CHAPTER 2

UNIT II

Arrays, Functions and Graphics

Syllabus

2.1 Creating and Manipulating Array, Types of Arrays- Indexed, Associative and Multi-dimensional arrays

2.2 Extracting data from arrays, implode, explode, and array flip.

2.3 Traversing Arrays

2.4 Function and its types - User defined function, Variable function and Anonymous function.

2.5 Operations on String and String functions: str_word_count() strlen(),strrev() strops() str_replace(), ucwords(),strtoupper(), strtolower(),strcmp().

2.6 Basic Graphics Concepts, Creating Images, Images with text, Scaling Images, Creation of PDF document.

2.1 Creating and Manipulating Array

Q. Describe the concept of Array in PHP. **(4 Marks)**

Q. Define Array. How can we declare one dimensional and two dimensional array in PHP ? **(4 Marks)**

(C/O – Answer not covered in this section plz check) refer 2.1.3 multidimensional array

Q. Explain different methods of creating arrays in PHP. **(4 Marks)**

– Arrays in PHP is a type of data structure that allows to store multiple elements of similar data type under a single variable thereby saving us the effort of creating a different variable for every data.

– An array in PHP is actually an *ordered map.* A map is a type that associates *values* to *keys*.

– The arrays are helpful to create a list of elements of similar types, which can be accessed using their index or key.

– Suppose we want to store five names and print them accordingly. This can be easily done by the use of five different string variables. But if instead of five, the number rises to hundred, then it would be really difficult for the user or developer to create so much different variables. Here array comes into play and helps us to store every element within a single variable and also allows easy access using an index or a key.

– An array is created using an **array()** function in PHP.

Types of Array :

There are basically three types of arrays in PHP :

1. Indexed or Numeric Arrays

2. Associative Arrays
3. Multidimensional Arrays

2.1.1 Indexed or Numeric Arrays

Q. Write a program to create Index based Arrays in PHP. **(2 Marks)**

– An array with a numeric index where values are stored linearly.

– Numeric arrays use number as access keys.

– An access key is a reference to a memory slot in an array variable.

– The access key is used whenever we want to read or assign a new value an array element.

Syntax for creating indexed/numeric array in php

```
<?php
$variable_name[n] = value;
?>
```

OR

```
<?php
$variable_name = array(n => value, …);
?>
```

Where,

"$variable_name…" is the name of the variable

"[n]" is the access index number of the element

"value" is the value assigned to the array element.

Example 1:

```
<?php
$course[0]="Computer Engg.";
$course[1]="Information Tech.";
$course[2]="Electronics and Telecomm.";
// Accessing the elements directly
echo $course[2], "<BR>";
echo $course[0], "<BR>";
echo $course[1], "<BR>";
?>
```

Output :

```
Electronics and Telecomm.
```

```
Computer Engg.
Information Tech.
```

Example 2 :

```
<?php
$course = array(0 => "Computer Engg.",1 => "Information Tech.", 2 => "Electronics and Telecomm.");
echo $course [1];
?>
```

Output :

```
Information Tech.
```

2.1.2 Associative Arrays

Q.	Write a program to create Associative Arrays in PHP.	**(2 Marks)**
Q.	How can we declare associative arrays?	**(2 Marks)**

– This type of arrays is similar to the indexed arrays but instead of linear storage, every value can be assigned with a user-defined key of string type.

– An array with a string index where instead of linear storage, each value can be assigned a specific key.

– Associative array differ from numeric array in the sense that associative arrays use descriptive names for id keys.

Syntax for creating associative array in php

```
<?php
$variable_name['key_name'] = value;

$variable_name = array('keyname' => value);
?>
```

Where,

"$variable_name..." is the name of the variable

"['key_name']" is the access index number of the element

"value" is the value assigned to the array element.

Example 1 :

```
<?php
$capital = array("Mumbai" => "Maharashtra","Goa" => "Panaji", "Bihar" => "Patna");
print_r($capital);
echo"<br>";
echo "Mumbai is a capital of  " .$capital ["Mumbai"];
?>
```

Output :

```
Array ( [Mumbai] => Maharashtra [Goa] => Panaji [Bihar] => Patna )
    Mumbai is a capital of Maharashtra
```

Note: print_r( ) is used for displaying the contents of an array.

By default, array starts with index 0. What if you wanted to start with index 1 instead? You could use the PHP =>operator like this:

```
$course = array ( 1 => "Computer Engg.",
                  2 => "Information Tech.",
                  3 => "Electronics and Telecomm.");
```

The => operator create key/value pairs in arrays-the item on the left of the operator => is the key and the item on the right is the value.

Example 2 :

```
<html>
<head>
<title>
array
</title>
</head>
<body>
<h1> Creating an array  </h1>
<?php
$course = array(1 => "Computer Engg.",
          5 => "Information Tech.",
          3 => "Electronics and Telecomm.");
echo $course [5];
?>
```

```
</body>
</html>
```

Output :

```
Creating an array
Information Tech.
```

Example 3 :

```
<html>
<head>
<title>
array
</title>
</head>
<body>
<h1> Creating an array  </h1>

<?php
$course = array("CO"=>549,
                 "IF"=>450,
                 "EJ"=>100);
echo "course[CO]=", $course["CO"],"<br>";
echo "course[IF]=", $course["IF"],"<br>";
echo "course[EJ]=", $course["EJ"],"<br>";
?>
</body>
</html>
```

Output :

```
Creating an array
course['CO']=549
course['IF']=450
course['EJ']=100
```

2.1.3 Multidimensional Arrays

Q.	Write a note on multidimensional array with example.	**(4 Marks)**

– These are arrays that contain other nested arrays.

– An array which contains single or multiple arrays within it and can be accessed via multiple indices.

– We can create one dimensional and two dimensional array using multidimensional arrays.

– The advantage of multidimensional arrays is that they allow us to group related data together.

Syntax for creating multidimensional array in php

```
array (
    array (elements...),
    array (elements...),
    ...
)
```

Example :

```
<?php
// Defining a multidimensional array
$person = array(
    array(
        "name" => "Yogita K",
        "mob" => "5689741523",
        "email" => "yogi_k@gmail.com",
    ),
    array(
        "name" => "Manisha P.",
        "mob" => "2584369721",
        "email" => "manisha_p@gmail.com",
    ),
    array(
        "name" => "Vijay Patil",
        "mob" => "9875147536",
        "email" => "Vijay_p@gmail.com",
    )
);

// Accessing elements
echo "manisha P's email-id is: " . $person[1]["email"], "<br>";
echo "Vijay Patil's mobile no: " . $person[2]["mob"];
```

```
?>
```

Output :

```
manisha P's email-id is: manisha_p@gmail.com
Vijay Patil's mobile no: 9875147536
```

Accessing multidimensional array elements: There are mainly two ways to access multidimensional array elements in PHP.

- Elements can be accessed using dimensions as array name['first dimension']['second dimension'].
- Elements can be accessed using for loop.
- Elements can be accessed using for each loop.

Example :

```
<?php
$mobile = array
 (
 array("LG",20,18),
 array("sony",30,13),
 array("Redme",10,2),
 array("Samsung",40,15)
 );
echo $mobile[0][0].": In stock: ".$mobile[0][1].", sold: ".$mobile[0][2].".<br>";
echo $mobile[1][0].": In stock: ".$mobile[1][1].", sold: ".$mobile[1][2].".<br>";
echo $mobile[2][0].": In stock: ".$mobile[2][1].", sold: ".$mobile[2][2].".<br>";
echo $mobile[3][0].": In stock: ".$mobile[3][1].", sold: ".$mobile[3][2].".<br>";
?>
```

Output :

```
LG: In stock: 20, sold: 18.
sony: In stock: 30, sold: 13.
Redme: In stock: 10, sold: 2.
Samsung: In stock: 40, sold: 15.
```

- Another way to extracting elements from multidimensional array.
- We can also use a for loop inside another for loop to get the elements of the $mobile array.

```
<?php
$mobile = array
 (
 array("LG",20,18),
 array("sony",30,13),
 array("Redme",10,2),
 array("Samsung",40,15)
 );

  for ($row = 0; $row < 4; $row++)
{
 echo "<p><b>Row number $row</b></p>";
 echo "<ul>";

 for ($col = 0; $col < 3; $col++)
 {
   echo "<li>".$mobile[$row][$col]."</li>";
 }

 echo "</ul>";
}
?>
```

Output :

```
Row number 0
      -      LG
      -      20
      -      18
Row number 1
      -      sony
      -      30
      -      13
Row number 2
      -      Redme
      -      10
      -      2
Row number 3
      -      Samsung
      -      40
```

```
    -     15
```

2.2 Extracting Data from Arrays

Q. How to extract data from arrays ? **(4 Marks)**

You can use the **extract()** function to extract data from arrays and store it in variables.

