
    echo "You are an adult.";
}
```

🔸 b) `if...else` Statement

Executes one block if condition is **true**, another if **false**.

```php
$marks = 35;
if ($marks >= 40) {
    echo "Pass";
} else {
    echo "Fail";
}
```

🔸 c) `if...elseif...else` Statement

Used to check **multiple conditions**.

```php
$score = 85;

if ($score >= 90) {
    echo "Grade A";
} elseif ($score >= 75) {
    echo "Grade B";
} else {
    echo "Grade C";
}
```

🔸 d) `switch` Statement

Used when you have **many conditions** for the same variable.

```php
```

```php
$day = "Tuesday";

switch ($day) {
    case "Monday":
        echo "Start of the week";
        break;
    case "Tuesday":
        echo "Second day of the week";
        break;
    default:
        echo "Other day";
}
```

🧮 3. Types of Arrays in PHP

There are **three main types of arrays** in PHP:

🔹 a) Indexed Array

- Stores data with **numeric indexes** (starting from 0).
- Access elements using the index number.

Example:

```php
$cars = array("BMW", "Audi", "Mercedes");

echo $cars[0];  // Output: BMW
```

OR

```php
$cars[0] = "BMW";
$cars[1] = "Audi";
$cars[2] = "Mercedes";
```

🔹 b) Associative Array

- Uses **named keys** instead of numeric indexes.

Example:

```php
$age = array("John" => 25, "Peter" => 30, "Mary" => 22);

echo $age["Peter"];  // Output: 30
```

🔹 c) Multidimensional Array

- Contains **one or more arrays inside another array** (like a table).

Example:

```php
$students = array(
    array("Ravi", 85, "A"),
    array("Priya", 78, "B"),
    array("Kiran", 92, "A+")
);

echo $students[0][0];  // Output: Ravi
echo $students[2][2];  // Output: A+
```

🧠 Summary Table

Type of Array	Index Type	Example Key	Example Access
Indexed	Numeric	0, 1, 2	`$arr[0]`
Associative	String	"name", "age"	`$arr["name"]`
Multidimensional	Arrays inside arrays	Both	`$arr[0][1]`

Would you like me to include **a complete PHP program** demonstrating all 3 types of arrays together (for your notes or practical exam)?

3 Explain database connectivity in PHP with reference to MYSQL with an example

Let's clearly explain this 👇

💡 Database Connectivity in PHP with MySQL

Database connectivity in PHP means connecting a **PHP script** to a **MySQL database** so that we can **insert**, **retrieve**, **update**, or **delete** data.

🧱 Steps for PHP–MySQL Connectivity

🔹 Step 1: Create a Database

Example SQL:

```sql
CREATE DATABASE studentDB;
```

🔹 Step 2: Create a Table

```sql
CREATE TABLE students (
    id INT(5) AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    age INT(3),
    course VARCHAR(50)
);
```

⚙️ Step 3: Connect PHP to MySQL

PHP provides two main ways:

1. **MySQLi (MySQL Improved)**
2. **PDO (PHP Data Objects)**

Here, we'll use **MySQLi**, which is simpler for beginners.

✅ Example: PHP Script for MySQL Connection

```php
<?php
// Step 1: Define connection variables
$servername = "localhost";
$username = "root";
$password = "";
$database = "studentDB";

// Step 2: Create connection
$conn = mysqli_connect($servername, $username, $password, $database);

// Step 3: Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully!";

// Step 4: Insert data into table
$sql = "INSERT INTO students (name, age, course) VALUES ('Ravi', 21, 'Computer Science')";

if (mysqli_query($conn, $sql)) {
    echo "<br>New record created successfully";
} else {
    echo "<br>Error: " . $sql . "<br>" . mysqli_error($conn);
}

// Step 5: Close connection
mysqli_close($conn);
?>
```

🧠 Explanation of Code:

Step	Function	Description
`mysqli_connect()`	Creates a connection between PHP and MySQL	
`mysqli_connect_error()`	Displays connection errors	
`mysqli_query()`	Executes SQL queries (SELECT, INSERT, UPDATE, etc.)	
`mysqli_close()`	Closes the database connection	

🔍 Alternative: Using PDO (for advanced use)

```php
<?php
try {
    $conn = new PDO("mysql:host=localhost;dbname=studentDB", "root", "");
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully!";
} catch(PDOException $e) {
    echo "Connection failed: " . $e->getMessage();
}
?>
```

🧾 Summary

Method	Function Used	Error Handling
MySQLi	`mysqli_connect()`	Simple and procedural
PDO	`new PDO()`	Object-oriented, better for large projects

Would you like me to show a version that **fetches and displays records from the database** as well (using `SELECT` query)?

2 Explain how to read form data in PHP from web form controls like text boxes,
radio buttons, and lists with examples.

Let's break this down clearly 👇

🧩 Reading Form Data in PHP

In PHP, form data is collected from HTML forms using the `$_GET` or `$_POST` **superglobal arrays**.
These arrays store values submitted by form controls like **text boxes**, **radio buttons**, and **lists (dropdowns)**.

⚙️ 1️⃣ Steps to Read Form Data

1. **Create an HTML form**
2. **Specify the method (`GET` or `POST`)**
3. **Access data in PHP using `$_GET['name']` or `$_POST['name']`**

🧱 Example Form (HTML + PHP)

🔹 HTML File: `form.html`

