PHP 1

Dit is de eerste groep lessen PHP voor MBO 1ste jaars, periode 3. Studenten hebben al een beetje kennis gemaakt met JavaScript. Onderwerpen: installeren XAMPP, Code Editor, eenvoudige datatypes, (html)forms, vergelijkingen, if-the-else, arrays en loops

Goede Nederlandse online cursus: https://phpbasis.jaapvdveen.nl

- 8 Associative Arrays
- 9 Hoofdstedenspel
- 10 Portfolio
- Overzicht PHP filmpjes

0 Introductie Software Development en PHP

In deze les leren we wat over het vak Software Developer en hoe je je kansen op werk en stage kan vergroten. Dan hebben we een oefening in zoeken op internet. Dit is tevens de huiswerkopdracht.

Daarna gaan we een begin maken aan het opzetten van onze ontwikkelomgeving.

Over Software Development...

Coderen is als fietsen

Je wilt software developer worden? Dat betekent dat je moet werken. Programmeren is als fietsen dat leer je niet door een filmpje te kijken of door naar de les te komen en te luisteren naar wat je docent daar te vertellen heeft. Fietsen moet je DOEN om het te leren en dat is met programmeren ook zo.

Kans op een baan

Het verkrijgen van een stage en een baan in als software developer kan lastig zijn. Dat komt ondermeer omdat kandidaten vaak worden getest op wat ze kunnen. Soms wordt er een assessment (test) gedaan. Als jij goed bent dan kun je zo'n test halen. Je krijgt tijdens deze opleiding alles aangereikt om aan het eind van de opleiding helemaal zelfstandig een complete web applicatie te maken die op een pc en een mobiel draait. Zelfs zonder baan, kun je daar geld mee verdienen.

Inzet

Maar... je moet het wel zelf doen en het zal af en toe moeilijk zijn! Alle AO-docenten willen je daarbij helpen maar jij moet wel inzet tonen en dat betekent:

- op tijd in de les komen;
- aanweizg zijn in de les;
- tijdens de les *actief* meedoen (dus mobieltjes weg en zeker geen Netflix!);
- huiswerk maken.

Als je dit doet dan doen wij er alles aan om deze opleiding voor jou een succes te maken!

Cijfer

je zult op alle punten die genoemd staan worden beoordeeld. Zo kan het voorkomen dat je voor je testen een onvoldoende haalt maar doordat je goed hebt meegedaan in de lessen en al je huiswerk serieus hebt gemaakt, je toch een voldoende scoort.

Harde werker of slimme gast?

De meeste bedrijven hebben liever een harde werker die wat moeite heeft met coderen dan een luie werker die het allemaal komt aanwaaien en die goed kan programmeren. De luie programmeur loopt namelijk vroeg of laat een leer vast terwijl de harde werker door hard werken langzaam maar zeker zichzelf blijft ontwikellen.

Denken en leren als een programmeur

De wereld verandert snel en dat geldt zeker voor de wereld van de ICT. Dat betekent dat je je leven lang blijft leren. Wij leren dus hier op school een paar programmeertalen (JavaScript, PHP, Java en Python) maar wat belangrijker is, is dat wij leren leren. Hoe leer je relen? Door het zelf te doen: te puzellen, zaken zelf op te zoeken, tutorials te volgen, dingen proberen, testen dus gewoon DOEN!

Help het werkt niet!

https://www.youtube.com/watch?v=VrSUe_m19FY

Dus als we in de les zitten kan ik jullie alles vertellen. Ik kan zaken voor doen en dat zal ik af en toe ook doen, maar uiteindlijk moet je zelf leren om zaken uit te zoeken en op te zoeken. De docenten zullen jullie dus helpen door niet (altijd) te helpen. Als je echt hulp nodig hebt dan kan je dat krijgen maar dan zal je wel moeten uitleggen wat je zelf hebt gedaan om jouw probleem op te lossen.

Opdracht, wat is PHP?

1. Zoek uit welke grote websites gebruik maken van PHP; noem er twee.

2. Hoe oud is PHP?

3. Is PHP een back-end of front-end taal? Leg uit!

4. Noem ten minste drie voordelen op van het gebruik van PHP.

5. Zoek op wat je kan verdienen als PHP developer kijk daarbij naar vacatures.

6. Als je naar PHP developer vacatures kijkt, welke technologieën zou je dan nog meer moeten leren?

Installatie

Wij gaan een op onze laptop een development omgeving inirchten. Dit is software dat er voor zorgt dat we doen alsof we op een server werken. We laten onze laptop daarbij dus als een back-end server werken.

We kunnen dit op verschillende manieren doen. We kunnen een VM Linux server isntalleren. Dit is bijna hetzelfde alsof je een echte 'remote' server inricht maar dit is ook best een beetje lastig en daarom gaan we in de Linux lessen doen. Voor de PHP lessen gaan we gebruik maken van XAMPP voor Windows. Dat is eenvoudiger om te installeren omdat je via een Windows installer alles in één keer installeert.

Er staan mooie tutorials op het internet hoe je XAMPP moet inststalleren, bijvoorbeeld op de site: www.wikihow.com

1. Ga naar wikihow (of een andere site) en zoek op hoe je XAMPP moet installeren.

2. Als je moet kiezen welke componenten je wilt installeren dan kies je alleen Apache en MySQL.
   Dit zijn de eerste twee opties. De rest hebben we niet nodig.
   Let op dat we vanjuit security oogpunt alleen maar installeren wat we nodig hebben; weet je waarom?

3. Houd het installatie-path zoals dat in de tutorial is weergegeven: C:\xampp

4. Na installatie ga je naar de folder c:\xampp daar vind je xampp-controll.exe Hiermee start je de PHP server
   Tip: Als de installer geen shortcut heeft gemaakt dan kun je van xampp-controll.exe een shortcut op je desktop maken.

5. Gefeliciteerd als het NIET is gelukt: dit geeft jou een kans om als een echte pro te gaan trouble shooten. Ga met iemand samen kijken wat er mogelijk niet goed is gegaan.
   Pas als je zelf hebt onderzocht waarom het niet werkt kun je hulp bij je docent vragen.

Install Code Editor

Er zijn drie code editors die je kan gebuiken en die min of meer gelijkwaardig zijn:

1. Sublime Text - popular but with nagware (of USD 70)
2. Atom - open source, veel extenties
3. Brackets - open source, cross platform, veel extenties

Installeer minimaal één van deze editors; voor PHP is het nu niet zo belangrijk welke je kiest. Je kan ook meerdere naast elkaar installeren.

Huiswerk inleveren

Huiswerk telt mee voor je eindcijfer maar misschien nog belangrijker is dat huiswerk is de 'road to succes'. Zonder tijd te besteden aan programmeren leer je niet coderen en zul je meer moeite hebben met een geschikte stage te vinden en zal het ook lastiger worden om je examen te halen.

Tenzij anders vermeld lever je je husiwerk in Teams in, in een TXT document.

Code dient netjes leesbaar te zijn en wordt betjes ingesprongen.

De reden hiervoor is dat text makkelijk te lezen is en ik kan eventuele code makkelijk kopieren en uit proberen. Een text document kun je met elk van de genoemde text editors maken.

Bijvoorbeeld, hoe je de antwoorden in moet leveren.

```
// antwoorden les PHP Herhaling

// Opgave 1
1 12
2 2
3 0
4 12
5 10
6 6
7 0
8 12

// Opgave 2
for($i=0; $i<10; $i++)
```

```
// Opgave 3
// PHP Code
$color = array('white', 'green', 'red', 'purple', 'black', 'grey', 'orange', 'brown');
for($i=0; $i<6; $i++)
{
echo $color[$i];
    echo "<br>";
}

// HTML Form
<form action="test.php" method="get">

Getal 1? <input type="text" name="userName"><br>
Getal 1? <input type="text" name="userName"><br>
  <input type="submit" value="Submit">

</form>

// Opgave 4
De opdracht heeft als uitkomst 12 omdat het array 11 elementen heeft

// Opgave 5
1 false
2 false
3 true
4 false
5 true
6 true
7 false
8 false
9 fasle
10 true
11 true
12 true

// opgave 6
var_dump(1==1);
```

--

1 PHP Code en Document Root

Embedding PHP in HTML

PHP kun je tussen html plaatsen. Kijk naar het volgende voorbeeld. Kopieer het en voer het uit.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Document</title>
</head>
<body>
  <h1>Welkom op deze pagina</h1>
  <?php echo "Hello world"; ?>
</body>
</html>
```

Als we een file maken en we dubbel klikke n die file dan zien we in de browser:
file:///C:/Users/bisschopm/www/test/test.php.html

Zie je dat de URL begint met file:// ? Dat betekent dat de browser de file opent. We gebruiken alleen de web browser en PHP draait niet op de browser.

> Weet je nog PHP in een back-end taal en geen front-end taal?

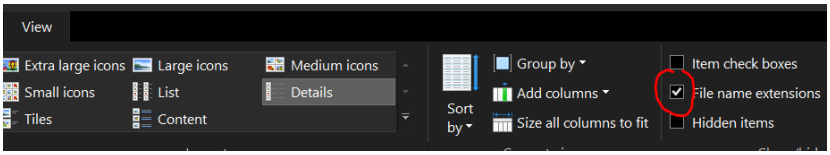
We moeten PHP dus op de 'server' draaien daarvoor hadden we XAMPP geinstalleerd. XAMPP instaleert een web server op jouw laptop. De HTML file en PHP files moeten op een bepaalde locatie staan. Dat heet de docuemnt root.

> Standaard is de document root voor XAMP c:/xampp/htdocs/

Maak een folder test in de document root van jouw webserver en zet daar de file in die je net heb gemaakt. Start de XAMPP web server en ga met je browser naar http://127.0.01/test - als het goed is zie je de directory test met daarin de html file met de php code. Open de file. Je zult zien dat de PHP code niet wordt uitgevoerd oro de server. Kijk maar eens in de "page source". Je ziet de php code gewoon; er is niets uitgevoerd.

Dat komt omdat de Apache webserver alleen PHP code uitvoert als de file de extentie php heeft. Verander de naam van de file die je net hebt gemaakt maar eens in bijvoorbeeld test.php.

Je kunt in de windows explorer (verkenner) de extentie zien als je de file name extentie laat zien (onder view, click file name extentions);



Ga nu weer naar http://127.0.0.1/test en open de test.php fle. Je zult zien dat de php code nu wordt uitgevoerd!

Het adress 127.0.0.1 is een ip address dt altijd wijst naar de eigen lokale server. Dat is zo afgesproken.

Wat hebben we geleerd?

1. PHP code wordt alleen uitgevoerd als deze in de document root van de web server staat.
2. PHP code wordt alleen uitgevoerd als de file waar de code in staat de extentie .php heeft.
3. PHP code staat tussen <?php en ?> in een html pagina
4. 127.0.0.1 is het ip address van de eigen server/werkstation; 127.0.0.1 verwijst dus altijd naar zichzelf.

--

2 PHP Simple Datatypes

Variabelen

php variabelen beginnen altijd met $ en zijn case sensitive. Verder mag je getallen en underscores gebruiken. In principe gebruiken we de Camelcase standaard.

> Camelcase: elke variabele begint met een kleine letter en elk nieuw woord begint met een hoofdletten.

Voorbeeld: $besteLeerling $hoogsteCijfer of $hoogsteCijferKlas

Je hoeft in PHP variabelen niet te declareren (zoals in JavaScript met var). Je kunt gewoon een variabele gebruiken. Variabelen zijn van het type string, integer of real. Het type wordt automatisch bepaald door PHP.

```
<?php
$stad="Parijs"; // $stad is een string
$aantal= 1;     // $aantal is een integer
$aantal="12";   //$aantal is een string
$prijs=1.95;    //$prijs is een real
$antwoord=False// $antwoord is boolean (true of false)
?>
```

Over booleans wordt in de volgende les meer verteld.

String Concatenate

In PHP plak je strings aan elkaar met een . (punt). Dit heet in het Engels string concatenate.

```
<?php
$naam="John";
echo "Welkom ".$naam;
?>
```

Operator

Net als in JavaScript kent PHP min of meer dezelfde operatoren.

Operator	Beschrijving	Voor variabel type(n)	voorbeeld
+	optellen	int, real	$a=$a+$b;
-	aftrekken	int, real	$a=$a-4;
/	delen	int, real	$x=12/3;
*	vermenigvuldigen	int, real	$y=12*3;
.	concatenate	string	$a="a"."b";
%	modulo	int, real	$x=21%10;
**	machtsverheffen	int, real	$y=2**3;
++	1 erbij optellen	int, real	$a++;
--	1 er van aftrekken	int, real	$a--

Typecasting

In PHP wordt het type variabele automatisch bepaald, maar als programmeur kun je een variabele een ander type geven. Dat noem je typecasting.

```
<?php
$myVar=12;
echo "Het nummer is ". (string)$myVar;

$nummer="125";
$nummmer=(int)$nummer+5;
echo "Het nummer is ".(string)$nummer;
?>
```

Typecasting gebeurt meestal vanzelf. PHP probeert te 'raden' wat je bedoeld. In sommigen gevallen kan het echter nodig zijn om een variabele te casten.

Omdat je soms niet weet welk type een variabele is kun je met var_dump() het type het type opvragen.

