**Task#04:**

Yes, there are other alternatives apart from the default argument hack to implement function overloading in PHP,there can be use four different methods for function overloading.

1. **Native Functions:**
2. **func\_num\_args** — Returns the number of arguments passed to the function.

Example: intfunc\_num\_args ( void )

Gets the number of arguments passed to the function.

This function may be used in conjunction with func\_get\_arg() and func\_get\_args() to allow user-defined functions to accept variable-length argument lists.

1. **func\_get\_args** — Returns an array comprising a function's argument list. array

Example: func\_get\_args ( void )

Gets an array of the function's argument list.

This function may be used in conjunction with func\_get\_arg() and func\_num\_args() to allow user-defined functions to accept variable-length argument lists.

**Example:**

<?php

function Arithmetic(){

if(func\_num\_args()==0){

echo "No Variables entered!!. Please enter atleast two numbers"."<br>"."<br>";

}

else if(func\_num\_args()==1){

echo "One Variable entered!!. Please enter atleast two numbers"."<br>"."<br>";

}

else if(func\_num\_args()==2){

echo "<b>Two Variables</b>"."<br>";

$arr=func\_get\_args();

$num1= $arr[0];

$num2=$arr[1];

$Addition=$num1+$num2;

$Subtraction=$num1-$num2;

$Multiplication=$num1\*$num2;

$Division=$num1/$num2;

echo "Addition of $num1 and $num2 is: ".$Addition."<br>";

echo "Subtraction of $num1 and $num2 is: ".$Subtraction."<br>";

echo "Multiplication of $num1 and $num2 is: ".$Multiplication."<br>";

echo "Division of $num1 and $num2 is: ".$Division."<br>"."<br>";

}

else if(func\_num\_args()==3){

echo "<b>Three Variables</b>"."<br>";

$arr=func\_get\_args();

$num1= $arr[0];

$num2=$arr[1];

$num3=$arr[2];

$Addition=$num1+$num2+$num3;

$Subtraction=$num1-$num2-$num3;

$Multiplication=$num1\*$num2\*$num3;

$Division=$num1/$num2/$num3;

echo "Addition of $num1, $num2 and $num3 is: ".$Addition."<br>";

echo "Subtraction of $num1, $num2 and $num3 is: ".$Subtraction."<br>";

echo "Multiplication of $num1, $num2 and $num3 is: ".$Multiplication."<br>";

echo "Division of $num1, $num2 and $num3 is: ".$Division."<br>";

}

}

Arithmetic();

Arithmetic(4);

Arithmetic(4,4);

Arithmetic(4,4,4);

?>

1. **Passing an Array:**

We prefer this method because using arrays in PHP is a frequent activity; therefore, all programmers are familiar with arrays and their behavior.

**Example:**

Pass an array and place the variable arguments inside the array:

// find the "average" of a group of numbers

function mean($numbers) {

// initialize to avoid warnings

$sum = 0;

// the number of elements in the array

$size = count($numbers);

// iterate through the array and add up the numbers

for ($i = 0; $i< $size; $i++) {

$sum += $numbers[$i];

}

// divide by the amount of numbers

$average = $sum / $size;

// return average

return $average;

}

$mean = mean(array(96, 93, 97));

1. **Splat Operator:**

We have a couple of new features coming in to PHP 5.6 with names that sound much less exciting than the features they actually represent: "variadic functions" sound positively academic, and "argument unpacking" isn't exactly catchy. However they both use a new operator in PHP which looks like an elipsis (three dots ...) and is referred to as either the splat operator or the scatter operator.

**Example:**

function concatenate($transform, ...$strings) {

$string = '';

foreach($strings as $piece) {

$string .= $piece;

}

return($transform($string));

}

echo concatenate("strtoupper", "I'd ", "like ",

4 + 2, " apples");

The parameters list in the function declaration has the ... operator in it, and it basically means " ... and everything else should go into $strings". You can pass 2 or more arguments into this function and the second and subsequent ones will be added to the $strings array, ready to be used.

1. **Magic Function:**

To implement overloading in php we will take help of magic method \_\_call. Magic method \_\_call invoked when method called by class object is not available in class. So here we will not create method exactly and will take help of \_\_call method. Now call method will provide us 2 argument, 1st name of the method called and parameter of the function. Now with the help of either switch , case or if else we will implement overloading in php. Following is very simple example of overloading in php.

**Example:**

<?php

class Shape {

const PI = 3.142 ;

function \_\_call($name,$arg){

if($name == 'area')

switch(count($arg)){

case 0 : return 0 ;

case 1 : return self::PI \* $arg[0] ;

case 2 : return $arg[0] \* $arg[1];}}}

$circle = new Shape();

echo $circle->area(5);

$rect = new Shape();

echo $rect->area(5,10);