Schreibe ein möglichst kurzes Programm in einer beliebigen Programmiersprache, welches das Wort „secunet“ ausgibt; und zwar entweder in kreativer Weise (z.B. als Laufschrift o.ä.), oder in kreativer Implementierung (z.B. ohne die Buchstaben des Wortes „secunet“ in einer Ausgabefunktion zu verwenden), oder beides.

## Die Aufgabe

secunet

*Chris Amthor* Hamburg

Essen *Stefan Dringenberg*

*Nils Magnus* Hamburg

Essen *Carsten Muck*

*Christian Perle* Dresden

Berlin *Frank Rustemeyer*

*Thomas Schoch* Essen

Hamburg *Andreas Soller*

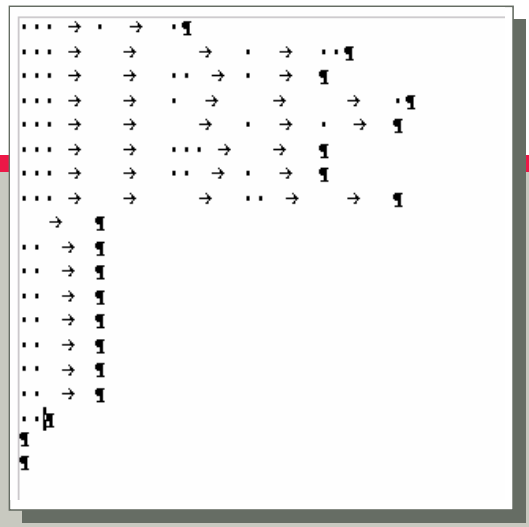
*Ulrich Tipp* Essen

Essen *Sascha Ziemann*

# Die Teilnehmer

Sprache: Whitespace
Autor: Chris Amthor
Ausgabe: secunet

*Code im Editor:*



# *Network Solutions and Support souffle.*

## *Ingredients.*

*115 blended SINA cds*
*101 g kwg24c garlic*
*99 firewall sausages*
*117 ml et player blood*
*110 g VPN apples*
*101 article 90 onions*
*116 old alfredos*
*10 l muellermilch*

## *Method.*

*Put muellermilch into the mixing bowl. Put old alfredos into the mixing bowl.*
*Put article 90 onions into the mixing bowl. Put VPN apples into the mixing bowl.*
*Put et player blood into the mixing bowl. Put firewall sausages into the mixing bowl.*
*Put kwg24c garlic into the mixing bowl. Put blended SINA cds into the mixing bowl.*

*Liquify contents of the mixing bowl.*
*Pour contents of the mixing bowl into the baking dish.*

## *Serves 1.*

Sprache: Chef
Autor: Stefan Dringenberg
Ausgabe: secunet

```
i,l;main(I,_)char**_;{for
(i=I=0; l=strlen(*(_+1));
i++) {if(!(i%l)) putchar(
 I++? '\r':'\r'); putchar
    _[1][(i+I)%l]);}}
```

Sprache: C

Autor: Nils Magnus

Aufruf: `./l secunet`

Ausgabe: secunet

*(Laufschrift)*

secunet

# secunet

Sprache: secunet © Nils Magnus
Autor: Nils Magnus
Aufruf: `secunet secunet`
Ausgabe: secunet

*secunet-Interpreter by Nils Magnus:*

*Version in C:*

```
int main(int argc, char **argv)
{if (!strcmp(argv[0], argv[1]) puts(argv[1]);}
```

*Version in Perl:*

```
#!/bin/perl -w
$secunet = shift;
print $secunet if ($secunet ~= /secunet/);
```

*Version in MICROSOFT BASIC V2.0*

```
10 INPUT S$
20 IF S$ = "secunet" THEN PRINT S$
30 END
```

```
Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook. Ook?
Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook.
Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook. Ook. Ook? Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook. Ook.
Ook. Ook. Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook. Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
Ook! Ook? Ook? Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook? Ook! Ook!
Ook? Ook! Ook? Ook. Ook! Ook. Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook! Ook. Ook. Ook? Ook. Ook?
Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook! Ook.
```