```
<?php
$teller="0";
var_dump($teller);

echo "<br>";
```

```
$teller=$teller+1;
var_dump($teller);
?>
```

Maar wanneer heb je typecasting nou echt nodig?

Omdat PHP zelf types omzet zul je typecasting niet vaak nodig hebben, maar onderstaand voorbeeld kun je het wel gebruiken:

```
$nummer=10;
echo 'De waarde is '. $nummer . ' en dat is goed';
```

Het type $nummer is int. Als je gewoon nummer afdrukt dan wordt het type van $nummer door PHP automatisch verandert naar een string. En dan plak je de drie strings aan elkaar. Niets aan de hand. Maar laten we een klein wijziging aanbrengen:

```
$nummer=10;
echo 'De waarde is '. $nummer+1 . ' en dat is goed';
```

Nu geef je PHP opdracht om 1 bij $nummer op te tellen. Dat betekent dat PHP $nummer zal (blijven) zien als een int. En een int kan je niet als een string aan elkaar plakken. Je kunt dit op verschillende manieren oplossen en type casting is er een van.

```
$nummer=10;
echo 'De waarde is '. (string)($nummer+1) . ' en dat is goed';
```

Opmerking: nu mag je zelfs (string) weer weglaten en alleen $nummer+1 tussen haakjes zetten. Je voert dan hetzelfde uit maar het is minder duidelijk wat er dan precies gebeurt.

En dan nog wat...

Er zijn nog een paar zaken die we niet heel erg duidelijk genoemd hebben. Maar dat komt omdat deze zaken hetzelfde zijn als dat we bij JavaScript hebben geleerd.

Zoals je hebt kunnen zien eindigt elke PHP-commando met een; net als in Javascript.

Je hebt in de voorbeelden ook gezien dat als je commentaar wilt toevoegen je // kan gebruiken.

Als je meer regels in commentaar wilt zetten dan kan je die tussen /* en */ zetten.

In de volgende lessen zullen we nog meer overeenkomsten met JavaScript tegenkomen.

We hebben ook het commando *echo* gezien dit is in JavaScript *document.write* maar in Unix Bash is het wel weer hetzelfde.

Opgaven

1. Maak een PHP-programmaatje. Maak twee variabelen en geeft deze de waarden 12 en 10. tel deze variabelen bij elkaar op en druk het resultaat af.

2. Maak een PHP-programmaatje. Maak twee variabelen en geeft deze de waarden 12 en 10. vermenigvudig deze variabelen en druk het resultaat af.

3. Maak een string en in PHP en zet daarin je naam. Druk daarna de boodschap af: "Welkom ...." op de plaats van de ... moet dan je naam worden afgedrukt.

4. Haal alle fouten uit de volgende code en schrijf de gecorrigeerde code op je antwoordblad.

```
<?php

// deze code bevat veel fouten
   haal alle fouten eruit!
//

$mijnNaam="Carl";
$jouwNaam=Camel;
echo $mijnNaam + " zit bij " + jouwNaam + " in de klas!"

?>
```

--

2.1 Strings en quotes

In deze les gaan we nog wat meer kijken naar strings en het verschil tussen " en ' (double quotes en single quotes).

Wat is een string?

Een string is een rijtje van karakters. Deze karakters kunnen van alles zijn: letters, cijfers, leestekens, spaties, ...

Als je een string maakt in PHP (en andere programmeertalen) dan moet je quotes gebruiken om aan te geven dat je een string bedoeld. Als je dat niet doet dan zou PHP niet weten wat je bedoeld:

Bijvoorbeeld echo en "echo" zijn twee verschillende dingen! Zonder quotes is het een commando en met quotes is het string.

Nog een voorbeeld $naam is wat anders dan "$naam". De eerste is de variabele en de tweede is een string.

Dus:

> `echo $naam;`
>
> is wat anders als
>
> `echo '$naam';`
>
> en
>
> `echo echo;`
>
> is wat anders als
>
> `echo "echo";`
>
> Probeer het maar!

Quotes

Een string hoort dus tussen quotes, maar wat is het verschil tussen double quotes en singel quotes?

Single quotes en escape

Dit is de meeste eenvoudige vorm. Een string tussen single quotes is gewoon een string en alles wat tussen de singel quotes staat is onderdeel van de string. Er is dan wel een klein probleempje: namelijk hoe maak je een string met een single quote?

Probeer maar: `echo 'Dit is een ' single quote'` werkt niet!

Dat komt omdat PHP denkt dat de string wordt afgesloten door de tweede quote.

Er geldt dus een uitzonderingsregel voor het maken van een string met een single quote. En die regel is dat als een singel quote onderdeel uitmaakt van een string je de singel quote moet 'escapen' en dat doe je als volgt:

```
echo 'Dit is een \' single quote.';


echo 'I\'ll be back!';
```

Double quotes

Double quotes zijn eigenlijk ingewikkelder omdat veel zaken die je tussen double quotes intypt worden gerenderd. Renderen betekent 'omzetten'. Als je bijvoorbeeld een variabele in een string tussen double quotes zet dan wordt niet de variabele naam, maar de waarde van de variabele afgedrukt.

Probeer de volgende code maar eens uit en zie wat het verschil is.

```
$naam="The Big Boss";
echo "Who is $naam";
echo "<br>";
echo 'Who is $naam';
```

Er zijn nog een paar zaken die tussen double quotes worden gerendered maar die zijn voor webtoepassingen niet zo belangrijk. Zo wordt een \n een nieuwe regel en \t een tab.
Als je een dubbele quote of een enkel quote wilt afdrukken moet je er een \ voor plaatsten en voor een backslash moet je zelf ook weer een \ plaatsten.

Dus samengevat:

expressie	tussen double quotes resulteert in	tussen singel quotes resulteert in
\"	"	"

\'	'	'
\\	\	\
$naam	de waarde van de variabele $naam	$naam
\n	nieuwe regel	\n
\t	tab	\t

Heredoc

In PHP zit een extra hulpje om grote blokken text (bijvoorbeeld html code) op een eenvoudige manier aan een variabele toe te kennen. Stel je wilt een tabel in een variabele zetten, dan kan dat op de volgende manier.

```
$html = '':
$html .= '<table><tr>';
$html .= '<td> kolom 1 </td>';
$html .= '<td> kolom 2 </td>';
$html .= '</tr><tr>';
$html .= '<td> kolom 1 </td>';
$html .= '<td> kolom 2 </td>';
$html .= '</tr><table>';

echo $html;
```

Dat werkt en ziet er best overzichtelijk uit, maar het kan ook zo:

```
$html = <<< END_HTML
<table>
<tr>
<td> kolom 1 </td>
<td> kolom 1 </td>
</tr><tr>
<td> kolom 1 </td>
<td> kolom 1 </td>
</tr>
</table>
END_HTML;

echo $html;
```

De END_HTML is een willekeurig gekozen identifier. Je mag elk woord gebruiken, maar het moet wel aan het begin van de regel staan anders werkt het niet als markering van het einde.

3 Vergelijkingen en forms

Vergelijkingen, bijvoorbeeld met if-then-else gaan min of meer hetzelfde als in JavaScript. We gaan in deze les een praktijk opdracht doen en gaan daarbij gebruik maken van forms. Let op forms zijn erg belangrijk en komen telkens terug, ze maken ook onderdeel uit van Laravel en van het examen.

Boolean

Een boolean is een data type in bijna alle programmeertalen en kent twee waarden true of false. Soms staan deze waarden gelijk aan 1 en 0; 1 is dan true en 0 is dan false. In PHP geldt dat alle waarden behalve een 0 of een lege waarde zijn true. Dit kan belangrijk zijn omdat een if-statement altijd met een boolean werkt.

> Dus: alles is true behalve een 0, een lege object/string, of de boolean false

Dat bekent dus bijvoorbeeld dat:

```
<?php
if ( 1 ) { ..... // dit is true en de if wordt uitgevoerd
if ( "" ) { ..... // dit is false en de if wordt niet uitgevoerd
if ( 3-3 ) { .... // dit is false en de if wordt niet uitgevoerd
if ( "Hallo" ) .. // dit is true en de if wordt dus uitvevoerd
```

Later gaan we nog meer oefenen met if statements.

Vergelijkingen

Een vergelijking resulteert altijd tot een boolean (true of flase). PHP kent de volgende belangrijkste vergelijkingen.

Vergelijkings Operator	Betekenis	Voorbeeld
==	is gelijk aan	$teller==3 of $naam=="John"
<> of !=	is ongelijk aan	$teller!=3 of $naam<>"John"

<	is kleiner dan	$nummer<10
>	is groter dan	$nummer>10
<=	is kleiner dan of gelijk aan	$nummer<=10
>=	is groter dan of gelijk aan	$nummer>=10

We kunnen een variabele ook *typecasten* naar een boolen met (bool) of met (boolean). Met vardump kunnen we dan het type variabele en de waarde afdrukken.

```
<?php
var_dump(1==1);

$naam="";
var_dump( (bool)$naam );

$naam="Maria";
var_dump( (bool)$naam );
?>
```

Opgave 1

Je kunt deze opgave maken door het Word document te downloaden en in te vullen.

1. Bepaal van de volgende waarden of ze *true* of *false* zijn en laat zien welke code je daarvoor hebt gebruikt. Probeer van te voren te voorspellen wat de uitkomst (true of false) is.

Opgave	Vergelijking	true of false?
1	("ROC")	
2	("true")	
3	("false ")	
4	(0)	
5	(1)	
6	("0")	
7	("1")	
8	(3+3)	

9	(12-12)	
10	(201*0)	
11	(12<12)	
12	(1201 != 1200)	
13	(1201 >= 1201)	

Als $a=0 en $b=1 wat is dan de uitkomst van de volgende statements?

Opgave	Vergelijking	true of false?
14	($a==1)	
15	($b==1)	
16	($a==1 or $b==1)	
17	($a==0 and $b<>1)	
18	($a<>1 and $b<>0)	
19	($a+$b==1 and $a*$b==0)	
20	($a+$b==1 or $a*$b==0)	
21	($a)	
22	($b)	
23	($a and $b)	
24	($a or $b)	
25	($a or $b*$a)	
26	(($a+2)*10>5 and $a)	

If-statement

Het if statement in PHP is hetzelfde als in JavaScript. Ook de && en || zijn ook hetzelfde, maar mogen ook als *and* en *or* worden geschreven. Dus:

```
if ( $leeftijd > 21 && $leeftijd < 30 ) { ...
// dit is hetzelfde als
if ( $leeftijd > 21 and $leeftijd < 30 ) { ...

if ( $naam == "" || $naam == "leeg" ) { ...
// dit is hetzelfde als
if ( $naam == "" or $naam == "leeg" ) { ...
```

Het if statement is verder ook hetzelfde als in JavaScript:

```
if ( $command == "execute" ) {
    // ......hier komt code voor als de gebruiker execute heeft gekozen.
} elseif ( $command == "cancel" ) {
    // ......hier komt code voor als de gebruiker cancel heeft gekozen.
} else {
    // ......hier komt code voor als de gebruiker geen execute of cancel heeft gekozen.
}
```

User input

De les over forms en user input is verplaatst naar de volgende les 3.1

--

3.1 Forms

User input

Met JavaScript kunnen we met het *prompt* commando de gebruiker om input vragen. Dat kan met PHP niet en dat komt, omdat PHP niet in de browser draait. PHP draait op de server en als je een venster zou openen dan zou dat venster op de server openen en daar heb je natuurlijk niets aan.

PHP kan alleen met de gebruiker communiceren door html naar de browser te sturen en de browser kan alleen met de server communiceren door met formulieren (forms) iets te posten.

Dat werkt als volgt:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Document</title>
</head>
<body>

  <!-- Hier begint het form -->
  <form action="action.php" method="get">

    Name: <input type="text" name="userName"><br>
    Geboortejaar: <input type="number" min="1900" max="2030" name="geboorteJaar"><br>
    Jarig geweest (ja/nee)? <input type="text" name="jarigGeweest"><br>
    <input type="submit" value="Submit">

  </form>

</body>
</html>
```

Dat is een heel stuk code maar waar het om gaat, is het formulier op regel 12 tot en met 17. De rest is een standaard HTML template en kun je vaak met een editor in één keer intoetsen

bijvoorbeeld met `!-tab`

De user input wordt door het form gevraagd en wordt naar de php file action.php gestuurd. Maak en een action.php en zet daar het volgende in:

```
<?php

$thisYear=date("Y");

echo "Naam: ".$_GET['userName'];
echo "<br>";

echo "Geboortejaar: ".$_GET['geboorteJaar'];
echo "<br>";

echo "Al jarig geweest?: ".$_GET['jarigGeweest'];
echo "<br>";

echo "Huidig jaar: ".$thisYear;
echo "<br>";

?>
```

Kijk wat er gebeurt als je het form post. Als het goed is zie je dat alle variabelen worden afgedrukt. Je kan nu uitrekenen hoe oud de persoon is.

Wat denk je dat het type van de variabele `$_GET['geboortejaar']` is? Check meet `var_dump` welk van welk type de variabele `$_GET['geboortejaar']` is.

