Assignment :- 1

1 Explain form elements with example.

Ans :- In PHP, form elements are used to collect user input on a web page. They allow users to enter data, make selections and submit the information to a server for processing. Here is basic example of form elements in PHP.

→ lets create a simple html form with two elements - a text input and a submit button.

```
<!DOCTYPE html>
<html>
<head>
<title> Simple Form </title>
</head>
<body>
    <h1> Enter your Name :- </h1>
<form method = "post" action = "process.php
    ">
<input type = "text" name = "name" placeholder
    = " your Name ">
<input type = "submit" value = "submit" >
</form>
</body>
</html>
```

In this example the form has an input element with the type "text" and the name "name". When the user enters their name in the text box and clicks the "submit" button, the form data will be sent to the server-side PHP script

specified in the "action" attribute of the form element.

→ Now lets create the "process.php" file to handle the submitted data

```
<? php
if ( $_SERVER [ "REQUEST_METHOD" ] == =
  "POST" ) {
  $ name = $_POST [ "name" ] ;
  echo " Hello, " . $ name . "! success ." ;
}
? >
```

→ we check if the form was submit using $_SERVER ["REQUEST_METHOD"] = = "POST".

→ This ensure that the code inside the if block executes only when the form is submitted via the post method.

→ we use the echo statement to display the value of the $ name on the screen.

→ we declare a variable $ name and assign the string the store $_POST on it.

→ The Form elements is collect the data through submitted via HTTP POST method.

2 Explain Passing Variables technique with example.

Ans :- In PHP, arguments to a function can be passed by value or passed by reference. By default values of actual arguments are passed by value to formal arguments which become local variables inside the function. Hence, modification to these variables doesn't change value of actual argument variable.

→ When arguments are passed by reference change in value of formal argument is reflected in actual argument variable because the former is a reference to latter.

→ Thus Pass by reference mechanism helps in indirectly manipulating data in global space.

→ it also helps in overcoming the fact that a function can return only one variable.

→ In PHP the variables are passed through two ways (1). Pass by value (2) Pass by references.

1. Pass By Value :-

In PHP, Primitive data types are passed by value, meaning a copy of the value is sent to the function.

- Example :-

```
function modifyValue ($num) {
    $num = $num * 2;
}
$value = 10;
modifyValue ($value);
echo $value;
```

→ the original variable '$value' remains unchanged because the function works with a copy of the value.

2. Pass By Reference :-

In PHP objects and arrays are passed by reference allowing changes inside a function to affect the original variable

- Example :-

```
function test (&$arr) {
    $arr[] = 4;
}
$arr = [1, 2, 3];
test ($arr);
print_r ($arr);
```

→ The function modifies the original 'array' because it passed by reference using the '&' symbol.

Assignment :- 2

8 Explain conditional statements of PHP

Ans :- Conditional statements are used to perform actions based on different conditions. Some times when we write a program, we want to perform some different actions for different actions. We can solve this by using conditional statements.

1. if Statement :-

The 'if' statements is used to execute a block of code if a certain condition is true. The condition is enclosed in parentheses and if it evaluates to true, the code inside the curly braces is executed.

Example :-

```
$age = 20;
if ($age >= 18) {
echo "you are eligible";
}
```

2. if - else statement :-

The 'if - else' statement allows you to execute one block of code if the condition is true and a different block of code if the condition is false.

Example :-

```
$temperature = 28;
if ($temperature > 30) {
```

```
    echo " It's hot outside ! ";
} else {
    echo " It's not too hot ";
}
```

3. if-else if-else statement :-

The 'if-else if-else' statement lets you test multiple conditions sequentially. The code block associated with the first true condition will be executed.

Example :-

```
$score = 85;
if ($score >= 90) {
    echo " you got An A. ";
} else if ($score >= 80) {
    echo " you got a B. ";
} else if ($score >= 70) {
    echo " you got a C. ";
} else {
    echo " you need to improove ";
}
```

4. switch statement :-

The 'switch' statement allows you to evaluate a single expression against multiple possible cases. it's an alternative to using a series of 'if-else if' statements.

Example :-

```
$day = "wednesday";
switch ($day) {
case "monday":
    echo "its the start of the week";
    break;
case "wednesday":
    echo "its the middle of the week";
    break;
case "Friday":
    echo "its the end of the week";
    break;
default:
    echo "its just another day";
    break;
}
```

4 Explain five MySQL connectivity functions.

Ans :- MySQL is a popular open-source relational database management system. it is widely used for storing, managing and retriving data in various application.

1. mysqli_connect() :-

This function is used to establish a connection to a MySQL database. it takes parameters such as the server name username, password and database name. it returns a connection object that can be used to perform various database operations.

2. mysqli_close() :-

This function is used to close the connection to the MySQL database. It takes the connection object as a parameter and terminates the connection.

3. mysqli_select_db() :-

This function is used to select a specific database to work with. it takes the connection object and the database name as parameters. once the database is selected, all subsequent queries and operation will be performed on that database.

4. mysqli_query() :-

This function is used to execute a SQL query on the selected database. it takes can be SELECT, INSERT, UPDATE, or DELETE statement. it returns a result object that can be used to fetch and manipulate the query result.

5. mysqli_fetch_assoc() :-

This function is used to fetch a row from the result set as an associative array. This function is commonly used in a loop to fetch multiple rows from the result set.