Example :

```
<html>
<head>
<title>
array
</title>
</head>
<body>
<h3> Extracting variables from array </h3>
<?php
$course["CO"]="Computer Engg.";
$course["IF"]="Information Tech.";
$course["EJ"]="Electronics and Telecomm.";
extract($course);
echo "CO=$CO<BR>";
echo "IF=$IF<BR>";
echo "EJ=$EJ<BR>";
?>
</body>
</html>
```

Output :

```
Extracting variables from array
CO=Computer Engg.
IF=Information Tech.
EJ=Electronics and Telecomm.
```

In fact, **List()**function is also used to extract variables from an array.

```
<html>
```

```
<head>
<title>
array
</title>
</head>
<body>
<h3> Extracting variables from array </h3>
<?php
$course[0]="Computer Engg.";
$course[1]="Information Tech.";
$course[2]="Electronics and Telecomm.";

list($one, $two)=$course;
echo $one,"<BR>";
echo $two;
?>
</body>
</html>
```

Output :

```
Extracting variables from array
Computer Engg.
Information Tech.
```

2.2.1 Compact() Function

This function is opposite of extract() function. It returns an array with all the variables added to it. Each parameter can be either a string containing the name of the variable, or an array of variable names. The compact() functions create associative array whose key value are the variable name and whose values are the variable values.

Example :

```
<?php
  $var1="PHP";
  $var2="JAVA";
  $var3=compact("var1", "var2");
  print_r($var3);
```

```
?>
```

Output :

```
Array ( [var1] => PHP [var2] => JAVA )
```

2.2.2 Implode and Explode

Q. Describe implode and explode **(4 Marks)**

– Imploding and Exploding are couple of important functions of PHP that can be applied on strings or arrays.

– The **implode( )** function implodes an array into a string, and the explode function explodes a string into an array. That's useful if you have stored your data in strings in files and want to convert those strings into array elements when your web application forms.

2.2.2(A) Implode()

– The **implode()** is a built-in function in PHP and is used to join the elements of an array.

– The implode() function returns a string from the elements of an array.

– If we have an array of elements, we can use the implode() function to join them all to form one string. We basically join array elements with a string. Just like join() function , implode() function also returns a string formed from the elements of an array.

Syntax :

```
string implode(separator, array);
```

– The **implode()** function accepts two parameter out of which one is optional and one is mandatory.

– **Separator :** This is an optional parameter and is of string type. The values of the array will be join to form a string and will be separated by the separator parameter provided here. This is optional, if not provided the default is "" (i.e. an empty string).

– **Array :** The array whose value is to be joined to form a string.

Example 1 : In which an array is "imploded" into a text string and displayed.

```
<html>
<head>
<title>
array
</title>
</head>
<body>
<h3> Extracting variables from array  </h3>
<?php
```

```
$course[0]="Computer Engg.";
$course[1]="Information Tech.";
$course[2]="Electronics and Telecomm.";
$text=implode("," , $course);
echo $text;
?>
</body>
</html>
```

Output :

```
Extracting variables from array
Computer Engg., Information Tech., Electronics and Telecomm.
```

Example 2 :

```
<?php
   // PHP Code to implement join function
       $InputArray = array('CO','IF','EJ');

   // Join without separator
   print_r(implode($InputArray));
   print_r("<BR>");

   // Join with separator
   print_r(implode("-",$InputArray));
?>
```

Output :

```
COIFEJ
CO-IF-EJ
```

2.2.2(B) Explode()

– The explode() function breaks a string into an array.

– The **explode()** is a built in function in PHP used to split a string in different strings.

– The explode() function splits a string based on a string delimeter, i.e. it splits the string wherever the delimeter character occurs. This function returns an array containing the strings formed by splitting the original string.

Syntax :

```
array explode(separator, OriginalString, NoOfElements);
```

The **explode( )**function accepts three parameters of which two are compulsory and one is optional. All the three parameters are described as follows :

1. **separator :** This character specifies the critical points or points at which the string will split, i.e. whenever this character is found in the string it symbolizes end of one element of the array and start of another.
2. **OriginalString :** The input string which is to be split in array.
3. **NoOfElements :** This is optional. It is used to specify the number of elements of the array. This parameter can be any integer (positive , negative or zero)
   - **(i)** **Positive (N) :** When this parameter is passed with a positive value it means that the array will contain this number of elements. If the number of elements after separating with respect to the separator emerges to be greater than this value the first N-1 elements remain the same and the last element is the whole remaining string.
   - **(ii)** **Negative (N) :** If negative value is passed as parameter then the last N element of the array will be trimmed out and the remaining part of the array shall be returned as a single array.
   - **(iii)** **Zero :** If this parameter is Zero then the array returned will have only one element i.e. the whole string.

When this parameter is not provided the array returned contains the total number of element formed after separating the string with the separator.

Example :

```
<html>
<body>
<?php
$str = "PHP: Welcome to the world of PHP";
print_r (explode(" ",$str));
?>
</body>
</html>
```

Output :

```
Array ( [0] => PHP: [1] => Welcome [2] => to [3] => the [4] => world [5] => of [6] => PHP )
```

2.2.2(C) Array_fip()

– The **array_flip()** function flips/exchanges all keys with their associated values in an array.

– This built-in function of PHP array_flip() is used to exchange elements within an array, i.e., exchange all keys with their associated values in an array and vice-versa.

Syntax :

```
array_flip(array);
```

Example :

```
<html>
<body>
<?php
$a1=array("CO"=>"Computer Engg","IF"=>"Information Tech","EJ"=>"Electronics and Telecomm");
$result=array_flip($a1);
print_r($result);
?>
</body>
</html>
```

Output :

```
Array ( [Computer Engg] => CO [Information Tech] => IF [Electronics and Telecomm] => EJ )
```

2.3 Traversing Arrays

The most common task with arrays is to do something with every element; for instance, sending mail to each element of an array of addresses, updating each file in an array of filenames, or adding up each element of an array of prices.

We can traverse an indexed array using loops in PHP. We can loop through the indexed array in two ways. First by using for loop and secondly by using for each.

For example,

```
<?php
// Creating an indexed array
$course = array("CO", "IF", "EJ", "ME", "CE");
// One way of Looping through an array using foreach
echo "Looping using foreach: <br>";
foreach ($course as $val)
{
echo $val. "<br>";
}

// count() function is used to count the number of elements in an array
$total = count($course);
echo "<br>The number of elements are $total <br>";
```

```
// Another way to loop through the array using for
echo "Looping using for: <br>";
for($n = 0; $n < $total; $n++)
{
   echo $course[$n], "<br>";
}
?>
```

Output :

```
Looping using foreach :
CO
IF
EJ
ME
CE
The  number of elements are 5
Looping using for :
CO
IF
EJ
ME
CE
```

2.3.1 Modifying data in Arrays

You can modify the values inside an array.

Example :

```
<HTML>
<HEAD>
<TITLE>
            Modifying an array
</TITLE>
</HEAD>

<BODY>
<H1>
            Modifying an array
</H1>
```

```
<?php
        $course[0] = "Computer Engg";
        $course[1] = "Information Tech";
        $course[2] = "Electronics and Telecomm";
       echo "<h2> Before modification</h2>";
       echo $course[0], "<BR>";
        echo $course[1], "<BR>";
        echo $course[2], "<BR>";

        echo "<br><h2>After modification</h2>";

        $course[2] = "Mechanical Engg";
        $course[] = "Civil Engg";
        for ($i= 0; $i< count($course); $i++)
        {
          echo $course[$i], "<BR>";
       }
    ?>
</BODY>
</HTML>
```

Output :

```
Modifying an array
Before modification
Computer Engg
Information Tech
Electronics and Telecomm

After modification
Computer Engg
Information Tech
Mechanical Engg
Civil Engg
```

2.3.2 Deleting Array Elements

Q. What is function name in PHP, use to delete an element from array? Give example. **(4 Marks)**

The **unset() function is used to remove element from the array**. The **unset()** function is used to destroy any other variable and same way use to delete any element of an array.