Opgave 1

```
$leeftijd = .... - ....;

if ( strtoupper($_GET['jarigGeweest']) ..... ) {
  $leeftijd++;
}

echo $_GET['userName']."", jouw leeftijd is: ".$leeftijd;
```

Plaats deze code onderaan in action.php en vul de code aan op de plaats van de puntjes.

- Je moet dus de leeftijd berekenen door twee variabelen van elkaar af te trekken.
- en je moet het if-statement afmaken

strtoupper()

Let op dat strtoupper($_GET['jarigGeweest']) de string ja of nee in hoofdletters zet. Dit zorgt ervoor dat als je *ja* of *Ja* of *JA* intypt dit allemaal wordt opgezet in *JA.* strtoupper() zet de string dus op in een string met alleen maar hoofdletters.

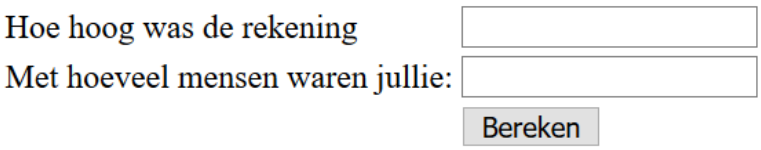
Zorg ervoor dat de code werkt en dat dit bijvoorbeeld de output is:

```
Naam: Ayoub
Geboortejaar: 1996
Al jarig geweest?: Nee
Huidig jaar: 2020
Jouw leeftijd is: 24
```

Opgave 2

Maak een kleine app waarin je berekent hoeveel iedereen moet betalen als je gezamenlijk heb gegeten en gedronken en de rekening wil verdelen.

Maak een form dat vraagt hoe hoog de rekening was en met hoeveel personen jullie waren. In de action van het form bereken je dan wat iedereen moet betalen.



Dit voorbeeld form is in een (html) table geplaatst om het netjes uitgelijnd te tonen. Probeer zelf ook of je het form in een table kunt zetten.

Dan vul je (bijvoorbeeld) 200 en 5 in en dan moet de volgende output verschijnen:

Jullie waren met 5 personen en de rekening was 200 Euro
Per persoon moet je 40 Euro betalen.

Succes!

Plaats de volgende bestanden in Teams (huiswerk):

Voor opgave 1, één file: de aangepaste file action.php
Voor opgave 2, twee files: opgave2.html en opgave2.php. De html-file bevat het form en de action-file bevat de php code.

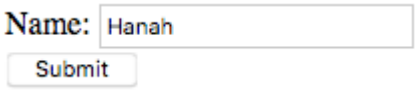
-_-

3.2 Opdracht met forms

Omdat forms erg belangrijk zijn gaan we nog een keer oefenen met forms.

Opdracht 1

Vraag door middel van een form de naam van de gebruiker.

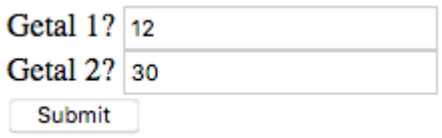


Stel je voert als naam Hanah in dan wordt de output van je programma code:

```
Welkom Hanah!
```

Opdracht 2

Vraag de gebruiker om twee getallen.

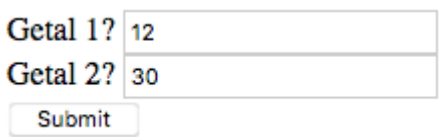


Tel de getallen bij elkaar op en druk het volgende af:

```
12 + 30 = 42
```

Opdracht 3

Vraag de gebruiker om twee getallen.



a) Bepaal het grootste getal en druk dat op de volgende manier af:

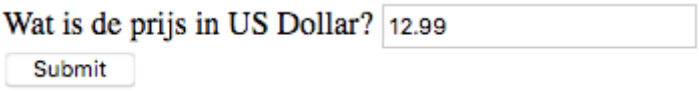
```
30 is groter dan 12
```

b) Pas de code aan zodat als de getallen gelijk het volgende wordt afgedrukt:

```
12 is gelijk aan 12; geen van de getallen is groter
```

Opdracht 4

Maak een form en vraag de gebruiker de prijs in US Dollars. Submit het form en bereken de prijs in Euro's. Gebruik daarbij een vast omrekenkoers van 1.11 Dus 22.20 Dollar = 22.20/1.11=20 Euro. Om van US Dollar naar Euros te gaan deel je dus door 1.11



Stel je hebt 12.99 ingevoerd in het form dan is dit de output van je programma-code.

```
12.99 US Dollar is gelijk aan 11,70 euro
```

4 Arrays

In deze les gaan we het over arrays hebben en gaan we leren wat een array is en hoe je er mee kan werken.

Arrays algemeen

Een array is een verzameling van variabelen. Een array bestaat uit een index of een keys (sleutels) en waarden. Bij een één dimensionaal array zijn de keys/indexes altijd 0,1,2,3,4.... Deze keys worden dus automatisch gemaakt en beginnen bij 0. Bij associatie arrays zullen we zien dat we de keys zelf kunnen bepalen.

https://www.youtube.com/embed/5lJLecI0BTA

In dit filmpje wordt uitgelegd wat een array is en wat het verschil is tussen een eenvoudig, one dimensional array en een assiociative array.

One Dimensional Array

De meest eenvoudige vorm van een array is een één dimensionaal array. Dit is eigenlijk niets meer als een lange rij variabelen.

Array maken

Een eenvoudig array kun je op verschillende manieren maken, de meest gebruikte methodes zijn:

```
$fruit = ['appel', 'banaan', 'citroen'];
$fruit = array('appel', 'banaan', 'citroen');
```

Regel 1 en 2 doen hetzelfde. Bij regel 1 maak je rechtstreeks een array en in regel 3 gebruik je er een functie voor.

Array uitlezen

Het opvragen van een waarde gaat als volgt:

```
echo $fruit[1];
```

Het element met key 1 wordt nu afgedrukt. Omdat keys bij 0 beginnen wordt hier *banaan* afgedrukt.

Array-waarde aanpassen

Als je een waarde wilt veranderen dan gaat dat als volgt.

```
$fruit[2]='druif';
```

Array uitbreiden

Een element aan het einde toevoegen kan als volgt:

```
$fruit[]='peer';
```

Bestudeer: https://www.w3schools.com/js/js_arrays.asp

Er zijn heel veel ingebouwde php functies waarmee je iets met een array kun doen.

De count() functie is belangrijk en wordt veel gebruikt, omdat je daarmee het aantal elementen van een array kan opvragen.

Opdracht 1

```
$myArray = ['auto','fiets','brommer','bus','vliegtuig','trein'];
```

Maak een programmaatje met dit array.

a) Voeg een waarde toe zonder de regel zoals die hierboven is gegeven aan te passen. Dus maakt een tweede regel waarin je een waarde toevoegt aan het array.

b) Druk alle waarden van het array af.

Zet de code op je antwoordenblad.

Opdracht 2

```
$myArray = ['auto','fiets','brommer','bus','vliegtuig','trein'];
```

Maak een programmaatje met dit array en bepaal het aantal elementen en druk dat af.

De output moet luiden:

```
Het array heeft 6 elementen.
```

Voeg een element toe aan het array en de output moet nu vanzelf veranderen naar:

```
Het array heeft 7 elementen.
```

Zet de code op je antwoordenblad.

Opdracht 3

```
1  $array = array("foo","bar","hello","world");
2  echo $array[1];
3
4  $array = array(1,2,3,4,5);
5  echo $array[3];
6
7  $cars = array("Volvo","BMW","Toyota");
8  echo $array[2];
9  echo count($cars);
10
11 $array = [ 5, 4, 3, 2, 1 ];
12 echo $array[0];
13
14 $numbers[0]="one";
15 $numbers[1]="two";
16 $numbers[2]="three";
17 $numbers[3]="four";
18 $numbers[4]="five";
19 echo $numbers[count($numbers)-1];
```

Bepaal wat er wordt afgedrukt op regel 2, regel 5 , regel 8, regel 9, regel 12 en regel 19.

Schrijf de 6 antwoorden op je antwoordenblad op.

Opdracht 4

a) Kijk naar het volgende voorbeeld waarin je een array hebt met bijvoorbeeld al jouw cijfers. Je ziet dat je steeds beter bent geworden in PHP coderen. Dat komt doordat je telkens je huiswerk hebt gemaakt :)

Het gemiddelde bereken je door alle getallen op te tellen en te delen door het aantal getallen. Dat kan makkelijk in PHP omdat er een array functie is om het aantal elementen in array te tellen (die heb je ook al bij opdracht 2 gebruikt) en er is een functie om alle getallen in array op te tellen.

En sta je een voldoende?

Bestudeer de code en probeer te begrijpen hoe het werkt.

```
<?php

$cijfersPHP = [4.4, 4.6, 5.6, 6.1, 7.6, 7.2];

// bereken gemiddelde door alle cijfers op te tellen en te delen door het aantal
$aantalCijfers = count($cijfersPHP);
$gemiddelde = array_sum($cijfersPHP) / $aantalCijfers;

echo "Gemiddelde is: ".$gemiddelde;

?>
```

Verander de code, zodat:

- Regel 6 en 7 worden gecombineerd tot één regel met één commando,
- en zorg ervoor dat het gemiddelde uit het bovenstaande voorbeeld wordt afgerond op 1 decimaal, dus bijvoorbeeld 8.1 of 8.0.

Lees hoe je kunt afronden op: https://www.w3schools.com/php/func_math_round.asp

Zet de aangepaste code op je antwoordenblad.

b) En wat is beter regel 6 en 7 samenvoegen zoals de opdracht was of zo laten staan. Schrijf op je antwoordenblad op, wat jouw voorkeur heeft en leg uit waarom. Dus:

Ik denk dat het beter is om de regels samen te voegen, omdat ……

en/of

Ik denk dat het beter is om de regels niet samen te voegen, omdat …..

--

5.1 For - Loops

In deze les gaan we kennismaken met loops in PHP. We leren wat een for-loop is en hoe we die in combinatie met arays kunnen toepassen.

In het onderstaande filmpje wordt de basis uitgelegd van hoe een for loop in werkt.

https://www.youtube.com/embed/dlgq69XKiR0

Loops (lus)

Een loop (in het Nederlands "lus") is een stuk code dat onder bepaalde voorwaarden uitgevoerd wordt. Zolang de voorwaarde waar (TRUE) is, wordt de code in de loop steeds opnieuw uitgevoerd. Als in de lus een variabele wordt aangepast, dan heeft deze variabele de nieuwe waarde bij de volgende keer dat de loop wordt uitgevoerd. Totdat de voorwaarde van de loop niet meer waar is. De voorwaarde moet zorgvuldig worden gekozen; een foutje kan er voor zorgen dat de loop niet meer stopt. Mocht dat toch gebeuren, dan zorgt PHP dat het script (standaard) na 30 seconden toch wordt gestopt.

De verschillende soorten lussen in PHP lijken op elkaar, alleen is steeds de voorwaarde anders opgebouwd. De while-lus is het meest eenvoudig, de for-lus is wat ingewikkelder en de foreach-lus is er speciaal voor het werken met arrays.

For Loop

For loops komen in alle programmeer talen voor. Een loop zorgt ervoor dat een stuk programma code X keer wordt uitgevoerd. De waarde van X hangt af van hoed e loop wordt geprogrammeerd. Eén van de meest gebruiken loops is de for -loop.

Dus een for-loop ziet er altijd als volgt uit:

> for(**lus- startwaarde** ; **lus-voorwaarde** ; **lus-teller**) { lus-code }

In de for loop bepaal je in ieder geval een begin- en een eind waarde bijvoorbeeld 1 t/m 10. De loop wordt dan 10 x uitgevoerd. Bijvoorbeeld:

```
<?php
for($i=0; $i<10; $i++)
{
  echo "Hello<br>";
}
echo "Einde loop<br>";
?>
```

In de for loop staan drie statements:

$i=0;	**lus-startwaarde**	Dit is de startwaarde van de index $i
$i<10;	**lus-voorwaarde**	De loop gaat door zolang deze vergelijking *true* is, dit heet de voorwaarde
$i++;	**lus-teller**	Na elke iteratie (uitvoering van de code in het blok van de loop) wordt dit uitgevoerd.
{ ....}	**lus-code**	Deze code hoort bij de loop en wordt X keer uitgevoerd.

Dit gebeurt er stap voor stap:

Iteratie	Wat gebeurt er?
1	$i=0
	$i < 10 // als dit waar is voer dan de code uit anders ga door met de code na de loop
	$i++ // $i wordt nu 1
2	$i < 10 // nog steeds waar dus voer de code die bij de loop hoort nog een keer uit.
	$i++ //$i wordt nu 2
3	$i < 10 // nog steeds waar dus voer de code die bij de loop hoort nog een keer uit.
	$i++ //$i wordt nu 3
...	...
9	$i < 10 // nog steeds waar dus voer de code die bij de loop hoort nog een keer uit.

	$i++ //$i wordt nu 9
10	$i < 10 // NIET meer waar dus de code in de loop wordt niet meer uitgevoerd, de loop stopt!