Sprache: Ook!
Autor: Carsten Muck
Ausgabe: `secunet`

```
l=A/D1E6E3W,ROFL1DT4Q5D.EZZ1T4U3B+C1WX6LJM5N-X1/5LOL6V:R1A4Y5X/_RGGBL1ZX6Z4I
while [ ${#l} -gt 0 ];do s=${l:1:1};case $s in [/A-Z/]) ;; *) x=$x$s ;; esac
l=${l#?};done;IFS=",.+-:_";set -- $x;IFS=" _";for l in $*;do s="$s"\\$l;done
echo -e "$s" ; : : : : : : : : : : : : : : : : : : chris@linuxinfotag.de
```

Sprache: Bash

Autor: Christian Perle

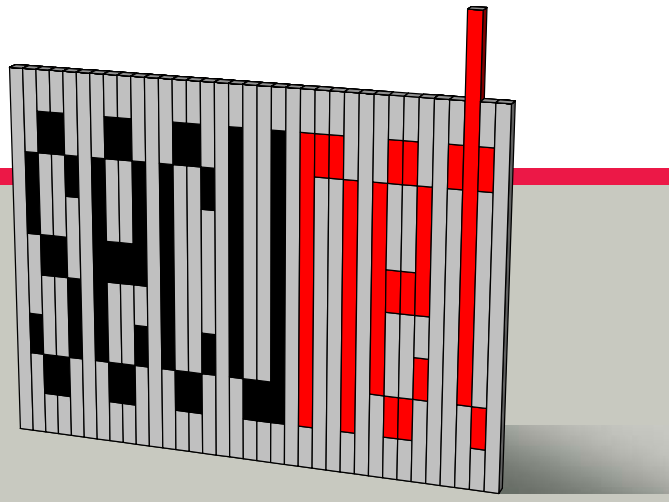
Ausgabe: secunet

```
(1)       DON'T IMPLEMENT CONFIDENCE
(2)       DO FORGET #1
          DO READ OUT ,1
          DO GIVE UP

          PLEASE COME FROM (1)
          DO ,1 <- #8

          DO ,1SUB#1 <- #50
          DO ,1SUB#2 <- #40
          PLEASE DO ,1SUB#3 <- #224
          DO ,1SUB#4 <- #24
          DO ,1SUB#5 <- #56
          PLEASE DO ,1SUB#6 <- #208
          DO ,1SUB#7 <- #120
          DO ,1SUB#8 <- #222

          DO (2) NEXT
```

Sprache: INTERCAL
Autor: Frank Rustemeyer
Ausgabe: secunet

```
9 2 1 1 2 9 2 1 1 2 9 2 1 1 2 9 2 1 1 1 9 1 7 7 1 9 2 1 1 2 9 7 2 1 9
0 1 1 1 2 0 5 1 1 1 0 5 1 1 1 0 6 1 1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 2 2 2 0 2 2 2 1 0 2 5 5 3 0 1 7 7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 1 1 1 0 0 1 1 3 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 2 2 2 0 0 2 2 2 0 0 1 1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 1 1 6 0 5 1 1 1 0 1 9 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 2 0 2 2 2 1 0 1 0 5 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 3 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0
```

secunet



Sprache: Excel + VBA

Autor: Frank Rustemeyer

Ausgabe: *(siehe rechts)*

```
int buchstabe(float f)
{    if (f > 6) { return '\n'; }
     printf ("%c", (char) (115.5 +
            (f*3067)/60 -
            (f*f*154971)/1080 +
            (f*f*f*108) –
            (f*f*f*f*411)/12 +
            (f*f*f*f*f*293)/60 –
            (f*f*f*f*f*f*31)/120));
     return buchstabe(f+1);
     }

void main(void)
{    printf("%c", buchstabe(0));
     }
```

