

Musterlösung zu Blatt 12:

1. Es ist eine Zahl x gegeben, schreiben Sie ein Programm, das die Fakultät der Zahl $= x * (x-1) * (x-2) * \dots * 1$ berechnet (z.B. $fak(4) = 4*3*2*1$). Bringen Sie Ihren Rechner zum 'Schwitzen' und testen Sie mal, bei welcher Zahl der Rechner die Fakultät nicht innerhalb von 2 Minuten berechnet.

```
#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: Fakultaetsrechner rekursiv
use strict;
use utf8;
use locale;

{
my $zahl;

print "Bitte geben Sie eine Zahl (groesser als 0) ein:>>>\n";
chomp($zahl = <>);
my $erg = &fak($zahl);
print "Die Fakultaet Ihrer Zahl lautet: $erg \n";
}

sub fak($){
    my $zahl=$_[0];
    if ($zahl == 0){
        return 1;
    }else{
        return ($zahl * &fak($zahl -1));
    }
}
```

2. Es ist eine Zahl x gegeben, schreiben Sie ein Programm, das die Summe der Zahlen $= x + (x-1) + (x-2) + \dots + 1$ berechnet (z.B. $sum(4) = 4+3+2+1$)

```
#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: Summenrechner rekursiv
use strict;
use utf8;
use locale;

{
my $zahl;

print "Bitte geben Sie eine Zahl (groesser als 0) ein:>>>\n";
chomp($zahl = <>);
my $erg = &sum($zahl);
print "Die Summe Ihrer Zahl lautet: $erg \n";
}
```

```

sub sum($){
    my $zahl=$_[0];
    if ($zahl == 0){
        return 0;
    }else{
        return ($zahl + &sum($zahl -1));
    }
}

```

3. Es sind eine Zahl x und eine Liste von Zahlen @liste gegeben. Schreiben Sie ein Programm, das testet, ob die Zahl x in @liste vorkommt.

```

#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: Testet ob Element in Liste ist rekursiv
use strict;
use utf8;
use locale;

{
    my $zahl;
    my @liste = (1,2,3,4,5);

    print "Bitte geben Sie Ihre Zahl ein:>>>\n";
    chomp($zahl = <>);

    my $erg = &testInListe($zahl, @liste);

    if ($erg == 1) {
        print "Ihre Zahl ist in der Liste.\n";
    }
    elsif ($erg == 0) {
        print "Ihre Zahl ist nicht in der Liste.\n";
    }
}

sub testInListe($@){
    my ($zahl,@liste) = @_;

    if (scalar(@liste) == 0){
        return 0;
    }
    if ($zahl == $liste[0]){
        return 1;
    }
    shift(@liste);
    return (&testInListe($zahl,@liste));
}

```

4. Führen sie die Multiplikation zweier Zahlen auf die Addition zurück: $2 * 3 = 2 + 2 + 2$. Schreiben Sie ein rekursives PERL Programm, das dies realisiert.

```
#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: Multipliziert zwei Zahlen rekursiv
use strict;
use utf8;
use locale;

{
my ($zahl1, $zahl2);

print "Bitte geben Sie Ihre Zahl ein:>>>\n";
chomp($zahl1 = <>);

print "Bitte geben Sie Ihre Zahl ein:>>>\n";
chomp($zahl2 = <>);

my $erg = &multi($zahl1, $zahl2);

print "Ihr Ergebnis lautet $erg.\n";
}

sub multi($$){
    my ($zahl1, $zahl2) = @_;
    if($zahl2 == 0) {
        return ($zahl1, 0);
    }
    else {
        return ($zahl1 + &multi($zahl1, ($zahl2-1)));
    }
}
}
```

5. In einer Liste @zahlen sind 10 Zahlen gespeichert. Schreiben Sie eine subroutine print_second(@zahlen) die jede zweite Zahl der Liste mit print ausgibt

```
#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: gibt jede zweite Zahl der Liste aus
use strict;
use utf8;
use locale;

{
my @zahlen = (1,2,3,4,5,6,7,8,9,10);
&print_second(@zahlen);
}
```

```

sub print_second(@){
    my @liste = @_;
    for (my $i = 1; $i < scalar(@liste); $i = $i+2){
        print "$liste[$i]\n";
    }
}

```

6. In einer Liste @zahlen sind 10 Zahlen gespeichert. Schreiben Sie eine subroutine my_reverse(@zahlen) die die Reihenfolge der Zahlen in der Liste umdreht.

```

#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: dreht die Reihenfolge der Zahlen in der Liste um
use strict;
use utf8;
use locale;
{
    my @zahlen = (1,2,3,4,5,6,7,8,9,10);
    my @zahlen_umgedreht = &my_reverse(@zahlen);
    foreach my $elem (@zahlen_umgedreht) {
        print "$elem \n";
    }
}

```

```

sub my_reverse(@){
    my @liste = @_;
    my @liste_umgedreht;
    for (my $i = scalar(@liste)-1; $i >= 0; $i--) {
        push(@liste_umgedreht, $liste[$i]);
    }
    return @liste_umgedreht;
}

```

7. In einer Liste @zahlen sind 10 Zahlen gespeichert. Schreiben Sie eine subroutine del_second(@zahlen) die jede zweite Zahl aus der Liste entfernt die Reihenfolge der Zahlen in der Liste umdreht.

```

#!/usr/bin/perl
# Autor: Nicola Greth
# Programm: entfernt jedes zweite Element der Liste und dreht die Reihenfolge
um
use strict;
use utf8;
use locale;
{
    my @zahlen = (1,2,3,4,5,6,7,8,9,10);
    my @zahlen_umgedreht = &del_second(@zahlen);
    foreach my $elem (@zahlen_umgedreht) {
        print "$elem \n";
    }
}

```

```
sub del_second(@){  
  my @liste = @_;  
  my @liste_umgedreht;  
  for (my $i = scalar(@liste)-1; $i >= 0; $i = $i-2) {  
    push(@liste_umgedreht, $liste[$i]);  
  }  
  return @liste_umgedreht;  
}
```