Opdracht 1

Bepaal van elk van de volgende loops hoe vaak die wordt uitgevoerd.

```
for($i=0; $i<10; $i++)
for($i=1; $i<10; $i++)
for($i=0; $i<100; $i++)
for($i=1; $i<100; $i++)
for($k=0; $k<1; $k++)
for($k=1; $k<1; $k++)
for($i=10; $i>0; $i--)
for($i=10; $i>0; $i=$i-1)
for($i=10; $i<>4; $i--) // <> betekent 'ongelijk aan' != mag ook
for($i=2; $i; $i--) // denk aan wanneer $i true of false is
for($i=0; $i>0; $i++)
for($i=1; $i>0; $i++)
```

Schrijf van elke regel op hoe vaak de loop wordt uitgevoerd.

Opdracht 2a

Maak een klein programmaatje met een for-loop waarin alle even getallen tussen 0 en 1000 worden afgedrukt. De output wordt dus: 2, 4, 6, 8, 10 ,12 ,14 ,16, ..., 996, 998, 1000

Opdracht 2b

Maak een klein programmaatje met een for-loop waarin alle oneven getallen aflopend tussen 1000 en 800 worden afgedrukt. De output wordt dus: 999, 997, 995, 993, 991, 989, ..., 807, 805, 803, 801

Opdracht 3

Voer onderstaande code uit en verander deze code door de regels 4 t/m 15 met behulp een loop af te drukken.

Let op, je mag het array $maanden niet aanpassen.

```
<?php
$maanden = ['Januari', 'Februari', 'Maart', 'April', 'Mei', 'Juni', 'Juli', 'Augustus', 'September', 'Oktober',
'November', 'December'];

echo 'Maand 1 is '.$maanden[0].'.<br />';
echo 'Maand 2 is '.$maanden[1].'.<br />';
echo 'Maand 3 is '.$maanden[2].'.<br />';
echo 'Maand 4 is '.$maanden[3].'.<br />';
echo 'Maand 5 is '.$maanden[4].'.<br />';
echo 'Maand 6 is '.$maanden[5].'.<br />';
echo 'Maand 7 is '.$maanden[6].'.<br />';
echo 'Maand 8 is '.$maanden[7].'.<br />';
echo 'Maand 9 is '.$maanden[8].'.<br />';
echo 'Maand 10 is '.$maanden[9].'.<br />';
echo 'Maand 11 is '.$maanden[10].'.<br />';
echo 'Maand 12 is '.$maanden[11].'.<br />';
?>
```

Opdracht 4a

In de loop kun je ook gebruik maken van de index-variabele $i. Bijvoorbeeld:

```
<?php
for($i=1; $i<11; $i++) {
  echo $i . " * 7 = " . ($i*7) . "<br>";
}
?>
```

$i wordt in deze code gebruikt om de tafel van 7 af te drukken.

Verander de bovenstaande code, zodat je een tafel van x kan afdrukken. Maar een variabele $x en geef die voor de loop een waarde. De tafel van $x wordt afgedrukt.

(Tip: vervang de 7 's in het voorbeeld met een variabele)

Opdracht 4b

Maak nu een form en vraag de gebruiker welke tafel die wilt afdrukken. Zorg er dan voor dat deze tafel wordt afgedrukt.

Stel de gebruiker geeft aan dat hij de tafel van 14 wil afdrukken, dan zou dit de output moeten worden:

```
1 X 14 = 14
2 X 14 = 28
3 X 14 = 42
4 X 14 = 56
5 X 14 = 70
6 X 14 = 84
7 X 14 = 98
8 X 14 = 112
9 X 14 = 126
10 X 14 = 140
```

Opdracht 5

Het programma hieronder bevat een fout.

```
<?php
$colors=array('black','red','white');

for($i=0; $i<=3; $i++) {
  echo $colors[$i]."<br>";
}
?>
```

De foutmelding is

Notice: Undefined offset: 3 in **/Applications/XAMPP/xamppfiles/htdocs/test.php** on line **5**

Wat betekent "Undefined offset"? Schrijf in je eigen woorden op wat dit betekent en leg uit waarom deze code een foutmelding geeft.

Opdracht 6

```
$dagen=['maandag','dinsdag','woensdag','donderdag','vrijdag','zaterdag','zondag'];
```

a) Maak gebruik van een loop en van bovenstaand array en druk alle dagen van de week op de volgende manier af:

```
Het is maandag
Het is dinsdag
Het is woensdag
Het is donderdag
Het is vrijdag
Het is zaterdag
Het is zondag
```

Schrijf de php code op je antwoordenblad.

b) Maak gebruik van een loop en van bovenstaand array en druk alle dagen van de week op de volgende manier af:
(je mag het array $dagen niet aanpassen).

```
Dag 1 van de week is maandag
Dag 2 van de week is dinsdag
Dag 3 van de week is woensdag
Dag 4 van de week is donderdag
Dag 5 van de week is vrijdag
Dag 6 van de week is zaterdag
Dag 7 van de week is zondag
```

Tip: In de loop heb je de loop variabele die telt. Gebruik deze variabele om de nummers 0 t/m 6 af te drukken. Denk dan na hoe je van 0 t/m 6 naar 1 t/m 7 kan komen.

Schrijf de php code op je antwoordenblad

Opdracht 7

Kijk naar het volgende programmaatje

```
<?php
...
for( ; $i<=10 ; ) {
  echo "$i"."<br>";
  ...
}
```

```
?>
```

Deze for-loop heeft in dit voorbeeld alleen een voorwaarde, de startwaarde en de lus-teller zijn weggelaten. Dat mag, maar de loop werkt niet meer. Vul nu op de plaats van de puntjes code toe, zodat de loop op deze manier werkt en de getallen 0 t/m 10 afdrukt.

Gelukt? Je hebt nu eigenlijk een soort while-loop gemaakt en dat gaan we in de volgende les verder behandelen.

--

5.2 While - Loop

While-Loop

De while loop is eigenlijk eenvoudiger uit te leggen. Hij kent één vergelijking, de lus-voorwaarde en zolang die true is blijft de loop doorlopen.

```
$i=0;
while($i<10){
$i++;
   ...
}
```

Met deze while-loop doe je eigenlijk hetzelfde als in de for loop: `for($i=0; $i<10; $i++)` Je ziet dat de startwaarde op regel 1 staat en de lus-teller op regel 3.

Deze loop is iets 'gevaarlijker', omdat de drie fases van de loop niet per sé bij elkaar staan. Dat is 'gevaarlijk', omdat je dan eerder het overzicht verliest en misschien het ophogen met de lus-teller (in het voorbeeld regel 3) vergeet. Ook kan het minder overzichtelijk zijn.

Bij een while-loop moet je net als bij een for-loop altijd nadenken over de 3 fasen:

1. Wat is de start-waarde; hoe begint de loop (in het voorbeeld $i=0)?
2. Wat is de lus-teller?
3. Wat is de lus-voorwaarde??

Stel je hebt een functie die een row uit de database haalt, deze functie heet getRow() en deze functie returned de row uit de database. Als er geen rows meer zijn dat wordt er 0 teruggegeven. Je kunt nu een while-loop gebruiken om alle regels af te drukken:

```
<?php
$dezeRegel = getRow();

while($dezeRegel<>0){
echo $dezeRegel;
   $dezeRegel=getRow();
}
?>
```

Het is gebruikelijk om dit op deze manier uit te voeren, maar het kan ook in een for-loop. Welke methode vind jij beter leesbaar?

```
<?php
for( $dezeRegel = getRow(); $dezeRegel<>0; $dezeRegel=getRow();){
echo $dezeRegel;
}
?>
```

In sommige gevallen wil je eindeloze loop maken. Een while loop kun je eenvoudig eindeloos maken: `while(true){...`

Do-While

De do-while loop is een variant op de while-loop. In deze loop staat de lus-voorwaarde aan het eind in plaats van in het begin zoals bij de while-loop en de for-loop.

```
<?php
$i = 0;
do {
    echo $i;
} while ($i > 0);
?>
```

Wat denk je dat deze code doet? Probeer deze code. Als je deze code uitvoert, dan zie je dat de loop 1x uitgevoerd wordt en dat terwijl $i nooit groter dan 0 is (en dus de vergelijking nooit true oplevert). Een Do-While loop wordt dus altijd minimaal 1x uitgevoerd. Dat geldt niet voor de 'gewone' while-loop of voor de for-loop.

Stroomdiagrammen

Hieronder zie je twee stroom-diagrammen. Welke hoort bij de while en welke bij de do-while? Welke zou het beste passen bij een for-loop?

Image result for do while loop php	Image result for while loop in php

Break

Met een break statement spring je uit de huidige loop. Je gaat direct naar het eind en begint met het statement dat vlak na de } staat aan het einde van de loop. De lus-code wordt dus niet verder uitgevoerd.

Continue

Met het continue statement stop je met de huidige iteratie en ga je naar het begin van de loop om de volgende iteratie te beginnen. De lus-code wordt dus weer vanaf het begin uitgevoerd. Bij een for-loop wordt dat we de lus-teller bijgewerkt en wordt gekeken of er nog aan de lus-voorwaarde wordt voldaan.

Opdracht 1

De volgende code moet alle getallen van 1 t/m 25 afdrukken, maar de programmeur heeft één regel code vergeten. Denk aan de lus-voorwaarde, lus-startwaarde en lus-teller; staan die er allemaal in? Kun jij een regel toevoegen, zodat de getallen 1 t/m 25 worden afgedrukt?

```
<?php
$count=0;
while (true) {
  echo $count . "<br>";
  if ( $count == 25 ) {
    break;
  }
}
?>
```

Opdracht 2a

Met de volgende code gooi je met twee dobbelstenen ($dice1 en $dice2). Beide dobbelstenen krijgen een willekeurige waarde van 1 t/m 6, net als een dobbelsteen.

```
$dice1=rand(1,6);
$dice2=rand(1,6);
echo $dice1. "-" .$dice2;
echo "<br>";
```

Zet de code in een loop, zodat er 10x met beide dobbelstenen wordt gegooid.

De output ziet er bijvoorbeeld als volgt uit:

```
1-2
5-6
```

```
4-2
5-3
4-3
3-1
5-6
6-1
3-2
1-3
```

Opdracht 2b

Verander de loop nu, zodat de code net zolang blijft 'gooien' totdat er twee maal 6 is gegooid. Op de laatste regel moet dus 6-6 komen te staan.

Zet een getal voor de worp, zodat je eenvoudig kan zien hoe vaak de code de dobbelstenen heeft 'gegooid'. Zet ook achter elke worp de som van de twee dobbelstenen. De output ziet er dan bijvoorbeeld als volgt uit. De laatste regel moet dus altijd twee zessen laten zien.

```
worp 1: 1-2 totaal 3
worp 2: 5-6 totaal 11
worp 3: 4-2 totaal 6
worp 4: 5-3 totaal 8
worp 5: 4-3 totaal 7
worp 6 :3-1 totaal 4
worp 7: 6-6 totaal 12
```

Opdracht 2c

Pas de code van opdracht 2b aan, zodat je met drie dobbelstenen gooit. De output iet er bijvoorbeeld als volgt uit:

```
worp 1: 3-4-3
worp 2: 1-1-6
worp 3: 2-2-5
worp 4: 3-2-3
...
```

'Gooi' nu net zolang met de drie dobbelstenen totdat je drie maal een 6 heb gegooid.

--

5.3 Opgaven loops en arrays

In deze video wordt uitgelegd hoe je een loop maakt waarmee je alle waardes van een array kan afdrukken

www.youtube.com

1 Opgave, terugtellen

Maak een PHP loop waarmee je alle getallen van 1000 tot en met 0 afdrukt op de volgende manier:

```
1000, 999, 998, 997, 996, 995, 994, 993, ..... 2, 1, 0
```

In plaats van de puntjes worden dus alle getallen afgedrukt.

Zet de PHP-code op je antwoordenblad.

2 Opgave array afdrukken

Druk het volgende array achtersevoren af.

```
$myArray = ['auto','fiets','brommer','bus','vliegtuig','trein'];
```

De output moet er als volgt uit zien:

```
trein, vliegtuig, bus, brommer, fiets, auto
```

Maak je code zo dat het bij elk array werkt dus ook als het meer of minder elementen zou hebben.

Zet de PHP-code op je antwoordenblad.

3 Opgave, oneven getallen

Maak een loop en druk alle oneven getallen tussen 0 en 100 af.

Dus de output moet worden:

```
1
3
5
7
9
11
13
..
..
97
99
```

Zet de PHP-code op je antwoordenblad.

4 Opgave, even getallen uit een array

Maak een loop en loop door het array. Druk alle getallen af uit het array $array die groter zijn dan 5.

```
$array=[3,4,5,8,9,12,13,6];
```

Zet de PHP-code op je antwoordenblad.

5 Opgave, piramide

Maak met behulp van een loop de volgende output:

```
1
2
3
4
5
6
7
8
9
```

Breid de code uit, zodat dit wordt afgedrukt. 1 wordt dus 1x afgedrukt, 2, 2X, 3 3X etc.

```
1
22
333
4444
55555
666666
7777777
88888888
999999999
```

Verander nu de code, zodat je het volgende afdrukt (let op dat dit 12 regels zijn).