The **unset ()** function accepts a variable name as parameter and destroy or unset that variable.

Syntax :

```
void unset ($var,...);
```

Example 1 :

```php
<?php

   $course = array("CO", "IF", "EJ", "ME", "CE");
   echo "<h2> Before deletion:</h2><br>";
   echo "$course[0] <br>";
   echo "$course[1] <br>";
   echo "$course[2] <br>";
   echo "$course[3] <br>";
   echo "$course[4] <br>";
   // unset command accepts 3rd index and thus removes the array element at
   // that position
   echo "<h2> Before deletion:</h2><br>";
   unset($course[3]);
   print_r($course);
   echo "<h2> Delete entire array elements:</h2><br>";
   unset($course); //delete all elements from an array
?>
```

Output :

```
Before deletion:
CO
IF
EJ
ME
CE
Before deletion:

Array ( [0] => CO [1] => IF [2] => EJ [4] => CE )
```

```
Delete entire array elements:
```

Another way to delete an element from array like this, just setting an array element t to an empty string.

Example 2 :

```
<HTML>
<HEAD>
<TITLE>
            Modifying an array
</TITLE>
</HEAD>

<BODY>
<H1>
            Deletion of array element
</H1>

<?php
        $course[0] = "Computer Engg";
        $course[1] = "Information Tech";
        $course[2] = "Electronics and Telecomm";
        echo "<h2> Before Deletion</h2>";
        echo $course[0], "<BR>";
        echo $course[1], "<BR>";
        echo $course[2], "<BR>";

        echo "<br><h2>After Deletion</h2>";

$course[2] = " ";//assign empty string to course[2]

        for ($i= 0; $i< count($course); $i++)
        {
          echo $course[$i], "<BR>";
        }
     ?>
```

```
</BODY>
</HTML>
```

Output :

```
Deletion of array element
Before Deletion
Computer Engg
Information Tech
Electronics and Telecomm
After Deletion
Computer Engg
Information Tech
```

2.3.3 The array_values()

Function returns all the values from the array and indexes the array numerically.

Syntax :

```
array array_values ( array $array )
```

2.3.4 Sorting Arrays

Sorting refers to ordering data in an alphabetical, numerical order and increasing or decreasing fashion according to some linear relationship among the data items depends on the value or key. Sorting greatly improves the efficiency of searching.

Following are the Sorting Functions for Arrays in PHP :

1. sort(): sorts arrays in ascending order
2. rsort(): sorts arrays in descending order
3. asort(): sorts associative arrays in ascending order, according to the value
4. ksort(): sorts associative arrays in ascending order, according to the key
5. arsort(): sorts associative arrays in descending order, according to the value
6. krsort(): sorts associative arrays in descending order, according to the key

Example :

```
<?php
$num = array(40, 61, 2, 22, 13);
echo "Before Sorting:<br>";
$arrlen= count($num);
for($x = 0; $x < $arrlen; $x++)
```

```
  {
   echo $num[$x];
   echo "<br>";
  }


sort($num);
echo "After Sorting in Ascending order:<br>";
$arrlen= count($num);
for($x = 0; $x < $arrlen; $x++)
  {
   echo $num[$x];
   echo "<br>";
  }
echo "After Sorting in Descending order:<br>";
rsort($num);
$arrlen= count($num);
for($x = 0; $x < $arrlen; $x++)
  {
   echo $num[$x];
   echo "<br>";
  }
?>
```

Output :

```
Before Sorting:
40
61
2
22
13
After Sorting in Ascending order:
2
13
22
```

```
40
61
After Sorting in Descending order:
61
40
22
13
2
```

The following function sorts an associative array in ascending order, according to the value by using **asort()** and key by using **ksort()** :

For example,

```
<?php
$percent = array("Manisha"=>"80", "Yogita"=>"78", "Siddhesh"=>"68");
echo "<b>Sorting according to Value:</b><br>";
asort($percent);

foreach($percent as $x => $x_value)
 {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}

echo "<br><br><b>Sorting according to Key:</b><br>";
ksort($percent);
foreach($percent as $x => $x_value)
 {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```

Output :

```
Sorting according to Value :
Key=Siddhesh, Value=68
Key=Yogita, Value=78
Key=Manisha, Value=80
```

```
Sorting according to Key :
Key=Manisha, Value=80
Key=Siddhesh, Value=68
Key=Yogita, Value=78
```

2.3.5 Splitting and Merging Arrays

You can also cut up and merge arrays when needed. For example, say you have 5 courses in array of course and want to get a subarray consisting of the last two items. You can do this with the **array_slice()** function, passing it the array you want to get a section of, the offset at which to start and the length of the array you want to create.

Syntax :

```
array_slice(array, start, length, preserve) ;
```

Where,

- **Array :** Specifies an array and this is required parameter.
- ***Start :*** This is required parameter and contain numeric value. Specifies where the function will start the slice. 0 = the first element. If this value is set to a negative number, the function will start slicing that far from the last element. -2 means start at the second last element of the array.
- **Length :** This is optional parameter and contain numeric value. Specifies the length of the returned array. If this value is set to a negative number, the function will stop slicing that far from the last element. If this value is not set, the function will return all elements, starting from the position set by the start-parameter.
- **Preserve:** This is optional parameter. Specifies if the function should preserve or reset the keys. Possible values :
  - true - Preserve keys
  - false - Default Reset keys

Example 1 :

```
<?php
        $course[0] = "Computer Engg";
        $course[1] = "Information Tech";
        $course[2] = "Electronics and Telecomm";
       echo "<h2> Before Splitting array:</h2>";
       echo $course[0], "<BR>";
        echo $course[1], "<BR>";
        echo $course[2], "<BR>";

        echo "<br><h2>After Splitting:</h2>";

       $subarray = array_slice($course, 1, 2);
        foreach ($subarray as $value)
```

```
         {
          echo "Course: $value<br>";
         }

      ?>
```

Output :

```
Before Splitting array:
Computer Engg
Information Tech
Electronics and Telecomm
After Splitting:
Course: Information Tech
Course: Electronics and Telecomm
```

You can also merge two or more arrays with **array_merge():**

Example 2 :

```
<?php
   $sem3 = array("Object Oriented Programming", "Principle of Database", "Data Structure");
   $sem4 = array("Database Management", "Java Programming", "Software Engg.", "Computer Network");

   $subject = array_merge($sem3, $sem4);

   foreach ($subject as $value)
{
echo "Subject: $value<br>";
}
?>
```

Output :

