Funktioner

Tänk på att inte fastna för länge på en övning, testa en annan övning, det är inte säkert att den är svårare för att den kommer senare i listan. Kör på tills du känner att det sitter.

Det är även ok att använda PHPs inbyggda funktioner, det är inte fusk bara man lär sig något. 🤓

- String functions
  - `substr`
  - `strlen`
  - `explode`
  - `str_replace`
  - `str_repeat`
- Array functions
  - `array_push`
  - `array_pop`
  - `array_sum`
  - `array_reverse`

En funktion ser oftast ut på följande sätt. Med `return` menar vi att ett värde ska skickas tillbaka till oss när vi kallar på funktionen **men inte skrivas ut**

```
function function_name($parameter){
    return $parameter;
}
```

Om vi vill direkt skriva ut ett värde måste vi använda `echo`:

```
function function_name($parameter){
    echo $parameter;
}
```

Det som är innanför paranteserna är en **parametrar**. De parametrar som skickas med kan användas inom funktionen utan att deklareras och vi kan använda parametern inom vår funktion. Funktionen är dock bara HUR programmet ska funka och funktionen gör ingenting förrän vi kallar på funktionen och skickar med värden. Detta betyder att vi kan skapa kod som KAN användas men används bara när vi bestämmer oss för att kalla på koden.

Om vi ska skicka med värden skickar vi med dem som **argument**. De argument vi skickar med till funktionen blir funktionens parametrar. Om vi ska kalla på en funktion samt skicka med ett argument gör vi på följande sätt:

```
function function_name($parameter){
    return $parameter;
}

$my_name =  function_name('Jesper');
echo $my_name;
//Output: Jesper
```

Övningar

1 .

- Skapa en funktion: `multiply`, som tar in 2 *parametrar*. I funktionen, multiplicera dessa nummer med varandra och `echo` resultatet av multiplikationen. Du får använda vilka namn du vill på parametrarna.
- Kalla på din funktion multiply med siffrorna 8,4 som *argument*
- Om du gjort rätt ska du få siffran 32.

2 .

- Skapa en funktion som heter `calculate` och som istället tar 3 parametrar och sedan multiplicerar de två första parametrarna med varandra och delar värdet med den tredje parametern, alltså: `param1 * param2 / param3`. Funktionen ska sedan `echo` ut svaret.
- Kalla sedan på din funktion med valfria värden.

3 .

- Skriv en funktion som heter `highest_number` som tar två tal som parametrar.
- Funktionen ska sedan jämföra vilket av talet som är störst och `echo` det största talet.
- Kalla på din funktion med två valfria värden.

4 . Koden ovan är dock inte optimal. För det mesta vill vi att funktioner endast returnerar värden. En funktions uppgift ska vara att returnera värden så att vi kan återanvända det värdet och sedan forsätta göra mer beräkningar på det. När vi har massa `echo` på det här sättet blir koden svårare att återanvända så vi vill nu skriva om våra funktioner så att de endast returnerar värden.

Refaktorera dina funktioner som du tidigare skapade:

- De två första funktionerna (multiply och calculate) ska **returnera** det slutgiltiga värdet av beräkningarna. Sedan måste du spara värdena för att sedan använda `echo` på dem.
- Den tredje funktionen `highest_number` ska **returnera** det högsta värdet av de två värden som skickas in som parametrar. Om dock värdet inte är ett nummer ska funktionen returnera `false` och om de båda parametrarna är samma värde ska funktionen returnera *"Samma värde"*

Spara värdena som returneras på följande sätt:

```
$multiply_answer = multiply(5,5);
echo $multiply_answer;
```

5.

Eftersom **alla** gillar katter så ska vi refaktorera vår Mjau Machine 😺

Du ska skriva en funktion som ska ha samma funktionalitet som vår ursprungliga Mjau Machine från tidigare övningar men nu i form av en funktion.

6 .

Använd while-loopen du skapade från en av de tidigare uppgiftern. Nummer 5 i `03_loops`

Denna loop ska du nu göra om till en funktion som tar 2 parametrar.

- Första parametern ska vara siffran som loppen ska räkna ner ifrån, alltså hur många värden funktionen ska gå igenom.
- Andra parametern ska vara om funktionen ska skriva ut jämna eller ojämna värden.

7 .

Skriv en funktion som tar 2 parametrar. Parametrarna ska vara två heltal. Funktionen ska multiplicera heltalen utan att använda `*`-operatorn.

8 . Skriv en funktion som tar in en parameter. Parametern ska vara en string. Funktionen ska sedan returnera strängens längd på detta sätt:

```
"Strängen du matade in är 14 tecken lång"
```

9 .

Skapa en funktion som heter `convert_string`, funktionen ska ta **två parametrar**. Den första parametern ska vara en sträng som skickas med, typ: *"Goodbye World"*. Den andra parametern som skickas med ska bestämma om strängen ska konverteras till bara stora bokstäver eller bara små bokstäver. Till detta kan man använda hjälpfunktionen: `strtolower($string)` och `strtoupper($string)`

10 . Skapa en funktion som tar en parameter, argumentet som skickas in ska vara en sträng. Funktionen ska sedan returnera det sista tecknet i strängen som skickas in.

