Experiment No. 2

Title: Arrays, Image and File handling functions, User Defined function in PHP

Batch:A3 **Roll No.:16010421073** **Experiment No.:2**

Aim: Write PHP script for demonstrating use of arrays, functions, image and file handling functions.

Resources needed: Windows OS, Web Browser, Editor, and XAMPP Server

---

Pre Lab/ Prior Concepts:

Students should have prior knowledge of HTML/CSS/Basic Programming.

Theory:

What is PHP Arrays
Arrays are complex variables that allow us to store more than one value or a group of values under a single variable name.

Types of Arrays in PHP
There are three types of arrays that you can create. These are:

1. Indexed array — An array with a numeric key.
2. Associative array — An array where each key has its own specific value.
3. Multidimensional array — An array containing one or more arrays within itself.

1. Indexed Arrays Example

```
<?php
        echo "<br><b> --CREATING & DISPLAYING NUMERIC ARRAY--
        </b><br><br>";
        $languages = array("PHP", "JAVA", "PYTHON", "C++");
        $languages[0]="C";   //will overwrite "PHP"at index 0
        $languages[4]="PHP";//will add PHP at index 4
        var_dump($languages);
?>
```

var_dump() function dumps information about one or more variables. The information holds type and value of the variable(s).

2. Associative Array Example

```
<?php
        echo "<br><br><b> --CREATING & DISPLAYING ASSOCIATIVE ARRAY-
        - </b><br>";
        $subjectcode=array(111=>"C",222=>"JAVA",333=>"PYTHON",444=>"C++",5
        55=>"PHP");
        echo "<br><br><b> ACCESSING ASSOCIATIVE ARRAY USING KEY
        </b><br>";
        echo "SUBJECT with code 333 is ::".$subjectcode[333];
        //echo $subjectcode[777]; //throw php error
```

```
echo "<br><br><b> DISPLAYING ASSOCIATIVE ARRAY USING KEY
THROUGH LOOPS</b><br>";
foreach ($subjectcode as $code => $subvalue){
   echo "SubjectCode $code ::$subvalue  <br>";}
 echo "<br><b> Displaying  raw value of array</b><br>";
print_r($subjectcode);
?>
```

print_r() function prints the information about a variable in a more human-readable way.

3. Multidimensional Array

```
<?php
 echo "<br><b> --CREATING & DISPLAYING MULTIDIMENSIONAL ARRAY --
</b><br>";
 $courses = array(
   array("code" => "001", "subject" => "PIC","sem" => "First"),
   array( "code" => "003","subject" => "PYTHON", "sem" => "Third"),
   array( "code" => "004", "subject" => "Advance JAVA/PYTHON", "sem" => "Fourth")
 );
 echo "<b> Display multidimensional array</b><br>";
$keys = array_keys($courses);
for($i = 0; $i < count($courses); $i++) {
   echo $keys[$i] . "<br>";
   foreach($courses[$keys[$i]] as $key => $value) {
      echo $key . " : " . $value . "<br>";
   } }

?>
```

array_keys() function returns an array containing the keys.

Functions in PHP

1. PHP Built-in Functions
   -A function is a self-contained block of code that performs a specific task.
   -PHP has a huge collection of internal or built-in functions that you can call directly within your PHP scripts to perform a specific task, like gettype(), print_r(), var_dump, etc.
2. PHP User-Defined Functions
   a. Syntax for creating Functions
      function functionName(){
         // Code to be executed
      }
   b. function myFunc($oneParameter, $anotherParameter){
         // Code to be executed
      }
   c. function myFunc($oneParameter, $anotherParameter=value){
         // Code to be executed
      }

```
//calling such functions
$myFunc("value1", "value2"); $myFunc("value1");// both will work
```

d.
```
function myFunc($oneParameter, $anotherParameter){
    return $returnvalue;
     }
```

Built in Functions:

Built in functions are functions that comes along with PHP installation package. The built in functions are what make PHP a very efficient and productive scripting language. The built in functions can be classified into many categories.

1) String Funtions
2) Date and Time Functions
3) Math Functions
4) Image Handling Functions
5) File Handling Funtions

File Handling Functions:

File handling in PHP is similar as file handling is done by using any programming language like C. PHP has many functions to work with normal files.

Those functions are:

1) fopen() – This function is used to open a file. First parameter of fopen() contains name of the file which is to be opened and second parameter tells about mode in which file needs to be opened, e.g.,

```
<?php
$file = fopen("demo.txt",'w');
?>
```

Files can be opened in any of the following modes :

"w" – Opens a file for write only. If file not exist then new file is created and if file already exists then contents of file is erased.
"r" – File is opened for read only.
"a" – File is opened for write only. File pointer points to end of file. Existing data in file is preserved.
"w+" – Opens file for read and write. If file not exist then new file is created and if file already exists then contents of file is erased.
"r+" – File is opened for read/write.
"a+" – File is opened for write/read. File pointer points to end of file. Existing data in file is preserved. If file is not there then new file is created.
"x" – New file is created for write only.