```
Subject: Object Oriented Programming
Subject: Principle of Database
Subject: Data Structure
Subject: Database Management
Subject: Java Programming
Subject: Software Engg.
```

Subject: Computer Network

2.3.6 PHP Array Functions

The array functions allow you to access and manipulate arrays.

Function	Description	Syntax	Example
Array_diff()	Compare the **values** of two arrays, and return the differences.	array_diff(*array1, array2, array3, ...*)	$a1=array('PHP','C','Java','Perl'); $a2=array('PHP','ASP','Java','Python'); array_diff($a1,$a2); output : Array ([1] => C [3] => Perl)
Array_intersect()	Compare the **values** of two arrays, and return the matches	array_intersect(*array1, array2, array3, ...*)	$a1=array('PHP','C','Java','Perl'); $a2=array('PHP','ASP','Java','Python'); array_intersect($a1,$a2); output : Array ([0] => PHP [2] => Java)
Array_chunk()	Split an array into chunks of two and preserve the original keys	array_chunk(*array, size, preserve_key*)	$a1=array('PHP','C','Java','Perl'); $ac=array_chunk($a1,2,true); Output : Array ([0] => Array ([0] => PHP [1] => C) [1] => Array ([2] => Java [3] => Perl))
Array_combine()	Create an array by using the elements from one "keys" array and one "values" array	array_combine(*keys, values*)	$a1=array("PHP","Java","Perl"); $a2=array("10","20","30"); $ac=array_combine($a1,$a2); Output : Array ([PHP] => 10 [Java] => 20 [Perl] => 30)
Array_unique()	Remove duplicate values from an array	array_unique(*array, sorttype*)	$a1=array("10","20","30","20"); $ac=array_unique($a1); Output : Array ([0] => 10 [1] => 20 [2] => 30)
Array_count_values ()	Count all the values of an array	array_count_values(*array*)	$a1=array("10","20","30","20"); print_r(array_count_values($a1)); Output : Array ([10] => 1 [20] => 2 [30] => 1)

Function	Description	Syntax	Example
Array_merge()	Merge two arrays into one array	array_merge(*array1, array2, array3, ...*)	$a1=array("10","20"); $a2=array("30","40"); print_r(array_merge($a1,$a2)); Output : Array ([0] => 10 [1] => 20 [2] => 30 [3] => 40)
Array_pop()	Delete the last element of an array	array_pop(*array*)	$a1=array("10","20","30"); array_pop($a1); Output : Array ([0] => 10 [1] => 20)
Array_product()	Calculate and return the product of an array	array_product(*array*)	$a1=array(10,20); echo(array_product($a1)); Output : 200
Array_push()	Inserts one or more elements to the end of an array.	array_push(*array, value1, value2, ...*)	$a1=array(10,20); array_push($a1,"PHP","Perl"); Output : Array ([0] => 10 [1] => 20 [2] => PHP [3] => Perl)
Array_reverse()	Return an array in the reverse order	array_reverse(*array, preserve*)	$a1=array("a"=>"PHP", "b"=>"Java","c"=>"Perl"); print_r(array_reverse($a1)); Output : Array ([c] => Perl [b] => Java [a] => PHP)
Array_search()	Searchs an array for a value and returns the key	array_search(*value, array, strict*)	$a1=array("a"=>"PHP", "b"=>"Java","c"=>"Perl"); echo(array_search("Java",$a1)); output : b
Array_sum()	Return the sum of all the values in the array	array_sum(*array*)	$a1=array(10,20,30); echo(array_sum($a1)); output : 60
Count()	Returns the number of elements in an array	count(*array)*	$a1=array(10,20,30); echo(count($a1)); output : 3

2.4 Functions and Its Types

Q. Define function. How to define user defined function? Explain with example **(4 Marks)**

- PHP functions are similar to other programming languages. A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value.
- They are built-in functions but PHP gives you option to create your own functions as well. A function will be executed by a call to the function. You may call a function from anywhere within a page.
- There are two parts which should be clear to you
  - Creating a PHP Function
  - Calling a PHP Function
- It's very easy to create your own PHP function. Suppose you want to create a PHP function which will simply write a simple message on your browser when you will call it. Following example creates a function called writeMessage() and then calls it just after creating it.
- A user-defined function declaration starts with the word function :

Syntax :

```
function functionName()
 {
   code to be executed;
}
```

Example :

```
<html>
<head>
<title>Writing PHP Function</title>
</head>
<body>

<?php
/* Defining a PHP Function */
function writeMessage()
{
        echo "Welcome to PHP world!";
}
```

```
/* Calling a PHP Function */
        writeMessage();
?>
</body>
</html>
```

Output :

```
Welcome to PHP world!
```

PHP Functions with Parameters :

PHP gives you option to pass your parameters inside a function. You can pass as many as parameters you like. These parameters work like variables inside your function. Following example takes two integer parameters and add them together and then print them.

Example :

```
<?php
        function addfunc($num1, $num2)
        {
           $sum = $num1 + $num2;
           echo "Sum of the two numbers is : $sum";
        }
        addfunc(50, 20);
     ?>
```

This will display following result :

```
Sum of the two numbers is : 70
```

PHP Functions returning value :

– A function can return a value using the **return** statement in conjunction with a value or object. return stops the execution of the function and sends the value back to the calling code.

– You can return more than one value from a function using **return array (1, 2, 3, 4)**.

– Following example takes two integer parameters and add them together and then returns their sum to the calling program. Note that **return** keyword is used to return a value from a function.

For example,

```
<?php
        function add($num1, $num2)
        {
           $sum = $num1 + $num2;
           return $sum;
```

```
    }
    $return_value = add(50, 20);
    echo "Returned value from the function : $return_value";
  ?>
```

This will display following result :

```
Returned value from the function : 70
```

2.4.1 Variable Function

– PHP supports the concept of variable functions. This means that if a variable name has parentheses appended to it, PHP will look for a function with the same name as whatever the variable evaluates to, and will attempt to execute it. Among other things, this can be used to implement callbacks, function tables and so forth.

– Variable functions won't work with language constructs such as echo, print, unset(), isset(), empty(), include, require and the like. Utilize wrapper functions to make use of any of these constructs as variable functions.

Example 1 :

```
<?php
function simple()
{
   echo "In simple()<br />\n";
}
function data($arg = '')
{
   echo "In data(); argument was '$arg'.<br />\n";
}
$func = 'simple';
$func();        // This calls simple()

$func = 'data';
$func('test');  // This calls data()
?>
```

Output :

```
In simple()
In data(); argument was 'test'.
```

Example 2 :

```
<?php
```

```
  // Declare a variable and initialize with function
$function = function()
 {
    echo 'InformationTechnology';
 };
  // Check is_callable function contains
// function or not
if( is_callable( $function ) )
 {
    echo "It is function";
}
else
{
    echo "It is not function";
}
  // Declare a variable and initialize it
$var = "IF";
echo "<br>";
  // Check is_callable function contains function or not
if( is_callable( $var ) )
{
    echo "It is function";
}
else
 {
    echo "It is not function";
}
?>
```

Output :

```
It is function
It is not function
```

Above example uses **is_callable()** function to verify whether the parameter is a function or not.

2.4.2 Anonymous or Lambda Function

Introduction to anonymous functions

When you define a function, you specify a name for it. Later, you can call the function by the name.

For example, to define a function that multiplies two numbers, you can do it as follows:

```
<?php

function multiply($x, $y)
{
        return $x * $y;
}
```
Code language: HTML, XML (xml)

The `multiply()` function accepts two arguments and returns the result. To call the `multiply()` function, you pass the arguments to it like this:

```
<?php

multiply(10, 20);
```
Code language: HTML, XML (xml)

In this example, the `multiply()` is a named function. And you can reuse it as many times as you want.

Besides named functions, PHP allows you to define anonymous functions.

An anonymous function is a function that doesn't have a name.

The following example defines an anonymous function that multiplies two numbers:

```
<?php

function ($x, $y) {
        return $x * $y;
};
```
Code language: HTML, XML (xml)

Since the function doesn't have a name, you need to end it with a semicolon (;) because PHP treats it as an expression.

This anonymous function is not useful at all because you cannot use it like a named function.

To use an anonymous function, you need assign it to a variable and call the function via the variable.

The following example assigns the anonymous function to the `$multiply` variable:

```
<?php
// ...
```

```
$multiply = function ($x, $y) {
            return $x * $y;
};
Code language: HTML, XML (xml)
```

And this calls the anonymous function via the $multiply variable:

```
echo $multiply(10, 20);
```

There are times when you need to create a small localized throw-away function consisting of a few lines for a specific purpose, such as a callback. It is unwise to pollute the global namespace with these kind of single use functions. For such an event you can create anonymous or lambda functions using create_function (). Anonymous functions allow the creation of functions which have no specified name. An example is as follows :

Example 1 :

```
<?php
 $str = "hello world!";
$lambda = create_function('$match', 'return "friend!";');
$str = preg_replace_callback('/world/', $lambda, $str);
echo $str ;
 ?>
```

Output :

```
hello friend!
```

– Here we have created a small nameless (Anonymous) function which is used as a callback in the ***preg_replace_callback( )*** function. preg_replace_callback( ) — Perform a regular expression search and replace using a callback.

– Although *create_function* lets you create anonymous functions. We create an unnamed function and assign it to a variable, including whatever parameters the functions accept and then simply use the variable like an actual function.

Example 2 :

```
<?php
$greet = function($name)
{
    printf("Hello %s\r\n", $name);
};

$greet('World');
echo"<br>";
$greet('PHP');
?>
```

Output :

```
Hello World
Hello PHP
```

Note the ending semicolon at the end of the defined function. This is because the function definition is actually a statement, and statements always end with a semicolon.

Example 3 :

```
<?php
 $str = "Hello World!!";
$func = function($match)
{
   return "PHP!!!";
};
 $str = preg_replace_callback('/World/', $func, $str);
echo $str ;
 ?>
```

Output :

```
Hello PHP!!!!!
```

2.5 Operations on String and String Functions

2.5.1 Operations on Strings

- PHP is a string oriented and it comes packed with many string functions.
- A string is a collection of characters. String is one of the data types supported by PHP.
- The string variables can contain alphanumeric characters.

Creating a string : There are 2 ways to create a string in php.

(i) Creating Strings Using Single quotes: The simplest way to create a string is to use single quotes. Let's look at an **example** that creates a simple string in PHP.

