Verzweigungen

Wie praktisch alle Programmiersprachen mit imperativen Sprachkonzepten bietet auch PHP Sprachelemente, sogenannte Kontrollstrukturen, um Verzweigungen zu realisieren. Dazu zählen die einfache Verzweigung mit `if` und die mehrfache Verzweigung mit `switch`.

Einfachverzweigung mit if/else

Mit der sogenannten if-Anweisung (if, engl. für „wenn“) kann man eine Bedingung prüfen. Das Sprachkonstrukt (es ist keine Funktion) wird mit dem Schlüsselwort `if` notiert, gefolgt von einem Paar runder Klammern, in dem eine Bedingung notiert wird. Diese Bedingung wird nach dem "Wenn-Dann-Prinzip" geprüft. Wenn sie zutrifft, dann soll etwas bestimmtes passieren. Dies ist ein Anweisungsblock, der innerhalb von geschweiften Klammern notiert und zur besseren Lesbarkeit eingerückt wird.

Für den Fall, dass die Bedingung nicht erfüllt ist, kann man einen Alternativzweig mit dem Schlüsselwort else (else, engl. für „ansonsten“) festlegen.

Normierte Sprache	Programmiersprache	Flussdiagramm

`WENN ( Schraube ist darunter )`
`DANN`
`  aufheben`
`*ENDE DANN`
`SONST`
`  ablegen`
`*ENDE SONST`
	`if (istAufGegenstand("Schraube"))`
	`{`
	`  aufnehmen();`
	`}`
	`else {`
	`  ablegen("Schraube");`
	`}`	

Codebeispiel:

beispiel01.php

```php
$haben = 10;
$limit = 20;

if($haben > $limit) {
  // gibt Ergebnis aus
  print "Einkaufen gehen";
}
else {
  print "Nicht einkaufen gehen";
}
```

Exkurs: Logische Operationen

Vergleichsoperatoren erlauben es - wie der Name schon sagt - zwei Werte miteinander zu vergleichen. Die wesentliche Eigenschaft von Vergleichsoperationen ist, dass es nur zwei mögliche Ergebnisse gibt: WAHR und FALSCH. Diese beiden Möglichkeiten nennt man auch "Wahrheitswerte". **Jede `Vergleichsoperation` liefert also einen der beiden Wahrheitswerte *Wahr* oder `Falsch` zurück**.

Beispiele:

```
3 == 5  liefert FALSCH
4 == 4  liefert WAHR
3 <  3  liefert FALSCH
3 <= 3  liefert WAHR
```

Es gibt eine ganze Anzahl von Vergleichsoperatoren für, verschiedene Datentypen, einen Überblick gibt das PHP Handbuch: http://www.php.net/manual/de/language.operators.comparison.php

Vergleiche können darüber hinaus noch logisch vernüpft werden, z.B. so:

```
4 == 5 ODER 3 == 3 liefert WAHR
```

denn bei der Verknüpfung mit ODER reicht es, wenn eine der beiden Vergleichsoperationen WAHR ist. Man kann dazu eine Wahrheitstabelle aufstellen:

ODER		**(PHP Notation ||)**
A	B	A||B
WAHR	WAHR	WAHR
WAHR	FALSCH	WAHR
FALSCH	WAHR	WAHR
FALSCH	FALSCH	FALSCH

Aufgabe

Erstelle für jeder der logischen Operatoren eine solche Wahrheitstabelle: http://php.net/manual/de/language.operators.logical.php

Experiment: Passwort prüfen

Wir legen (in einem neuen Unterverzeichnis, z.B. "passwort") zwei neue Dateien an, `eingabe.html` und `ausgabe.php`:

Zunächst ein neues Formular, welches eingegebene Daten an `ausgabe.php` versenden wird. In das Passwortfeld müssen wir später unser Passwort eintragen und mit dem Button das Formular abschicken.

eingabe.html

```
<form action="ausgabe.php" method="post">
 <label>Passwort: <input type="password" name="pw"></label>
 <button>Senden</button>
</form>
```

Zur Verarbeitung legen wir die Datei `ausgabe.php` an, die unsere vom Formular übermittelten Informationen verarbeitet:

ausgabe.php

```
if ($_POST["pw"] == "test") {
  print "Herzlich Willkommen im geschützten Bereich";
}
else {
  print "Falsches Passwort";
}
```

Per if-Anweisung überprüfen wir, ob das eingegebene Passwort dem Wert "test" entspricht, wenn ja, dann hat man Zutritt zum geschützten Bereich, ansonsten nicht.

