❖ Arrays In PHP

- Array is one kind of variable which can hold more than one value at a time.
- Every element in an array has an index number. The index of array starts from 0.
- So if we have to get any element from array then we can easily get it through index number.
- In PHP arrays are created through array().
  - Example : $languages = array(‘PHP’,’JAVA’,’JAVASCRIPT’);
- We can also create array using square brackets like,
  - Example : $friends = [‘sumant’,’ankit’,’mayur’];
- There are mainly 3 types of array available in php.

- **1) Indexed array**

  - Indexed arrays are the array with numerical index numbers.
  - When we don’t provide any key or when we create an array without keys at that time indexed array is generated.
  - We can access the elements of indexed array using numeric index numbers.
  - We can also remove an element from array using unset() function
  - If we want to print whole array at that time we have to use print_r() function.
  - When we want to count the length of array at that time we can use count() as well.
  - Let’s see an complete example of indexed array.
  - Example

```
<?php

        // create array

  $programming_languages = array('PHP' , 'JAVA' , 'C' , 'C++' , 'C#');


  // print whole array using print_r() function

  print_r($programming_languages);


  // echo single value of array

  echo "<br>element at index 0 is : " . $programming_languages[0];

  echo "<br>element at index 2 is : " . $programming_languages[2] .
"<br>";
```

```
    // add new value to array

    $programming_languages[5] = 'JAVASCRIPT';

    print_r($programming_languages);


    // count the number of elements in array

    echo "<br> length of array is : " . count($programming_languages);


    // unset () is used to remove array element from array.

    unset($programming_languages[4]);

    print_r($programming_languages);


    // short syntax for array creation

    $friends = ['sumant','ankit','mayur'];

?>
```

- **2) Associative array**

  - Generally associative arrays are the arrays with named keys.
  - Unlike indexed array associative arrays can have string keys.
  - We can create associative array just like indexed array but with keys.
  - We can skip keys of some values as well
  - When we have to access some values at that time we have to provide keys in place of index number. Ex : echo $arr['key'];
  - Now let's see the example of associative array.
  - Example :

```
<?php

    // create an associative array

    $founders = array('PHP'=>'Rasmus Lerdorf' , 'JAVA'=>'James Goling',

    'C'=>'Dennis Ritchie', 'C++'=>'Bjarne Stroustrup',

    'JAVASCRIPT'=>'Brendan Eich');
```

```
   // print whole array

   print_r($founders);

   // echo single value from associative array

   echo "<br>Founder of PHP is : " . $founders['PHP'];

   echo "<br>Founder of JAVASCRIPT is : " . $founders['JAVASCRIPT'];


   // add another key value pait into associative array

   $founders['PYTHON'] = 'Guido Van Russom';

   print_r($founders);


   // count the length of associative array

   echo "<br>The length of array is : " . count($founders);

?>
```

- **3) Multi-dimensional array**

    - In multi-dimensional array we have more than one array.
    - Simply the multi-dimensional array is an array which contains one or more arrays.
    - Multi-dimensional array can be two dimensional , three dimensional and so on.
    - For two dimensional array we need two indices to access it’s element , for three dimensional array we need three indices to access it’s element and so on.
    - The array inside one array can be indexed array or associative array.
    - We can also give a key to whole array as well.
    - Let’s see an example of multi-dimensional array.
    - Example :

```
<?php

   // create a multi-dimensional array

   $frameworks = array(
   'PHP'=>array('Laravel','Symfony','CodeIgniter','CakePHP','Phalcon') ,
   'JAVA'=>array('Spring','Hibernet','Struts','Blade','Grails') ,
   'JAVASCRIPT'=>array('Angular','Express','React','Vue.js','Ember.js') ,
   'PYTHON'=>array('CubicWeb','Django','Glotto','Web2py'));
   // print array
   print_r($frameworks);

   //access single element from array
   echo "<br><br>PHP frameworks are : ";
   print_r($frameworks['PHP']);

   //access single elements as well as element from inner array
   echo "<br>" . $frameworks['PHP'][0];

   // add value or element in array
   $frameworks['PHP'][5] = 'Zend';
   echo "<br>" . $frameworks['PHP'][5];

   foreach($frameworks as $language => $framework_list) {
       echo '<br><br><strong>' . $language . '</strong><br>';
       foreach($framework_list as $framework) {
           echo '<br>' . $framework;
       }
   }
?>
```

❖ Sorting arrays

- **1) Sort() :** sort arrays in ascending order
- **2) rsort() :** sort arrays in descending order
- **3) assort() :** sort associative arrays in ascending order, according to the value
- **4) ksort() :** sort associative arrays in ascending order, according to the key
- **5) arsort() :** sort associative arrays in descending order, according to the value
- **6) krsort() :** sort associative arrays in descending order, according to the key

❖ Foreach loop

- The foreach loop - Loops through a block of code for each element in an array.
- The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.
- Example :

```
<?php

        // created names array

        $names = array('rohit','sumant','ankit','mayur','naimish','jay');


        // echo each name one by one using foreach loop

        foreach($names as $name) {

           echo $name . '<br>';

        }


        $ages = array('rohit'=>22 , 'sumant'=>23 , 'ankit'=>29 , 'mayur'=>35
      , 'naimish'=>40 , 'jay'=>36);


        echo '<br><br><b>List of all names with their ages</b><br><br>';

        foreach($ages as $name=>$age) {

           echo $name . ' is ' . $age . ' years old. <br>';

        }

?
```

