Web Programming

#5 PHP - Functions

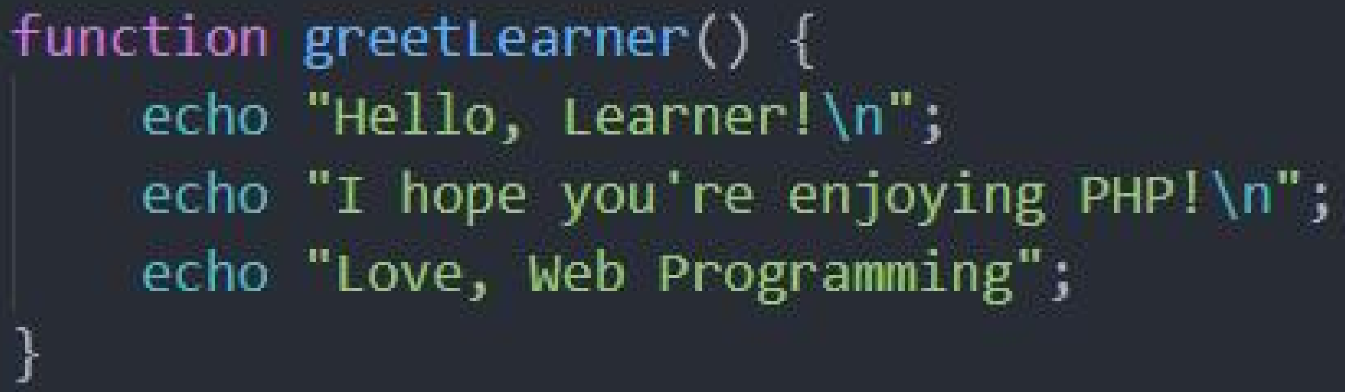
Herman Kabetta, M.T.

hermanka.github.io

What is Function?

- A function is a set of instructions we package as a unit, often with a name, so that we can reuse it.
- Function name are **CASE-INSENSITIVE**, though it is usually good form to call functions in the same case as they appear in their declaration.

Defining Functions

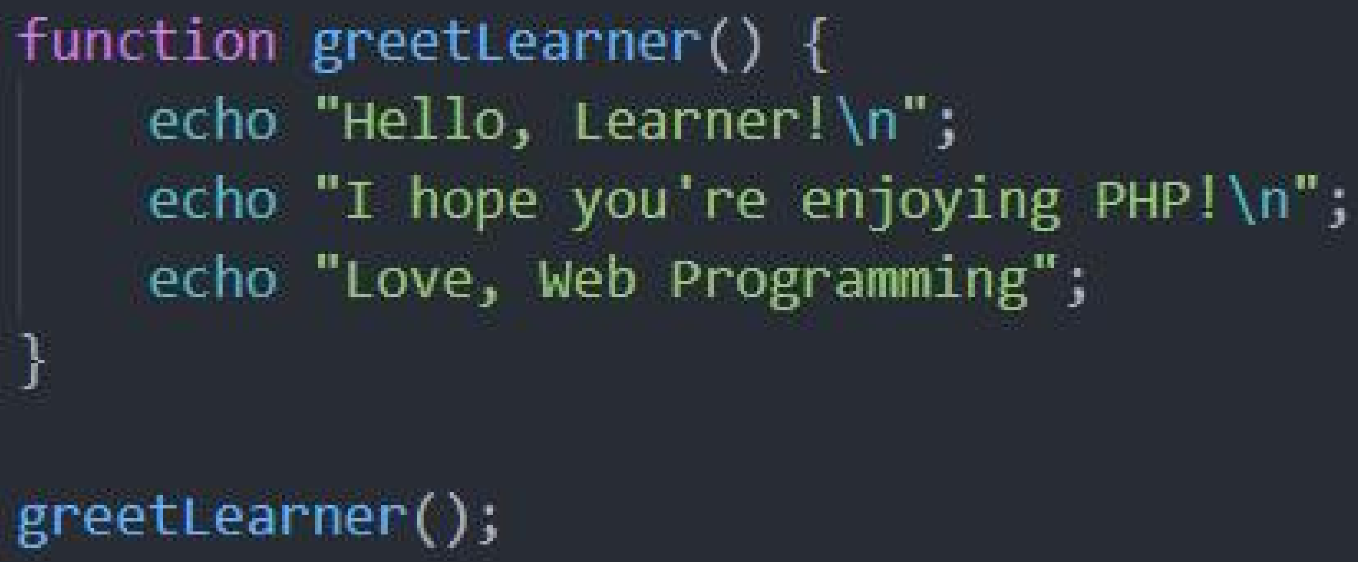


```
function greetLearner() {
    echo "Hello, Learner!\n";
    echo "I hope you're enjoying PHP!\n";
    echo "Love, Web Programming";
}
```

