

Входно/изходни операции.

Управляващи структури.

Операции над масиви и хешове.



Perl

Входно/исходни операции



Функцията print

`print` - вградена функция за извеждане на низ/списък от низове.

Общ синтаксис:

```
print FILEHANDLE LIST;
```

`FILEHANDLE` - дескриптор на изходния поток/файл. По дефиниция е 'STDOUT'.

`LIST` - списък от низове, които ще се изведат в съответния изходен поток.

Например:

```
print STDOUT 'Iva', ' ', 'Popova';
```

е еквивалентно по действие на:

```
print 'Iva Popova';
```

Оператор за вход: <>

<FILEHANDLE> - четене от файл/поток.

По дефиниция входния поток е STDIN и може да се пропуска. Т.е.

<STDIN> е еквивалентно на <>*. Например:

```
$name = <>; # в $name ще се съхрани въведения от потребителя текст, плюс знака за нов ред: '\n'.
```

Ако искаме да премахнем знака за нов ред в края, може да използваме функцията `chomp()`. Например:

```
chomp ($name); # в $name ще остане само въведения текст.
```

Пример:

```
print "Enter your name: ";
chomp ($name=<>); # защото = връща lvalue
print "$name, Hello !";
```

Perl

Управляващи структури



истина или лъжа ?

Винаги се изчислява в скаларен контекст

Всеки низ е 'истина', освен ако не е празен низ или 0

Всяко число е 'истина', освен ако не е 0.

Всеки указател е 'истина'

Недефинираната стойност е винаги 'лъжа'

IF

```
if ($city eq "Sofia") {  
    print "Sf.\n";  
}  
elseif ($city eq "Pernik") {  
    print "Pk.\n";  
}  
elseif ($city eq "Botevgrad") {  
    print "Bgr.\n";  
}  
else {  
    print "where?\n";  
}
```

Unless

```
unless ($destination eq $home) {  
    print "Няма да се прибирам\n";  
}
```

Или на един ред:

```
print "Няма да се прибирам\n" unless $destination eq $home ;
```

! Блока се изпълнява само когато условието е лъжа

For, While, Until

For

```
for ( $i=1; $i<6; $i++ ){  
    print "$i\n";  
}
```

While

```
$i=1;  
while ($i < 6) {  
    print "$i\n"; $i++;  
}
```

Until

```
$i=1;  
until ($i == 6) {  
    print "$i\n"; $i++;  
}
```

Foreach

Най-често се използва за обхождане на масиви / хешове:

```
@alphabet = ('a' .. 'z');  
foreach $letter ( @alphabet ) {  
    print " $letter\n";  
}
```

Perl

Операции над масиви и хешове.



Обхождане на елементите на масив

for :

```
for ( $i=0; $i<=$#plodove; $i++ ){  
    print "$plodove[$i]\n";  
};
```

Предпочитан вариант:

foreach :

```
foreach $plod ( @plodove ) {  
    print "$plod\n";  
};
```

Обхождане на елементите на хеш

```
while ( ($key, $value) = each %hash ) {  
    print "$key - $value\n";  
}
```

или

```
foreach $key ( keys %pazar ) {  
    print "$pazar{$key}\n";  
}
```

Няма гаранция че елементите ще се изпишат в реда в който сме ги задали при инициализирането на хеша - при хешовете няма ред! Но можем да сортираме

Сортиране на масиви и хешове

sort SUBNAME LIST

сортиране на масив по азбучен ред (А-Я) :

```
foreach $plod ( sort @plodove ) { print "$plod\n"}
```

сортиране на масив по азбучен ред (Я-А) :

```
foreach $plod ( sort { $b cmp $a } @plodove ) {  
    print "$plod\n";  
}
```

сортиране на хеш по азбучен ред (Я-А) спрямо СТОЙНОСТИТЕ му:

```
foreach $value ( sort{ $b<=>$a } values %pazar) {  
    print "$value\n";  
}
```

сортиране на хеш по азбучен ред (Я-А) спрямо КЛЮЧОВЕТЕ му:

```
foreach $value ( sort{ $b<=>$a } keys %pazar) {  
    print "$value\n";  
}
```