```html
<!DOCTYPE html>
<html>
<head>
  <title>Form Example</title>
</head>
<body>
  <h2>Student Information Form</h2>
  <form action="readform.php" method="post">
```

```
    <!-- Text Box -->
    Name: <input type="text" name="studentName"><br><br>

    <!-- Radio Buttons -->
    Gender:
    <input type="radio" name="gender" value="Male"> Male
    <input type="radio" name="gender" value="Female"> Female<br><br>

    <!-- List (Dropdown) -->
    Course:
    <select name="course">
      <option value="BCA">BCA</option>
      <option value="B.Tech">B.Tech</option>
      <option value="MCA">MCA</option>
    </select><br><br>

    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

🔹 PHP File: `readform.php`

```
php

<?php
// Read data sent from form using POST method
$name = $_POST['studentName'];
$gender = $_POST['gender'];
$course = $_POST['course'];

echo "<h3>Student Details</h3>";
echo "Name: " . $name . "<br>";
echo "Gender: " . $gender . "<br>";
echo "Course: " . $course . "<br>";
?>
```

🧠 Explanation

Form Control	HTML Tag	How to Access in PHP	Example Output
Text Box	`<input type="text" name="studentName">`	`$_POST['studentName']`	"Ravi"
Radio Button	`<input type="radio" name="gender" value="Male">`	`$_POST['gender']`	"Male"
Drop-down List	`<select name="course">`	`$_POST['course']`	"BCA"

🗒️ Alternative: Using `$_GET` Method

If you use `method="get"` in the form, the values will appear in the **URL** and can be read using:

```php
$name = $_GET['studentName'];
```

✅ Output Example (After Form Submission)

```makefile
Student Details
Name: Ravi
Gender: Male
Course: BCA
```

Would you like me to add **checkbox input** (for selecting multiple hobbies) to this form as well?

1 Write a PHP script to open, close, read and write into a file and Write a PHP

program for a simple calculator.

Let's go step-by-step 👇

🗒️ Question 1: PHP Script to Open, Read, Write, and Close a File

PHP provides **file handling functions** that allow you to create, open, read, write, and close files.

✅ Example: `file_handling.php`

```php
<?php
// File name
$filename = "sample.txt";

// --- Open or Create File ---
$file = fopen($filename, "w") or die("Unable to open file!");

// --- Write to the File ---
$text = "Hello, this is a sample text written using PHP.\n";
fwrite($file, $text);
fwrite($file, "File handling in PHP is easy!\n");

// --- Close the File ---
fclose($file);
echo "Data written successfully!<br>";

// --- Open the File for Reading ---
$file = fopen($filename, "r") or die("Unable to open file!");

// --- Read the File ---
echo "<h3>File Content:</h3>";
while (!feof($file)) {
    echo fgets($file) . "<br>";
}

// --- Close the File ---
```

```
fclose($file);
?>
```

🧠 Explanation:

Function	Description
`fopen(filename, mode)`	Opens a file in specified mode (`r`, `w`, `a`, etc.)
`fwrite(file, text)`	Writes data to the file
`fgets(file)`	Reads one line from the file
`feof(file)`	Checks for end-of-file
`fclose(file)`	Closes the file

File Modes:

Mode	Meaning
`r`	Read only
`w`	Write only (clears old content)
`a`	Append to file
`r+`	Read and write

🧮 Question 2: Simple PHP Calculator

A calculator performs basic operations like **Addition, Subtraction, Multiplication, and Division**.

✅ Example: `calculator.php`

```php
<!DOCTYPE html>
<html>
<head>
  <title>Simple Calculator</title>
</head>
<body>
  <h2>PHP Simple Calculator</h2>
  <form method="post">
    Number 1: <input type="text" name="num1" required><br><br>
    Number 2: <input type="text" name="num2" required><br><br>

    <input type="submit" name="operator" value="Add">
    <input type="submit" name="operator" value="Subtract">
    <input type="submit" name="operator" value="Multiply">
    <input type="submit" name="operator" value="Divide">
  </form>

  <?php
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $num1 = $_POST['num1'];
    $num2 = $_POST['num2'];
    $op = $_POST['operator'];

    if (is_numeric($num1) && is_numeric($num2)) {
      switch ($op) {
        case "Add":
          $result = $num1 + $num2;
          break;
        case "Subtract":
          $result = $num1 - $num2;
          break;
        case "Multiply":
          $result = $num1 * $num2;
          break;
        case "Divide":
          if ($num2 == 0)
            $result = "Cannot divide by zero!";
          else
            $result = $num1 / $num2;
          break;
      }
```

```
        echo "<h3>Result: $result</h3>";
    } else {
        echo "<h3>Please enter valid numbers.</h3>";
    }
}
?>
</body>
</html>
```

🧠 Explanation:

Concept	Description
`$_POST`	Reads data from form fields
`switch`	Decides which operation to perform
`is_numeric()`	Ensures the inputs are valid numbers
`$_SERVER["REQUEST_METHOD"]`	Checks if form is submitted using POST

✅ Sample Output

```yaml
Number 1: 10
Number 2: 5
Operation: Multiply

Result: 50
```

Would you like me to combine **both programs** (file handling + calculator) in one PHP file as a single project example (for your practical exam)?