***Array functions (15 example)***

***In vs code***

|  |  |  |
| --- | --- | --- |
| ***Pass by value*** | | ***Pass by reference*** |
| * By default, PHP functions use pass by value. When you pass a variable to a function in PHP, a copy of the value of that variable is created and passed to the function. Any modifications made to the parameter within the function do not affect the original variable outside the function.   function by\_value($x)  {  $x = $x + 1;  echo "Inside function: $x\n";  }  $num = 5;  echo "Before function call: $num\n";  by\_value($num);  echo "After function call: $num\n";   * Output   Before function call: 5  Inside function: 6  After function call: 5  **$num** was incremented inside **the by\_value()** function, its value remains unchanged outside the function because it was passed by value. | | * In PHP, you can explicitly pass arguments by reference using the & symbol in the function definition. This allows the function to modify the original variable.   function by\_reference(&$x)  {  $x = $x + 1;  echo "Inside function: $x\n";  }  $num = 5;  echo "Before function call: $num\n";  by\_reference ($num);  echo "After function call: $num\n";   * Output   Before function call: 5  Inside function: 6  After function call: 6  **$num** by reference (**&$x**), the value of **$num** is modified directly inside the **by­\_reference()** function, and this modification persists outside the function as well. |
| ***Array\_map()*** | ***Array\_filter()*** | ***Array\_reduce()*** |
| * The **array\_map()** function in PHP applies a callback function to each element of an array and returns a new array containing the results.   $numbers = [1, 2, 3, 4, 5];  $doubled=array\_map(function($num)  { return $num \* 2;}, $numbers);  print\_r($doubled); | * function in PHP filters elements of an array using a callback function and returns a new array with only the elements that satisfy the callback condition.   ***$numbers = [1, 2, 3, 4, 5];***  ***$evens = array\_filter($numbers, function($num) {***  ***return $num % 2 === 0;});***  ***print\_r($evens);*** | * function in PHP iteratively reduces an array to a single value using a callback function.   $numbers = [1, 2, 3, 4, 5];  $sum = array\_reduce  ($numbers, function($accumulator, $currentValue) {  return $accumulator + $currentValue;}, 0);  echo $sum; |