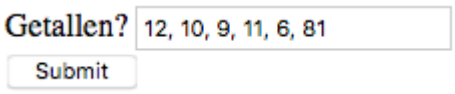
```
*
**
***
****
*****
******
*******
********
*********
**********
***********
************
```

Zet de PHP-code van de laatst gelukte opdracht op je antwoordenblad.

6 Opgave, gemiddelde

Maak een form waarbij je de gebruiker om een reeks getallen vraagt. Deze getallen kunnen worden ingevoerd als één regel waarbij de getallen met een komma worden gescheiden.

Bijvoorbeeld:



Met de volgende code kan je de getallen die je hebt ingevoerd in een array zetten:

```
$arrayGetallen = explode(",", $_GET['getallen']);
```

Maak code, zodat het volgende wordt uitgevoerd:

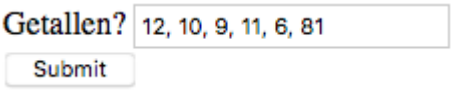
```
Aantal ingevoerde getallen: 6
Totaal van alle getallen: 129
Gemiddelde van alle getallen: 129
```

Zet de PHP-code op je antwoordenblad.

7 Opgave, sorteren

Maak een form waarbij je de gebruiker om een reeks getallen vraagt. Deze getallen kunnen worden ingevoerd als één regel waarbij de getallen met een komma worden gescheiden.

Bijvoorbeeld:



Met de volgende code kan je de getallen die je hebt ingevoerd in een array zetten:

```
$arrayGetallen = explode(",", $_GET['getallen']);
```

Met de functie `sort($array`) kun je een array sorteren. Gebruik deze functie om het array dat de gebruiker heeft te sorteren.

Druk het gesorteerde array af met behulp van een loop en gebruik daarbij de count() functie om te bepalen hoeveel elementen moeten worden afgedrukt.

Zet de PHP-code op je antwoordenblad.

8 Opgave tabel

Een tabel bestaat uit regels en kolommen. Stel je wilt een tabel maken met één regel en 10 kolommen, zoals:

```
<table border=1>
<tr>
<td> 1 </td>
   <td> 2 </td>
   ....
   <td> 10 </td>
</tr>
</table>
```

Zoals je ziet, wordt in elke kolom een getal afgedrukt.

a) Maak PHP-code die deze tabel maakt met behulp van een loop.
b) Breidt de code uit, zodat de tabel uit 12 identieke regels bestaat. Alles tussen <tr> en </tr> wordt dus 12 x afgedrukt.

Zet de code voor opgave (b) op je antwoordenblad.

9 Opgave, tafel van 13 in tabel

Maak PHP-code die de tafel van 13 netjes in een tabel afdrukt.

De output bestaat uit 5 kolommen:

1	X	13	=	13
2	X	13	=	26
3	X	13	=	78
...	...	...	...	...
10	X	13	=	130

Zet de PHP-code op je antwoordenblad.

6.1 Functies

In deze les gaan we keer uitleggen wat functies zijn. Dit onderwerp is behandeld bij Java Script en andere programmeer vakken en in PHP werken functies eigenlijk niet anders. We kijken nog een keer naar wat een functie doet:

> "*Je stopt er wat in en er komt wat uit*".

Aan de hand van filmpjes kijken we nog een keer naar de theorie en daarna is er een aantal opgaven om te oefenen.

Uitleg

Functies zijn stukjes code die een bepaalde functie hebben; je stop er wat in er gebeurt wat en er komt wat uit.

Functies worden gebruikt om stukjes code op te delen, te groeperen. Je deelt een groot probleem op in kleinere problemen en elk klein probleempje maak je een functie van. Soms zijn de kleinere problemen nog steeds te groot en deel je die ook weer op in nog kleinere stapjes. Door functies *moduleer* je je code, je geeft er structuur aan. In dit filmpje wordt dit verder teogelicht.

https://www.youtube.com/embed/XfnH3AEF5Z8

Op deze site staat ook nog een goede uitleg in het Nederlands over functies. Lees tot aan recursieve functies. Recursieve functies hoef je dus (nog) niet te weten.

Voorbeeld

Stel je koopt iets in via internet in China. Je moet dan invoerrechten betalen als het bedrag hoger is dan 22 euro. De invoerrechten bestaan uit 21% btw en inklaringskosten. PostNL rekent 13 euro. Als het bedrag hoger is dan 150 euro dan komt er nog eens 6% bij aan douanerecht.

We kunnen een functie maken om te berekenen wat de totale kosten van het ingevoerde product zouden worden.

```php
<?php

function importKosten($prijs) {
  $extra=0; // extra kosten initialiseren

  if ( $prijs > 22 ){
    $extra = $prijs * 0.21; // btw 21%
    $extra = $extra + 13  ; // inklaringskosten PostNL 13 euro
  }
  if ( $prijs > 150 ) {
    $extra = $prijs * 0.06; // Douanekosten 6%
  }

  return($prijs+$extra);
}

$itemPrijs=130;
echo "De prijs na invoeren uit China is ".importKosten($itemPrijs);

?>
```

Opgave 1

Pas de functie aan, zodat de prijs die de functie returned op twee decimalen wordt afgerond. De prijs dient altijd twee decimalen te bevatten (dus 12 euro wordt 12.00). Om dit op te lossen moet je zelf op zoek gaan naar een oplossing (dit is niet behandeld in de les).

Schrijf de regel(s) code die je hebt, verandert op je antwoordenblad.

Opgave 2

De bestellingen die je doen lopen niet meer via PostNL, maar via EMS. EMS rekent 17.50 aan inklaringskosten. Pas de code aan, zodat het nieuwe tarief van EMS wordt verwerkt.

Schrijf de regel(s) code die je hebt verandert op je antwoordenblad.

Opgave 3

(zet de inklaringskosten weer terug naar 13 euro)

Je hebt een array met prijzen:

```
$prijzen=[12,45,23,121,302,14,21];
```

Gebruik een loop en gebruik de functie die om van al deze prijzen de prijzen na invoer te berekenen. De output moet er als volgt uit zien:

```
De prijs van 12 na invoeren uit China is 12,00
De prijs van 45 na invoeren uit China is 67,45
De prijs van 23 na invoeren uit China is 40,83
De prijs van 121 na invoeren uit China is 159,41
De prijs van 302 na invoeren uit China is 320,12
De prijs van 14 na invoeren uit China is 14,00
De prijs van 21 na invoeren uit China is 21,00
```

Wat voor een loop heb je gebruikt? Zoek op hoe een foreach loop werkt en gebruik deze om het array te doorlopen.

Schrijf de loop inclusief de code die in de loop wordt gebruikt op je antwoordenblad.

Opgave 4

Je importeert spullen uit China en kan met de functie importKosten uitrekenen wat de producten jou kosten. Jij besluit de ingevoerde producten te verkopen via Martplaats en je wilt 20% winst maken.

Maak nu een *tweede functie* die de als *input* een prijs heeft en als *output* de prijs inclusief jouw winst. Dat is dus de prijs maal 1.2

Gebruik nu weer het array uit opgave 3 en druk de prijs na invoer en na inclusief (met) jouw winst. Je moet hierbij dus beide functies gebruiken!

De output gaat er dus als volgt uitzien:

```
De prijs van 12 na invoeren uit China is 14.40
De prijs van 45 na invoeren uit China is 80.94
De prijs van 23 na invoeren uit China is 49.00
De prijs van 121 na invoeren uit China is 191.29
De prijs van 302 na invoeren uit China is 384.14
De prijs van 14 na invoeren uit China is 16.80
De prijs van 21 na invoeren uit China is 25.20
```

Schrijf de complete code op je antwoordenblad.

Opgave 5

Kijk naar dit filmpje en beantwoord daarna de vragen.

https://www.youtube.com/embed/7bXfUBjLR5I

Een variable $bedrag wordt meegegeven aan een functie.

```
function myFunctie($bedrag) {
$bedrag=0;
    return($bedrag);
}

$bedrag=100;
$resultaat=myFunctie($bedrag);

echo $bedrag;
```

a) Wat wordt er in regel 9 afgedrukt?

b) Je verandert regel 6 in $bedrag=200, wat wordt er dan in regel 9 afgedrukt?

c) Je verandert regel 2 in $bedrag=12, wat wordt er dan in regel 9 afgedrukt?

d) Zie je een patroon in de vragen a,b, en c? Zo ja wat is het patroon, of wat kun je zeggen over de variabelen op regel 2 en op regel 6? Zijn die hetzelfde?

Schrijf de antwoorden op je antwoordenblad.

Opgave 6

Pas nu de functie aan, zodat je als input een *array* met prijzen meegeeft. De output moet ook een *array* met nieuwe prijzen zijn.

Dus de aanroep wordt:

```
$prijzenNaImport = importKosten($prijzen);
```

Druk dan vervolgens de prijzen af en gebruik daarvoor het array $prijzenNaImport. Maak weer gebruik van de foreach loop. De output moet er hetzelfde uitzien als bij opgave 3.

Dus:

De input van de functie importKosten is:

```
$prijzen=[12,45,23,121,302,14,21];
```

en de output is:

```
$prijzenNaImport['14.4','80.94','49','191.29','384.14','16.8','25.2',]
```

Dus je roept de functie aan als volgt $prijzenNaImport=importKosten($prijzen);

Schrijf de complete code op je antwoordenblad.

Het volgende filmpje kan ook helpen bij deze opgave:
https://www.youtube.com/watch?v=INHM0S_8GHA

-

6.2 Functies en parameters

In de vorige lessen hebben we geoefend met functies. In deze les gaan we functies nog beter leren en we gaan vooral kijken naar de functie parameters en de return values. In deze les gaan we een functie maken die een array als (input) parameter krijgt en die ook een return als return waarde terug geeft.

Allereerst kijk naar dit filmpje:

https://www.youtube.com/embed/INHM0S_8GHA

In dit filmpje wordt uitgelegd en voorgedaan hoe we een functie maken waarin we een array als input parameter geven en hoe we een array als return value terug geven.

Opgave 1

```
<?php

$getallen=[3,5,2,7,8,5,6,9,2,1,8];

$even=alleenOnevengetallen($getallen);

echo "<pre>";
print_r($even);

function alleenOnevengetallen($array) {
  ...
  ...
  ...
  ...
  ...
  ...
  return($output);
 }
?>
```

Kijk goed naar het voorbeeld in het filmpje en maak dan de code af. Maak de functie alleenOnevengetallen. Deze functie krijgt als input een array mee met getallen en de return value is een array met oneven getallen.

Kijk goed naar het voorbeeld in het filmpje.

Opgave 2

(Als je 6 punten voor les 6.4 had dan mag je deze opgave overslaan. Zet dan op je antwoordenblad "Ik had 6 punten voor 6.4".)

We gaan nu nog een keer opgave 6 van les 6.4 maken.

We hadden deze functie:

```
<?php

function importKosten($prijs) {
  $extra=0; // extra kosten initialiseren

  if ( $prijs > 22 ){
    $extra = $prijs * 0.21; // btw 21%
    $extra = $extra + 13  ; // inklaringskosten PostNL 13 euro
  }
  if ( $prijs > 150 ) {
    $extra = $prijs * 0.06; // Douanekosten 6%
  }

  return($prijs+$extra);
}



$prijzen=[12,45,23,121,302,14,21];

$prijzenNaImport=importKosten($prijzen);

echo "<pre>";
print_r($prijzenNaImport);

?>
```

Verander deze functie, zodat die als parameter een array mee krijgt en als return value ook een array.

--

6.3 functies oefeningen

Opgave 1

Kijk naar het filmpje: https://www.youtube.com/watch?v=INHM0S_8GHA

En maak daarna een functie `onvoldoendes($PHPCijfers);` die alle onvoldoendes afdrukt uit het onderstaande array.

```
$PHPCijfers=[4,5,4,5,6,6,5,8,7,6,4,8];
```

Dus de output van de functie wordt:

```
4,5,4,5,5,4
```

Schrijf de functie op je antwoordenblad (.php file).

Opgave 2

Maak nu een functie die de onvoldoendes niet afdrukt maar terug geeft als een array.

Dus de functie `$resulstaat=onvoldoendes($PHPCijfers);` geeft een array $resultaat terug.

Gebruik de volgende code om je $resultaat array af te drukken:

```
echo "<pre>";
print_r($resultaat);
```

Schrijf de functie op je antwoordenblad (.php file).

Opgave 3

Het is belangrijk om goed te begrijpen wat lokale scoping van variabele is. Dit werk in alle computertalen (min of meer) hetzelfde.

In dit filmpje (Engels) wordt uitgelegd wat lokale scoping is: https://www.youtube.com/watch?v=5vJwM-_YHfk

In dit filmpje wordt het door mij in het Nederlands uitgelegd: https://youtu.be/hp8-MrfuwoI

De volgende vraag gaat over lokale scoping. De functie berekent de omtrek van de cirkel als de straal wordt gegeven. De straal is zeg maar de lengte van je spaak van je fiets en de omtrek is de afstand die je aflegt als je wiel precies één keer ronddraait.