<u>The PSR :</u>

Use camelCase for our function names to easily tell the difference between variables.

Invoking Functions



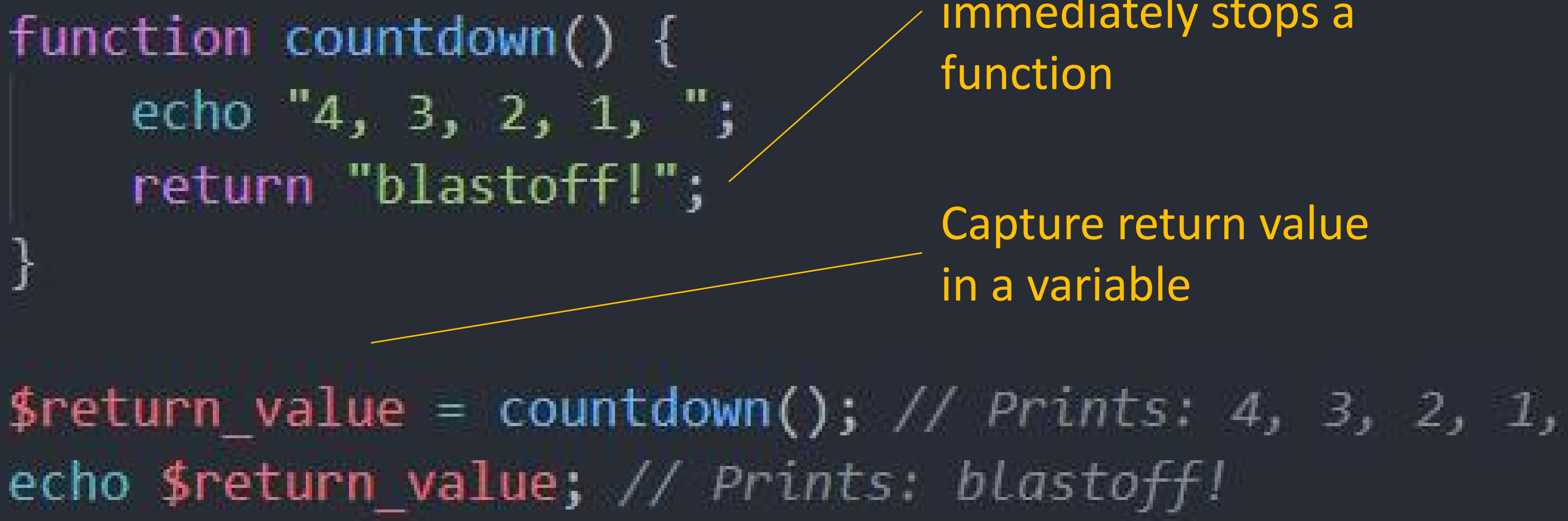
```php
function greetLearner() {
    echo "Hello, Learner!\n";
    echo "I hope you're enjoying PHP!\n";
    echo "Love, Web Programming";
}

greetLearner();
```

Defining & Invoking Functions - Exercise

- Define the function `inflateEgo()` which prints a compliment.
- Invoke your function!
- Edit your code so your function is invoked twice!

