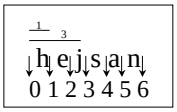
Påminnelse

- Skriv luftigt!
- Kommentera
- Tänk på kontexten: $, @ och %
- Använd deskriptiva variabelnamn
- Använd –w i Shebangen
- Provkör ofta
- Använd print för att skriva ut variabler
- Vägled alltid användaren

substr - förklaring



$hej = ’hejsan’;

Första bokstaven:

```
$ini = substr($hej,0,1);
# $ini blir ’h’
```

Andra bokstaven:

```
$andra = substr($hej,1,1);
#$andra blir ’e’
```

Allt utom första bokstaven:

```
$rest = substr($hej,1);
#$rest blir ’ejsan’
```

Och allt utom de 3 första:

```
$svans = substr($hej,3);
#$rest blir ’san’
```


Val

```
print ”Jag tänker på en siffra mellan 1-10.
        Gissa vilken: ”;
$a = <STDIN>;
chomp($a);
if ($a == 10) {
      print ” 10! Rätt!\n”;
}
print ”Nu vill jag inte leka mer! Hejdå!\n”;
```


Körning

> Jag tänker på en siffra mellan 1-10.
> Gissa vilken: *9*
> Nu vill jag inte leka mer! Hejdå!

> Jag tänker på en siffra mellan 1-10.
> Gissa vilken: *10*
> 10! Rätt!
> Nu vill jag inte leka mer! Hejdå!

Val forts

```
if ($a == 10) {
      print ” 10! Rätt\n”;
}
else {
      print ”Fel Fel Fel!!!\n”
}
```

```
unless ($a == 10)  { print ”Fel Fel Fel!!!\n”; }
if(!$a==10) { print ”Fel Fel Fel!!!\n”; }
if($a!=10) { print ”Fel Fel Fel!!!\n”; }
```


Körning

> Jag tänker på en siffra mellan 1-10. Gissa vilken: *9*
> Fel Fel Fel!!!
> Nu vill jag inte leka mer! Hejdå!

> Jag tänker på en siffra mellan 1-10. Gissa vilken: *10*
> 10! Rätt
> Nu vill jag inte leka mer! Hejdå!

> Jag tänker på en siffra mellan 1-10. Gissa vilken: *9*
> Fel Fel Fel!!!
> Nu vill jag inte leka mer! Hejdå!

> Jag tänker på en siffra mellan 1-10. Gissa vilken: *10*
> Nu vill jag inte leka mer! Hejdå!

Val forts

```
if ($a == 10) {
        print ” 10! Helt Rätt!\n”;
}
else {
        if ($a == 9) {
                print ” 9! Nästan helt rätt!\n”;
        }
        else {
                print ”Fel Fel Fel!!!\n”;
        }
}
```


Val forts

```
if ($a == 10) {
        print ” 10! Helt rätt!\n”;
}
elsif ($a == 9) {
        print ” 9! Nästa helt rätt!\n”;
}
else {
        print ”Fel Fel Fel!!!\n”;
}
```


Körning

```
> Jag tänker på en siffra mellan 1-10. Gissa vilken: 8
Fel Fel Fel!!!
Nu vill jag inte leka mer! Hejdå!

> Jag tänker på en siffra mellan 1-10. Gissa vilken: 9
9! Nästan helt rätt!
Nu vill jag inte leka mer! Hejdå!

> Jag tänker på en siffra mellan 1-10. Gissa vilken: 10
10! Helt Rätt!
Nu vill jag inte leka mer! Hejdå!
```


Iterationer

while - Så länge som

```
print ”Vilket tals fakultet ska jag räkna fram åt dig: ”;
$fak_tal = <STDIN>;
chomp($fak_tal);
$raknare = 1;
$tot_fak = 1;
while ($raknare <= $fak_tal) {
        $tot_fak *= $raknare;
        $raknare++;
}
print ”$fak_tal fakulteten är $tot_fak.\n”;
```