Sprache: C
Autor: Frank Rustemeyer
Ausgabe: secunet

```
;3%5;3%5;3%5;2 ;5 ; ;%4;3%5;4 ;5
;2 ;open _ ,$0;2 ; ;5 ; ;3;2 ;1%2;4 ; ;5 ;3 ;5
;2 ;7 ;5 ; ;;while (<_>) { ;5 ; ;5 ;1%6;2
;3%5;2%7; ;7 ;2;s;%; ;g;3 ;1%7;2; ;5
;8 ; ;;s'([; ])(\d)' $1x$2 ; 'ge;; ; s'\S';'g; ;4;
s; ;#;g; ; ;5 ; ;5 ; ;5 ; ;5 ; s;\;; ;g; ;3 ;
;3%5;3%5;3%5;3%5;; print ;}%5;5%3;2
```

Sprache: Perl

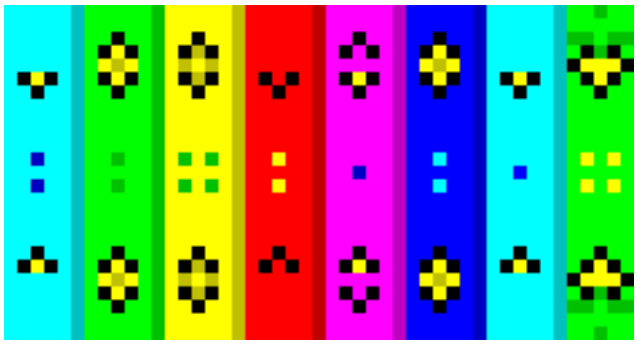
Autor: Thomas Schoch

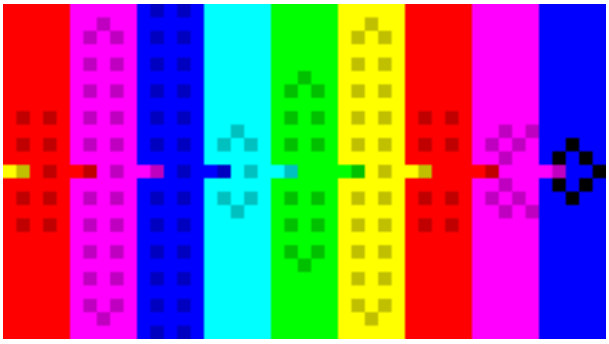
Ausgabe:

```
 #####   #####   #####  #    # # ####   #####    #
#     # #     # #     # #    # ##    # #     #   #
#       #     # #       #    # #    # #     # ######
 #####  ####### #       #    # #    # ####### #
      # #       #       #    # #    # #       #
#     # #     # #     # #    # #    # #     #   #   #
 #####   #####   #####   #####  #    #  #####    ###
```

```
# ================================================================ #
  /[+-]/&&eval"\$c$_$_"||ord==46&&print(chr($c))while($_=getc(DATA))
# ================================================================ #

                                __DATA__
                                                                  ++
 +++++++   +++++++   +++++++  ++     ++ + ++++++   +++++++  ++++++++
++     ++ ++     ++ ++     ++ ++     ++ +++    ++ ++     ++   ++
++        ++     ++ ++        ++     ++ ++     ++ ++     ++   ++
 +++++++  +++++++++ +.        --     -- --     -- ------.--   .+
       ++ ++        ++        ++     ++ ++     ++ ++           +.
--     -- --     -. --     -- --     -- -.     ++ ++     ++   ++   ++
 +++++.-   -------   -------   -------  --     --  -------     -----

 ----------------------------------------------------------------------.
```

