

Instukcje sterujące Liczby

Informatyka

Jolanta Bachan
2008-05-08

if/unless

```
print "How old are you?" ;
```

```
$age = <STDIN> ;
```

```
chomp($age) ;
```

```
if ($age<24) {
```

```
    print "So you are younger than our teacher." ;
```

```
} elsif ($age==24) {
```

```
    print "Hey, you are at our teacher's age! :-)" ;
```

```
} else {
```

```
    print "Wow! You could be our teacher!!!" ;
```

```
}
```

if/unless

```
print "How old are you?" ;  
$age = <STDIN> ;  
chomp($age) ;
```

```
if ($age<24) {  
    print "So you are younger than our teacher." ;  
} elsif ($age==24) {  
    print "Hey, you are at our teacher's age! :-)" ;  
} else {  
    print "Wow! You could be our teacher!!!" ;  
}
```

if/unless

```
print "How old are you?" ;
```

```
$age = <STDIN> ;
```

```
chomp($age) ;
```

```
unless ($age<18) {
```

```
    print "Cool! You are old enough to vote!" ;
```

```
} else {
```

```
    print "So you are not old enough to vote. :-( " ;
```

if/unless

```
print "How old are you?" ;
```

```
$age = <STDIN> ;
```

```
chomp($age) ;
```

```
unless ($age<18) {
```

```
    print "Cool! You are old enough to vote!" ;
```

```
} else {
```

```
    print "So you are not old enough to vote. :-( " ;
```

if/unless

- Ćwiczenie: Napisz własny program z instrukcją `if` lub `unless`. Masz na to 5min.

while/until

- Aby wykonać instrukcje while, Perl ocenia najpierw wyrażenie kontrolne. Jeśli wyrażenie jest ocenione jako prawdziwe, wtedy wykonywane są wyrażenia zawarte w ciele instrukcji while jeden raz. Ten krok jest powtarzany tak długo, dopóki wyrażenie kontrolne jest ocenione jako fałszywe.
- Czasami łatwiej jest powiedzieć “dopóki coś jest prawdziwe” zamiast “podczas gdy to nie jest prawdziwe” i w tym przypadku while zastępowane jest until.

Pętla `while` w tablicach

```
@fred = (1,2,3) ;
```

```
$length = @fred ;
```

```
$i = 0 ;
```

```
while ($i<$length) {
```

```
    $fred[$i] = $fred[$i] + 5 ;
```

```
    print $fred[$i] ;
```

```
    $i++ ;
```

```
}
```

Pętla for

```
for (initial_exp; test_exp; increment)
{
    code to repeat ;
}
```

Pętla for

```
for ($i=1; $i<=5; $i++) {  
    print $i ;  
}
```

Pętla for

Wyrażenie początkowe

Wyrażenie kontrolne

Inkrementacja

```
for ($i=1; $i<=5; $i++) {  
    print $i ;  
}
```

Pętla for w tablicach

```
@fred = (1,2,3) ;  
$fred_length = @fred ;  
for ($i=0; $i<$fred_length; $i++) {  
    print $fred[$i] ;  
}
```

The foreach statement

- This statement is used to operate on arrays as in the following example.

```
foreach $cookie (@cookies) {  
    print $cookie, "\n" ;  
}
```

- The `@cookies` value is an array and the `$cookie` scalar variable is set to a value of an element of the array and the loop continues until all elements of the array have been operated on. Therefore the above example will print every element of the `@cookies` array.

The foreach statement

- More examples on foreach statement:

```
@a = (1,2,3,4,5) ;  
foreach $b (reverse @a) {  
    print $b . "\n" ;  
}
```

The foreach statement

- More examples on foreach statement:

```
@a = (3, 5, 7, 9) ;
```

```
@b = (10, 20, 30) ;
```

```
$x = 17 ;
```

```
foreach $one (@a, @b, $x) {
```

```
    $one *= 3 ;
```

```
}
```

```
@a is now (9, 15, 21, 27) ;
```

```
@b is now (30, 60, 90) and $x is 51.
```

Ćwiczenia

1. Napisz program, który prosi o wartość temperatury na zewnątrz i drukuje “too hot”, jeśli temperatura jest powyżej 25°C, “too cold” jeśli temperatura jest poniżej 18°C i “just right”, jeśli temperatura jest pomiędzy 25 i 18°C.
2. Napisz program, który czyta listę liczb w oddzielnych wierszach do czasu, kiedy wcztana jest liczba 999. Jeśli wczytana jest liczba 999, drukuj sumę liczb (nie dodając liczby 999.)

Do zobaczenia za tydzień!