```
<?php
   var_dump('PHP is string-oriented');
?>
```

Output:

```
string(22) "PHP is string-oriented"
```

(ii) PHP Create Strings Using Double quotes

- The double quotes are used to create relatively complex strings compared to single quotes.

– Variable names can be used inside double quotes and their values will be displayed.

Example :

```
<?php
$name='PHP';
echo "$name is string-oriented";
?>
```

Output :

```
PHP is string-oriented
```

(A) Converting to and from Strings

– Because data is sent to you in string format, and because you will have to display your data in string format in the user's browser, converting between strings and numbers is one of the most important interface tasks in PHP.

– To convert to a string, you can use **cast(string)** or the **strval()**, which returns the string value of the item you pass to it.

– Strings in PHP can be converted to numbers (float / int / double) very easily. In most use cases, it won't be required since PHP does implicit type conversion. There are many methods to convert string into number in PHP some of them are as follows :

(i) Using number_format() Function :

– The **number_format ()** function is used to convert string into a number.

– It returns the formatted number on success otherwise it gives E_WARNING on failure.

Example :

```
<?php
$num = "20100.3145";
// Convert string in number using number_format()
echo number_format($num), "<br>";
// Convert string in number using number_format()
echo number_format($num, 3);
?>
```

Output :

```
20,100
20,100.315
```

(ii) Using type casting :

Typecasting can directly convert a string into float, double or integer primitive type. This is the best way to convert a string into number without any function.

Example :

```
<?php

// Number in string format
$num = "20100.3145";

// Type cast using int
echo (int)$num, "<br>";

// Type cast using float
echo (float)$num, "<br>";

// Type cast using double
echo (double)$num;
?>
```

Output :

```
20100
20100.3145
20100.3145
```

(iii) Using intval() and floatval() Function :

The intval() and floatval() functions can also be used to convert the string into its corresponding integer and float values respectively.

Example 1 :

```
<?php
  // Number in string format
$num = "1300.314";
  // intval() function to convert string into integer
echo intval($num), "<br>";
  // floatval() function to convert string to float
echo floatval($num);
?>
```

Output :

```
1300
1300.314
```

(B) Formatting Text Strings :

– Because all the data sent to your PHP scripts and the data you send back to the browser is in text form, formatting that data is one of the most important things you'll be doing in PHP.

– For example, what if you want to make sure that price you're displaying has exactly two places after the decimal point? It turns out that there are two PHP functions that specifically handle formatting of text strings, and here they are- **printf( ) and sprintf( ).**

– **printf( )** function prints a string (much like echo), and **sprintf( )** also "prints" its data, but in this case, the output is a string-that is, it returns a string.

Syntax :

```
printf (format [ , args] );

sprintf (format [ , args] );
```

The following is a brief discussion of the Formats and Datatypes that can be specified in PHP. Each one of them is implemented with a preceding percentile symbol or '%'.

Formatting Values

– **Sign specifier** can be used to forcibly display the sign (- or +) to be used on a number. By default, only the – sign is displayed on negative numbers. Using this specifier positive numbers are shown with a preceding +. This can be achieved using a + symbol and can be implemented only upon numeric values.

Example :

%+d // Specify the integer along with it's sign (+ or -).

– **Padding specifier** can be used to specify what character will be used for padding the results to any defined string size. By default, spaces are used as padding. An alternate padding character can be specified by prefixing it with a single quote or '.

Example :

%'0d // Pad with 0s to achieve the right length.

– **Alignment specifier** can be used to specify the alignment of the result i.e. whether left-justified or right-justified. By default it is right-justified. Using a – character makes it left-justified.

Example :

%-s // Specifies the alignment as left-justified.

– **Width specifier** can be used to specify the minimum number of characters to be present in the result itself. It can be specified using any number denoting the minimum width. It is seen in use with padding specifier the most.

Example :

%'05d // Specifies there should be at least 5 digits, if less, then 0s are filled to get the desired result.

– **Precision Specifier** can be used to specify the precision while working with real numbers. A period or '.' followed by an optional decimal digit string that refers to the decimal digits to be displayed after the decimal. When using this specifier on a string, it specifies the maximum character limit on the string. Example,

%.5f // Defines Real Number Precision.

%.2s // Maximum Character to be allowed in a string.

– **Type Specifier** that says what type the argument data should be treated as.

Here are the possible **Type specifiers** :

- % : To display %. No directive is required.
- b : The directive refers to an integer and displayed as a binary number.
- c : The directive refers to an integer and displayed as the corresponding ASCII character.
- d : The directive refers to an integer and displayed as a decimal number.
- e : The directive refers to scientific notation (e.g. 2.12e+3).
- E : Alias of 'e'.
- f : The directive refers to a float and displayed as a real number (locale aware).
- F : The directive refers to a float and displayed as a real number (non-locale aware).
- o : The directive refers to an integer and displayed as an octal number.
- s : The directive is treated and displayed as a string.
- u : The directive refers to an integer and displayed as an unsigned decimal number.
- x : The directive refers to an integer and displayed as a hexadecimal number (with lowercase letters).
- X : The directive refers to an integer and displayed as a hexadecimal number (with uppercase letters).

For example,

```
<?php
printf("I have %s pens and %s pencils. <br><br>", 4,18);
$str=sprintf("After using I have %s pens and %s pencils.<br>",2,9);
echo $str, "<br>";
$y=2019;
$m=11;
$date=4;
echo "The date is:";
printf("%04d-%02d-%02d<br>", $y, $m, $date);


?>
```

Output :

```
I have 4 pens and 18 pencils.
After using I have 2 pens and 9 pencils.

