APLIKACJE INTERNETOWE
PHP

TEMAT 1-03: Instrukcje warunkowe.

Autor dokumentu: Wojciech Galiński	piątek, 16 września 2016 r.	351203 Technik informatyk

ŹRÓDŁA WIEDZY: http://www.php.net/manual/pl, http://webmaster.helion.pl/index.php/kurs-php, http://pl.wikipedia.org/, http://pl.wikibooks.org/wiki/PHP, http://phpkurs.pl/, http://kursphp.com/.

Zagadnienia obowiązkowe

1. **Instrukcje sterujące** – dzielimy na:
   - ➔ **INSTRUKCJE PROSTE**: instrukcja przypisania „=”;
   - ➔ **INSTRUKCJE WARUNKOWE**: podstawowa instrukcja warunkowa „**if**”, instrukcja wyboru „**switch**”, operator przetwarzania warunkowego „**(?:)**”;
   - ➔ **INSTRUKCJE ITERACYJNE (PĘTLE)** – wyróżniamy: pętla dopóki „**while**”, pętla dla „**for**”, pętla dla każdego „**foreach**”, pętla przetwarzaj dopóki „**do while**”.

   W niniejszym temacie omówione zostaną instrukcje warunkowe.
2. **Operatory relacji (porównywania)** – wyróżniamy: „**==**” – równe; „**!=**”, „**<>**” – różne; „**<=**” – mniejsze lub równe; „**>=**” – większe lub równe, „**===**” – identyczne (ta sama wartość i typ); „**!==**” – nieidentyczne.

   PRZYKŁADY:
```
var_dump(5=='5');  var_dump(5!='5');    var_dump(5==='5');  var_dump(5!=='5');
var_dump(0===0.0);   var_dump('abc'=='ABC');   var_dump('abc'==='ABC');
```
3. **Operatory logiczne** – służą głównie do połączenia wielu wyrażeń logicznych złożonych z operatorów relacji. Wyróżniamy: „**!**” – negacja logiczna; „**||**”, „**or**” – alternatywa logiczna; „**&&**”, „**and**” – koniunkcja logiczna; „**xor**” – logiczna alternatywa wykluczająca.

   PRZYKŁAD:
```
var_dump(!2);   var_dump(2 || 0);   var_dump(2 or 0);
var_dump(2 && 0);   var_dump(2 and 0);
$rybki = true;   $akwarium = true;   var_dump($rybki xor $akwarium);
var_dump(5<=2+3 || 4-2>0 && !0);   var_dump(0==='0' || 0==false);
```
4. **Warunek** – wyrażenie zwracające wartość logiczną: „true” (warunek jest spełniony) albo „false” (warunek nie jest spełniony). Każdy warunek składa się głównie z operatorów relacji i logicznych.
5. **Instrukcja „if”** – oto jej podstawowa wersja:

```
if (warunek)
   instrukcja;

if (warunek)
{
   // instrukcje
}
```

PRZYKŁAD:
```
$k = -5;
if ($k)
   echo 'Zmienna k różna od 0.';
if ($k < 0)
{
   $k = -$k;   echo $k;
}
```

W powyższych instrukcjach najpierw sprawdzany jest warunek. Jeśli jest on spełniony, instrukcja lub instrukcje są wykonywane. W przeciwnym razie powyższe instrukcje są pomijane.

Oto rozszerzone wersje tej instrukcji:

```
if (warunek)
   instrukcja1;
else
   instrukcja2;

if (warunek)
{
   // instrukcje A
}
else
{
   // instrukcje B
}
```

PRZYKŁAD:
```
$k = -5;
if ($k)
   echo 'Zmienna k jest RÓŻNA od 0.';
else
   echo 'Zmienna k jest RÓWNA 0.';
if ($k >= 0)  echo $k;
else
{
   $k = -$k;
   echo $k;
}
```

W powyższych instrukcjach także najpierw sprawdzany jest warunek. Jeśli jest on spełniony, wykonywane są: „instrukcja1” lub „instrukcje A”. Gdy warunek nie jest spełniony, wykonywane są: „instrukcja2” lub „instrukcje B”.

