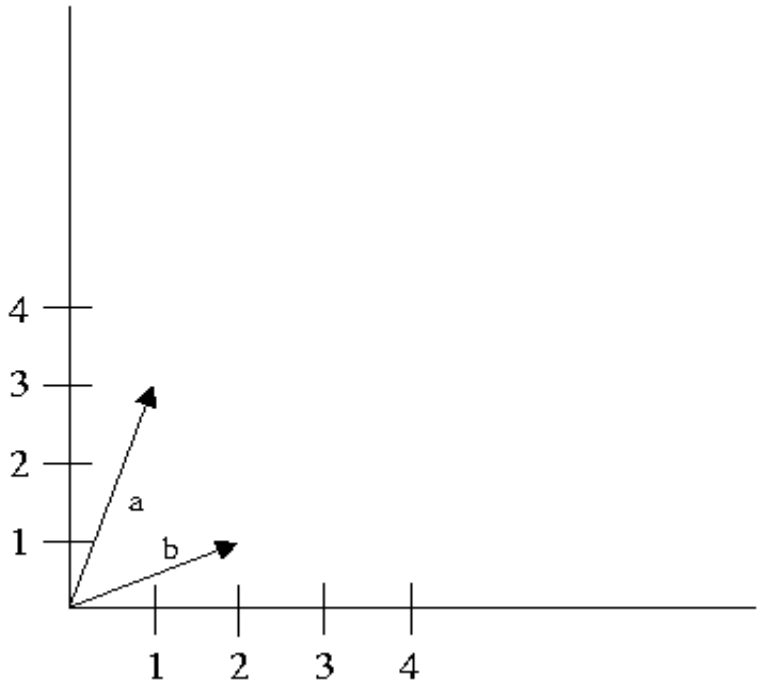
Arrays



Problem:

Volatilitätenbeispiels in der Programmiersprache JavaScript.

Der Benutzer unseres Programms soll die Aktienkurse für beliebig viele Tage eingeben können. Dabei soll er zunächst sagen, wie viele Tage berücksichtigt werden sollen. Sodann kann er für jeden dieser Tage einen Kurs eingeben. Ausgegeben werden soll zunächst der Durchschnitt der Kurse, dann die Varianz, dann die Volatilität und zum Schluss noch einmal die eingegebenen Werte.

JavaScript Array-Regeln

Deklaration eines Arrays :

```
var meinArray;
```

Erzeugung eines Arrays :

```
meinArray=new Array();
```

Erzeugung eines Arrays mit Angabe der Anzahl Koordinaten:

```
meinArray=new Array(5);
```

Der Array-Index wird von 0 hochgezählt.

Zugriff auf Felder über eckige Klammern:

```
meinArray[0]=1;
```

Die Elemente eines Arrays werden bei Benutzung erzeugt.

JavaScript Array-Regeln (2)

Die Elemente eines Array müssen nicht vom gleichen Typ sein.

```
meinArray[0]=10
meinArray[1]="diesesFeldIstEinString";
```

Die Werte mehrerer Felder eines Arrays können mit einer Anweisung zugewiesen werden. Dies geschieht, indem die Werte in eckigen Klammern eingeschlossen und durch Kommata getrennt dem Arraynamen zugewiesen werden:

```
meinArray = [0,1,2];
```

ist gleichbedeutend mit:

```
meinArray[0] = 0;
meinArray[1] = 1;
meinArray[2] = 2;
```

Typen von Arrays in JavaScript

- in JavaScript gibt es eine Array-Klasse

Einfache Arrays

```
myArray= new Array(3);
myArray[0]=10;
myArray[1]=11;
myArray[2]=12;

document.write(myArray[0]);
myArray= new Array(10,11,12);
```

Assoziative Arrays

```
myArray=new Array();
myarray=["Eins"]="Prof";
myarray=["Zwei"]="Student";
myarray=["Drei"]="Mitarbeiter";

document.write(myArray["Eins"]);
```