Return Statements

As we build more complicated functions, we'll often be using them to process data. In order for the data to be useful, functions have the ability to return a value in addition to performing instructions.



```php
function countdown() {
    echo "4, 3, 2, 1, ";
    return "blastoff!";
}

$return_value = countdown(); // Prints: 4, 3, 2, 1,
echo $return_value; // Prints: blastoff!
```

The return keyword immediately stops a function

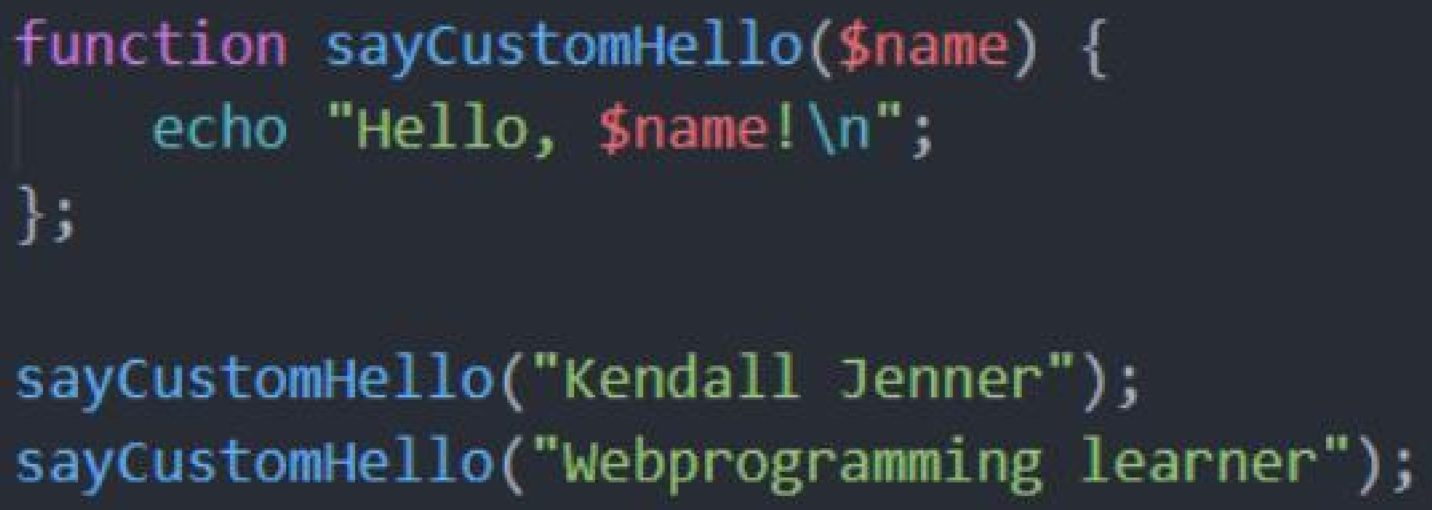
Capture return value in a variable

Return Statements - Exercise

1. Write a function `printStringReturnNumber()` which prints a string and returns a number value.
2. Capture your function’s return value in a variable named `$my_num`.
3. Use echo to print your `$my_num` variable.

Parameters

A parameter is a variable which serves as a placeholder throughout the function’s code block.