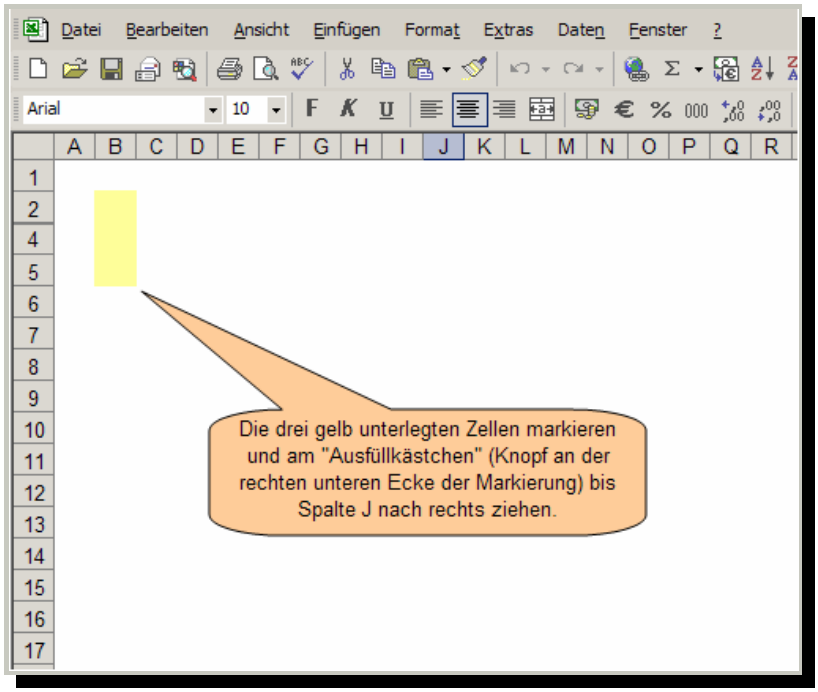
Sprache: Perl, Brainf*** *(2fach polyglott)*
Autor: Thomas Schoch
Ausgabe: `secunet`





Sprache: Piet
Autor: Thomas Schoch
Ausgabe: `secunet`
`secunet`

Sprache: Piet
Autor: Thomas Schoch
Ausgabe: `secunet`




Sprache: Excel

Autor: Thomas Schoch

Ausgabe:

| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | | | | | |
| 2 | | | | | | | | | | | |
| 4 | | s | e | c | u | n | e | t | | | |
| 5 | | | | | | | | | | | |
| 6 | | | | | | | | | | | |

```
%q=(q<{>=>q<+>,q{>}=>q=-=),$_=q({73>e>2{12>7>9{f>6a),$p=q
{print(chr($c},s{(.)(\w+)}[$p$q{$1}=hex(qq{$2}))),]g,eval
```

Sprache: Perl
Autor: Thomas Schoch
Ausgabe: `secunet`

# secunet

*Asthma-Interpreter by Thomas Schoch:*

```
#!/usr/bin/perl

@buf = ();
interpret ($_) while (<STDIN>);

sub interpret {
  $_ = shift;
  $ptr = 0;
  for (split //) {
    $buf[$ptr] = $_ if not ($hidden or /h/);
    $c = $_;
    $_ = lc $_;
    if ($nop) {
      $buf[$ptr] = $c if /h/;
      $nop = 0;
    } else {
      @undo = @buf if not /u/;
      /s/ and $ptr = 0;
      /n/ and $nop = 1;
      /h/ and $hidden = 1;
      /t/ and do { print $_ for @buf, "\n"     };
      /u/ and do { @buf = @undo; push @buf, $_ };
      /c/ and do { $_ = " " for @buf[0..$ptr]  };
      /e/ and do { interpret (@buf[0..$ptr-1]) };
    }
    $ptr++ if not $hidden;
    $hidden = 0 unless /h/;
  }
}
```

Sprache: Asthma © Thomas Schoch

Autor: Thomas Schoch

Ausgabe: secunet

```
#!/bin/ksh

C=2#11100000 e=$(print '\\033[')
p=${0##.*.} && while ((${#p}>0))
do typeset -R${#p} s=&& x=${p#?}
c=${p%$x} p=$x; print -n "${e}1\
;3$(((C>>=1)&1))m$c${e}0m" >$s$c
done && print>>$s$c; eval echo *

cat *
```

Sprache: Kornshell
Autor: Thomas Schoch
Aufruf: `./.secunet`
Ausgabe:
```
s e c u n e t
secunet
```

```
eval " " ;      eval 'exec      perl $0‘_______________
if $_ > $_++    ||      $_++    == ++$_;        @_ = q
([>+++ + + [    <       < + +   + +     >_______________
>- ] < -        ]       < -     -       -       - -___
.> + + [        <       -       - -     - -     -_____
-> - ] <        .       - - . > +       +_______________
[> + + +        [       < < +   + +     >_______________
>- ] < -        ]       <       . - -   -       -_____
--      -_____________________________________________
.- - -  -_____________________________________________
-- - -  -_____________________________________________
.> + +  +_____________________________________________
[< + +  +_____________________________________________
++ > -  ]_____________________________________________
<. > +  +_____________________________________________
++ [ >  +_____________________________________________
++ + +_______________________________________________
[<_____________________________________________________
<-_____________________________________________________
---->>-]<-]<------);print`./bf<$0`;print`./ws<$0`#____
```

