**Leren Programmeren**

**Overzicht leerstof PHP**

# Basics

## Operatoren

**+ optellen**

**- aftrekken**

**\* vermenigvuldigen**

**/ delen**

**\*\* machten**

**intdiv() gehele deling**

**% modulo**

## Waarden & Types

**Integer**

**Float**

**String**

**Boolean**

## Standaardfuncties

**print()**

**gettype()**

# Variables, Expressions and Statements

## Assignment (toewijzing)

Wat is een assignment?

Bv.

**$x**=2;

## Variabelen

Wat is een variabele?

Naamgeving van variabelen

**Keywords** (gereserveerde woorden)

## Expressions (expressies)

Wat is een **expressie**?

Bv.

**$x** + **$y** + 6 + (**int**)3.99

## Volgorde van operaties

Haakjes / Machten / Vermenigvudigen en delen / Optellen en aftrekken

## String operatoren

. samenvoegen (**concatenate**)

## Commentaar toevoegen

**# Dit is commentaar  
// Dit is commentaar  
/\* Dit is commentaar \*/**

## Soorten fouten

**Syntax** error

**Runtime** error

**Logische fout** (Semantic error)

# Functions

## Function calls – functie aanroep

Standaardfuncties: **gettype**(), **strlen(), count(), …**

**Return value** (terugkeerwaarde)

## Gebruikersfuncties

Functiedefinitie

**function OppervlakteRechthoek( $lengte, $breedte )  
{  
 return ($lengte** \* **$breedte)**;  
**}**

## Flow of execution (volgorde van uitvoering)

* **Van boven naar beneden**
* **Function calls** onderbreken deze logische volgorde
* In een functie mag je een andere functie oproepen

## Parameters en argumenten

Bij een **function call** kan je **argumenten** doorgeven

Bv. **oppervlakte(23)**

**23** is het **argument** van de functie **oppervlakte**()

In een **functie definitie** kan je **parameters** voorzien:

**function oppervlakte( $straal )  
{  
 $opp** = *round***($straal** \*\* 2 \* *pi***(),** 2**)**;  
 **return $opp**;  
**}**

**$straal** is een **parameter** van de functie **oppervlakte**()

Argumenten zijn **waarden** (of kunnen omgezet worden naar waarden)

Parameters zijn **variabelen**

## Parameters en lokale variabelen

**Scope** (‘bereik’ of ‘geldigheid’) van parameters en variabelen:

* **lokaal**
* **globaal**

## Functies met en zonder return

* Sommige functies geven een **return value**
* Sommige functies geven **geen return value**

!!! Opgelet: je mag **output** (bv. met een **print** statement) niet verwarren met een **return value**

## Het nut van functies

* Functies laten toe om **dezelfde bewerking meermaals uit te voeren**, met dezelfde code; ze maken je code **compacter**, en zorgen voor **minder kans op fouten**
* De code is **makkelijker** **aanpasbaar**
* Door functies een **zinvolle naam** te geven, documenteer je meteen ook je code: ze wordt beter leesbaar en beter te begrijpen

# For loop

## Eenvoudige lussen

For-loop

Bv.

**for ( $x**=0; **$x** < **$aantal\_spaties**; **$x**++**)  
{  
 print " "**;  
**}**

# Keyword argumenten

**$x = oppervlakte( $straal=23 )**

# Conditionals

## Booleaanse expressies

**5 == 5**

**$x != 3**

**$a >= $b**

…

## Logische operatoren

**AND**

**OR**

**NOT**

## Condities

**if (** *strpos***($tekst, "<h1>")** === **false )  
{  
 print "Ik vind geen heading in de tekst"**;  
**}  
else  
{  
 print "Joehoe! Heading gevonden!!!"**;  
**}**

## Geneste condities

## Invoer vragen aan gebruiker

**$x = readline( "Wat is uw naam?" );**

# Soorten functies

## Return values - met of zonder

## Booleaanse functies

**function DeelbaarDoor( $getal, $deler )  
{  
 if ( $getal** % **$deler** == 0 **) return true**;  
 **else return false**;  
**}**

## Parameters controleren (Checking types)

Het is zinvol om de argumenten die doorgegeven worden aan een functie, te controleren op type, geldigheid, …

# Iteration (herhaling)

## Reassignment

**$x=3**

**$x=27**

## Increment/decrement

**$x++  
$x += 3  
$x--  
$x -= 3**

## While statement

Bv.

**$n** = 1;  
  
**while ( $n** <= 20 **)   
{  
 print $n** . **" "** . random\_int**(**100**,** 999**)** . **"\n"**;  
 **$n**++; **//increment  
}**

## break statement

## continue statement

# Strings

## strlen()

## substr()

## str\_replace()

## strpos()

# Arrays

## Voorbeeld

**$lijst1** = **[** 14**,** 5**,** 8**,** 9**,** 13**,** 18**,** 16**,** 25 **]**;

**$lijst2** = **[ "jan", "piet", "lies", "els" ]**;

## Een nieuwe lege lijst maken

**$lijst** = **[]**;

## Een lijst overlopen

**$lijst** = **[ "jan", "piet", "lies", "els" ]**;  
  
**foreach ( $lijst as $naam )  
{  
 print $naam** . **"\n"**;  
**}**

## Een element uit een lijst aanspreken

**print $lijst[**0**]** . **"\n"** ;  
**print $lijst[**3**]** . **"\n"** ;

**0** is een **sleutel** of **key**

de **waarde** of **value** daarvan wordt geprint, bv. **"jan"**

## Een nieuw element aan een lijst toevoegen

**$lijst[]** = **"annemie"**;

## Een bestaand element in een lijst overschrijven

**$lijst[**0**]** = **"jan-willem"**;

# Associatieve arrays: keys en values

## Voorbeeld

**$lijst** = **[ "jan"** => 38**, "piet"** => 27**, "lies"** => 41**, "els"** => 18 **]**;

**"jan"** is een **sleutel** of **key**

**38** is een **waarde** of **value**

## Een associatieve array overlopen

**foreach( $lijst as $naam** => **$leeftijd )  
{  
 print "$naam is $leeftijd jaar oud\n"**;  
**}**

## Een element uit een associatieve array aanspreken

**print $lijst["jan"]** . **"\n"** ;  
**print $lijst["piet"]** . **"\n"** ;

## Een nieuw element aan een associatieve array toevoegen

**$lijst["marie"]** = 78;

## Een bestaand element in een lijst overschrijven

**$lijst["lies"]** = 29;

# Array functions

Zie <https://www.php.net/manual/en/ref.array.php>