11 . Skriv en funktion med namnet make_paragraph som skriver ut en sträng som HTML-elementet `<p>`. Exempel: *"hej"*_ ska skrivas ut som "`<p>hej</p>`". Funktionen ska ha en parameter, som är strängen som ska skrivas ut, och den ska inte returnera något bara `echo` a ut.

12 . Funktionen `make_paragraph()` är lite begränsad. Tänk om vi vill göra `<h1>`-taggar? Eller `<h2>`, `<h3>` osv. Skriv en ny funktion med namnet make_heading. Funktionen behöver veta strängen som ska skrivas ut och vilken heading det ska vara. Den behöver alltså två parametrar.

13 . Nu har vi två funktioner som vi kan använda för att skapa HTML-paragrafer och headings. Men det blir väldigt många funktioner om vi ska ha en funktion för varje möjligt HTML-element. Vi behöver en funktion som kan göra flera sorters element. Skriv en funktion make_tag som kan göra alla sorters HTML-element.

14 . Förbättra `make_tag` så att man kan ange inline styles också. (Eller href för länkar) Exempel: `<p style="color: hotpink;">Exempeltext</p>`

15 . Skriv en funktion som gör om alla nyrader i en sträng till `<br>`-element. Funktionen ska ta strängen som parameter och returnera en ny sträng. En nyrad i PHP skrivs `'\n'`.

16 . Skriv en funktion som returnerar en array med slumpade tal. Använd `mt_rand()` för att göra slumptal. Hur många parametrar behöver funktionen?

17 . Skriv en funktion som gör om en array till en lista i HTML. Använd funktionen make_tag. Exempel: `make_list( [1, 2] )` → `"<ul> <li>1</li> <li>2</li> </ul>"`

18 . Skriv en funktion med namnet `capitalize` som gör om ett användarnamn till stora bokstäver.

19 . Förbättra funktionen så att den bara gör första bokstaven stor. Tips är att använda `substr()`

20 . Skriv en funktion som genererar en random färg.

21.

Skriv en funktion som avrundar en float till närmaste heltal med hjälp av typecast.

```
Exempel: round(3.9) → 3, round(5.5) → 6.
```

22 . Skriv en funktion som gör om ett decimaltal till en sträng. Strängen ska använda decimalkomma i stället för decimalpunkt. Exempel: float_to_string(75.5) → "75,5".

23 . Skriv en funktion som räknar ut summan av alla tal i en array. Skriv en annan som räknar ut medelvärdet.

24 . Skriv en funktion som tar en sträng som motsvarar en veckodag som parameter och returnerar en siffra. Om strängen är "måndag" ska funktionen returnera 1, "tisdag" ska bli 2 och "söndag" ska bli 7. Funktionen ska fungera oavsett om veckodagen står med små eller stora bokstäver.

Lösningsförslag

1.

```php
/*
 * x, y are parameters
 */
function multiply($x,$y){
   echo $x * $y;
}

//Use case, when called, it will echo the result of the arguments
multiply(5,5);
```

2.

```php
function calculate($x,$y,$z){
    echo ($x*$y) / $z;
}

//use case
calculate(5,10,2);
```

3.

```php
function highest_number($x,$y){
    /* If x is higher than y, return x */
    if( $x > $y){
        echo $x;
    }
    /* or else y is higher */
    else{
        echo $y;
    }
}

//Use case
highestNumber(5,4);
```

4.

```php
function multiply($x,$y){
   return $x * $y;
}

//Use case, we must first store the value
$answer = multiply(5,5);
echo $answer;
//or we can print it directly
//echo multiply(5,5);

function calculate($x,$y,$z){
    return ($x*$y) / $z;
}

//use case
$answer = calculate(5,10,2);
echo $answer
```

5.

```
function mjau_machine($number_of_mjau){
    if($number_of_mjau == 0){
        return '😿';
    } else {
        $all_the_mjaus = '';
        for($i = 0; $i <= $number_of_mjau; $i++){
            $all_the_mjaus = $all_the_mjaus . 'mjau ';
     }
    return $all_the_mjaus;
    }
}

echo $mjau_machine(10);
```

6.

```
function even_or_odd($number, $is_even){
    $numbers = '';
    while($number < 10){
        //If it's a even number, we wont get rest value, we will have 0
        if( $number % 2 == $is_even){
            $numbers = $numbers . $number;
        }
        $number++;
    }
    return $numbers;
}

even_or_odd(10, 0)
```

7.

```
function calcy($x,$y){
    $sum = 0;
    /* x * y är samma sak som att addera x, y antal gånger. Vi kör alltså loopen
    i det här fallet y antal gånger, alltså 4 i vårat fall.
    Värdet som läggs till i sum är 5 i detta fall, alltså:
    5 + 5 + 5 + 5 */
    for ($i = 0; $i <= $y ; $i++){
        $sum = $sum + $x;
    }
    return $sum;
}

$calcyAnswer = calcy(5,4);
echo $calcyAnswer;
```