Sprache: Shell, Perl, Brainf***, Whitespace *(4fach polyglott)*

Autor: Thomas Schoch *(Brainf***: Sascha, Whitespace: Chris)*

Ausgabe: `secunet`

`secunet`

```
# define  o(o) printf(o) /*
sub o {print(reverse((split
(//,shift))[0..6]) . "\n")}
o `cc -oo $0>&-&&./o;rm o`;
o <<'main()';
tenuces */
main()
{ o("tenuces"); }
```

Sprache: C, Perl *(2fach polyglott)*
Autor: Thomas Schoch
Ausgabe: secunet
secunet
secunet

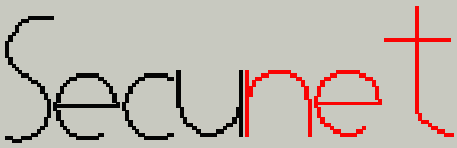
```
print(($_)eq'.'?qq(\n):chr(ord($_)+13*(ord($_)109?-1:1
)))for(split//,(reverse(split(m([\\/]|pl),$0)))[0]));
```

Sprache: Perl
Autor: Thomas Schoch
Aufruf: `./frpharg.pl`
Ausgabe: `secunet`

```
setbg 7 setpc 0 pu rt 20 fd 10 pd arc 180 10 pu
fd 20 rt 180 pd arc 180 10 pu fd 30 lt 20 lt 90
fd 15 lt 90 fd 10 rt 90 fd 10 rt 60 pd arc 300
10 pu rt 120 fd 10 rt 180 pd fd 20 pu fd 2 rt 90
fd 10 rt 180 fd 10 rt 90 fd 10 rt 60 pd arc 240
10 pu fd 10 lt 60 fd 2 lt 90 fd 10 rt 90 fd 10
pd arc 180 10 pu fd 10 lt 90 fd 10 rt 180 pd fd
20 pu rt 90 fd 20 rt 90 fd 10 pd fd 10 pu rt 180
fd 20 lt 90 fd 20 fd 2 lt 90 setpc 4 fd 20 rt 90
fd 20 rt 90 fd 10 rt 90 fd 10 pd arc 180 10 pu
fd 10 lt 90 fd 10 rt 180 pd fd 20 pu rt 90 fd 20
rt 90 fd 10 pd fd 10 pu rt 180 fd 20 lt 90 fd 20
fd 2 lt 90 pd pu fd 20 lt 90 fd 20 fd 2 fd 2 lt
90 fd 10 rt 90 fd 10 rt 60 pd arc 300 10 pu rt
120 fd 10  rt 180 pd fd 20 pu fd 2 rt 90 fd 10
rt 180 fd 10 rt 90 fd 20 rt 90 pd arc 90 10 pu
rt 90 fd 10 rt 90 pd fd 30 pu rt 180 fd 10 rt 90
fd 10 rt 180 pd fd 20 pu fd 2 rt 90 fd 30 rt 180
```

Sprache: Logo

Autor: Andreas Soller

Ausgabe: 

```
a:::*+32*+#v:#+3#*5#3*#3-#:<
                >#v:#-3#*4#6+#3+#:<
                   >#v:#+3#*1#5-#3+#:<
                      >1-:v
                         v_@
                          ,:
                          >^
```