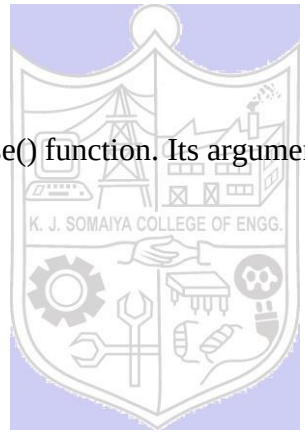
2) fread() — After file is opened using fopen() the contents of data are read using fread(). It takes

two arguments. One is file pointer and another is file size in bytes, e.g.,

```
<?php
$filename = "demo.txt";
$file = fopen( $filename, 'r' );
$size = filesize( $filename );
$filedata = fread( $file, $size );
?>
```

3) fwrite() – New file can be created or text can be appended to an existing file using fwrite() function. Arguments for fwrite() function are file pointer and text that is to written to file.

```
<?php
$file = fopen("demo.txt", 'w');
$text = "Hello world\n";
fwrite($file, $text);
?>
```



4) fclose() – file is closed using fclose() function. Its argument is file which needs to be closed, e.g.,

```
<?php
$file = fopen("demo.txt", 'r');
fclose($file);
?>
```

Image Handling Function:
imagecreate() returns an image identifier representing a blank image of specified size. In general, we recommend the use of imagecreatetruecolor() instead of imagecreate() so that image processing occurs on the highest quality image possible.

imagecreate (int $width, int $height)

```
<?php
header("Content-Type: image/png");
$im = @imagecreate(110, 20)
    or die("Cannot Initialize new GD image stream");
$background_color = imagecolorallocate($im, 0, 0, 0);
$text_color = imagecolorallocate($im, 233, 14, 91);
imagestring($im, 1, 5, 5,  "A Simple Text String", $text_color);
imagepng($im);
imagedestroy($im);
```

?>

The above example will output something similar to:

Image Upload using File Upload

The process of uploading a file follows these steps −

The user opens the page containing a HTML form featuring a text files, a browse button and a submit button.
The user clicks the browse button and selects a file to upload from the local PC.
The full path to the selected file appears in the text filed then the user clicks the submit button.
The selected file is sent to the temporary directory on the server.
The PHP script that was specified as the form handler in the form's action attribute checks that the file has arrived and then copies the file into an intended directory.
The PHP script confirms the success to the user.
An uploaded file could be a text file or image file or any document.

```
<?php
  if(isset($_FILES['image'])){
    $errors= array();
    $file_name = $_FILES['image']['name'];
    $file_size =$_FILES['image']['size'];
    $file_tmp =$_FILES['image']['tmp_name'];
    $file_type=$_FILES['image']['type'];
    $file_ext=strtolower(end(explode('.',$_FILES['image']['name'])));

    $extensions= array("jpeg","jpg","png");

    if(in_array($file_ext,$extensions)=== false){
      $errors[]="extension not allowed, please choose a JPEG or PNG file.";
    }

    if($file_size > 2097152){
      $errors[]='File size must be excately 2 MB';
    }

    if(empty($errors)==true){
      move_uploaded_file($file_tmp,"images/".$file_name);
      echo "Success";
    }else{
      print_r($errors);
    }
```

```
  }
?>
<html>
  <body>

    <form action="" method="POST" enctype="multipart/form-data">
      <input type="file" name="image" />
      <input type="submit"/>
    </form>

  </body>
</html>
```

PHP GD

GD is an open-source code library that is required to create and manipulate images in PHP. It is used for creating PNG, JPEG, and GIF images. It is commonly used to create charts, graphics, thumbnails, etc, and website development is the most common application of GD

Installing PHP GD in Windows

To install the PHP GD follow the following steps:

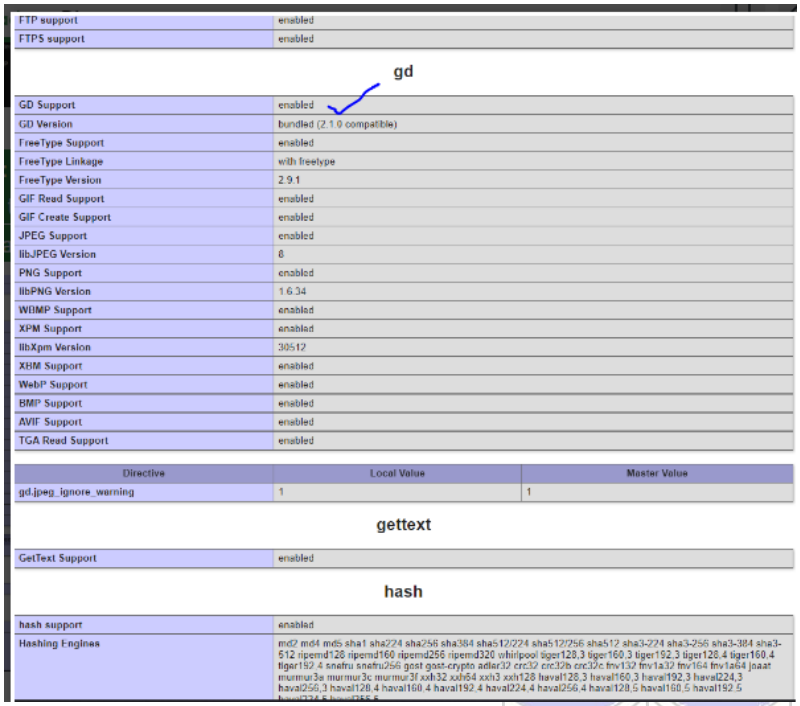
Step 1: Install XAMPP in your windows system.
Step 2: Verify if GD is already installed or not. So to verify GD we need to follow the following steps:

- Open XAMPP and click on the start button in front of Apache and MySQL to start php server, and go to the admin.
- A web page will open. Go to the 'PHPInfo' option on the top of the page.
- A PHPInfo dashboard will open up.
- Scroll down the page and search for 'gd'. If present, GD is already installed
  - If 'gd' is not present on the phpinfo page, you can follow the next to install GD.

Step 3: Locate and open php.ini in your editor.

Step 4: Find **;extension=gd.**

Step 5: Remove semicolon from **;extension=gd** and save the file.
Step 6: Go to php folder. It is usually present in **C:\xampp**.
Step 7: Look for **php_gd.dll** in the ext folder.
Step 8: Copy php_gd.dll and paste it into the following folder.
C:\Windows\System32
Step 9: Restart the XAMPP server. Now use the phpinfo() method as mentioned in step 2 to check whether the GD library is installed in the PHP server. It will show information about the PHP's configuration.