Iterationer, forts

until - Tills

```
print ”Vilket tals fakultet ska jag räkna fram åt dig: ”;
$fak_tal = <STDIN>;
chomp($fak_tal);
$raknare = 1;
$tot_fak = 1;
until ($raknare > $fak_tal) {
        $tot_fak *= $raknare;
        $raknare++;
}
print ”$fak_tal fakulteten är $tot_fak.\n”;
```


do {..} while/until

```
print ”Jag tänker på en siffra mellan 1-10. Gissa vilken: ”;

do {
        $a = <STDIN>;
        chomp;
        if ($a != 10) { print ”Fel fel fel! Försök igen:”;}
} while ($a != 10);
print  ”Äntligen rätt!!”;
```

```
do { …
} until ($a == 10);
```

for - bra när något ska göra ett visst antal gånger

```
print ”Hur många gånger ska jag skriva ’hej’?  ”;
$a = <STDIN>;
chomp($a);
for ($i = 0, $i <= $a, $i++) {
        print ”hej\n”;
}
```

```
> Hur många gånger ska jag skriva ’hej’? 4
hej
hej
hej
hej
```


Listor

(’a’,’e’,’i’,’o’,’u’,’y’,’å’,’ä’,’ö’)

@vokaler = (’a’,’e’,’i’,’o’,’u’,’y’,’å’,’ä’,’ö’);

(’a’,’e’,’i’,’o’,’u’,’y’,’å’,’ä’,’ö’)
0 1 2 3 4 5 6 7 8

Listor forts

$vokaler[0] är första elementet på listan @vokaler. **$vokaler[1]** är andra, etc

@vokaler[2,3] är en ’dellistan’ av listan @vokaler som innehåller de element vars **INDEX** är 2 och 3. (dvs (’o’,’u’)).

Tar man ut delar av listorna MÅSTE man tänka på om det handlar om **skalär-** eller **listdata** och ange prefix efter det!!! Tänk på kontexten!!!

```
$variabel[0];     # skalär kontext
@variabel[2,3];# listkontext
```


”Bygga upp” listor

```
@ental = (0 .. 9);
# @ental blir (0,1,2,3,4,5,6,7,8,9)

@sv_vok = qw(å ä ö);
# @sv_vok blir (’å’,’ä’,’ö’)
@vokaler = (’a’,’e,’i’,’o’,’u’,’y’,@sv_vok);
# @vokaler blir (’a’,’e,’i’,’o’,’u’,’y’,’å’,’ä’,’ö’)
```

Perllistor kan INTE innehålla andra listor!!
Listorna ”plattas ut” till en lista!!!

Listor - tilldelning

jmf:

```
$ett = 1;
# en skalär variabel med värdet ’1’. I Prolog:
atomen 1.

@ett = 1;
# en listvariabel med värdet av en lista med bara
ett element, detta element är ’1’, dvs lista ’(1)’. I
Prolog: listan [1].
```


Listors längd och sista element

```
@a = (’a’,’b’,’c’);

$a = @a;  # $a blir 3, dvs @a:s LÄNGD

$s_index = $#a; # $s_index blir 2, som är
      #index för listan @a sista element!!!

$s_element = $a[$#a];  # $s_element blir ’c’
```

Arbeta med listor

```
@elever = qw(anna pelle lisa);
@bup = @elever;
Anna slutar:
($slut,@elever) = @elever;
Lotta börjar:
@elever = (@elever,'lotta');
Lotta vill stå först:
(@elever,$jag) = @elever;

@elever = ($jag,@elever);
```

@elever är:
('anna','pelle','lisa')
@bup är:
('anna','pelle','lisa')

('pelle','lisa')
('anna','pelle','lisa')
('pelle','lisa','lotta')
('anna','pelle','lisa')