Typen von Arrays in JavaScript

Mehrdimensionale Arrays

```
myArray=new Array;
myArray[0]=new Array("Professor","Student","Mitarbeiter");
myArray[1]=new Array("EinsLive","WDR2","WDR3","WDR4","WDR5");
myArray[2]=new Array("Audi","BMW","Mercedes", "Ford", "Opel", "VW");

document.write(myArray[0][2]+"<br>");
document.write(myArray[2][0]+"<br>");

var raueme;
   raeume=new Array(27);
   for(i=0;i<raume.lenght;i++)
   {
      raeume[i]=new Array(3);
   }

   raueme[0][0]=10;
```

Wichtig:
meinArray.length gibt die Länge (Anzahl Elemente) des Arrays aus.

Volatilitätenbeispiel

```
<script language = "JavaScript">
        var anzahlTage;
        var i;
        var durchschnitt;
        var varianz;
        var dt;
        var kurs;
        kurs=new Array();
        durchschnitt=0;
        varianz=0;
        anzahlTage=parseInt(prompt("Bitte geben Sie die Anzahl der Tage ein!"));
        for(i=0;i<anzahlTage;i++)
        {
                kurs[i]=parseFloat(prompt("Nächster Kurs bitte!"));
                durchschnitt+=kurs[i];
        }
        durchschnitt/=anzahlTage;
        for(i=0;i<anzahlTage;i++)
        {
                        varianz+=(kurs[i]-durchschnitt)*(kurs[i]-durchschnitt);
        }
        varianz/=(anzahlTage-1);
        dt=anzahlTage/250;
        //Volatilitaet
        volatilitaet=varianz/Math.sqrt(dt);
```

Typen von Arrays in PHP

Einfache Arrays

```
$myArray=array();
$myarray=(1,2,3,4,5,"test", "ok", 68,34);
```

Assoziative Arrays

```
$myArray=array();
$myarray=("ErsterWert"=>"Prof", "ZweiterWert"=>"Student", "DritterWert"=>"LBA");
```

Mehrdimensionale Arrays

```
$myArray=array();
$myarray=("Eins"=>array("Test", "OK"), "zwei"=>array("foo"=>array("bar", "doe"));
```

PHP-Array Regeln

Keine Deklaration, kein Zwang zur Erzeugung

```
$meinArray[0]=0;
$meinArray[1]=1;
```

Mögliche Erzeugung

```
$meinArray=Array();
```

Vorbelegung bei Erzeugung

```
$meinArray=array(23, 45);
```

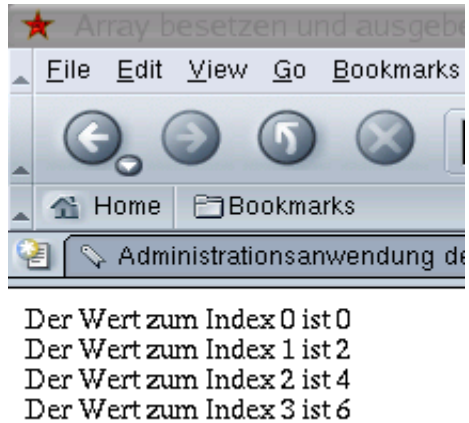
Auch in php müssen die Elemente eines Arrays nicht vom gleichen Typ sein.
Wichtig:
Arrays in PHP können gesplittet, sortiert, verkettet, in einzelne Variable aufgeteilt werden .
Sehen Sie in die php-Funktionsreferenz unter dem Stichwort Array.
Beispiel: Die Funktion count

```
$meinArray=array(23, 45);
echo count($meinArray);
```

Die for - Schleife

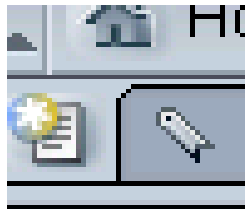
```
for(Initialisierungsanweisung;Bedingung;Änderungsanweisung)
{
   anweisung1;
   ...
   anweisungN
}
```

```
<?php
   for($i=0;$i<4;$i++)
   {
      $meinArray[$i]=$i*2;
   }
   for($i=0;$i<count($meinArray);$i++)
   {
      echo "Der  Wert zum Index $i ist $meinArray[$i]<br>";
   }
?>
```