FTP support	enabled
FTPS support	enabled

gd

GD Support	enabled
GD Version	bundled (2.1.0 compatible)
FreeType Support	enabled
FreeType Linkage	with freetype
FreeType Version	2.9.1
GIF Read Support	enabled
GIF Create Support	enabled
JPEG Support	enabled
libJPEG Version	8
PNG Support	enabled
libPNG Version	1.6.34
WBMP Support	enabled
XPM Support	enabled
libXpm Version	30512
XBM Support	enabled
WebP Support	enabled
BMP Support	enabled
AVIF Support	enabled
TGA Read Support	enabled

Directive	Local Value	Master Value
gd.jpeg_ignore_warning	1	1

gettext

GetText Support	enabled

hash

hash support	enabled
Hashing Engines	md2 md4 md5 sha1 sha224 sha256 sha384 sha512/224 sha512/256 sha512 sha3-224 sha3-256 sha3-384 sha3-512 ripemd128 ripemd160 ripemd256 ripemd320 whirlpool tiger128,3 tiger160,3 tiger192,3 tiger128,4 tiger160,4 tiger192,4 snefru snefru256 gost gost-crypto adler32 crc32 crc32b crc32c fnv132 fnv1a32 fnv164 fnv1a64 joaat murmur3a murmur3c murmur3f xxh32 xxh64 xxh3 xxh128 haval128,3 haval160,3 haval192,3 haval224,3 haval256,3 haval128,4 haval160,4 haval192,4 haval224,4 haval256,4 haval128,5 haval160,5 haval192,5

Activity: Write PHP program for implementing different types of arrays and their associated functions. Also write program for implementing image and file handling functions.

Output(Code with result Snapshot)

Arrays and its associated functions

Code:

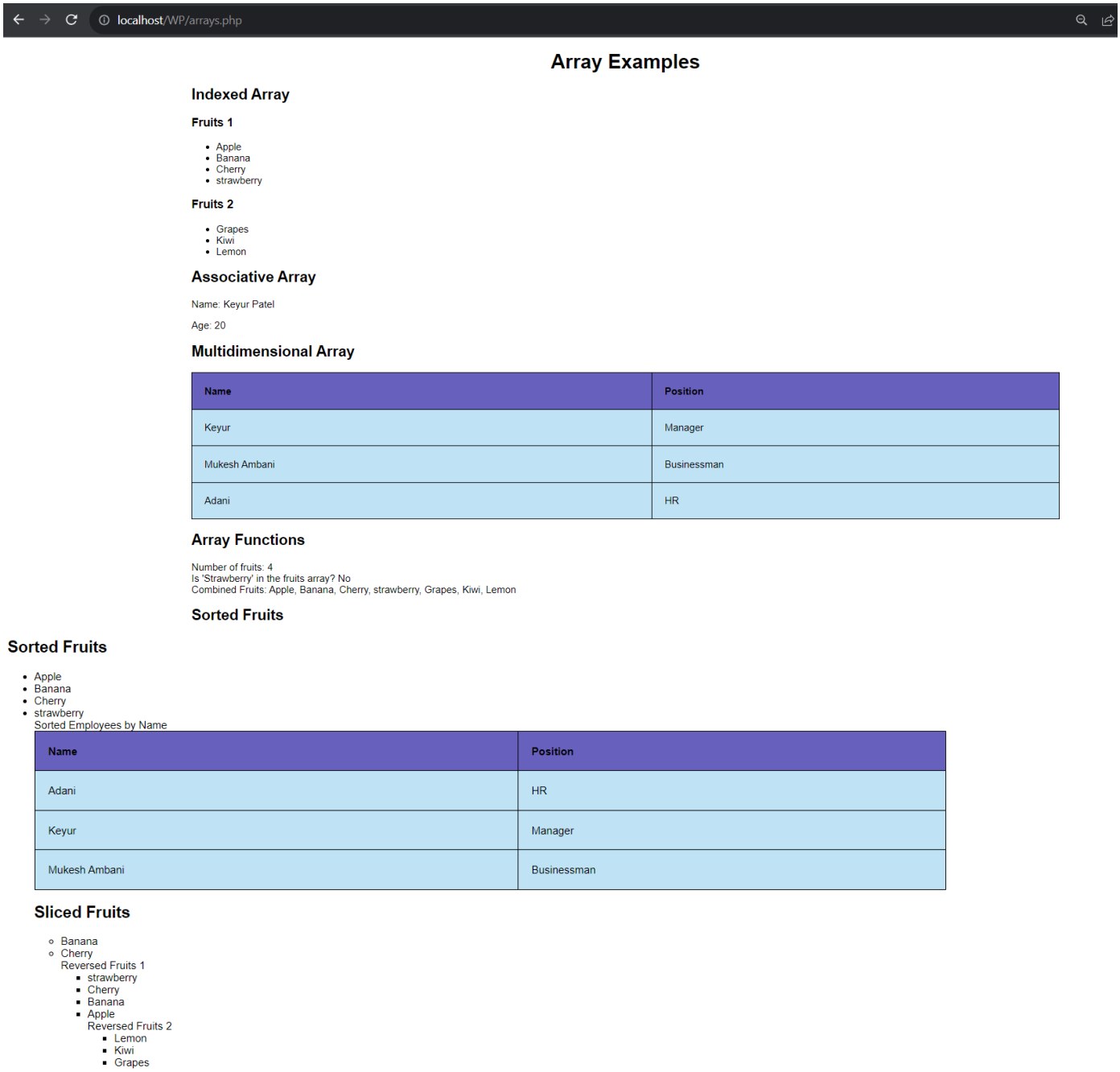
```
<!DOCTYPE html>
<html>

<head>
    <style>
        body {
            font-family: Arial, sans-serif;
        }

        .container {
            width: 70%;
            margin: 0 auto;
        }

        h1 {
            text-align: center;
        }

        table {
            width: 100%;
```

```
            border-collapse: collapse;
        }

        th, td {
            padding: 20px;
            text-align: left;
            border-bottom: 1px solid #ddd;
            border: 2px solid black;
            background-color:#bfe3f5 ;

        }

        th {
            background-color: #6860bd;
        }
    </style>
</head>

<body>
    <div class="container">
        <h1>Array Examples</h1>

        <?php
        // Indexed Array
        $fruits = array("Apple", "Banana", "Cherry", "strawberry");
        $moreFruits = array("Grapes", "Kiwi", "Lemon");

        // Associative Array
        $person = array("first_name" => "Keyur", "last_name" => "Patel", "age" => 20);

        // Multidimensional Array
        $employees = array(
            array("name" => "Keyur", "position" => "Manager"),
            array("name" => "Mukesh Ambani", "position" => "Businessman"),
            array("name" => "Adani", "position" => "HR")
        );

        echo "<h2>Indexed Array</h2>";
        echo "<h3>Fruits 1</h3>";
        echo "<ul>";
        foreach ($fruits as $fruit) {
            echo "<li>$fruit</li>";
        }
        echo "</ul>";

        echo "<h3>Fruits 2</h3>";
        echo "<ul>";
```

```
        foreach ($moreFruits as $fruit) {
            echo "<li>$fruit</li>";
        }
        echo "</ul>";

        echo "<h2>Associative Array</h2>";
        echo "<p>Name: " . $person["first_name"] . " " . $person["last_name"] . "</p>";
        echo "<p>Age: " . $person["age"] . "</p>";

        echo "<h2>Multidimensional Array</h2>";
        echo "<table>";
        echo "<tr><th>Name</th><th>Position</th></tr>";
        foreach ($employees as $employee) {
            echo "<tr>";
            echo "<td>" . $employee["name"] . "</td>";
            echo "<td>" . $employee["position"] . "</td>";
            echo "</tr>";
        }
        echo "</table>";

        // Using Array Functions
        echo "<h2>Array Functions</h2>";

        // Count the number of elements in the indexed array
        echo "Number of fruits: " . count($fruits) . "<br>";

        // Check if an element exists in the indexed array
        $searchFruit = "Strawberry";
        echo "Is '$searchFruit' in the fruits array? " . (in_array($searchFruit, $fruits)
? "Yes" : "No") . "<br>";

        // Combine two arrays (indexed)
        $combinedFruits = array_merge($fruits, $moreFruits);
        echo "Combined Fruits: " . implode(", ", $combinedFruits) . "<br>";

        // Sorting Indexed Array
        sort($fruits);
        echo "<h2>Sorted Fruits</h2>";
        echo "<ul>";
        foreach ($fruits as $fruit) {
            echo "<li>$fruit</li>";
        }
        echo "</ul";

        // Sorting Multidimensional Array
        function sortEmployeesByName($a, $b) {
            return $a["name"] < $b["name"] ? -1 : 1;
```

```
        }

        usort($employees, "sortEmployeesByName");
        echo "<h2>Sorted Employees by Name</h2>";
        echo "<table>";
        echo "<tr><th>Name</th><th>Position</th></tr>";
        foreach ($employees as $employee) {
            echo "<tr>";
            echo "<td>" . $employee["name"] . "</td>";
            echo "<td>" . $employee["position"] . "</td>";
            echo "</tr>";
        }
        echo "</table>";

        $slicedFruits = array_slice($fruits, 1, 2); // Get elements from index 1 to 2
(exclusive)
        echo "<h2>Sliced Fruits</h2>";
        echo "<ul>";
        foreach ($slicedFruits as $fruit) {
            echo "<li>$fruit</li>";
        }
        echo "</ul";

        // Array Reverse
        $reversedFruits = array_reverse($fruits);
        echo "<h2>Reversed Fruits 1</h2>";
        echo "<ul>";
        foreach ($reversedFruits as $fruit) {
            echo "<li>$fruit</li>";
        }
        echo "</ul";

        $reversedmoreFruits = array_reverse($moreFruits);
        echo "<h2>Reversed Fruits 2</h2>";
        echo "<ul>";
        foreach ($reversedmoreFruits as $moreFruit) {
            echo "<li>$moreFruit</li>";
        }
        echo "</ul";
        ?>
    </div>
</body>

</html>
```

Output:



File Handling

file_choose.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>File Upload</title>
    <style>
        body {
            font-family: Arial, sans-serif;
```

```
            background-color: #f4f4f4;
            text-align: center;
        }

        .container {
            background-color: #fff;
            padding: 80px;
            border-radius: 5px;
            box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.2);
            width: 300px;
            margin: 0 auto;
        }

        .upload-form {
            display: flex;
            flex-direction: column;
        }

        .upload-input {
            margin-bottom: 30px;
        }

        .upload-button {
            background-color: #4CAF50;
            color: white;
            border: none;
            padding: 10px;
            border-radius: 5px;
            cursor: pointer;
        }

