## Q1. Write a PHP script to get the PHP version and configuration information

Ans.

Sure, here's a PHP script that retrieves the PHP version and configuration information:

```
<?php


// Get PHP version
$phpVersion = phpversion();
// Get PHP configuration information
$phpConfig = phpinfo();
// Display PHP version
echo "PHP Version: $phpVersion <br><br>";
// Display PHP configuration information
echo "PHP Configuration Information: <br>";
echo $phpConfig;
?>
```

Output

```
PHP Version 8.2.12
```

Save this script with a .php extension, for example, php_info.php, and then you can run it on your server to see the PHP version and configuration information. Keep in mind that the PHP phpinfo() function will output a lot of information, so you might want to use it cautiously, especially on a public servers

## Q .2Create a simple HTML form and accept the user name and display the name through PHP echo statement.

Ans.

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>PHP Name Display</title>
</head>
<body>
  <h2>Enter Your Name</h2>
  <form method="post" action="">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required>
    <button type="submit">Submit</button>
  </form>

  <?php
  // Check if form is submitted and name is set
  if (isset($_POST['name'])) {
    // Get the submitted name
    $name = $_POST['name'];
    // Display the name using PHP echo statement
    echo "<h2>Hello, $name!</h2>";
  }
  ?>
```

```
</body>

</html>
```

Output

```
Hello, deepak!
```

This HTML form allows users to enter their name and submit the form. Once the form is submitted, the PHP code checks if the name is set in the $_POST superglobal array. If it is, the name is retriev ed and displayed using a PHP echo statement.

## Q .3 Write a PHP script to display string, values within a table:

## Note: Use HTML table elements into echo.

Ans.

Certainly! Here's a PHP script that displays strings and their corresponding values within an HTML table:

Code

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>String Values Table</title>

  <style>

    table {

      width: 50%;

      border-collapse: collapse;

    }

    th, td {
            border: 1px solid black;

            padding: 8px;

            text-align: left;

        }

    </style>

</head>

<body>


<?php

// Define an associative array with string keys and corresponding values

$stringValues = array(

    "Name" => "John Doe",

    "Age" => 30,

    "Country" => "USA",

    "Email" => "john.doe@example.com"

);

?>


<h2>String Values Table</h2>


<table>

    <tr>

        <th>String</th>

        <th>Value</th>

    </tr>

    <?php

    // Loop through the associative array and display each string and its corresponding
value in a table row
```

```
  foreach ($stringValues as $string => $value) {

    echo "<tr>";

    echo "<td>$string</td>";

    echo "<td>$value</td>";

    echo "</tr>";

  }

  ?>

</table>


</body>

</html>
```

| **String** | **Value** |
|---|---|
| Name | John Doe |
| Age | 30 |
| Country | USA |
| Email | john.doe@example.com |

Output

This script defines an associative array called $stringValues, where each key represents a string and its corresponding value. It then generates an HTML table with two columns: "String" and "Value". Inside a loop, it iterates over the associative array and outputs each string and its corresponding value in a table row.

## Q .4 Arithmetic operations on character variables: $d = 'A00'. Using this variable print the following numbers.

## Sample Output : A01 A02 A03 A04 A05

## Ans.

You can achieve this by converting the character variable to a numeric value, performing arithmetic operations, and then formatting the result back to the desired format. Here's how you can do it in PHP

```
<?php


// Initial character variable

$d = 'A00';
```

```
// Loop to print numbers from A01 to A05

for ($i = 1; $i <= 5; $i++) {

   // Convert character variable to numeric value

   $numericPart = intval(substr($d, 1)) + $i;

   // Format the result back to the desired format (Axx)

   $result = $d[0] . sprintf("%02d", $numericPart);

   // Print the result

   echo $result . "\n";

}


?>
```

```
Output

A01

A02

A03

A04

A05
```

It first extracts the numeric part from the character variable, converts it to an integer, adds the loop index ($i), formats it back to a two-digit number, and concatenates it with the first character ('A') to get the desired output.