8.

```
function string_checker($string_to_check){
    //Use the built in function `strlen`
    $string_length = strlen($string_to_check);
    return "Strängen du matade in är " + $string_length + " tecken lång";
}

//Vi skickar in en valfri sträng som argument
echo string_checker('Huburru Hubburru');
```

9.

```
function string_converter($string_to_convert, $up_or_down){
    if($up_or_down == 1){
        return strtoupper($string_to_convert);
    } else{
        return strtolower($string_to_convert);
    }
}
```

```
echo string_converter("Hello", 1);
```

10.

```
function string_checker($string_to_check){
    //Use substring, which extract a part of a string,
    //we can use a negative index to start from the back
    return substr($stringToCheck, -1);
}

echo string_checker("Hubburu"); //u
```

11, 12, 13

```
function make_paragraph($hej) {
    echo "<p> $hej </p>";
}
function make_heading($text, $number) {
    /*
     *  Alternative syntax:
     *  echo "<h" . $number . ">"
     *  . $text . "</h" . $number . ">";
     */
    echo "<h$number> $text </h$number>";
}
function make_tag($text, $tag) {
    echo "<$tag> $text </$tag>";
}
```

14.

```
function make_tag($text, $tag, $style, $href) {
    //if values are empty, return tag as is
    if ( $style === '' && $href === '' ) {
        echo "<$tag> $text </$tag>";
    }
    /*
     * Escape "" with \ like this: \" so they are treated as actual quotes and
     * not a part of the string in PHP.
     */
    else if ( $style === '' && $href !== '' ) {
        echo "<$tag href=\"$href\">$text</$tag>";
    }
    else if ( $style !== '' && $href === '' ) {
        echo "<$tag style=\"$style\">$text</$tag>";
    }
    else {
        echo "<$tag style=\"$style\" href=\"$href\">$text</$tag>";
    }
}
```

15.

```
function replace_linebreaks($string) {
    /* str_replace, built in function to replace every occurence of a character */
    $resultat = str_replace("\n", '<br />', $string);
    return $resultat;
}
//Use case
$proper_linebreaks = replace_linebreaks("en\nsträng\nmed\nmassa\nradbrytningar");
echo $proper_linebreaks;
```

16.

```php
function random_number_array_generator() {
    //array to store the values
    $array = [];
    //create 5 random numbers
    for ($i=0; $i < 5; $i++) {
        /*
         * `mt_rand`, built in function that returns number between
         * a range, in this case: 1 to 100
         */
        $random = mt_rand(1, 100);
        //push each value to the array, add it at the end
        array_push($array, $random);
    }
    //When the array is created, when the loop is finished. Return it.
    return $array;
}

//Store the  returned array in `$my_array`
$my_array = random_number_array_generator();
//Then loop through each value
foreach( $my_array as $value ) {
    echo "<p>Random value is: $value</p>";
}
```

17.

```php
function make_list($array) {
    //Open up a `<ul>`-element
    $list = "<ul> ";
    foreach( $array as $value ) {
        //$text, $tag, $style, $href, function we created before
        $li = make_tag($value, "li", "", "");
        $list = $list . $li;
    }
    //close the list when all items are added
    $list = $list . " </ul>";
    //Return the list
    return $list
}

//Use case
$my_array = random_number_array_generator();
$html_lista = make_list($my_array);
echo $html_lista;
```

18, 19.

```php
function capitalize($text) {
        // "david" ska bli "David"
        // plocka ut första bokstaven
        // och gör den stor
        // lägg ihop med resten av strängen
        // returnera
        $first = substr($text, 0, 1);
        $first = strtoupper($first);
        $rest = substr($text, 1);
        return $first . $rest;
    }
//Use case
$text = capitalize($text);
echo "<p>$text</p>";
```

20.

```php
function random_color(){
    //A color is a combination of red, green and blue values
    $r = mt_rand(0, 256);
    $g = mt_rand(0, 256);
    $b = mt_rand(0, 256);
    echo "rgb($r,$g,$b)";
}
```

21 .

```php
function my_round($x) {
    //(int) casts variable to int if not int, will convert float to int
    return (int)($x + 0.5);
}
```

22 .

```php
function float_to_string($float) {
    //converts our float to a string so we can replace certain
    //values in the string, for example switch . to ,
    return str_replace(".", ",", (string)$float);
}
```

23.

```php
function sum($array) {
    $sum_of_array = 0;
    for( $i=0; $i < count($array); $i++ ) {
        $sum_of_array = $sum_of_array + $array[$i];
    }
    return $sum_of_array;
}
```

24.

```php
function weekday_to_number($weekday) {
    /*
     * Always be sure to convert to lowercase or uppercase
     * when comparing case sensitive strings
     */
    $weekday = strtolower($weekday);
    switch($weekday) {
        case "måndag":
            return 1;
        case "tisdag":
            return 2;
        case "onsdag":
            return 3;
        case "torsdag":
            return 4;
        case "fredag":
            return 5;
        case "lördag":
            return 6;
        case "söndag":
            return 7;
        default:
            return "ERROR";
        }
    }
```