        .upload-button:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>File Upload</h2>
        <form class="upload-form" method="POST" action="upload.php"
enctype="multipart/form-data">
            <input class="upload-input" type="file" name="fileToUpload"
id="fileToUpload">
            <input class="upload-button" type="submit" value="Upload File">
        </form>
    </div>
</body>
```

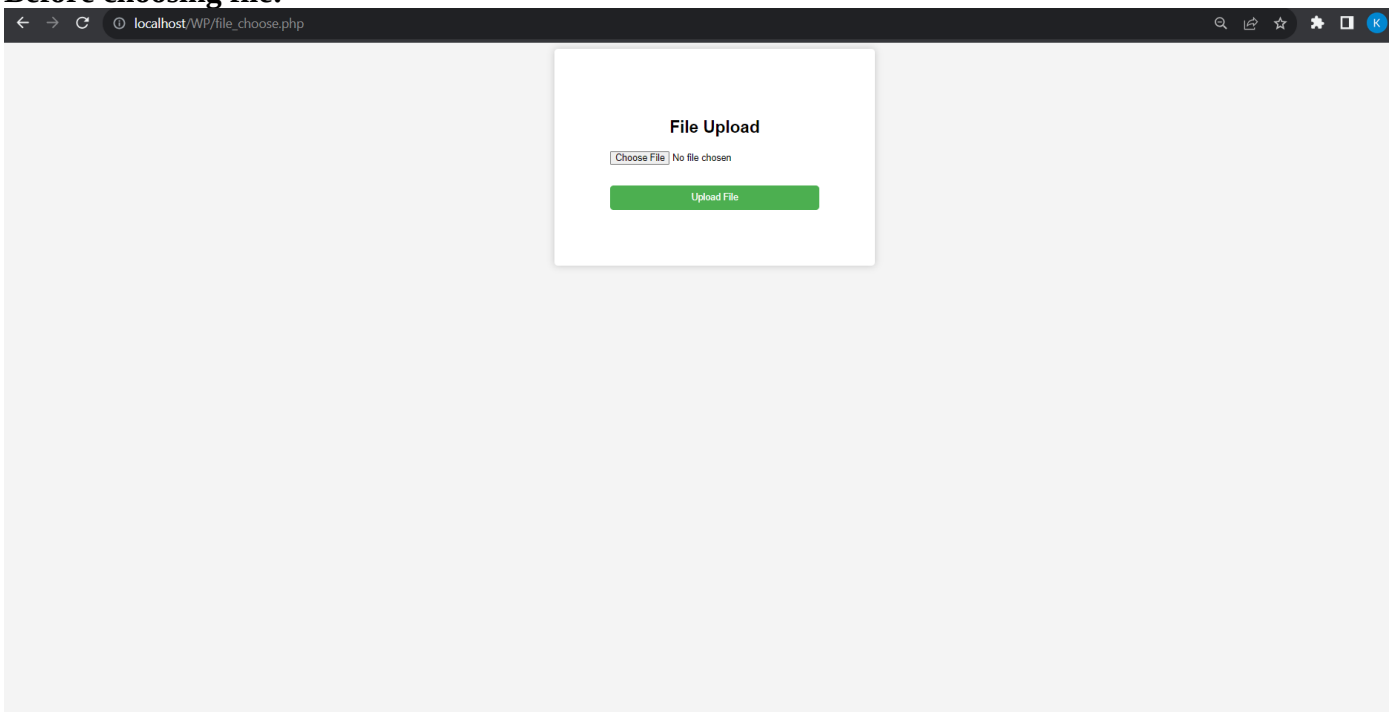
```
</html>
```

upload.php

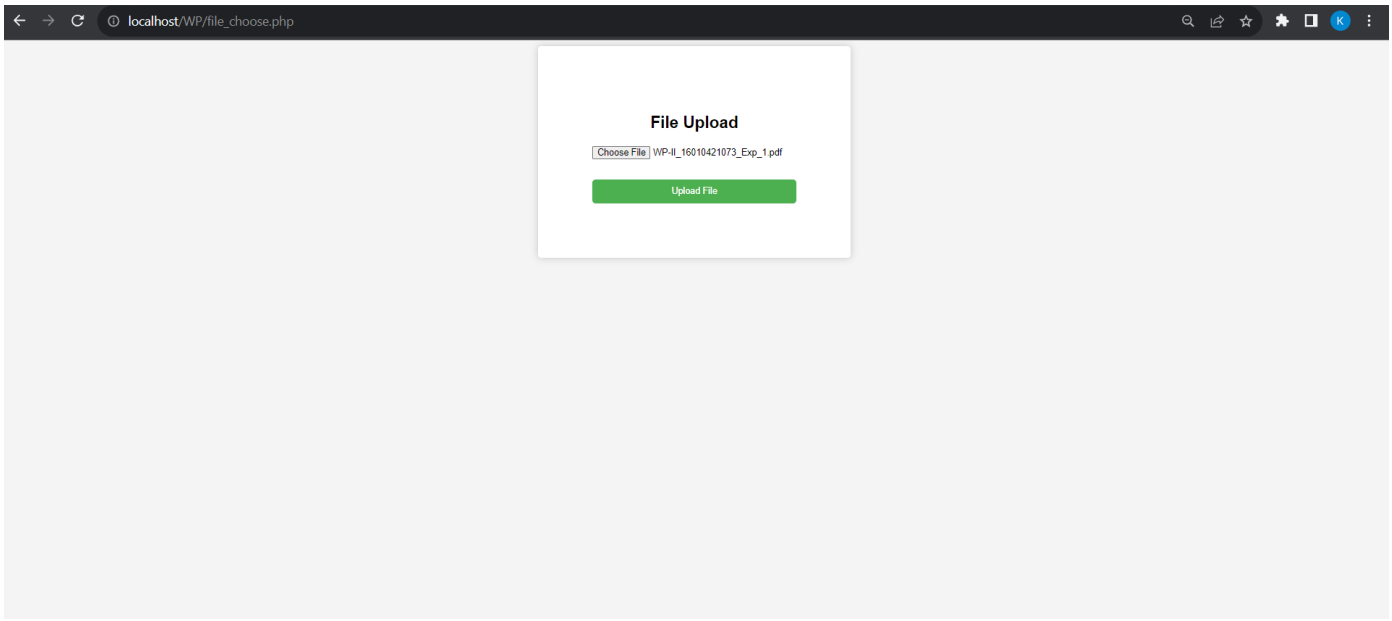
```
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_FILES["fileToUpload"]))
{
    $targetDirectory = "uploads/";
    $targetFile = $targetDirectory . basename($_FILES["fileToUpload"]["name"]);

    if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], $targetFile))
    {
        echo "The file " . basename($_FILES["fileToUpload"]["name"]) . " has been
uploaded.";
    }
    else
    {
        echo "Sorry, there was an error uploading your file.";
    }
}
?>
```

Output:

Before choosing file:



After choosing file:



File uploaded:

The file WP-II_16010421073_Exp_1.pdf has been uploaded.

Image Handling:

image_choose.php