## Q 5. Write a PHP script to get the full URL.

## Ans.

You can use PHP superglobal variables to get the full URL. Here's a PHP script to achieve that:

```
<?php


// Get the protocol

$protocol = isset($_SERVER['HTTPS']) && $_SERVER['HTTPS'] === 'on' ? "https" : "http";


// Get the host

$host = $_SERVER['HTTP_HOST'];
```

```
// Get the requested URI

$uri = $_SERVER['REQUEST_URI'];


// Combine the parts to form the full URL

$fullUrl = $protocol . "://" . $host . $uri;


// Display the full URL

echo "Full URL: " . $fullUrl;


?>
```

Output

```
Full URL:
http://localhost/pratical_php/05.php
```

This script uses $_SERVER superglobal variables to obtain the protocol (HTTPS if secure, otherwise HTTP), the host (HTTP_HOST), and the requested URI (REQUEST_URI). Then, it concatenates these parts to form the full URL and displays it using a PHP echo statement.

## Q 6. Write a PHP program to print out the multiplication table up to 6*6.

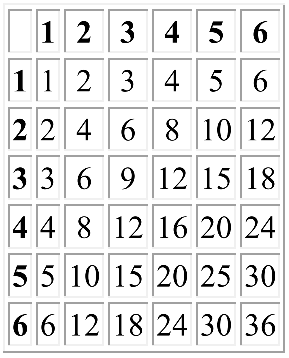
## Ans.

Here's a PHP program to print out the multiplication table up to 6*6:

```
<?php


// Define the maximum number for the multiplication table

$max = 6;


// Print the multiplication table headers
```

```
echo "<h2>Multiplication Table up to $max * $max</h2>";

echo "<table border='1'>";

echo "<tr><th>&nbsp;</th>";


// Print column headers

for ($i = 1; $i <= $max; $i++) {

   echo "<th>$i</th>";

}

echo "</tr>";


// Print table rows

for ($i = 1; $i <= $max; $i++) {

   echo "<tr>";

   echo "<th>$i</th>"; // Row header

   // Print the multiplication results for each column

   for ($j = 1; $j <= $max; $j++) {

      echo "<td>" . ($i * $j) . "</td>";

   }

   echo "</tr>";

}

echo "</table>";



?>
```

Output



| | **1** | **2** | **3** | **4** | **5** | **6** |
|---|---|---|---|---|---|---|
| **1** | 1 | 2 | 3 | 4 | 5 | 6 |
| **2** | 2 | 4 | 6 | 8 | 10 | 12 |
| **3** | 3 | 6 | 9 | 12 | 15 | 18 |
| **4** | 4 | 8 | 12 | 16 | 20 | 24 |
| **5** | 5 | 10 | 15 | 20 | 25 | 30 |
| **6** | 6 | 12 | 18 | 24 | 30 | 36 |

This program generates an HTML table representing the multiplication table up to 6*6. It first prints the table headers, including the numbers from 1 to 6 as column headers and row headers. Then, it calculates and prints the multiplication results for each cell in the table.

## Q 7. Write a PHP program that implements multiple catch blocks to handle different types of exceptions.

```
<?php


try {
   // Code that may throw exceptions
   $result = 10 / 0; // This will throw a division by zero error
} catch (DivisionByZeroError $e) {
   // Catch the specific DivisionByZeroError exception
   echo "Caught DivisionByZeroError: " . $e->getMessage();
} catch (Exception $e) {
   // Catch any other type of exception
   echo "Caught Exception: " . $e->getMessage();
}

?>
```

Output

```
Caught
DivisionByZ
eroError:
Division by
zero
```

- We have a try block where we execute code that may throw exceptions.
- Inside the try block, we attempt to perform a division by zero operation, which will throw a DivisionByZeroError.
- We have two catch blocks:
- The first catch block catches the specific DivisionByZeroError exception and prints a message.
- The second catch block catches any other type of exception (as it extends the base Exception class) and prints a generic message.
- This setup allows you to handle different types of exceptions separately, providing specific error handling for each type.