The date is:2019-11-04
```

2.5.2 String Functions

Q.	Explain any four string functions in PHP	**(4 Marks)**
Q.	What are the use of strlen() and strops() functions	**(4 Marks)**

Q. PHPPHPPHPPHP. Which string function is used to get this type of output ? **(2 Marks)**

PHP string functions are used to manipulate string values.

Function	Description	Syntax	Example
str_word_count( )	Count the number of words in a string	Str_word_count(String)	<?php echo str_word_count("Welcome to PHP world!"); ?> Output: 4
strlen()	Returns the length of a string	Strlen(String)	<?php echo strlen("Welcome to PHP"); ?> Output: 14
strrev()	Reverses a string	Strrev(String)	<?php echo strrev("Information Technology"); ?> Output: ygolonhceT noitamrofnI
strpos()	Returns the position of the first occurrence of a string inside another string (case-sensitive)	Strops(String, text)	<?php echo strpos("PHP contains for loop, for each and while loop","loop"); ?> Output: 17
str_replace()	Replaces some characters in a string (case-sensitive)	Str_replace(string tobe replaced, text, string)	<?php echo str_replace("Clock","Click","Click and Clock"); ?> Output: Click and Click
ucwords()	Convert the first character of each word to	Ucwords(String)	<?php echo ucwords("welcome to php

Function	Description	Syntax	Example
	uppercase		world"); ?> Output: Welcome To Php World
strtoupper()	Converts a string to uppercase letters	Strtoupper(String)	<?php echo strtoupper("information technology "); ?> Output: INFORMATION TECHNOLOGY
strtolower()	Converts a string to lowercase letters	Strtolower(String)	<?php echo strtolower("INFORMATION TECHNOLOGY"); ?> Output: information technology
Str_repeat()	Repeating a string with a specific number of times.	Str_repeat(String, repeat)	<?php echo str_repeat("*",10); ?> Output: **********

strcmp()	Compare two strings (case-sensitive).
	If this function returns 0, the two strings are equal.
	If this function returns any negative or positive numbers, the two strings are not equal .
		Strcmp(String1, String2)	<?php
			echo strcmp("Hello world!","Hello world!");
			?>
			Output:
			0
			
			Another example:
			<?php
			echo strcmp(" world!","Hello PHP!");
			?>

Function	Description	Syntax	Example
			Output: -40
Substr()	substr() function used to display or extract a string from a particular position.	Substr(String, start, length)	<?php echo substr("Welcome to PHP",11)." "; echo substr("Welcome to PHP",0,7)." "; ?> Output: PHP Welcome
Str_split()	To convert a string to an array	str_split(string,length);	$str="PHP" Str_split($str); Output: Array ([0] => P [1] => H [2] => P)
Str_shuffle()	To randomly shuffle all the character of a string	str_shuffle (string $str)	$str="PHP" Str_shuffle($str); Output: PPH
Trim()	Removes white spaces and predefined characters from a both the sides of a string.	trim(string,charlist)	$str=" Welcome " Output" Welcome
Rtrim()	Removes the white spaces from end of the string	string rtrim (string $str [, string $character_mask])	$str="Hello PHP" Rtrim($str,"PHP") Output: Hello
Ltrim()	Removes the white spaces from left side of the string	ltrim(string,charlist);	$str=" PHP" Output: PHP
Chop()	Remove whitespace or other predefined character from the right end of a string.	chop(string,charlist);	$str="Hello PHP" Cho($str,"PHP"); Output: Hello
Chunk_split()	Splits a string into smaller parts or chunks.	chunk_split(string,length,end)	$str="Hello PHP";

Function	Description	Syntax	Example
			Chunk_split($str,6,"…"); Output: Hello… PHP…

2.5.3 PHP Maths Functions

PHP provides many predefined functions that can be used to perform mathematical operations.

Function	Description	Syntax	Example
abs()	Returns absolute value of given number.	number abs (mixed $number)	Abs(-7)= 7
ceil()	Rounds fractions up.	float ceil (float $value)	Ceil(3.3)=4 Ceil(-4.6)=-4
floor()	Rounds fractions down.	float floor (float $value)	Floor(3.3)=3 Floor(-4.6)=-5
sqrt()	Returns square root of given argument.	float sqrt (float $arg)	Sqrt(16)=4
decbin()	Converts decimal number into binary. It returns binary number as a string.	string decbin (int $number)	Decbin(2)=10 Decbin(10)=1010
dechex()	Converts decimal number into hexadecimal. It returns hexadecimal representation of given number as a string.	string dechex (int $number)	Dechex(10)=a Dechex(22)=16
decoct()	Converts decimal number into octal. It returns octal representation of given number as a string.	string decoct (int $number)	Decoct(10)=12 Decoct(22)=26
bindec()	Converts binary number into decimal.	number bindec (string $binary_string)	Bindec(10)=2 Bindec(1010)=10
sin()	Return the sine of a number.	float sin (float $arg)	Sin(3)= 0.1411200080598
cos()	Return the cosine of a number.	float cos (float $arg)	Cos(3)= -0.989992496600
tan()	Return the tangent of a number.	float tan (float $arg)	Tan(10)= 0.64836082745
Base_convert()	Convert any base number to any base number. For example, you can convert hexadecimal number to	string base_convert (string $number ,	Base_convert($n1, 10,2)=1010

Function	Description	Syntax	Example
	binary, hexadecimal to octal, binary to octal, octal to hexadecimal, binary to decimal etc.	int $frombase , int $tobase)	
Pi()	Returns the value of pi.	pi()	3.14159265
Exp()	Returns exponent of e	exp(x);	Exp(0)=1
Fmod()	Returns the floating point remainder.	float fmod (float $x , float $y);	Fmod(5,2)=1
Hexdec()	Convert a hexadecimal to a decimal number.	number hexdec (string $hex_string)	Hexdec(a)=10
Log()	To find the natural logarithm of a number	log(number,base);	Log(2)= 0.69314718055
Max()	To find the highest value.	max(array_values);	Max(20,30,5,10)= 30
Min()	To find the lowest value.	min(value1,value2, value3...);	Min(20,30,5,10)= 5
Octdec()	To convert an octal number to a decimal number.	octdec(octal_string);	Octdec(36)=30
Pow()	It raises the first number to the power of the second number.	number pow (number $base , number $exp)	Pow(3,2)=9

Round(): This function takes numeric value as argument and returns the next highest integer value by rounding up value if necessary.

– **PHP_ROUND_HALF_UP :** (set by Default) Rounds number up to precision decimal, when it is half way there. Rounds 1.5 to 2 and -1.5 to -2

– **PHP_ROUND_HALF_DOWN :** Round number down to precision decimal places, when it is half way there. Rounds 1.5 to 1 and -1.5 to -1

– **PHP_ROUND_HALF_EVEN :** Round number to precision decimal places towards the next even value.

– **PHP_ROUND_HALF_ODD :** Round number to precision decimal places towards the next odd value.

Example :

```
round(1.95583, 2)= 1.96
round(1241757, -3)= 124200
round(9.5, 0, PHP_ROUND_HALF_UP)=10
round(9.5, 0, PHP_ROUND_HALF_DOWN)=9
round(9.5, 0, PHP_ROUND_HALF_EVEN)=10
round(9.5, 0, PHP_ROUND_HALF_ODD)=9
```

2.5.4 PHP Date Functions

PHP date function is an in-built function that works with date data types. The PHP date function is used to format a date or time into a human readable format.

Syntax: date(format, timestamp)

Example:

```
<?php
echo "Today's date is :";
$today = date("d/m/Y");
echo $today;
?>
```

Output :

```
Today's date is :11/11/2019
```

Format: Specifies the format of the outputted date string. The following characters can be used to display various formatted output :

Format	Description
D	The day of the month (from 01 to 31)
D	A textual representation of a day (three letters)
J	The day of the month without leading zeros (1 to 31)
l (lowercase 'L')	A full textual representation of a day
N	numeric representation of a day (1 for Monday, 7 for Sunday)
S	The English ordinal suffix for the day of the month (2 characters st, nd, rd or th. Works well with j)
Z	The day of the year (from 0 through 365)
W	A numeric representation of the day (0 for Sunday, 6 for Saturday)
W	week number of year (weeks starting on Monday)
F	A full textual representation of a month (January through December)
M	A numeric representation of a month (from 01 to 12)
M	A short textual representation of a month (three letters)

Format	Description
T	The number of days in the given month
Y	A four digit representation of a year
Y	A two digit representation of a year
A	Lowercase am or pm
A	Uppercase AM or PM
G	12-hour format of an hour (1 to 12)
G	24-hour format of an hour (0 to 23)
H	12-hour format of an hour (01 to 12)
H	24-hour format of an hour (00 to 23)

Time(): The time() function is used to get the current time as a Unix timestamp (the number of seconds since the beginning of the Unix epoch: January 1 1970 00:00:00 GMT).

Example:

```
<?php
$timestamp = time();
echo($timestamp);
echo "<br/>";
echo(date("F d, Y h:i:s A", $timestamp));
?>
```

Output :

```
1573445225
November 11, 2019 05:07:05 AM
```

2.6 Basic Graphics Concepts

Basic Graphics concepts :

The Web is more than just text. Images appear in the form of logos, buttons, photographs, charts, advertisements, and icons PHP supports graphics creation with the GD and Imlib2 extensions. In this chapter we'll show you how to generate images dynamically with PHP, using the GD extension.

2.6.1 Creating an Image

To create an image in memory to work with, you start with the GD2 **imagecreate ()** function.

Syntax :

```
imagecreate(x_size, y_size);
```

The x_size and y_size parameters are in pixels.

Here's how to create a first image.

```
$img_height = 100;
$img_width=300;
$img=imagecreate($img_height,  $img_width);
```

Next, to set colors to be used in the image, you use the **imagecolorallocate( )** function.

Syntax :

```
imagecolorallocate(image, red, green, blue);
```

You pass this function the image you're working with, as well as the red, green, and blue components as 0- 255 value. For example, if you want solid red, you'd pass imagecolorallocate a red value of 255, and blue and green values of 0.