```
<!DOCTYPE html>
<html>
<head>
    <title>Image Upload and Display</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            margin: 0;
            padding: 0;
            display: flex;
```

```
            justify-content: center;
            align-items: center;
            height: 100vh;
        }

        .container {
            background-color: #fff;
            border-radius: 5px;
            box-shadow: 0 2px 10px rgba(0, 0, 0, 0.2);
            padding: 20px;
            width: 400px;
            text-align: center;
        }

        h1 {
            margin: 0 0 20px;
        }

        form {
            margin: 0;
        }

        input[type="file"] {
            margin: 0 0 10px;
        }

        input[type="submit"] {
            display: block;
            margin: 10px auto 0;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Upload and Display Image</h1>
        <form action="fileupload.php" method="post" enctype="multipart/form-data">
            <input type="file" name="image" id="fileToUpload">
            <input type="submit" value="Upload Image" name="submit">
        </form>
    </div>
</body>
</html>
```

fileupload.php

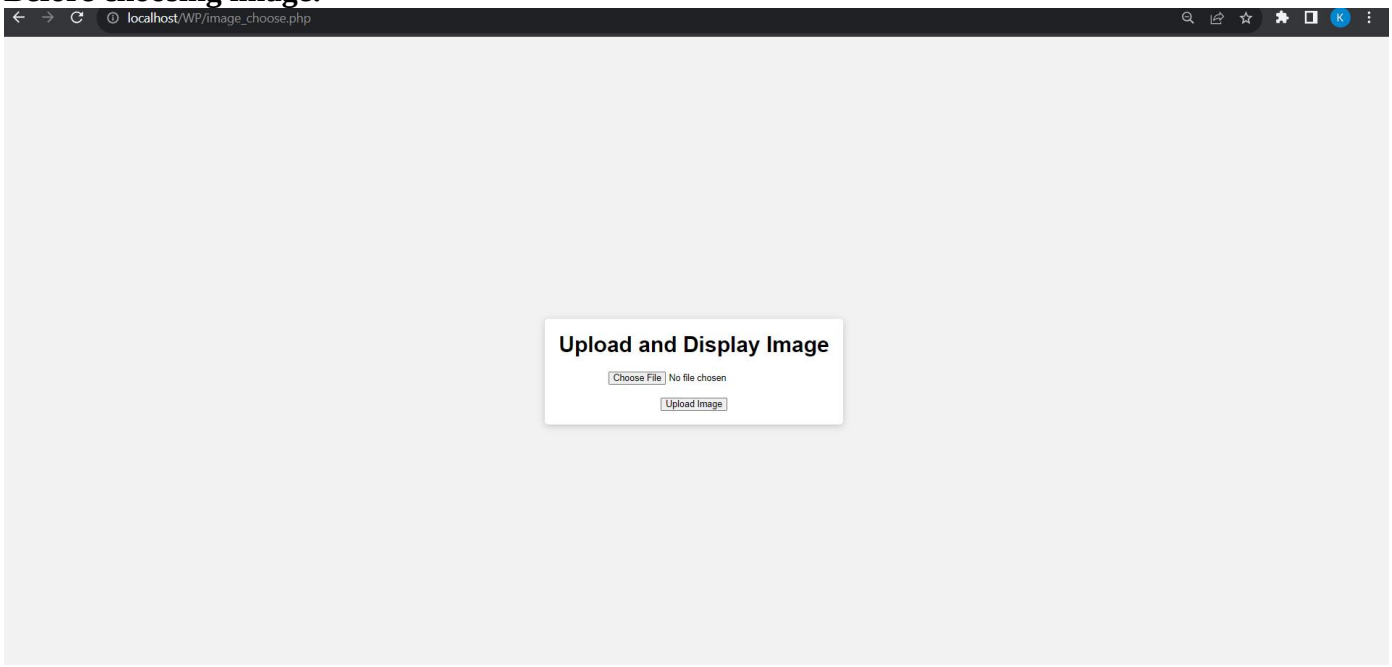
```
<!DOCTYPE html>
<html>
```

```
<head>
    <title>Uploaded Image</title>
</head>
<body>
<h1>Uploaded Image</h1>

<?php
if (isset($_POST['submit'])) {
    $targetDir = "uploads/"; // Create an "uploads" directory to store uploaded images
    $targetFile = $targetDir . basename($_FILES["image"]["name"]);
    $uploadOk = 1;
    $imageFileType = strtolower(pathinfo($targetFile, PATHINFO_EXTENSION));

    // Check if the file is an actual image
    $check = getimagesize($_FILES["image"]["tmp_name"]);
    if ($check !== false) {
        echo "File is an image - " . $check["mime"] . ".<br>";
        $uploadOk = 1;
    } else {
        echo "File is not an image.<br>";
        $uploadOk = 0;
    }

    // Check file size (you can change the size limit)
    if ($_FILES["image"]["size"] > 500000) {
        echo "Sorry, your file is too large.<br>";
        $uploadOk = 0;
    }