Instrukcja „**if**” może być zagnieżdżona:

```
if (warunek1)
   instrukcja1;
else
   if (warunek2)
      instrukcja2;
   else
      instrukcjaN;
```

PRZYKŁAD:

```
$dzien = 3;
if ($dzien>=1 && $dzien<=5)
   echo 'Dzień roboczy (poniedziałek - piątek).';
else
   if ($dzien==6)  echo 'sobota';
   else
     if ($dzien==7)  echo 'niedziela';
     else  echo 'Nieprawidłowy numer dnia tygodnia.';
```

6. **Instrukcja „switch”** – powstała, aby uprościć wielokrotne zagnieżdżanie instrukcji „**if**” (tak jak pokazano to w poprzednim punkcie). Ogólna postać tej instrukcji jest przedstawiona w poniższej ramce. Oto opis działania tej instrukcji:
   - ➔ najpierw sprawdzana jest wartość wyrażenia „**wyrazenie**” (jego typ to liczba lub tekst – mniej restrykcyjnie, niż w C++);
   - ➔ jeśli wartość tego wyrażenia występuje po jednym ze słów kluczowych „**case**” to zostają wykonane wszystkie instrukcje umieszczone za słowem kluczowym „**case**”, po którym umieszczona była ta wartość;
   - ➔ jeśli wartość tego wyrażenia nie występuje po żadnym ze słów kluczowych „**case**” to zostają wykonane wszystkie instrukcje umieszczone za słowem kluczowym „**default**”;
   - ➔ do powyższego schematu jest jeden wyjątek: po napotkaniu instrukcji „**break**” wykonywanie instrukcji w bloku „**switch**” zostaje przerwane – następuje przeskok do pierwszej instrukcji za blokiem „**switch**”;
   - ➔ instrukcje „**default**” i „**break**” nie są wymagane, dlatego w schemacie oznaczono je czcionką pochyłą. Liczba instrukcji „**case**” jest dowolna.

```
switch (wyrazenie)
{
case wartosc1:
   // instrukcje1
   break;
case wartosc2:
   // instrukcje2
   break;
   /* … */
default:
   // instrukcjeN
}
```

PRZYKŁAD:

```
$pora_roku = 3;
switch ($pora_roku)
{
   case 1: echo 'wiosna';
   break;
   default:
   echo 'nieznana pora roku';
   break;
}
```

7. **Operator przetwarzania warunkowego „?:”** – to przybliżony odpowiednik instrukcji „**if**”. Oto ogólny schemat tej instrukcji:

```
(warunek? wyrazenie1: wyrazenie2);
```

W odróżnieniu od instrukcji „**if**” powyższa instrukcja umożliwia obliczenie jednego z 2 różnych wyrażeń (zamiast wykonania jednej z 2 różnych instrukcji) w zależności od wartości warunku. Jeśli warunek jest spełniony, obliczone zostanie „**wyrażenie1**”, w przeciwnym razie – „**wyrazenie2**”. Wynik można wysłać na ekran lub do zmiennej.

```
echo (warunek? wyrazenie1: wyrazenie2);
$wynik = (warunek? wyrazenie1: wyrazenie2);
```

PRZYKŁAD: `$k = -9;  echo ($k<0? -$k: $k);  $wynik = ($k<0? -$k: $k);  echo $wynik;`

8. **Operator „or” jako alternatywa dla instrukcji „if”** – poniższe zapisy są równoważne:

```
a or b; ⇔ if (!a) b;
```

PRZYKŁAD:

```
ini_set('display_errors', 0);  //  wyłączenie w skrypcie wyświetlania błędów
echo $k or die('Brak zmiennej "k"');
```

Zadania

1. Prześlij do skryptu liczbę (za pomocą adresu skryptu) i zbadaj czy liczba ta jest większa od 0, np. „**123** jest większe od **0**”, „**0** nie jest większe od **0**”.
2. Prześlij do skryptu 2 liczby i porównaj je ze sobą: x1<x2, x1==x2, x1>x2.
3. Wczytaj tekst i sprawdź, czy jest to polskojęzyczny dzień tygodnia: poniedziałek, wtorek, itd.
4. Dane są 3 parametry: (**a**, **b**) – współrzędne punktu, **r** – promień okręgu o środku w punkcie (0,0). Sprawdź, gdzie leży punkt (a, b): wewnątrz okręgu, na okręgu, czy na zewnątrz okręgu.
5. Pobierz 2 parametry: **a** i **b** i poinformuj, czy są to liczby z przedziału [-1E6, 1E6].
6. Pobierz współrzędne 4 punktów i sprawdź, czy tworzą one kwadrat lub prostokąt.