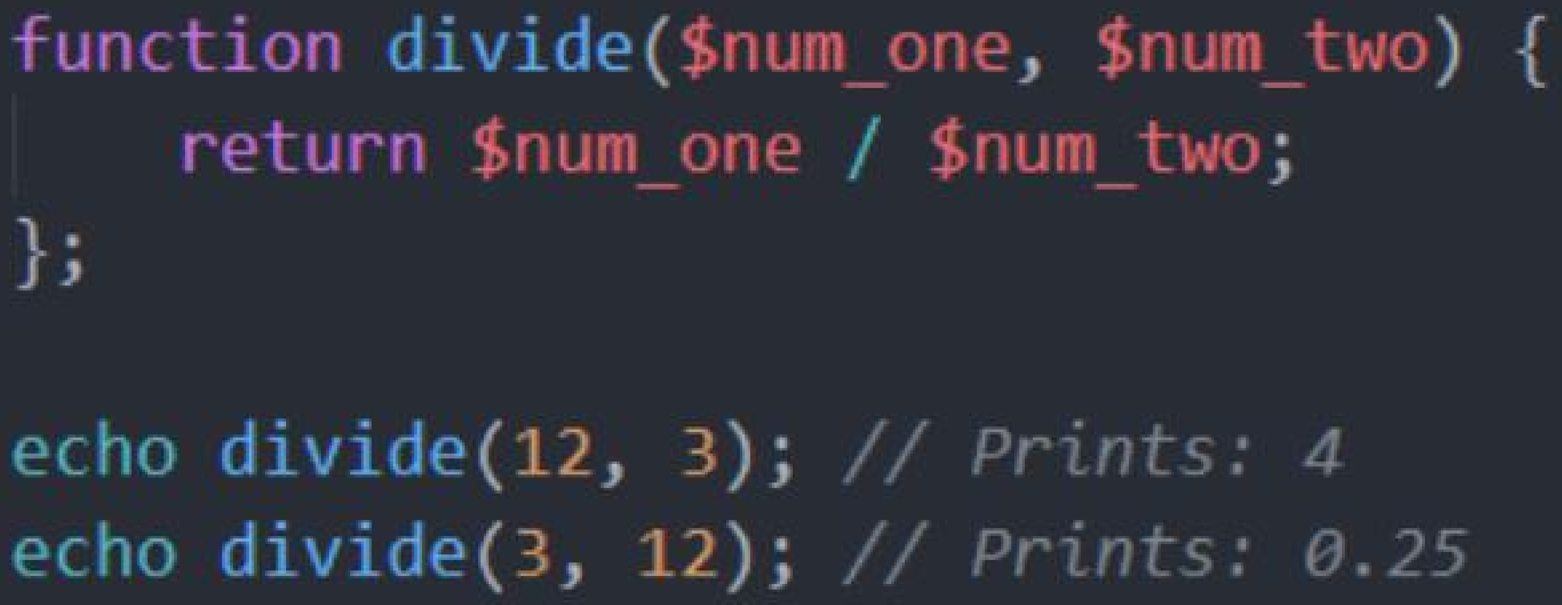
```
function sayCustomHello($name) {
    echo "Hello, $name!\n";
};

sayCustomHello("Kendall Jenner");
sayCustomHello("Webprogramming learner");
```

Parameters - Exercise

1. Write a function `increaseEnthusiasm()` which takes in a string parameter and returns that string appended with an exclamation mark.
2. Use echo to print the result of invoking your `increaseEnthusiasm()` function with a string of your choice.
3. Write a function `repeatThreeTimes()` which takes in a string parameter and returns that string repeated three times
4. Use echo to print the result of invoking your `repeatThreeTimes()` function with a string of your choice.
5. Use echo to print the result of invoking your `increaseEnthusiasm()` with the result of invoking `repeatThreeTimes()` as the paramater passed into `increaseEnthusiasm()`. You can choose any string you like for the argument to `repeatThreeTimes()`.

Multiple Parameters

We can also define functions with multiple parameters.



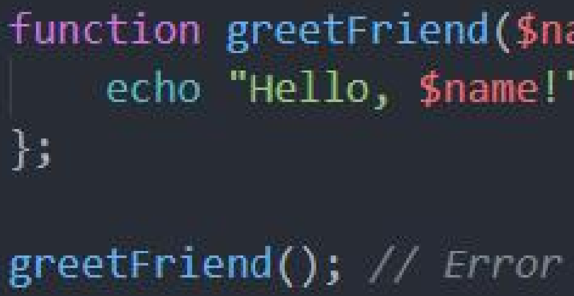
```php
function divide($num_one, $num_two) {
    return $num_one / $num_two;
};

echo divide(12, 3); // Prints: 4
echo divide(3, 12); // Prints: 0.25
```