    // Allow certain image file formats (you can add more)
    if ($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg" &&
$imageFileType != "gif") {
        echo "Sorry, only JPG, JPEG, PNG, and GIF files are allowed.<br>";
        $uploadOk = 0;
    }

    if ($uploadOk == 0) {
        echo "Sorry, your file was not uploaded.<br>";
    } else {
        if (move_uploaded_file($_FILES["image"]["tmp_name"], $targetFile)) {
            echo "The image " . htmlspecialchars(basename($_FILES["image"]["name"])) . "
has been uploaded and displayed below:<br>";
            echo "<img src='" . $targetFile . "' width='300'>";
        } else {
            echo "Sorry, there was an error uploading your file.<br>";
        }
    }
```

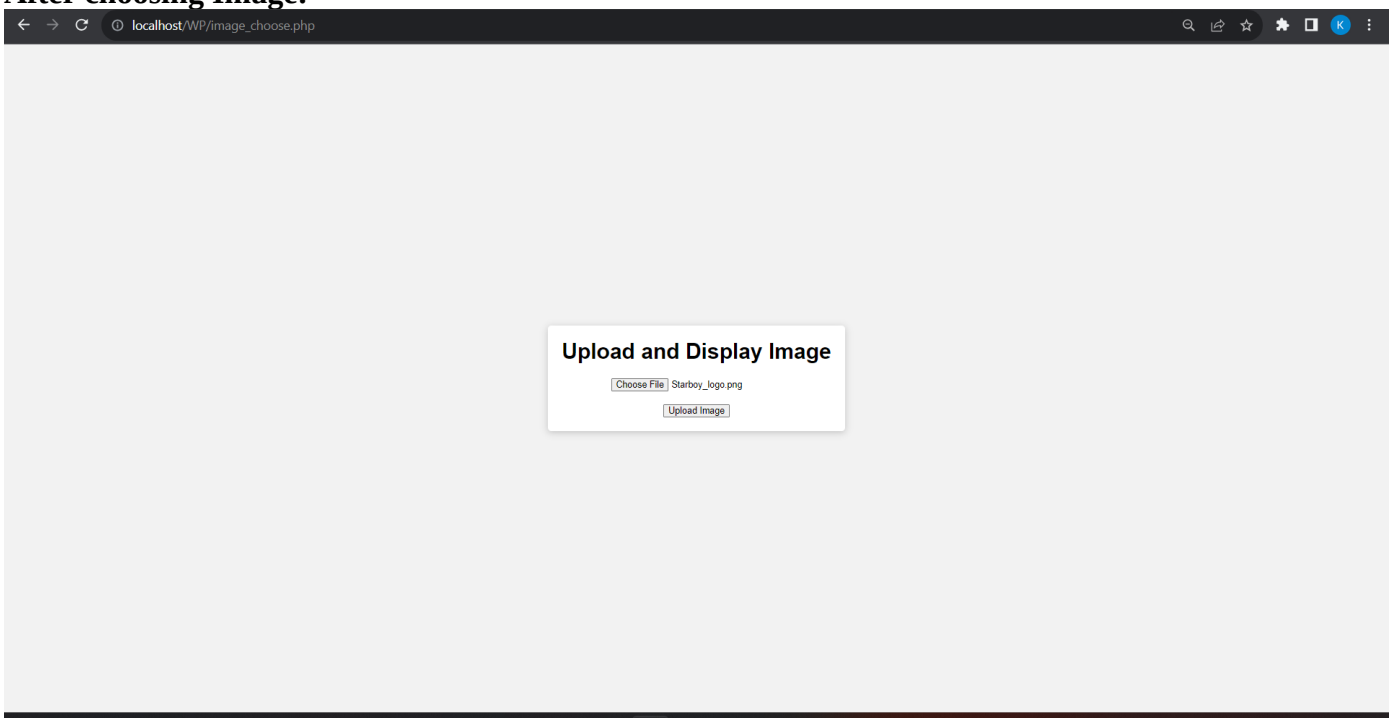
```
}
?>

</body>
</html>
```

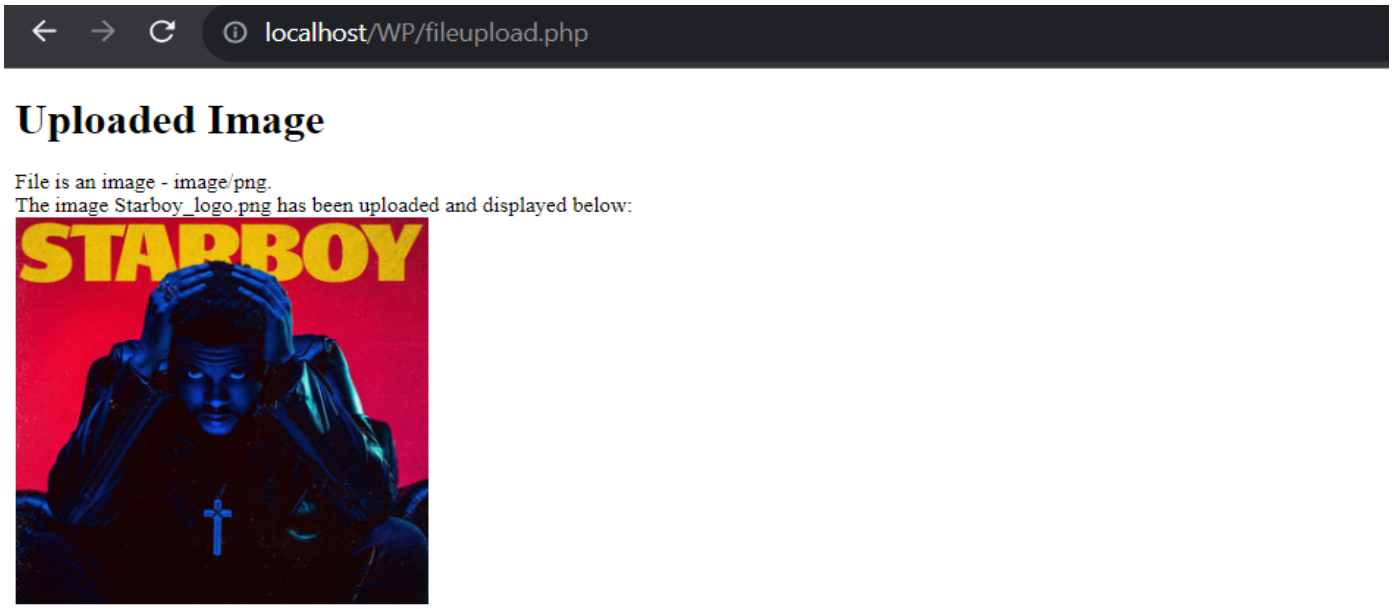
Output:

Before choosing image:



After choosing Image:





Questions:-

1. What is difference between array_merge and array_combine? Answer:

Aspect	`array_merge`	`array_combine`
Purpose	Merges two or more arrays into one.	Creates an array by using one array for keys and another for values.
Parameters	Accepts multiple arrays as input.	Requires two arrays as input, one for keys and one for values.
Result	Combines all values into a single array.	Creates a new array with keys and values paired together.
Key Handling	If keys are not specified, they are reindexed numerically.	Uses one array for keys and another for values; keys must be unique.
Example	`` `php $merged = array_merge($array1, $array2);` ``	`` `php $combined = array_combine($keys, $values);` ``
Use Case	Combining multiple arrays into one.	Creating associative arrays from separate arrays.

2. Write a program based on functions passing reference as parameter.

Answer: In this program:

1. We define a function incrementByReference that takes two parameters: a reference to a number and a value by which the number should be incremented.
2. We declare the $originalNumber variable and set it to 10, and we specify the $incrementValue as 5.
3. We call the incrementByReference function, passing $originalNumber by reference. This means any changes made to $number inside the function will affect the original variable.
4. Inside the function, we increment the value of $number by the specified increment value.
5. After the function call, we print the original number to show that it has been modified by the function.

When we run this program, we'll see that the original number is incremented by the specified value, demonstrating the use of passing variables by reference to modify them inside a function.