Achtung

Achtung: Die Passwort-Prüfung ist in der vorliegenden Form noch keine sinnvolle Lösung für eine Zugangsbeschränkung (daher das "Experiment:" im Titel). Hier muss bei jedem Seitenaufruf das Passwort erneut eingegeben und das Formular abgeschickt werden um an die geschützten Inhalte zu gelangen.

Aufgabe

Erweitere das "Passwort prüfen" Beispiel:

- Ermittle mit der "date" Funktion von PHP die Stunde der aktuellen Uhrzeit Hilfe.
- Baue im Zweig mit dem korrekten Passwort noch eine Unterscheidung ein, so dass der Benutzer je nach Tageszeit anders begrüßt wird (z.B. "Guten Morgen" bis 12:00 Uhr, "Guten Tag" ab 12 Uhr).

Mehrfachverzweigung mit switch

Mit der Switch-Anweisung lassen sich quasi mehrere If-Abfragen bündeln, wenn man eine Variable auf unterschiedliche Inhalte abfragen möchte.

Beispiel: Mehrfachverzweigung mit switch

beispiel.php

```php
$a = 30;
$b = 12;
$op = "-";

switch ($op){
    case "+":
        echo "$a plus $b ergibt: ", $a + $b;
        break;
    case "-":
        echo "$a minus $b ergibt: ", $a - $b;
        break;
    case "*":
    case "x":
        echo "$a mal $b ergibt: ", $a * $b;
        break;
    case "/":
    case ":":
        echo "$a durch $b ergibt: ", $a / $b;
        break;
    default:
        echo "Das Rechenzeichen '$op' ist mir unbekannt.";
}
```

Mit dem Schlüsselwort `switch` wird die Struktur eingeleitet. In runden Klammern wird ein Ausdruck notiert (hier eine Variable), der zu prüfen ist.

Die einzelnen Fallunterscheidungen werden mit der Kombination aus `case` und `break` notiert. Nach dem Schlüsselwort `case` steht ein Wert, gefolgt von einem Doppelpunkt. Danach folgen beliebig viele Anweisungen, die für diesen Fall gelten. Mit dem Schlüsselwort `break` (unbedingt mit abschließendem Semikolon!) wird der jeweilige Fall abgeschlossen.

Das Schlüsselwort `default` (default, engl. Vorgabe) ist für alle die Fälle gemeint, die keinem der notierten Fälle entsprechen, weshalb hier kein Wert vor dem Doppelpunkt notiert wird. In diesem Code-Beispiel steht es an letzter Stelle, weshalb auf das abschließende break verzichtet werden kann.

Folgt einem `case` kein `break`, so werden alle nachfolgenden Anweisungen der folgend notierten Fälle abgearbeitet, bis wieder ein break steht. Damit kann man Anweisungen zusammenfassen:

Beispiel: Mehrfachverzweigung mit "Durchrutschen"

beispiel.php

```php
$note = 1;

echo "Sie haben eine ";

switch ($note){
    case 1:
        echo "sehr ";
    case 2:
        echo "gute ";
    default:
        echo "Note ";
}

echo $note," bekommen.";
// Ergebnisse
// $note = 1: "Sie haben eine sehr gute Note 1 bekommen."
// $note = 2: "Sie haben eine gute Note 2 bekommen."
// $note = 3: "Sie haben eine Note 3 bekommen."
```

Das Fehlen von `break` nach jedem `case` führt dazu, dass alle folgenden echo-Anweisungen ausgeführt werden.

Aufgabe

Ergänze das Taschenrechner-Beispiel durch ein Formular, so dass du im Unterverzeichnis "taschenrechner" auf deinem Webspace einen funktionierenden Taschenrechner erhältst:

- Erstelle das passende Formular in einer HTML Datei (3 Eingabefelder).
- Verändere das case-Beispiel so, dass es die entsprechenden $_POST-Variablen verwendet.

Aufgabe

Implementiere einen "Mitternachtsformel" Rechner mit einem Formular mit 3 Eingabefeldern und einer entsprechenden PHP-Verarbeitungsdatei. Welches ist das Unterscheidungskriterium für die case Anweisung?