Multiple Parameters - Exercise

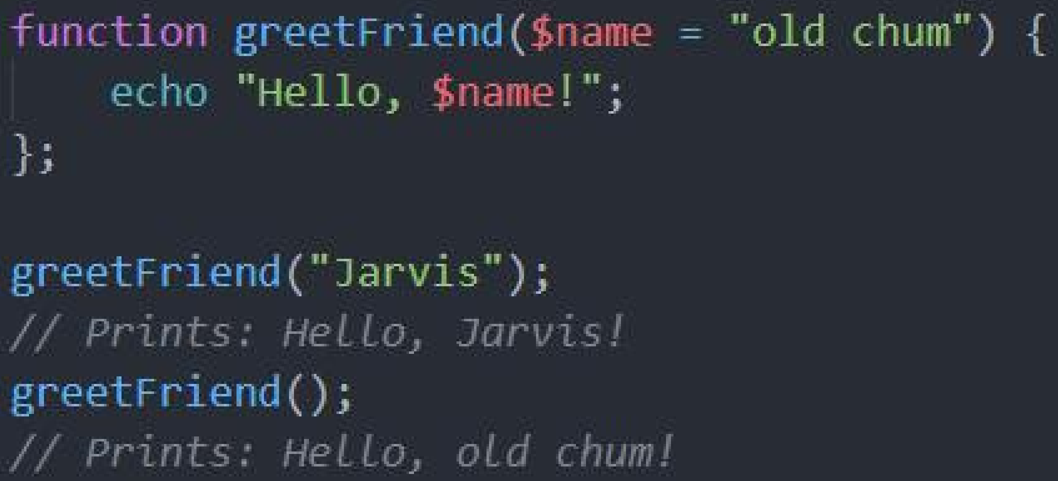
1. Write a function `calculateArea()` that takes in two number arguments—representing the height and width of a rectangle—and returns the area of that rectangle.
2. Use `echo` to print the result of invoking your `calculateArea()` function with two number arguments.
3. Write a function `calculateVolume()` that takes in three number arguments—representing the height, width, and depth of a box—and returns the volume of that box.
4. Use `echo` to print the result of invoking your `calculateVolume()` function with three number arguments.

Default Parameters

How to invoke parametered function without an argument that not cause an error.



```php
function greetFriend($name) {
    echo "Hello, $name!";
};

greetFriend(); // Error
```



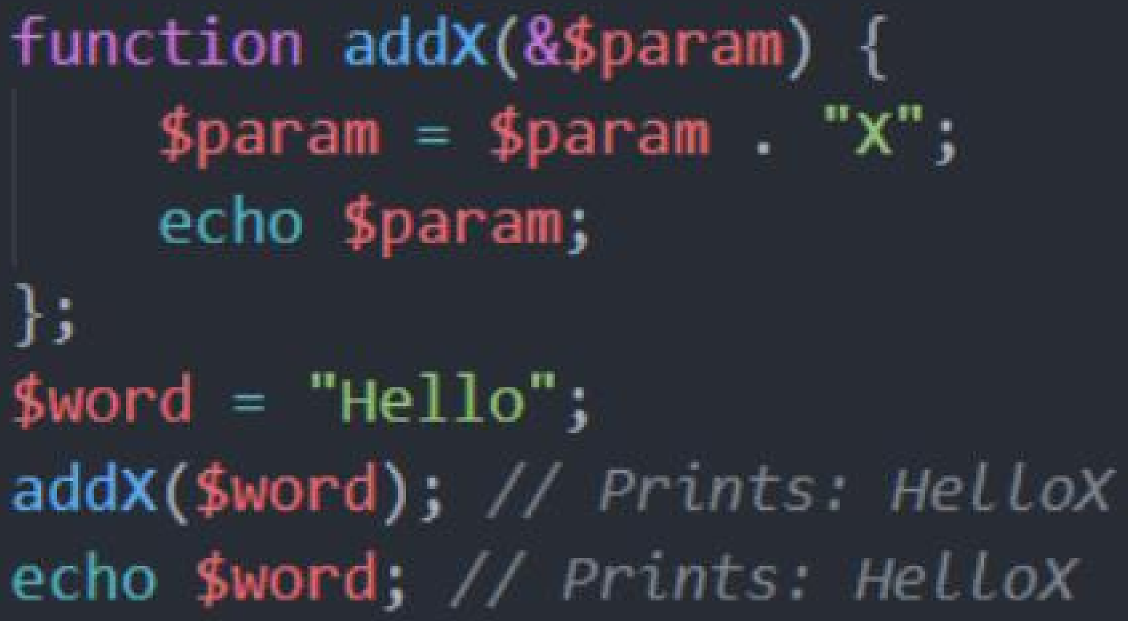
```php
function greetFriend($name = "old chum") {
    echo "Hello, $name!";
};

greetFriend("Jarvis");
// Prints: Hello, Jarvis!
greetFriend();
// Prints: Hello, old chum!
```