Besetzen und Auslesen eines Arrays mit einer for-schleife

```
<?php
   for($i=0;$i<4;$i++)
   {
      for($j=0;$j<3;$j++)
      {
         $meinArray[$i][$j]=$i*$j;
      }
   }
   for($i=0;$i<count($meinArray);$i++)
   {
      for($j=0;$j<count($meinArray[$i]);$j++)
      {
         echo "{$meinArray[$i][$j]}  ";
      }
      echo "<br>";
   }
?>
```



```
000
012
024
036
```

Aufgabe

Aufgabe:

1. Erzeugen Sie ein zweidimensionales Array in php und geben Sie es in einer Tabelle aus.

2. Schreiben Sie nun eine Funktion, die ein zweidimensionales Array in einer Funktion ausgibt und benutzen Sie diese.

3. Lagern Sie das Array in eine Klasse HtmlDarstellung aus!

Lösung 1

```
for($i=0;$i<4;$i++)
{
    for($j=0;$j<3;$j++)
    {
        $meinArray[$i][$j]=$i*$j;
    }
}
echo "<table>";
for($i=0;$i<count($meinArray);$i++)
{
    echo "<tr>";
   for($j=0;$j<count($meinArray[$i]);$j++)
    {
        echo "<td>";
          echo $meinArray[$i][$j];
          echo "</td>";
    }
    echo "</tr>";
}
echo "</table>";
```

Lösung 2

```
<?php
    function stelleArrayAlsHtmlTabelledar($meinArray, $border)
    {
            echo "<table border='$border'>";
            for($i=0;$i<count($meinArray);$i++)
            {
                echo "<tr>";
               for($j=0;$j<count($meinArray[$i]);$j++)
                {
                    echo "<td>";
                      echo $meinArray[$i][$j];
                      echo "</td>";
                }
                echo "</tr>";
            }
            echo "</table>";
    }
```

Lösung 2-2

```
for($i=0;$i<4;$i++)
   {
      for($j=0;$j<3;$j++)
      {
         $meinArray[$i][$j]=$i*$j;
      }
   }
      stelleArrayAlsHtmlTabelledar($meinArray, "2");
?>
```

Lösung 3 – Die Klasse

```
class HtmlDarstellung
{
    function stelleArrayAlsHtmlTabelledar($meinArray, $border)
    {
        echo "<table border='$border'>";
        for($i=0;$i<count($meinArray);$i++)
        {
            echo "<tr>";
            for($j=0;$j<count($meinArray[$i]);$j++)
            {
                echo "<td>";
                echo $meinArray[$i][$j];
                echo "</td>";
            }
            echo "</tr>";
        }
        echo "</table>";
    }
}
```

Lösung 3 – Der Aufruf

```
<?php
    require_once("./klassen/HtmlDarstellung.inc.php");

  for($i=0;$i<4;$i++)
  {
     for($j=0;$j<3;$j++)
     {
        $meinArray[$i][$j]=$i*$j;
     }
  }
    HtmlDarstellung::stelleArrayAlsHtmlTabelledar($meinArray, "2");
?>
```

Regeln

Nach der for-Anweisung darf kein Semikolon stehen.

Initialisierungsanweisung, Bedingung und Änderungsanweisung werden durch Semikolons getrennt.

Es kann mehrere Initialisierungsanweisungen und Änderungsanweisungen in einer for-Schleife geben.

Sie werden dann jeweils durch Kommata getrennt.