```
$back_color= imagecolorallocate ($image, 200, 0, 0);
```

To send a JPEG image back to the browser, you have to tell the browser that you're doing so with the header function to set the image's type, and then you send the image with the **imagejpeg()** function like this:

```
$img_height = 100;
$img_width=300;
$img=imagecreate($img_height,  $img_width);
back_color= imagecolorallocate ($image, 200, 0, 0);
header('Content-Type:image.jpeg');
imagejpeg($img);
```

Here are some of the image-creating functions for various image formats you can use:

- imagegif(): Output a GIF image to browser or file.
- imagejpeg(): Output a JPEG image to browser or file.
- imagewbmp(): Output a WBMP image to browser or file.
- imagepng(): Output a PNG image to browser or file.

After sending the image, you can destroy the image object with the **imagedestroy()** function.

2.6.2 Images with text

The imagestring() function is an inbuilt function in PHP which is used to draw the string horizontally. This function draws the string at given position.

Syntax:

bool imagestring($image, $font, $x, $y, $string, $color)

- $image: The imagecreate or imagecreatetruecolor() function is used to create an image in a given size.
- $font: This parameter is used to set the font size. Inbuilt font in latin2 encoding can be 1, 2, 3, 4, 5 or other font identifiers registered with imageloadfont() function.
- $x: This parameter is used to hold the x-coordinate of the upper left corner.

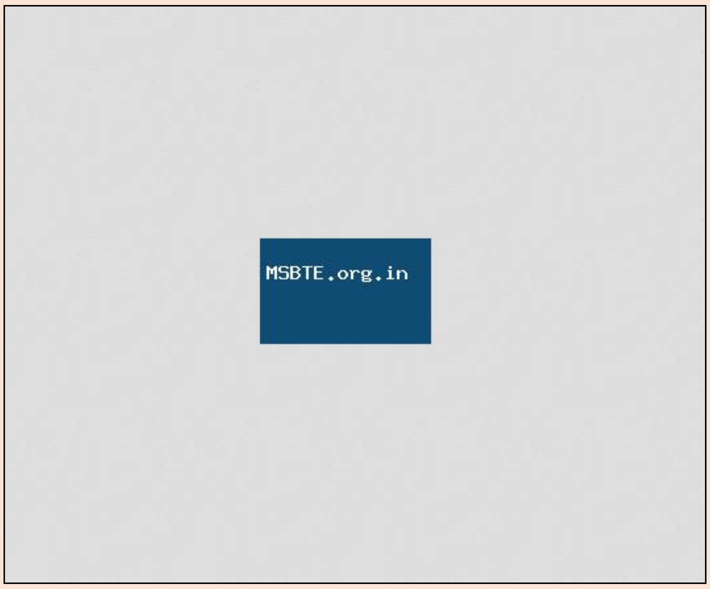
- $y: This parameter is used to hold the y-coordinate of the upper left corner.
- $string: This parameter is used to hold the string to be written.
- $color: This parameter is used to hold the color of image.

Example: img2.php

Q.	Write a program to display Text in a Image	**(4 Marks)**

```
<?php
header ("Content-type: image/png");
$img = ImageCreate (130, 50);
$bg_color = ImageColorAllocate ($img, 240, 240,140);
$txt_color = ImageColorAllocate ($img, 0, 0, 0);
ImageString ($img, 5, 5, 18, "MSBTE.org.in", $txt_color);
imagepng ($img);
?>
```

Output :



```
MSBTE.org.in
```

Displaying images in HTML pages :

The image created by img2.php is a standard PNG image, so there's no reason you can't embed it in a Web page. For example, if you had a PNG on the server, pqr.png, you could display it this way in a web page:

```
<img src="pqr.png">
```

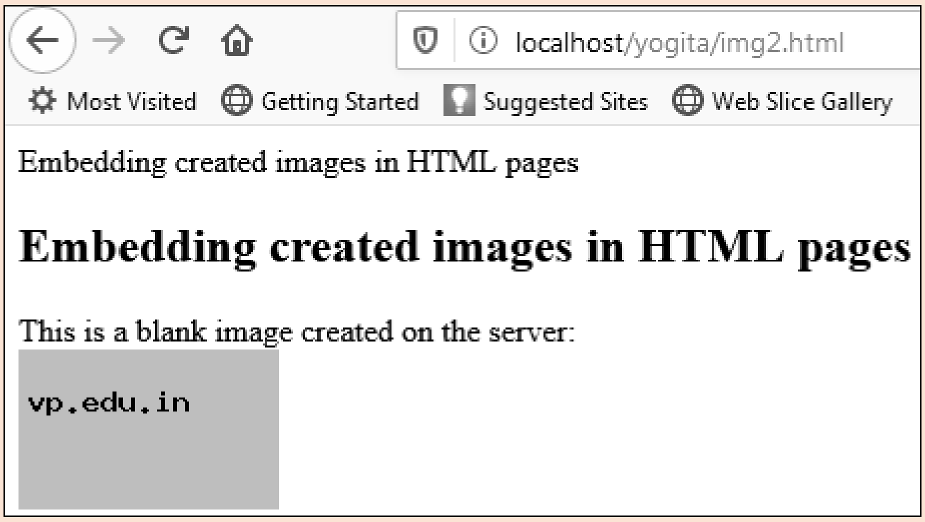
In the same way, you can give the name of the script that generates a PNG image, img2.php, as the src attribute like this :

```
<img src="img2.php">
```

Here's what a Web page, img2.html that displays the blank box created by img2.php looks like:

```
<html>
<head>
<tilte>
Embedding created images in HTML pages
</title>
</head>
<body>
<h2> Embedding created images in HTML pages </h2>
This is a blank image created on the server:
<br>
<img src="img2.php">
</body>
</html>
```

Output :



Embedding created images in HTML pages

Embedding created images in HTML pages

This is a blank image created on the server:

vp.edu.in

2.6.3 Scaling Images

- There are two ways to change the size of an image. The **ImageCopyResized( )** function is available in all versions of GD. The **ImageCopyResampled( )** function is new in GD 2.x and features pixel interpolation to give smooth edges and clarity to resized images (it is, however, slower than ImageCopyResized( )).
- **imagecopyresampled()** copies a rectangular portion of one image to another image, smoothly interpolating pixel values so that, in particular, reducing the size of an image still retains a great deal of clarity.

– In other words, **imagecopyresampled()** will take a rectangular area from src_image of width src_w and height src_h at position (src_x,src_y) and place it in a rectangular area of dst_image of width dst_w and height dst_h at position (dst_x,dst_y).

– If the source and destination coordinates and width and heights differ, appropriate stretching or shrinking of the image fragment will be performed. The coordinates refer to the upper left corner. This function can be used to copy regions within the same image (if dst_image is the same as src_image) but if the regions overlap the results will be unpredictable.

Syntax:

```
imagecopyresampled ( resource$dst_image , resource$src_image , int$dst_x , int$dst_y , int$src_x ,
int$src_y , int$dst_w , int$dst_h , int$src_w , int$src_h )
```

Parameters

dst_image : Destination image link resource.

src_image : Source image link resource.

dst_x: x-coordinate of destination point.

dst_y: y-coordinate of destination point.

src_x: x-coordinate of source point.

src_y: y-coordinate of source point.

dst_w: Destination width.

dst_h: Destination height.

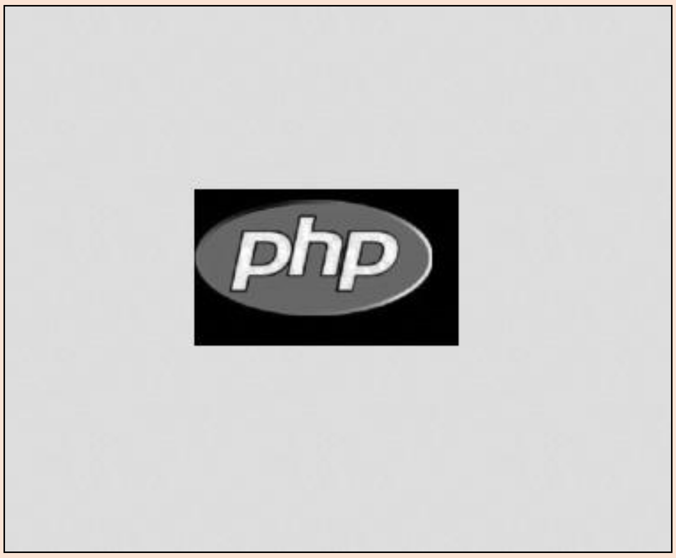
src_w: Source width.

src_h: Source height.

Example :

```
<?php
$src = ImageCreateFromJPEG('php.jpg');
 $width = ImageSx($src);
 $height = ImageSy($src);
 $x = $width/2;
 $y = $height/2;
 $dst = ImageCreateTrueColor($x,$y);
 ImageCopyResampled($dst,$src,0,0,0,0,$x,$y,$width,$height);
 header('Content-Type: image/png');
 ImagePNG($dst);
?>
```

Output :



- **imagecopyresized()** copies a rectangular portion of one image to another image. dst_image is the destination image, src_image is the source image identifier.
- In other words, **imagecopyresized()** will take a rectangular area from src_image of width src_w and height src_h at position (src_x,src_y) and place it in a rectangular area of dst_image of width dst_w and height dst_h at position (dst_x,dst_y).
- If the source and destination coordinates and width and heights differ, appropriate stretching or shrinking of the image fragment will be performed. The coordinates refer to the upper left corner. This function can be used to copy regions within the same image (if dst_image is the same as src_image) but if the regions overlap the results will be unpredictable.

Syntax :