Default Parameters - Exercise

1. Write a function `calculateTip()` which takes a number representing the total cost of a meal as its first argument. It should also take a second, optional argument—an integer representing the percent tip desired (eg. 25 will indicate a 25% tip should be calculated). If no second argument is passed in, the function should default to a 20% tip. The function should return the new total—the previous total plus the calculated tip.
2. Use echo to test your function with one and two arguments to make sure it works as expected in each case.

Passing by Reference

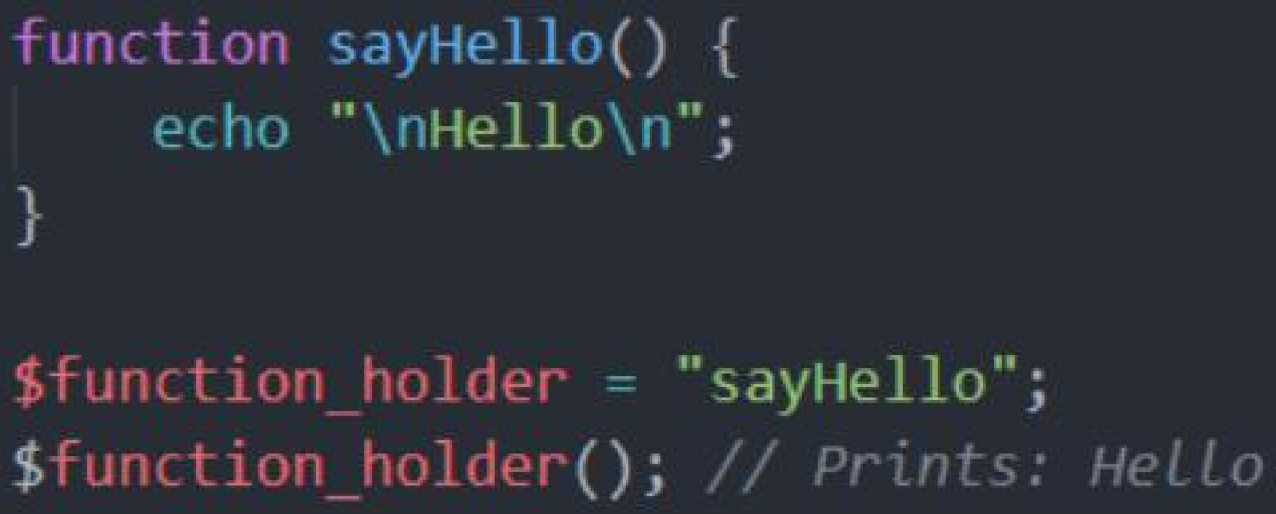
If we do want to make permanent changes to a variable within a function, we can prepend the parameter name with the reference sign (&).



```php
function addX(&$param) {
    $param = $param . "x";
    echo $param;
};
$word = "Hello";
addX($word); // Prints: HelloX
echo $word; // Prints: HelloX
```

Dynamic Function Calls

It is possible to assign function names as strings to variables and then treat these variables exactly as you would the function name itself.



```php
function sayHello() {
    echo "\nHello\n";
}

$function_holder = "sayHello";
$function_holder(); // Prints: Hello
```