Assignment :- 3

Q What is array ? explain any five array functions.

Ans :- An array in PHP is a data structure that can hold multiple values of different types. It allows you to store and access multiple values using a single variable.

1. count() :-

The count() function is used to count the number of elements in an array. It takes the array as a parameter and returns the total count of elements in the array.

2. array_push() :-

The array_push() function is used to add one or more elements to the end of an array. The function modifies the original array by adding the elements at the end.

3. array_pop() :-

The array_pop() function is used to remove and return the last element of an array. It takes the array as a parameter and removes the last element from the array. The removed element is returned as the function's result.

4. array_merge() :-

The array_merge() function is used to merge two or more arrays into a single array. it takes multiple arrays as parameters and combines them into a new array. The resulting array contains all the elements from the input arrays.

5. array_search() :-

The array_search() function is used to search for a specific value in an array and returns the corresponding key if found. it takes the value to be searched for as the first parameter, followed by the array to be searched.

6. Explain class with example in PHP

Ans :- A class in PHP is a blueprint for creating objects. it defines the properties and methods that an object of that class can have.

→ The properties represent the state or characteristics of the object, while the methods define the behaviour or actions that the object can perform.

→ By creating objects from the class, we can assign specific values to the properties and call the methods to perform actions specific to each object.

→ For example, let's say we have a class called "Person". it could have properties like "name", "age" and "gender" and methods like "sayHello" or "calculate Age()".

→ Classes help organize and structure code, making it easier to manage and reuse.

- Example of class :-

```
class car {
    public $brand;
    public $color;
    public $price;

    public function startEngine() {
        return "The car engine is starting!";
    }
    public function accelerate() {
        return "The car is accelerating!";
    }
    public function brake() {
        return "The car is braking!";
    }
}
    $car1 = new car1();
    $car1 -> brand = "Fortuner";
    $car1 -> color = "Red";
    $car1 -> price = "25000";
echo $car1 -> startEngine();
echo $car1 -> accelerate();
echo $car1 -> brake();
```

Assignment :- 4

7. How to execute PHP script using Python and Python script using PHP ?

Ans :- To execute a PHP script using Python in more detail you can use the sub process module's run() function.

```
import subprocess

result = subprocess.run(['PHP', 'Path/to/your/script.php'], capture_output=True, text=True)

output = result.stdout
```

→ In this example we use the 'subprocess.run()' function to run the PHP script.

→ The 'C:\PHP', 'Path/to/your/script.php'] argument specifies the command to execute.

→ where 'PHP' is the PHP interpreter and 'Path/to/your/script.php' is the path to your PHP script.

→ By setting 'capture_output = True' and 'text = True' we capture the output of the PHP script and decode it as text.

→ The output of the PHP script is stored in the 'stdout' attribute of the 'result' object.

→ To execute a python script using PHP exec ('python path/to/your/script.py', $output, $return);

```
<?php
exec('python path/to/your/script.py', $output, $return);
$output_string = implode("\n", $output);
?>
```

→ In this example, we use the 'exec()' function to run the python script.

→ The 'python path/to/your/script.py' argument specifies the command to execute, where 'path/to/your/script.py' is the path to your python script.

→ By passing the '$output' and '$return' variables as references, we can capture the output of the python script in the '$output' array.

→ We then use 'implode("\n", $output)' to concatenate the array elements into a string, '$output_string', which contains the output of the python script.

→ Remember to replace '/path/to/your/script.py' with the actual paths to your PHP and python scripts, respectively.

8 Explain Flask request object with example.

Ans :- The Flask 'request' object provides access to incoming request data in Flask applications. it allows you to access data sent by the client, such as form data, query, parameters, and headers.

```
python
from flask import Flask, request

app = Flask (__name__)

@app.route ('/example', methods = ['POST'])
def example ():
    name = request.form.get ('name')
    age = request.args.get ('age')
    user_agent = request.headers.get ('User-Agent')

return f "Hello {name}, you are {age} years old. your user-Agent is {user-agent}."

if __name__ == '__main__':
    app.run ()
```

→ In this example, we define a Flask route '/example' that accepts a POST request.

→ Inside the route function, we can access the incoming request data using the 'request' object.

→ To access form data we use 'request.form.get ('name')', where 'name'

is the name of the form field. Similarly to access query parameters we use 'request.args.get('age')', where 'age' is the name of the query parameter.

→ We can also access headers using 'request.headers.get('User-Agent')', where 'User-Agent' is the name of the header we want to retrive.

→ Finally, we process the data and return a response using the retrived values.

→ In this example, we return a string that includes the name, age, and user-Agent.

→ Remember to import the necessary modules ('Flask' and 'request') and run the Flask application using 'app.run()'.

- **Flask object :-**

  1. **request.form :-**

  This attribute give you access to the form data submitted with a POST request. you can retrive form values using the 'get()'.

2. request.args :-

This attributes allows you to access the query Parameters sent with the request.

3. request.headers :-

This attributes provides access to the headers sent with the request.

4. request.method :-

This attribute gives you the HTTP method used in the request, such as GET, POST, PUT, DELETE etc.

5. request.files :-

This attributes allows you to access files uploaded with the request you can retrieve file data using the get() method.