❖ Include , require , include_once , require_once

▪ 1) include

- When we want to use same file or same code in multiple files then we can use that file into multiple files by writing just one line.
- We just have to use include statement to use other file into current file.
- Include will take all the code from that file and put it into the current file.
- We can also use variables of included file also.
- Example :

```
<?php

        include 'php_starter.php';

        $sub_heading = 'include';

        echo '<h2>'.$sub_heading.'</h2>';

        echo 'this way we can include any file in PHP';

?>
```

- **Note: when we use include to get any file and that file is no available then it will give a warning and rest of the code will be executed.**

▪ 2) require

- Require is same as include
- Include and require both work same
- But there is just a small difference between them
- When we use require to get a file and that file is not available then it will generate an error while include will give warning
- Example :

```
<?php

        require 'php_starter.php';

        $sub_heading = 'require';

        echo '<h2>'.$sub_heading.'</h2>';

        echo 'this way we can require any file in PHP';

?>
```

  - **Note: when we use require to get any file and that file is no available then it will give an error and rest of the code will not be executed**

- **3) include_once**

  - Include_once is same as include but it ensures that the file will included only once.
  - When we use include_once , at that time it will check the previous code to ensure that the file included only once.
  - Example :

```
<?php

        include 'php_starter.php';

        include_once('php_starter.php');

        $sub_heading = 'include_once()';

        echo '<h2>'.$sub_heading.'</h2>';

        echo 'this way we can use include_once() to include any file in PHP';

?>
```

- **4) require_once**

  - Require_once is same as include_once.
  - It also ensures that the file required only once.
  - Example :

```
<?php

        require 'php_starter.php';

        require_once('php_starter.php');

        $sub_heading = 'require_once()';

        echo '<h2>'.$sub_heading.'</h2>';

        echo 'this way we can use require_once() to require any file in PHP';

?>
```

❖ String Functions

▪ 1) strlen( )

- Strlen() function is used to get the length of string.
- It will return the length of string in number.
- It takes one required argument which is string itself.

- Example :

```
<?php
          include 'php_starter.php';
          $sub_heading = 'strlen()';
          echo '<h2>'.$sub_heading.'</h2>';

          $str = 'hello everyone, good afternoon';
          echo 'string lenght is : '.strlen($str);
?>
```

▪ 2) strtoupper( )

- Strtoupper() is used to convert the string into upper case
- Whenever we want to convert the string to upper case at that time we can use strtoupper()
- It takes one required argument which is string itself.
- Example :

```
<?php
  include 'php_starter.php';
  $sub_heading = 'strtoupper()';
  echo '<h2>'.$sub_heading.'</h2>';
  $str = 'My name is rohit lalwani';
  echo strtoupper($str);   ?>
```

- **3) strtolower( )**

  - Strtolower() is used to convert the string into lower case
  - Whenever we want to convert the string into lower case at that time we can sue strtolower()
  - It takes one required argument which is string itself
  - Example :

```
<?php
  include 'php_starter.php';
  $sub_heading = 'strtoupper()';
  echo '<h2>'.$sub_heading.'</h2>';
  $str = 'MY NAME IS ROHIT LALWANI';
  echo strtolower($str);
?>
```

- **4) strpos( )**

  - Strops() is used when we want to get position of a word in the string.
  - It will return the numeric position of that word
  - It takes two required arguments , first is string and second is word whose position we want to find.
  - Example :

```
<?php
  include 'php_starter.php';
  $sub_heading = 'stringpos()';
  echo '<h2>'.$sub_heading.'</h2>';
  $str = 'we say good morning , good afternoon , good evening and good night as greeting when we meet someone';
  $find = 'good';
  echo 'the first position of <b>good</b> is at : ' . strpos($str , $find);
  // let's find all position
  echo '<br><br>All occurances of good';
  $i=1;
```

```
 $offset = 0;

 $findLength = strlen($find);

 while($position = strpos($str , $find , $offset)) {

    echo '<br>'.$i.'. <b>good</b> found at : '.$position;

    $offset = $position + $findLength;

    $i++;

 }

?>
```

- **5) substr_replace( )**

  - Substr_replace() is used to replace a particular word in string.
  - First it will get to that word then it will change that particular word in string.
  - If there are so many same words and we want to change only one occurance of that word at that time we can use substr_replace()
  - It takes four required arguments, first is string , second is the new word that we want to place , third is the position of he old word and fourth is the length of old word that we want to change.
  - Example :

```
<?php

 include 'php_starter.php';

 $sub_heading = 'substr_replace()';

 echo '<h2>'.$sub_heading.'</h2>';

 $str = "hello, this is rohit lalwani.";

 echo substr_replace($str , 'sanket' , 15 , 5 );

?>
```

- **6) str_replace( )**

  - Str_replace() is used when we want to change all the occurrences of a particular word.
  - It will change all the occurrences of a given word.

- If we want to change more than one word then we can also pass an array of words too.
- Example :

```
<?php
   include 'php_starter.php';
   $sub_heading = 'str_replace()';

   echo '<h2>'.$sub_heading.'</h2>';
   $str = 'we say nice morning , nice afternoon , nice evening and nice night as greeting when we meet someone';
   $new_str = str_replace('nice' , 'good' , $str);
   echo '<br>'.$new_str;

   // replace multiple different words
   $odl_values = array('we', 'nice');
   $new_values = array('I','good');
   $new_string = str_replace($odl_values , $new_values , $str);
   echo '<br>'.$new_string;
?>
```