```
imagecopyresized ( resource$dst_image , resource$src_image , int$dst_x , int$dst_y , int$src_x , int$src_y
, int$dst_w , int$dst_h , int$src_w , int$src_h )
```

Parameters:

dst_image: Destination image link resource.

src_image: Source image link resource.

dst_x: x-coordinate of destination point.

dst_y: y-coordinate of destination point.

src_x: x-coordinate of source point.

src_y: y-coordinate of source point.

dst_w: Destination width.

dst_h: Destination height.

src_w: Source width.

src_h: Source height.

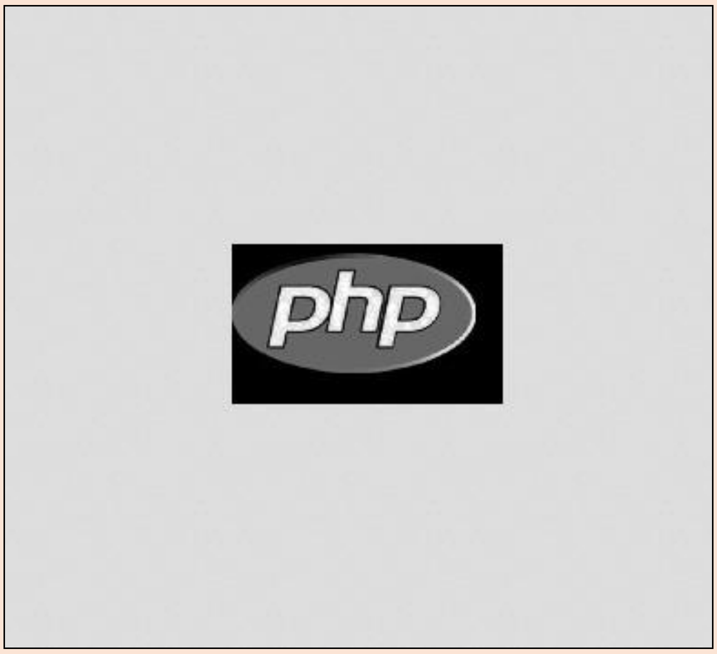
Example :

```
<?php
```

```
$src = ImageCreateFromJPEG('php.jpg');
 $width = ImageSx($src);
 $height = ImageSy($src);
 $x = $width/2;
 $y = $height/2;
 $dst = ImageCreateTrueColor($x,$y);
 imagecopyresized($dst,$src,0,0,0,0,$x,$y,$width,$height);
 header('Content-Type: image/png');
 ImagePNG($dst);

?>
```

Output :



2.6.4 Creation of PDF Document

FPDF is a PHP class which allows to generate PDF files with PHP code. F from FPDF stands for Free: It is free to use and it does not require any API keys. you may use it for any kind of usage and modify it to user needs.

Advantages of FPDF :

- Choice of measure unit, page format and margins
- Allow to set Page header and footer management
- It provides automatic line break, Page break and text justification
- It supports Images in various formats (JPEG, PNG and GIF)

– It allows to setup Colors, Links, TrueType, Type1 and encoding support

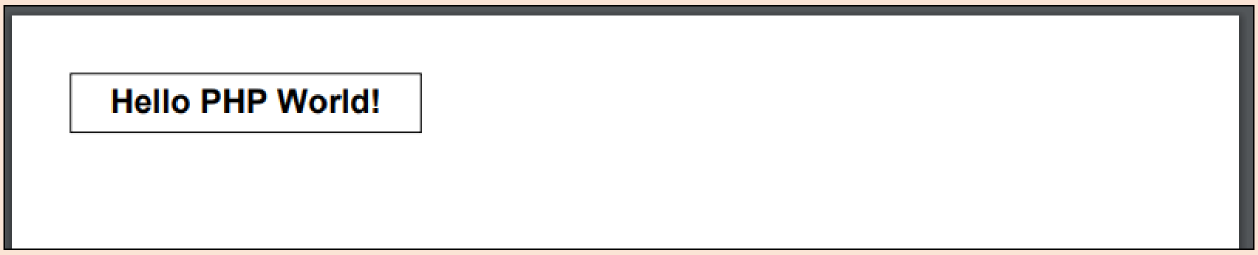
– It allows Page compression

Link to Download latest version of FPDF class: http://www.fpdf.org/en/download.php

Example:

```
<?php
 require('fpdf.php');
 $pdf=new FPDF();
 $pdf->AddPage();
 $pdf->SetFont('Arial','B',16);
 $pdf->Cell(60,10,'Hello PHP World!',1,1,'C');
  $pdf->Output();
?>
```

Output :



Hello PHP World!

The include() and require() function :

Q. What's the difference between include and require? **(4 Marks)**

– We can include the content of a PHP file into another PHP file before the server executes it. Two functions are used to do this :

– **The include() function :** The include() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the **include()** function generates a warning but the script will continue execution.

– **The require() function :** The require() function takes all the text in a specified file and copies it into the file that uses the include function. If there is any problem in loading a file then the **require()** function generates a fatal error and halt the execution of the script.

Syntax : Include 'filename' or require 'filename'

– **Cell parameters:** Prints a cell (rectangular area) with optional borders, background color and character string.

Syntax : cell(float w [,float h, [,String txt [, **mixed** border [, **int** ln [, **string** align [, **boolean** fill [, **mixed** link]]]]]]])

w	Cell width. If 0, the cell extends up to the right margin.h
h	Cell height. Default value: 0.

txt	String to print. Default value: empty string.
border	Indicates if borders must be drawn around the cell. The value can be either a number: 0: no border 1: frame or a string containing some or all of the following characters (in any order): L: left T: top R: right B: bottom Default value: 0.
ln	Indicates where the current position should go after the call. Possible values are: 0: to the right 1: to the beginning of the next line 2: below Putting 1 is equivalent to putting 0 and calling Ln() just after. Default value: 0.
align	Allows to center or align the text. Possible values are: L or empty string: left align (default value) C: center R: right align
fill	Indicates if the cell background must be painted (true) or transparent (false). Default value: false.
link	URL or identifier returned by AddLink().

Programs

1. Write a PHP program to print factorial of number using function (4 Marks)

```
?php
function Factorial($number)
{
    $factorial = 1;
    for ($i = 1; $i <= $number; $i++){
      $factorial = $factorial * $i;
    }
    return $factorial;
}



$number = 5;
$fact = Factorial($number);
echo "Factorial = $fact";
?>
```

Output:

```
Factorial = 120
```

2. Write a PHP program to check whether number is even or odd using function

```
<?php
function check($number){
    if($number % 2 == 0){
        echo "Even";
    }
    else{
        echo "Odd";
    }
}
$number = 20;
check($number)
?>
```

Output :

```
Even
```

3. PHP program to calculate the sum of digits using function

```
<?php

function sum($num) {
    $sum = 0;
    for ($i = 0; $i < strlen($num); $i++){
        $sum += $num[$i];
    }
    return $sum;
}



$num = "234";
echo sum($num);
?>
```

4. PHP program to check whether a number is prime or Not

```
<?php

function primeCheck($number)
{
    if ($number == 1)
    return 0;
    for ($i = 2; $i <= $number/2; $i++)
{
        if ($number % $i == 0)
            return 0;
    }
    return 1;
}

// Driver Code
$number = 13;
$flag = primeCheck($number);
if ($flag == 1)
    echo "Prime";
else
    echo "Not Prime"
```

```
?>
```

Output :

```
Prime
```

Review Questions

Q, 1. Explain any four math function available in PHP.

Q, 2. How will you create arrays in PHP.

Q, 3. Write notes on formatting strings with PHP.

Q, 4. Describe about drawing a new image in PHP.

Q, 5. What is anonymous function.

Q, 6. Write the difference between built in function & user defined function.

Q, 7. Explain any four string manipulating function with example.

Q, 8. Explain exploding & imploding function with example.

Q, 9. How to remove leading and trailing spaces of the strings.

Q, 10. How to traversing array with different ways.

Q, 11. How to search an element from array using built in function.

Q, 12. Write a short note sorting on array?

Q, 13. How to remove any element from array?

Q, 14. How will you find number of elements in array?

Q, 15. List string related function? Explain any two.

Q, 16. List date related functions. Explain any two.

Q, 17. Write a PHP program to create one dimensional array?

❑❑❑