Voer de onderstaande code uit.

```
<?php
function omtrek($straal) {
  $omtrek=$straal*2*pi();
  return($omtrek);
}

$straal=10;
$omtrek($straal);
echo $omtrek;
?>
```

a) waarom krijg je een foutmelding op regel 9?

b) pas de code aan, zodat deze werkt. Voeg *geen* regels toe, pas alleen de bestaande regel(s) aan. Zet de werkende code op je (.php) anwoordenblad.

Opgave 4

Schrijf alle variabele met een *lokale scope* van onderstaande code op je antwoordenblad. In de video's die bij opgave 3 (hierboven) genoemd worden, wordt dit uitgelegd.

Dus welke variabelen uit de onderstaande code hebben een *lokale scope*?

```
<?php
function bedragmetWinst($bedrag) {
  return($bedrag * 1.5);
}

$prijs=12;
$verkoopPrijs=bedragmetWinst($prijs);
```

```
echo $verkoopPrijs;


?>
```

Opgave 5

Stel je hebt twee functie functieA() en functieB(). In functieA() heb je een variabele $btw en $bedrag en je wilt deze variabele gebruiken in functieB(). Hoe kun je dit doen?

Om je een beetje op weg te helpen, probeer antwoord te geven op de volgende vragen:

(tip 1) Op welke manier kun je informatie (variabelen) in en uit een functie krijgen? Dit wordt beschreven in https://youtu.be/XfnH3AEF5Z8 deze video staat aan het begin van les 6.4

(tip 2) Nu weet je hoe je met één variabele informatie van functionA naar functionB kan krijgen. Blijven er twee vragen over:

(tip 3) Hoe geef je twee of meer variabelen mee als je ene functie aanroept. Dit is vrij eenvoudig; zoek dit op.

(tip 4) Je kunt met een return() maar één variabele terug geven. *Maar een array is ook één variabele. En in een array zitten doorgaans meerdere waarden.*

Als je er echt niet uit komt beantwoord dan de vraag met één variabele. Dus in plaats van $btw en $bedrag heb je nu maar één variabele $btw die je in functionA hebt en die je in functionB wilt gebruiken.

Geef een code voorbeeld door onderstaande code uit te breiden waarbij je de variabele $btw (en het liefst ok $bedrag) kunt gebruiken in functionB.

```
<?php


function functionA(...) {
    $btw=21;
    $bedrag=119;
    return(...);
 }


 function functionB(...) {
echo "De btw is $btw en het bedrag is $bedrag";
 }
```

```
...
...
functionB(...); // er wordt afgedrukt
                // De btw is 21 en het bedrag is 119



?>
```

--

6.4 Generieke Functies

Let op: voor deze opgave moet de opgave Controle Bankrekeningnummer worden uitgevoerd.

We hebben al geoefend met functies, maar we gaan nog een keer theoretisch kijken naar functies; wanneer gebruik je functies, wat zij de voordelen en waaraan voldoet een goede functie?

We hebben het her over functies, maar alles wat voor functies geldt ook voor methods in een class.

Wat is een (goede) functies

Een functie is een stukje code dat **hergebruikt** kan worden. Dat betekent dat functies generiek zijn. Dat betekent dat een goede functies een toepassing heeft die algemeen toepasbaar is.

Als je in grotere teams samenwerkt aan een groot project dan kun je verschillende developers laten werken aan verschillende functies. Als je de input en output goed beschrijft, dan hoef je niet precies te weten hoe de functie (intern) werkt. Je kunt het gewoon als een stukje code gebruiken. Je moet dan wel zorgen dat de functie algemeen toepasbaar is; generiek dus.

Van een functie is het heel belangrijk wat de input en de output is. Als je de input (parameters) en de output (return value) goed beschrijft, dan hoef je niet te weten hoe de functie (intern) werkt om hem te kunnen gebruiken.

Voorbeeld

Laten we eens kijken naar de (ingebouwde) PHP-functie rand().

```
rand ( int $min , int $max ) : int


Parameters
$min: De minimale waarde
$max: De maximale waarde


Return Values
Een willekeurig getal tussen min en max.
```

(bron: https://www.php.net/manual/en/function.rand.php)

Zoals je kan zien geef je twee waardes mee, beide een integer. De eerste integer bepaald de minimale waarde, de tweede de maximale. En de return value is een willekeurig getal (ook int)

tussen de min en max waarde.

Deze functie is generiek want die kan voor heel veel doeleinden worden gebruikt. Met de parameters kun je als het ware het gedrag van de functie sturen en bepaal je wat de functie doet. Wil je een dobbelsteen simuleren, of wil je een getal tussen 0 en 100, of wil je kop/munt spelen, dat kan allemaal met deze functie.

Een voorbeeld van een minder generieke functie zou zijn:

```
rand () : int

Parameters
Geen

Return Values
Een willekeurig getal tussen 1 en 10.
```

Om met deze functie een dobbelsteen te simuleren wordt een stuk lastiger.

Een voorbeeld van een nog minder generieke functie zou zijn:

```
rand ()

Parameters
Geen

Return Values
Geen

De functie print een getal tussen 0 en 10
```

In dit voorbeeld is er geen parameter en geen return value. De functie print (echo) een getal tussen 0 en 10 want je kunt bijvoorbeeld niet meer rekenen met de uitkomst van deze functie.

Hoe maken we functies generiek?

Daar zijn geen kei-harde regels voor, maar er is wel een aantal zaken waar je aan kan denken:

1. Is de output (return value) van een functie altijd hetzelfde?

Als dat zo is, dan is de functie waarschijnlijk niet heel erg generiek.

2. Zit er een echo/print in een functie?

Met een echo ligt het formaat van de output vast, bijvoorbeeld de taal. Ook kun je met het resultaat niet meer verder rekenen. Een echo kan er dus ook voor zorgen dat een functie minder generiek wordt.

3. Is de output generiek?

Is de output algemeen toepasbaar? Stel je hebt een functie die het gemiddelde berekend en je return ziet er als volgt uit:

```
return "Gemiddelde is: ".$som/$aantal;
```

De return value is nu taalafhankelijk en met de output is het lastig verder rekenen. Stel je wilt het gemiddelde vermenigvuldigen met 2, hoe zou je dat dan doen? Het is generieker om alleen een getal te returnen.

Sommige functies hebben als output een ja of nee. Bijvoorbeeld een functie die terug geeft of je gemiddelde een voldoende is. Als dat het geval is gebruik je bij voorkeur als return value true of false.

Je kunt dan bij het aanroepen van de functie gelijk een if statement maken:

```
if ( isgemiddeldeVoldoende($array) ) {
echo "Je hebt een voldoende!";
} else ...
```

4. Is de output afhankelijk van de input?

Bepaal je met de input van de functie de output? Stuur je als het ware wat de functie doet met input? Als dat zo is dan maakt dat de functie meer generiek.

Een generieke functie kan dus op meerdere plaatsen in je code worden gebruikt. Je kunt ook een library maken met functies en die dan met anderen delen. Dat is wel erg generiek en als dat gebeurt dan wil je ook je functie robuust is en niet zomaar verkeerde resultaten geeft.

Hoe maken we een functie robuust?

4. Wordt er gecontroleerd of de parameter een goede waarde hebben?

Een generieke functie heeft parameters. Er zijn gevallen dat niet alle parameters tot een goed resultaat leiden. Als je bijvoorbeeld en functie hebt die een getal afrond op $n decimalen dan wil je

dat $n groter of gelijk aan nul is. Dit zorgt ervoor dat de gebruiker van de functie fouten kan maken en dat die dan netjes worden afgevangen of in ieder geval niet leiden tot andere fouten.

Video

In dit voorbeeld hieronder wordt nog een keer gekeken naar de functie waarbij een bankrekeningnummer wordt gecontroleerd. De functie wordt generiek gemaakt door de return value aan te passen.

https://www.youtube.com/embed/Z7OKbmQYPgY

Voorbeelden

Met deze functie 'gooien' we met twee dobbelstenen.

```
function dobbel() {
    $steen1 = rand(1,6);
    $steen2 = rand(1,6);
    echo $steen1 + $steen2;
}
```

Kijken we naar de 3 richtlijnen dan zien we dat er een echo in de functie zit. We kunnen de functie dus generieker maken.

```
function dobbel() {
    $steen1 = rand(1,6);
    $steen2 = rand(1,6);
    return $steen1 + $steen2;
}
```

Nu is de functie al generieker, maar het kan nog beter want wat als we met 3 of 4 dobbelstenen zouden willen gooien; dat kan nu niet?

```
function dobbel($aantal) {
    $totaal = 0;
    for($i=0; $i<$aantal; $i++){
        $gooi=rand(1,6);
        $totaal = $totaal + $gooi;
```

```
    }
    return $totaal;
}
```

Nu hebben we een functie waarbij de gebruiker kan aangeven met hoeveel dobbelstenen hij gaat gooien. Een goede functie zorgt ervoor dat er niet fout kan gaan. Een goede functie is robuust. Is de bovenstaande functie robuust? Wat gebeurt er als je bijvoorbeeld -1 als parameter meegeeft? De functie loopt niet vast of zo en hij returned 0. Dat is (toevallig) goed, een 0 of een -1 als return value duidt soms op een fout.

Opgave 1

De function stringGelijk($string1, $string2) vergelijkt twee strings onafhankelijk van het hoofdletter gebruik. Dus "Appel" is gelijk aan "appel".

Is de functie generiek? Kun je hem misschien generieker maken? Pas de code aan.

```
<?php

function stringGelijk($string1, $string2) {

  if ( strtoupper($string1) == strtoupper($string2) ) {
    return "Gelijk";
  } else {
    return "Ongelijk";
  }

}

echo stringGelijk("Hallo", "hallo");
```

Opgave 2

De function berekenGemiddelde() berekend het gemiddeld van een array.

Pas de code aan, zodat die generieker wordt.

```
function berekenGemiddelde() {
$aantal = 0;
    $som = 0;
```

```
$array = [5,4,7,6,5,6,5,7,6,7,8,3,4,7,6,7];
    for($i=0; $i<count($array); $i++) {
    $som = $som + $array[$i];
        $aantal++;
    }
    return $som/$aantal;
}
```

--

6.5 Dag Functie

In deze les gaan wee nog een paar functies maken. Houd daarbij rekening met de vorige les waarin is beschreven dat functies generiek dienen te zijn.

Opgave 1

Maak een functie die aan de hand van een nummer een dag van de week terug geeft.

```
dagNaam( int $dagNummer ) : string
```

De functie krijgt een getal mee en maakt daar een weekdag (string) van, bijvoorbeeld:

```
echo dagNaam(1) // dit drukt ma af (maandag)
echo dagNaam(2) // dit drukt di af (dinsdag)
echo dagNaam(3) // dit drukt wo af (woensdag)
..
echo dagNaam(7) // dit drukt zo af (zondag)
```

Dus afhankelijk van het nummer dat je als parameter meegeeft aan de functie wordt de return waarde ma, di, wo ,do, vr, za of zo.

Denk ook aan fout-afvang (punt 4 vorige les).

Opgave 2

Maak een functie die aan de hand van een string een dagnummer terug geeft. Dit is dus eigenlijk het omgekeerde van de vorige functie.

```
dagNummer( string $dagNaam ) : int
```

De functie krijgt een dagnaam mee (ma, di, wo, do, vr, za of zo) en maakt daar een dag nummer van.

Bijvoorbeeld:

```
echo dagNummer("ma") // dit drukt 1 af
echo dagNummer("di") // dit drukt 2 af
```

```
..
echo dagNummer("zo") // dit drukt 7 af
```

Denk ook aan fout-afvang (punt 4 vorige les).

Opgave 3

Zorg dat je beide functies van opgave 1 en 2 hebt en voeg deze code toe.

```
$nummer=rand(1,7);


echo $nummer;
echo "<br>";
echo dagNummer(dagNaam($nummer);
```

Wat kun je zeggen over de variabele $nummer en het getal wat wordt afgedrukt op regel 5?

Wat gebeurt er, leg in je eigen woorden uit wat deze code doet.

Opgave 4

Maak een functie die bepaald wat het morgen is.

```
morgen(string $vandaag) : string
```

De functie krijg een dag mee (ma, di, wo, do, vr, za, zo) en bepaald dan wat het morgen is. De output is weer een string. Bijvoorbeeld:

```
$dag=morgen("do");
echo $dag         // er wordt vr afgedrukt.
echo morgen($dag) // er wordt za afgedrukt ($dag is immers vr)
```

Opgave 5

We gaan de functie uit opgave 4 generieker maken. Stel dat je bijvoorbeeld niet morgen maar overmorgen wil bepalen, of de dag na overmorgen.

We gaan de functie welkeDag maken:

```
welkeDag( string $dag, int $erbij) : string
```

De functie krijgt weer een dag mee (ma, di, wo, do ,vr, za, zo) en de tweede parameter $erbij geeft aan hoeveel dagen je erbij wilt tellen. Voor morgen moet $erbij dus 1 zijn. De return waarde is weer een dag (ma, di, wo, do, vr, za, zo).

Voorbeeld:

```
echo welkeDag("ma", 1) // dit drukt di af omdat di 1 dag na ma is.
echo welkeDag("di", 2)  // dit drukt do af omdat do 2 dagen na di is.
eche welkeDag("zo", 2) // dit drukt di af omdat di 2 dagen zo is.
```

--

7 Praktijkopdracht - Simulatie

Praktijkopdrachten worden individueel uitgevoerd. Je moet dus je eigen code maken.
Praktijkopdrachten worden apart beoordeeld (los van het huiswerk).