('pelle','lisa')
('anna','pelle','lisa')
('lotta','pelle','lisa')
('anna','pelle','lisa')

Listfunktioner

@elever = qw(anna pelle lisa); # Quote Words

Push & Pop - Påverkar listan från HÖGER

Lisa slutar:
pop(@elever);
#@elever är ('anna','pelle')

Lotta börjar:
push(@elever,'lotta');
#@elever är ('anna','pelle','lotta')

Shift & Unshift - Påverkar listan från VÄNSTER

Anna slutar:
shift(@elever);
#@elever är ('pelle','lisa')

Lotta börjar och vill stå först:
unshift(@elever,'lotta');
#@elever är ('lotta','pelle','lisa')

Listfunktioner

@elever = qw(anna pelle lisa);

Reverse - vänder ordningen på listans element

```
@revele = reverse(@elever);
# @revele blir ('lisa','pelle','anna')
@revele = reverse('anna','pelle','lisa');
# samma som ovan
$baklanges = reverse('lisa är glad');
# $baklanges blir 'dalg rä asil'
```

Sort - ordnar elementen alfabetiskt

```
@alf_elever = sort(@elever);
# @alf_elever blir ('anna','lisa','pelle')
```


Foreach

Foreach är en iterationsfunktion speciell för listor. Den går genom varje element i lista och slutar när alla element har behandlats!

Listfunktioner

```
print  "Nu ska vi börja lägga in dina elever i en lista!\n";
do {
        print "Ange namn på en elev: ";
        $eleven = <STDIN>;
        push(@elever,$eleven);
        print "Har du fler elever? (j/n): ";
        $svar = <STDIN>;
        chomp($svar);
} while ($svar eq 'j');

print "Du har mata in färdigt, nu ska vi skriva ut dina elever!\n";
chomp(@elever);
foreach $elev (sort(@elever)) {
        print "$elev \n";
}
```


Körning

```
1: Nu ska vi börja lägga in dina elever i en lista!
2: Ange namn på en elev: pelle
3: Har du fler elever? (j/n): j
4: Ange namn på en elev: anna
5: Har du fler elever? (j/n): j
6: Ange namn på en elev: lisa
7: Har du fler elever? (j/n): n
8: Du har matat in färdigt, nu ska vi skriva ut dina elever!
9: anna
10: lisa
11: pelle
```

Efter inmatningen är @elever ('pelle\n','anna\n','lisa\n'). Efter chomp är @elever ('pelle','anna','lisa').

Att läsa in från en fil och skriva till en fil!

```
open(IN,infil);
open(UT,'>utfil');
$i = 1;
while (<IN>) {
        chomp;
        $a = $_;
        print UT "rad $i är: $a\n";
        $i++;
}
close(UT);
close(IN);
```

$_ är en special-variabel vars värde är "current input". Många funktioner använder den som default.

Omvandling

Sträng till Lista - split

```
@ord = split(' ','hej, hur mår du?');
 # @ord blir ('hej,','hur','mår','du?')
@suc = split('>','n=1>Att>IE>att>')
# @suc blir ('n=1','Att','IE','att')
```

Lista till sträng - join

```
$hälsning = join(' ',@ord);
 # $hälsning blir ' hej, hur mår du?'
$ny_suc = join(' ',@suc);
# $ny_suc blir 'n=1 Att IE att'
```



```
while (<IN>) {
      chomp;
      $rad = $_;
      if (/^<w/) {        # är sann om raden börjar på '<w '

         s/<w //;         # tar bort '<w '
         s/<ana><ps//;  #tar bort '<ana><ps'
         s/<m//;          #tar bort '<m'
         s/<b//;          # tar bort '<b'
         s/<\/w//;        # tar bort '</w'
```

Raden '<w n=1>Att<ana><ps>IE<b>att</w>' ser nu ut 'n=1>Att>IE>att>'. Den kan splittras som ovan.

```
@rad = split('>',$rad);
```