Q 8. Write a PHP program that implements multiple catch blocks to handle different types of exceptions

Ans .

```
<?php


try {
  // Code that may throw exceptions
  $randomNumber = rand(1, 3);


  if ($randomNumber == 1) {
    throw new InvalidArgumentException("Invalid argument exception occurred");
  } elseif ($randomNumber == 2) {
    throw new OutOfRangeException("Out of range exception occurred");
  } else {
    throw new RuntimeException("Runtime exception occurred");
  }
} catch (InvalidArgumentException $e) {
  // Catch and handle InvalidArgumentException
  echo "Caught InvalidArgumentException: " . $e->getMessage() . "<br>";
} catch (OutOfRangeException $e) {
  // Catch and handle OutOfRangeException
  echo "Caught OutOfRangeException: " . $e->getMessage() . "<br>";
} catch (RuntimeException $e) {
  // Catch and handle RuntimeException
  echo "Caught RuntimeException: " . $e->getMessage() . "<br>";
} catch (Exception $e) {
  // Catch any other type of exception
  echo "Caught Exception: " . $e->getMessage() . "<br>";
}


      ?>
```

- It has a try block where the code that may throw exceptions is placed.
- Inside the try block, we generate a random number and throw different types of exceptions based on its value.
- We have multiple catch blocks:
- Each catch block handles a specific type of exception by catching the exception and displaying a corresponding error message.
- The last catch block catches any other type of exception and displays a generic error message.

Output

```
Caught InvalidArgumentException:
Invalid argument exception occurred
```

## Q 9. Write a PHP class called 'Shape' with an abstract method 'calculate Area ()'.

## Create two subclasses, 'Triangle' and 'Rectangle', that implement the 'calculate Area ()' method.

Ans.

```
<?php


abstract class Shape {
  abstract public function calculateArea();
```

```
}

class Triangle extends Shape {
  private $base;
  private $height;

  public function __construct($base, $height) {
    $this->base = $base;
    $this->height = $height;
  }

  public function calculateArea() {
    return 0.5 * $this->base * $this->height;
  }
}

class Rectangle extends Shape {
  private $length;
  private $width;

  public function __construct($length, $width) {
    $this->length = $length;
    $this->width = $width;
  }

  public function calculateArea() {
    return $this->length * $this->width;
  }
```

```
}


// Example usage

$triangle = new Triangle(5, 8);

echo "Area of Triangle: " . $triangle->calculateArea() . "\n";


$rectangle = new Rectangle(4, 6);

echo "Area of Rectangle: " . $rectangle->calculateArea() . "\n";


?>
```

Output

```
Area of Triangle: 20 Area of
Rectangle: 24
```

This code defines an abstract class Shape with an abstract method calculateArea(). Then, two subclasses Triangle and Rectangle are created which extend the Shape class and implement the calculateArea() method according to their respective shapes. Finally, examples of usage are provided to calculate the areas of a triangle and a rectangle.

## Q 10. Write a PHP class called 'Student' with properties like 'name', 'age', and 'grade'.

## Implement a method to display student information.

Ans.

Here's a PHP class Student with properties for name, age, and grade, along with a method to display student information:

```
<?php


class Student {

   private $name;

   private $age;
```

```
    private $grade;

    public function __construct($name, $age, $grade) {
        $this->name = $name;
        $this->age = $age;
        $this->grade = $grade;
    }

    public function displayInformation() {
        echo "Name: " . $this->name . "\n";
        echo "Age: " . $this->age . "\n";
        echo "Grade: " . $this->grade . "\n";
    }
}

// Example usage
$student1 = new Student("John Doe", 18, "12th");
echo "Student 1 Information:\n";
$student1->displayInformation();
echo "\n";

$student2 = new Student("Jane Smith", 17, "11th");
echo "Student 2 Information:\n";
$student2->displayInformation();

?>
```

```
Student 1 Information:
Name: John Doe Age: 18
Grade: 12th Student 2
Information: Name: Jane
Smith Age: 17 Grade: 11th
```

In this code, the Student class has private properties for name, age, and grade. The constructor initializes these properties when a new Student object is created. The displayInformation() method is used to display the student's information. Finally, two instances of the Student class are created and their information is displayed using the displayInformation() method.