Inleiding

Stel je wordt gevraagd of je mee wilt doen met een spel. Het voorstel is dat je voor 1 euro met drie dobbelstenen mag gooien. Als je 18 gooit, dat zijn dus drie zessen, dan win je 100 euro. Is het slim om dit spel te spelen? Denk je dat je winst maakt?

In deze praktijkopdracht leer je om uit te vinden wat de kans is om 18 te gooien. Als je weet hoelang het gemiddeld duurt voordat je 18 gooit met drie dobbelstenen dan weet je of dit een lucratief (=winstgevend) spel is.

In deze les komt alles aan de orde wat we tot nu hebben geleerd. Er komen ook functies aan de orde. Deze heb je bij de andere vakken (Java Script, Java) al gehad, maar er zal ook nog kort worden uitgelegd hoe een functie in PHP werken.

Praktijkopdracht - Simulatie

In opdracht 2b van een vorige les (PHP-1, 6.2) gooi je net zo lang met 3 dobbelstenen totdat je drie maal een 6 heb gegooid. Hoe groot is nu de kans om 3x een 6 te gooien? Vanuit de wiskunde kun je dit berekenen, maar als programmeur kun je dit ook bepalen met een simulatie.

We gaan nu via een simulatie bepalen wat de kans is om 3 x 6 te gooien. Omdat te doen gaan we 1000 keer net zolang gooien totdat we 3 x 6 hebben gegooid. We hebben dan 1000 uitkomsten. Elke uitkomst geeft aan hoe lang het duurde om 3 x 6 te gooien.

Om dit probleem aan te pakken delen we het op in kleine stapjes.

Functies

We gaan gebruik maken van functies.

Functies werken hetzelfde als in andere talen en in de JavaScript lessen hebben we het hier over gehad. Functies hebben we nog niet gehad in PHP, maar in het volgende filmpje wordt uitgelegd

hoe wat een functie ook alweer is en hoe je die in PHP moet gebruiken: Uitleg over functions.

Stap 1 - maak functie throwDice($numberOfDice)

Maak een functie die je aanroept met `throwDice($numberOfDice)` De functie krijgt het aantal dobbelstenen mee dat moet worden gegooid en de functie returned het totaal aantal ogen dat je hebt gegooid.

Stap 2 - voer de functie net zolang uit totdat de uitkomst 18 is

Drie maal zes geeft een uitkomst van 18. Voer de functie dus net zolang uit totdat de uitkomst 18 is. Tel hoeveel keer je de functie hebt aangeroepen.

Maak ook hiervan een functie. Noem de functie `waitForResult($result)`. De functie krijgt nu 18 mee als parameter en returned hoe vaak het heeft moeten 'gooien' om tot het resultaat te komen. Vanuit deze functie `waitForResult` roepen we dus de functie `throwDice` aan.

Dus bijvoorbeeld:

Je roept waitForResult(18) aan en je krijgt 101 terug. Dat betekent dat de computer 101 x heeft gegooid en dat er toen pas 18 was gegooid.

Of je roeptwaitForResult(7) en je krijgt 12 terug. Dat betekent dat de computer 12 keer heeft gegooid en dat er toen 7 was gegooid.

Stap 3 - voer de simulatie tig-keer uit.

In stap 2 voeren we de simulatie één keer uit. We wachten één keer op een bepaalde uitkomst bij het werpen van de dobbelstenen.

De uitkomst hangt af van geluk. Het kan zijn dat we gelijk in één keer 18 gooien, maar het kan ook meer dan 100 worden duren. Als we nu heel vaak gooien en het gemiddelde berekenen dan filteren we de geluksfactor er een beetje uit. We gaan in deze stap dan ook 1000x de functie waitForResult(18) uitvoeren en bepalen het gemiddelde van de return waarde van deze functie.

De functie geeft terug hoeveel worpen het kostte om 18 te gooien. Als we dat 1000 keer doen en we bepalen het gemiddelde van al deze aantal worpen dan weten we hoelang je gemiddeld moet wachten op een uitkomst van 18 als je met drie dobbelstenen gooit.

Stap 4 - nog een stapje verder

En wat is het antwoord? Met je huidige simulatieprogramma kun je ook heel eenvoudig bepalen hoe lang het duurt om bijvoorbeeld 11 te gooien of 10, of 9,....

Kun je van alle mogelijke uitkomsten 3 tot en met 18 nu in een loop bepalen hoe lang het gemiddeld duurt om dit aantal ogen met drie dobbelstenen te gooien?

De output ziet er als volgt uit:

```
To throw a total of 3 with 3 dice, it took on average ... throws.
To throw a total of 4 with 3 dice, it took on average ... throws.
To throw a total of 5 with 3 dice, it took on average ... throws.
To throw a total of 6 with 3 dice, it took on average ... throws.
To throw a total of 7 with 3 dice, it took on average ... throws.
To throw a total of 8 with 3 dice, it took on average ... throws.
To throw a total of 9 with 3 dice, it took on average ... throws.
To throw a total of 10 with 3 dice, it took on average ... throws.
To throw a total of 11 with 3 dice, it took on average ... throws.
To throw a total of 12 with 3 dice, it took on average ... throws.
To throw a total of 13 with 3 dice, it took on average ... throws.
To throw a total of 14 with 3 dice, it took on average ... throws.
To throw a total of 15 with 3 dice, it took on average ... throws.
To throw a total of 16 with 3 dice, it took on average ... throws.
To throw a total of 17 with 3 dice, it took on average ... throws.
To throw a total of 18 with 3 dice, it took on average ... throws.
```

Voor de duidelijkheid de vertaling in het Nederlands luidt:

> Om in totaal 18 te gooien met 3 dobbelstenen, duurde het gemiddeld ... worpen.

--

8 Associative Arrays

Arrays zijn er in meerdere soorten. In deze les gaan we het hebben over associatieve arrays.

We hebben het al uitgebreid gehad over 'gewone' of 'eenvoudige' arrays in PHP. Deze 'gewone' arrays heten ook wel een indexed array omdat je de elementen via de index (0,1,2,3,4,5.....) kunt opzoeken.

Er bestaat nog een type array: het assiociative array (in Python heet dit een 'Dictionary'). Het wordt ook wel een key-value array genoemd. Het assiociative array lijkt veel op JSON. JSON is een bestandsformaat waarin je data kan opslaan. Je zou het een soort database kunnen noemen.

	Andere namen	Hoe benader je een element?	Voorbeeld
Indexed Array	gewoon array, 1 dimensionaal array, eenvoudig array	via de index, die is altijd 0,1,2,3,4....	$myArray[13]
Assiociative Array	key-value array, Dictionary (Python), hash table	via de key, dit moet een unieke string zijn waarvan je zelf de waarde bepaald.	$myArray['james'];

In het onderstaande filmpje wordt eerst nog een keer herhaald hoe een 'gewoon' array ook alweer werkt en vanaf 3'00'' in het filmpje wordt uitgelegd hoe een associative array in PHP werkt.

https://www.youtube.com/embed/5lJLecI0BTA

In dit filmpje wordt aan de hand van een ander voorbeeld hetzelfde uitgelegd in het Engels.

Indexed Array

Bij een indexed array heb je als index een nummer. Elk array element heeft een nummer, te beginnen met 0.

```
$age[0] = "35";
$age[1] = "37";
$age[2] = "43";
```

Je kunt dit array ook op een ander manier maken, weet je nog?

```
$age = array(35,37,43);
```

Het bovenstaande array heeft drie elementen: $age[0], $age[1] en age{2]. Het getal tussen de vierkante haken heet de index.

Associative Array

Stel dat je het array dat hierboven staat wilt gebruiken om de leeftijd van personen vast te leggen. $age[0] is de leeftijd van Peter, $age[1] is de leeftijd van Ben en $age[2] is de leeftijd van Joe.

```
$age[0] = "35";
$age[1] = "37";
$age[2] = "43";
```

Dit is wel wat onhandig want je moet nu onthouden dat de 0 bij Peter hoort, de 1 bij Ben en de 2 bij Joe. Dat kan anders!

Associative Array aanmaken

Met een assiociative array kun je de index vervangen door een key. En de key bestaat uit een string.

Bijvoorbeeld:

```
$age['Peter'] = "35";
$age['Ben'] = "37";
$age['Joe'] = "43";
```

Je kunt het array ook op een andere manier maken

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
```

Associative Uitlezen

```
echo $age['Ben'];
```

In plaats van de index bij een indexed array, gebruik je nu de key die je zelf hebt gemaakt.

Associative Veranderen

```
$age['Ben']=38;
```

Dit gaat ook hetzelfde als bij een indexed array. Hier gebruik je de key die jezelf hebt gemaakt voor het array.

Associative Uitbreiden

```
$age['Mirna']=28;
```

Je kunt een assiociative array eenvoudig uitbreiden door een nieuwe key te gebruiken. Je moet wel zeker weten dat de key nieuw is anders overschrijf je de oude waarde. Als in het voorbeeld $age['Mirna'] al bestond dan heb je de oude waarde nu overschreven met de nieuwe waarde. Een key kan dus maar één keer voorkomen.

Opgave 1, theorievragen

a) In het (Nederlandse) filmpje over associative arrays staat 'ma' => 8 waar staat de 'ma' voor en waar staat de 8 voor? Dus wat betekent de 'ma' (Maarten, of wat?) en wat is de 8; 8 peren, 8 kilo, 8-wat?

b) Waaruit bestaat een index bij een index array; uit een string of een getal?

c) Waaruit bestaat een key bij een assiociative array; uit een string of uit een getal?

d) Stel een indexed array bestaat uit drie elementen. Wat zijn de indexen die dan worden gebruikt in dit array?

e) Kan dezelfde key bij een assiociative array meer dan één keer voorkomen?

f) Kan de key bij een assiociative array uit een getal zijn?

Opgave 2

Op deze pagina staan van alle maanden van het jaar de normale gemiddelde temperatuur in Nederland vermeld. Zo kun je zien dat januari de koudste maand is met gemiddeld 3.1 graden en de warmste maand is juli met 17.9 graden.

a) Maar een assiociative array waarin van alle maanden van het jaar (januari, februari, etc.) de normale gemiddelde temperatuur staan. Je hoeft alleen het array te maken.

b) Gebruik het assiociative array dat je hebt gemaakt en druk de temperatuur af van de maand april.

Opgave 3

Begin met de volgende code

```
<?php
$places = array(
"Japan" => "Tokyo",
   "Mexico" => "Mexico City",
   "USA" => "New York City",
   "India" => "Mumbai",
   "Korea" => "Seoul",
   "China" => "Shanghai",
   "Nigeria" => "Lagos",
   "Argentina" => "Buenos Aires",
   "Egypt" => "Cairo",
   "UK" => "London");
?>
```

- Maak code die het volgende afdrukt:
  de hoofdstad van China is Shanghai
  Hierbij moet de naam Shanghai worden opgezocht uit het array. Je code mag de naam Shanghai dus niet bevatten.

- Bestudeer: https://www.w3schools.com/php/php_arrays_associative.asp en zoek uit hoe je een associative array kunt uitprinten met een loop. Maak dan code die met het array van hierboven het volgende uitprint. Gebruik daarvoor een loop.

```
De hoofdstad van Japan is Tokyo
De hoofdstad van Mexico is Mexico City
De hoofdstad van USA is New York City
De hoofdstad van India is Mumbai
De hoofdstad van Korea is Seoul
De hoofdstad van China is Shanghai
De hoofdstad van Nigeria is Lagos
De hoofdstad van Argentina is Buenos Aires
De hoofdstad van Egypt is Cairo
De hoofdstad van UK is London
```

Opgave 4

Gebruik het array dat je hebt gemaakt bij opgave 2a en druk de temperatuur van alle maanden af. Gebruik hiervoor een loop en zorg ervoor dat de output er als volgt uit ziet.