Die foreach - Schleife

```
foreach($meinArray As $wert)
{
    anweisung1
    anweisung1;
    ...
    anweisungN
}
```

```
<?php
    for($i=0;$i<4;$i++)
    {
        $meinArray[$i]=$i*2;
    }
    foreach($meinArray As $arrayWert)
    {
        echo "Der nächste Wert des Arrays ist $arrayWert
<br>";
    }
?>
```

mehrdimensionaler Arrays mit der foreach-Schleife

```
<?php
   for($i=0;$i<4;$i++)
   {
      for($j=0;$j<3;$j++)
      {
         $meinArray[$i][$j]=$i*$j;
      }
   }
   foreach($meinArray As $arrayVector)
   {
      foreach($arrayVector As $arrayWert)
      {
         echo "$arrayWert  ";
      }
      echo "<br>";
   }
?>
```

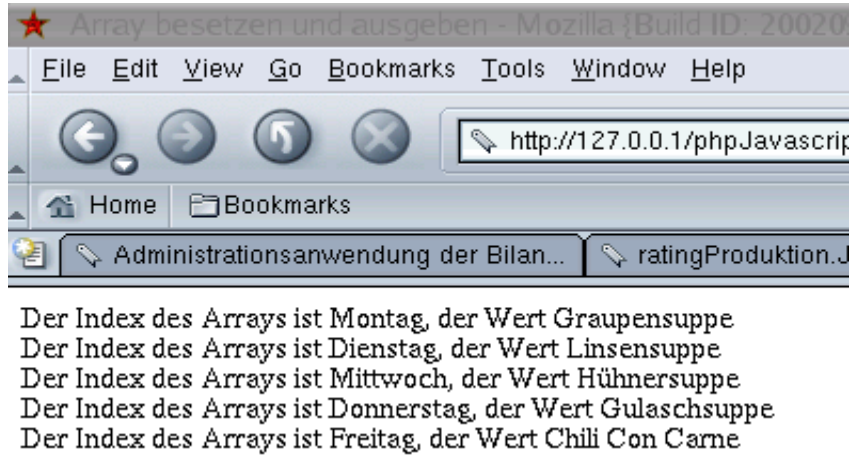
Die foreach-Schleife für assoziative Arrays in php

```
foreach($meinArray As $arraySchluessel => $arrayWert)
{
    anweisung1
    anweisung1;
    ...
    anweisungN
}
```

Beispiel

```
<?php
   $essen=array('Montag'=>'Graupensuppe',
                'Dienstag'=>'Linsensuppe',
                'Mittwoch'=>'Hühnersuppe',
                'Donnerstag'=>'Gulaschsuppe',
                'Freitag'=>'Chili Con Carne');

   foreach($essen As $arraySchluessel => $arrayWert)
   {
      echo "Der Index des Arrays ist $arraySchluessel, der Wert $arrayWert <br>";
   }
?>
```



Aufgabe

Aufgabe

Erweitern Sie Ihre Klasse HtmlDarstellung um eine Funktion, die ein assoziatives Array in einer html-Tabelle ausgibt.

Rufen Sie diese Funktion mit dem assoziativen Array aus dem Beispiel auf!

Lösung: Die Klassenerweiterung

```
<?php
class HtmlDarstellung2
{
    function stelleAssoziativesArrayAlsHtmlTabelledar($meinArray, $border)
    {
        echo "<table border='$border'>";
        foreach($meinArray As $key => $value)
        {
            echo "<tr>";
            echo "<td>";
            echo $key;
            echo "</td>";
            echo "<td>";
            echo $value;
            echo "</td>";
            echo "</tr>";
        }
        echo "</table>";
    }
```

Lösung: Die Klassenerweiterung

```
function stelleArrayAlsHtmlTabelledar($meinArray, $border)
{
    echo "<table border='$border'>";
```

Lösung: Der Aufruf

```
<?php
   require_once("./klassen/HtmlDarstellung2.inc.php");
   $essen=array('Montag'=>'Graupensuppe',
            'Dienstag'=>'Linsensuppe',
            'Mittwoch'=>'Hühnersuppe',
            'Donnerstag'=>'Gulaschsuppe',
            'Freitag'=>'Chili Con Carne');
   HtmlDarstellung2::stelleAssoziativesArrayAlsHtmlTabelledar($essen, $border)


?>
```