Sprache: befunge
Autor: Ulrich Tipp
Ausgabe: secunet

```
%!
% Buchstabenhöhe = Umsatz
% Buchstabenbreite = Anzahl Mitarbeiter
% Schieflage = Jahresergebnis

/hs 7 def /ws 8.6 def
/he 11.1 def /we 14.9 def
/hc 18 def /wc 24 def
/hu 22.4 def /wu 20.4 def
/hn 23.1 def /wn 18.1 def
/he2 21.1 def /we2 17.3 def
/ht 17.2 def /wt 21 def
72 700 moveto

/basefont /Lucida-Console findfont 3 scalefont def
/space 30 def

basefont [ws 0 -0.7 hs space 0] makefont setfont (s) show
basefont [we 0 -2.5 he space 0] makefont setfont (e) show
basefont [wc 0 -3.4 hc space 0] makefont setfont (c) show
basefont [wu 0 -4.8 hu space 0] makefont setfont (u) show
basefont [wn 0 0.3 hn space 0] makefont setfont (n) show
basefont [we2 0 -4.5 he2 space 0] makefont setfont (e) show
basefont [wt 0 0 ht space 0] makefont setfont (t) show
showpage
```

Sprache: PostScript
Autor: Ulrich Tipp
Ausgabe:

secunet

```
open IN, "$0";while(<IN>) { if($.>1)
   {s/[^\ \/|\\\[<>_\n]//g;print}
   else {s/.*//}};

$x=                      __LINE__;

if($x<3) { $a[0]=
  s%/a |b |c |d\t | f%%;
  $a[1]= m(a |b);}

if( $x>3) {$a[0]=
  s%\f* \wxvd_uifgs/|c \w|r%%;
  $a[1]=m(c  |d);}
```

Sprache: Perl

Autor: Ulrich Tipp

Ausgabe:

```
<
 > [ / | | |\| [ ___
   [ \ \_/ | \| [  |
```

```
set in [open $argv0 r]; gets $in
gets $in; set t [gets $in];
set c $t
gets $in; puts "[join [split
  [join [split $c  "$[string
  index $c end]"] ""]] ""][join
  [split [lindex [split [gets
  $in] ] 0]
  [string index $c 0] ] ""]"
unset c
```

Sprache: TCL
Autor: Ulrich Tipp
Ausgabe: secunet

secunet

# secunet Security Networks AG

*eu-Interpreter by Ulrich Tipp:*

```
#!/usr/bin/perl

open (in, $ARGV[0]) ||
  die "Can't open scriptfile $ARGV[0]!";
while (<in>) {
  @c = split;
  while ($#c>=0) {
    $_ = shift @c;
    if ($_ eq "AG") {
      while ($_=shift @out) {print "\L$_";}
      print "\n\033[0m";
    } else {
      while (s/u//) {
        @_ = ($',join ("", reverse( split "",$`)));
        while ($_ = shift @_) {
          while (s/([cn])e(.?)//) {
            $d = shift @c;
            $_ = $` .
              (($1 eq "c")?"\033[31m":"\033[0m") .
              substr( $d,0,ord($2)-112) . $';
          }
          push @out,$_;
        }
      }
    }
  }
}
```

Sprache: eu © Ulrich Tipp

Autor: Ulrich Tipp

Ausgabe: secunet

```
exec 5<>/dev/tcp/195.226.172.54/80
echo -e 'GET / HTTP/1.0\r\n\r\n' >&5
sed -n 's/^<title>\([a-z]\+\).*/\1/p'<&5
```

Sprache: Bash + HTTP

Autor: Sascha Ziemann

Ausgabe: secunet

```
      >++++++[>+
    ++++[<<++++>>-
  ]<-]<-----.>++[<--
  -----          >-]<.
  --.>++[
    >+++[<<++
      +>>-]<-]<.
          -------.-
                -------
  -.>++           +[<++
  +++>-]<.>++++[>+++
    ++[<<-----<<-]
      <-]<------.
```

Sprache: Brainf***
Autor: Sascha Ziemann
Ausgabe: secunet

```
(for-each
 (lambda (x)
   (display (integer->char x)))
 (map
  (lambda (l)
    (apply * (map
              (lambda (x)
                (list-ref (list 2 3 5 11 13
                                23 29 101)
                          x))
              l)))
  '((5 2) (7) (3 1 1) (4 1 1)
    (3 2 0) (7) (6 0 0) (2 0))))
```

Sprache: Scheme
Autor: Sascha Ziemann
Ausgabe: secunet