```
De normale gemiddelde temperatuur van januari in Nederland is 3.1 graden.
De normale gemiddelde temperatuur van februari in Nederland is 3.3 graden.
De normale gemiddelde temperatuur van maart in Nederland is 6.2 graden.
De normale gemiddelde temperatuur van april in Nederland is 9.2 graden.
De normale gemiddelde temperatuur van mei in Nederland is 13.1 graden.
De normale gemiddelde temperatuur van juni in Nederland is 15.6 graden.
De normale gemiddelde temperatuur van juli in Nederland is 17.9 graden.
De normale gemiddelde temperatuur van augustus in Nederland is 17.5 graden.
De normale gemiddelde temperatuur van september in Nederland is 14.5 graden.
De normale gemiddelde temperatuur van oktober in Nederland is 10.7 graden.
De normale gemiddelde temperatuur van november in Nederland is 6.7 graden.
De normale gemiddelde temperatuur van december in Nederland is 3.17graden.
```

--

9 Hoofdstedenspel

Praktijkopdrachten worden individueel uitgevoerd. Je moet dus je eigen code maken.
Praktijkopdrachten worden apart beoordeeld (los van het huiswerk).

We gaan een programma maken waarmee je de hoofdsteden van Europa kunt leren. We gaan in deze les oefenen met onder meer html forms, loops, arrays en assiociative arrays. We leren ook een groot probleem op te delen in kleinere deel-problemen.

Doel

Het uiteindelijke doel is om een programma te maken dat alle Europese landen en hoofdsteden kent. Het programma kiest een willekeurig land en vraagt aan jou de hoofdstad van dat land te noemen. Je kunt kiezen uit een lange lijst van hoofdsteden die alfabetisch zijn gesorteerd.

Een uitgewerkt voorbeeld is te vinden op: http://www.softwaredeveloper.ovh/max/hoofdsteden/

Plan van aanpak - stapjes

Bij een dergelijk probleem dienen we het probleem in kleinere (deel)probleempjes op te delen. Dit kan op verschillende manieren. Zo lees je in de opdracht dat je een willekeurig land moet kiezen. We zullen dus moeten onderzoeken hoe we een willekeurig item uit een lijst kunnen kiezen. We zien ook dat we lijst moeten maken waaruit we een hoofdstad moeten kiezen. We moeten dus uitzoeken hoe we zo'n lijst moeten maken. We zien ook dat we een lijst moeten sorteren. Zo zijn er dus allemaal deelproblemen te definiëren. Deze opdracht is opgedeeld in 6 kleinere deelprobleempjes. Los deze problemen ieder afzonderlijk op en kom zo stapje voor stapje tot de uiteindelijke oplossing. Let op als je een stap niet in één keer kunt oplossen, kun je dit stapje zelf ook weer opdelen in kleinere stapjes.

Stap 1 - form met drop down

Maak een form in HTML waarbij je een drop down maakt.
Check deze pagina en probeer te begrijpen hoe het werkt:

https://www.w3schools.com/html/tryit.asp?filename=tryhtml_elem_select

```
<form action="/action_page.php">
  <select name="cars">
    <option value="volvo">Volvo</option>
```

```
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
 </select>
 <br><br>
 <input type="submit">
</form>
```

Maak nu zelf een form met PHP waarbij je de waarden uit het array $cars haalt.

```
array=$cars('Volvo','Saab','Fiat','Audio');
```

De regels 3,4,5 en 6 uit het bovenstaande voorbeeld worden dus vervangen door PHP code.

Stap 2 - random array element selecteren

We gaan het array iets uitbreiden.

```
array=$cars('Volvo','Saab','Fiat','Audio','BMW','Porsch','Hyundai','Opel','Kia','Tesla','VW');
```

Maak nu een PHP programma die één willekeurig automerk uit het array oppikt.

Dus je maakt een programma dat

`echo $cars[$random];`

uitvoert.

Zoek zelf uit hoe de rand() werkt waarmee je een random (willekeurig) getal kan berekenen.

Als eind resultaat heb je een php programmaatje dat elke keer als je een reload uitvoert een ander automerk laat zien.

Stap 3 - willekeurig land met hoofdstad

Gebruik het array zoals hieronder is aangegeven.

```
$ceu = array(
"Italy"=>"Rome", "Luxembourg"=>"Luxembourg", "Belgium"=> "Brussels", "Denmark"=>"Copenhagen",
"Finland"=>"Helsinki", "France" => "Paris", "Slovakia"=>"Bratislava", "Slovenia"=>"Ljubljana",
"Germany" => "Berlin", "Greece" => "Athens", "Ireland"=>"Dublin", "Netherlands"=>"Amsterdam",
"Portugal"=>"Lisbon", "Spain"=>"Madrid", "Sweden"=>"Stockholm", "United Kingdom"=>"London",
"Cyprus"=>"Nicosia", "Lithuania"=>"Vilnius", "Czech Republic"=>"Prague", "Estonia"=>"Tallin",
```

```
"Hungary"=>"Budapest", "Latvia"=>"Riga", "Malta"=>"Valetta", "Austria" => "Vienna",
"Poland"=>"Warsaw") ;
```

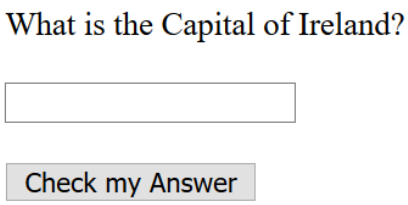
Maak een php programma dat een willekeurig land laat zien en dat dan het volgende afdrukt:

> The Capital of Finland is Helsinki

Elke keer als je de pagina ververst zie je dus een ander land.

Stap 4 - Hoofdstad raden

Maak nu een programma dat een willekeurig land uit het array landen $ceu selecteert en de gebruiker via een form vraagt:



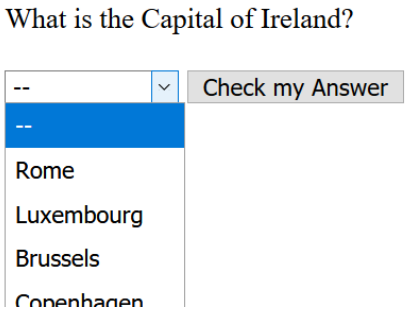
Controleer het antwoord en toon 'Correct' of 'Incorrect' afhankelijk van of het gegeven antwoord juist is.

Stap 5 - drop down

We gaan nu ons programma uit de vorige stap uitbreiden.

Maak een form dat een willekeurig land uit het array pikt. Stel je kiest 'Sweden'. Laat dan een form zien die vraagt wat de hoofdstad van Sweden is.

Zet alle hoofdsteden in een *drop down list* zodat de gebruiker een hoofdstad kan uitkiezen.



De gebruiker kiest dan een hoofdstad en als je op de knop *check my answer* drukt dan bepaald het PHP programma of het goed is.

Het programma antwoord:

Indien het antwoord juist is:

```
The Capital of Sweden is Stockholm, your answer was correct.
```

Indien het antwoord niet goed is:

```
The Capital of Sweden is Stockholm, your answer Riga was wrong.
```

Stap 6 - sorteren

Als laatste stap zorgen we ervoor dat de drop down lijst met hoofdsteden wordt gesorteerd.

Hiervoor zijn diverse strategieën te bedenken. Lees en bestudeer de volgende pagina en kies jouw strategie.

https://www.bitdegree.org/learn/php-sort-array

Denk hier ook aan het opdelen in kleine stapjes. Voordat je sorteren gaat toevoegen in jouw programma, kun je eerst met eenvoudige kleine stukjes code testen hoe het sorteren werkt.

Tips

Als je met een form werkt dan is het handig om te beginnen met een action method GET.

Voorbeeld staat in deze les. Op deze manier kun je op de URL (internet adres bovenaan in je browser) zien welke variabelen er worden meegegeven.

Je vraagt bijvoorbeeld wat de hoofdstad is van France (Paris). Stel je antwoord is Paris. Je post jouw antwoord en in de code die wordt uitgevoerd heb je nu jouw antwoord (Paris), maar is dat voldoende om te controleren of je het goed hebt? Nee dus je moet ook weten wat de vraag was, dus van welk land moest je de hoofdstad benoemen?Je zult dat dus ook mee moeten geven via de form variabelen aan jouw (action) script.

https://www.maxdata.ovh/cnt.php?id=php1-hoofdsteden1

10 Portfolio

Bij het zoeken naar stage of werk kan het als software developer helpen als je kan laten zien wat je hebt gedaan. Dit doe je door een portfolio op te bouwen. Het is natuurlijk erg mooi als je deze portfolio on-line kan laten zien. In deze les wordt uitgelegd hoe je zelf je eigen site online kan zetten.

Maak hiervoor een lokale website en stel je zelf hier in voor. Plaats een link naar je LinkedIn pagina en naar de projecten die je tijdens PHP hebt gemaakt. Het 'Hoofdsteden' spel is een leuk voorbeeld dat je kan plaatsten. Je kunt ook zelf een variant hierop maken of je kunt het hoofdsteden-spel uitbreiden.

Hoe maak je een portfolio?

Je kunt veel technieken die je hebt geleerd gebruiken; HTM, JavaScript en PHP. Je kunt een ook een front-end framework als bootstrap gebruiken. Daarover later meer.

Als je jouw site lokaal hebt draaien dan moet je jouw site online zetten.

Hoe zet je jouw site online?

Als je een pagina hebt gemaakt dan kun je via email een Linux account aanvragen. Je krijgt dan een login en een wachtwoord. Met de Linux kennis en met behulp van MobaXterm kun je jouw site dan online setten. In het onderstaande filmpje wordt uitgelegd hoe dat werkt.

https://www.youtube.com/embed/meQgmc3a2VU

Bootstrap

Bootstrap is een front-end framework waarmee je snel een eenvoudig je site kunt opleuken. je kunt werkelijk in een paar stappen van een eenvoudige lijkende site een site maken die er heel knap uit ziet.

Kijk maar eens naar het verschil: zonder bootstrap en met bootstrap.

In het volgende filmpje leg ik heel kort uit hoe je kan beginnen met bootstrap. Uiteraard kun je alle informatie zelf ook op de Bootstrap site vinden.

https://www.youtube.com/embed/HQImds_qf-M

Ideeën

Een leuk idee om een eenvoudige page counter aan je site toe te voegen: hier staat beschreven hoe je dat op een eenvoudige manier kan doen.

Voorbeelden

arjen.softwaredeveloper.ovh

harmohat.softwaredeveloper.ovh

qaher.softwaredeveloper.ovh

jessie.softwaredeveloper.ovh

lucas.softwaredeveloper.ovh

max.softwaredeveloper.ovh (hierin staan twee voorbeelden van het hoofdstedenspel)

--

Overzicht PHP filmpjes

Alle PHP Filmpjes:

https://www.youtube.com/playlist?list=PLBU_XXNTKkBrJ2kDna9tAAEJPbJBFQlZ-

For-loop

De for loop in PHP wordt uitgelegd. Waarom is een loop zo handig? Wat is de **startwaarde**, wat is de **voorwaarde** en hoe **tel** ik in een loop?

In dit filmpje worden vele voorbeelden van een **for-loop** getoond.

https://youtu.be/dlgq69XKiR0

Hoe kunnen we met een loop door een array heen lopen? Dit filmpje laat twee methodes zien: met een for-loop en met een foreach-loop. Ook wordt uitgelegd waarom een count() handig is als je met een for-loop door een array heen wilt lopen.

https://youtu.be/hWNfuxD5AoA

Functies

War is een functie en waarom zou je die willen gebruiken? In dit filmpje wordt een eenvoudige functie gemaakt waarin je Liters kan omrekenen naar (Amerikaanse) Gallons. Je ziet een 'echte' programmeur aan het werk die niet eens weet hoe je round() in moet typen :)

https://youtu.be/XfnH3AEF5Z8

Een ander voorbeeld van een functie die een **array** meekrijgt en die alleen de even getallen er uit haalt en deze terug geeft als **return waarde**.

In dit filmpje wordt een **for-loop** en de PHP functie **count()** gebruikt. Ook komt de **modulo** functie (**%**) terug.

https://youtu.be/INHM0S_8GHA

Functions en Scoping

Een korte uitleg over scoping.

Wat is lokale scope met de **functie** doelets() gaan we op onderzoek uit.

https://youtu.be/hp8-MrfuwoI

Local scope van variabelen in PHP Functions. Communiceren met functies doe je alleen met parameters/argumenten en meet de return value; de input en de output.In dit voorbeeld wordt een functie gebruikt om invoerkosten te berekenen. We gaan goed kijken naar de **return** value en variabele scoping.

https://youtu.be/7bXfUBjLR5I

PHP Assiociative Arrays

In deze video wordt het verschil tussen een gewoon array en een assiociative array uitgelegd.

https://youtu.be/5lJLecI0BTA

Stel we hebben een associative array met daarin een lijstje cijfers van een aantal personen, hoe kunnen we die doorlopen en daar de gemiddelden van berekenen? Het gaat hier om een associative array met daarin arrays. Dit is een complexe data structuur, omdat het om een combinatie vat arrays gaat.

https://youtu.be/5lJLecI0BTA

Hoe ga je om met een complexe datastructuur in PHP? In het voorbeeld gebruiken we een assiociative array waarvan de value een array is. Het gebruik van **count()** is nodig als je met een dynamische data structuur werkt. In dit voorbeeld hebben een aantal personen een verschillend aantal cijfers gehaald. Bepaal het gemiddelde van deze dynamische data structuur.

https://youtu.be/kvrKIa68Rv0

Inspringen Code

Waarom en hoe inspringen?

https://youtu.be/2ja9gu3ANME

Quiz - HTML Form

Eenvoudige quiz maken met HTML-form en PHP-code.

Het voorbeeld in dit filmpje is een form waarin de gebruiker een antwoord moet geven op een eenvoudig sommetje. Via PHP-code moet het antwoord dan worden gecontroleerd.

Hoe stuur je informatie over van de ene naar de andere pagina?
Wat is het verschil tussen POST en GET?
En wat zijn form variabelen en hoe kan ik die gebruiken?
Hoe weet de code die het antwoord moet controleren of het antwoord goed is?

https://youtu.be/8dE-4u94LlU

Eigen site

Hie zet je jouw site on line op een Linux platform via SSH en met behup van MobaXterm?

https://youtu.be/meQgmc3a2VU

--