```php
<?php
function incrementByReference(&$number, $value) {
    $number += $value;
}

$originalNumber = 10;
$incrementValue = 5;

echo "Original Number: $originalNumber<br>";
```

```php
incrementByReference($originalNumber, $incrementValue);

echo "Number after increment: $originalNumber";
?>
```

3. How can you display a file download dialog box using PHP?

Answer: To display a file download dialog box using PHP, we can use the header function to set the necessary HTTP headers. Here's a simple example of how to do this:

```php
<?php
// Define the file path
$file = 'path_to_your_file.pdf'; // Replace with the path to your file

// Check if the file exists
if (file_exists($file)) {
    // Set the appropriate headers for file download
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename=' . basename($file));
    header('Expires: 0');
    header('Cache-Control: must-revalidate');
    header('Pragma: public');
    header('Content-Length: ' . filesize($file));

    // Output the file
    readfile($file);
    exit;
} else {
    // File not found
    echo 'File not found.';
}
?>
```

In this PHP code:

1. Replace 'path_to_your_file.pdf' with the actual path to the file you want to make available for download.
2. The code first checks if the file exists using file_exists.
3. If the file exists, it sends the necessary HTTP headers to prompt the user's browser to display the file download dialog box.
4. Headers like Content-Disposition with the attachment value specify that the file should be treated as an attachment to download. The basename($file) provides the filename that will be displayed in the download dialog.
5. It sets other cache control headers to ensure the file is not cached.
6. The readfile function is used to output the file to the browser.

7. The exit function is called to prevent any further execution of the script.If the file does not exist, it will display "File not found." When a user accesses the PHP script, the browser will display a download dialog box for thespecified file, allowing the user to download it to their local machine.

4. What is the purpose of php.ini file?

Answer: The php.ini file in PHP serves as the configuration file that controls various settingsand options for the PHP interpreter. It plays a crucial role in defining how PHP behaves on your web server. Here are the primary purposes of the php.ini file:

1. **Configuration Settings:** The php.ini file contains a wide range of configuration settings forPHP. These settings determine how PHP interacts with the server, handles error reporting, processes data, manages resources, and more. It allows you to customize PHP to meet the specific requirements of your applications.
2. **Error Handling:** You can specify how PHP should handle errors and log error messages.This includes options for displaying errors on the screen, writing errors to log files, and configuring error reporting levels.
3. **Security:** PHP configuration settings in php.ini can be used to enhance the security of yourPHP applications. You can enable or disable features that may pose security risks, set permissions, and control access to certain functions.
4. **Resource Management:** You can configure resource limits, such as memory limits, execution time limits, and file upload limits. This helps in preventing excessive resourceconsumption or long-running scripts that could impact server performance.
5. **Extension Configuration:** PHP extensions (e.g., MySQL, GD, cURL) are often controlledthrough php.ini. You can enable or disable specific extensions and set their configuration options.
6. **Output Buffering:** PHP offers output buffering, and you can configure its behavior in php.ini. This can be useful for improving performance and managing the content sent to thebrowser.

Outcomes: CO1: Illustrate use of basic PHP concepts to develop applications.

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)

Understood how to create Indexed, Associative, Multi dimensional arrays in PHP and use thedifferent array functions linked with them.

Learned how to use built-in PHP functions and how to define our own functions with or withoutparameters and returning value if needed.

Also learned to implement the file and image handling functions in php.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge

---

with dateReferences:

1. Instruction Link to install php GD –
https://www.geeksforgeeks.org/how-to-install-php-gd-in-windows/

Books:

1. Thomson PHP and MySQL Web Development Addison-Wesley Professional , 5th Edition2016.
2. Peter MacIntyre, Kevin Tatroe Programming PHP O'Reilly Media, Inc, 4th Edition 2020
3. Frank M. Kromann Beginning PHP and MySQL: From Novice to Professional, Apress 1stEdition, 2018