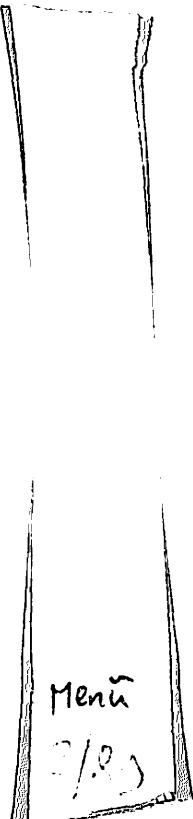


Diskette/Platte in Laufwerk C besitzt den Kennsatz S01
Verzeichnis von C:\CLV\VF\C16

.	<VER>	28.06.89	16.02
..	<VER>	28.06.89	16.02
MENUE	174080	27.02.89	13.03
EH-TERM 387	174081	26.02.89	22.07
UF38-1	174080	26.02.89	22.50
UF38-2	174080	26.02.89	23.40
UF38-3	174080	27.02.89	0.25
UF38-4	174080	27.02.89	2.48
TC38	174080	27.02.89	10.19
EDIT	174080	27.02.89	11.17
DSK-GRF 487	174080	27.02.89	13.55
EXE	<VER>	28.06.89	16.20
12 Datei(en)		9211904 Bytes noch frei!	



owner='S01:WK:HFI'

SSSSS	00	1		W	W	K	K	H	
S	S	0 0	11	W	W	K	K	H	
S	0	0	1 1	**	W	W	K	**	
S	0	0	1	**	W	W	K	**	
SSSSS	0 0	1		W	W	K K		HHHHHE	
S	0	0	1	W	W	W	KKK	H	
S	0	0	1	**	W	W	W	**	
S	0	0	1	**	WW	WW	K K	**	
S	S	0 0	1		W	W	K	K	H
SSSSS	00	111		W	W	K	K	H	

Despooler of 3Share - version 1.3.1

file spooled: 28-jun-89, 16:22:52
file printed: 28-jun-89, 16:40:32
copies=1, priority=50, form type=1

0

85

1

86

0	\ Directory Menue	clv18aug88
1		
2	d- split join	2
3	VArray	3
4	linked List	4 s
5	col! cole	5
6	scr! scrd	6
7	scrPush (test)	7
8	Menue	10
9	Screen UtI	13
10	Item Activate	30
11	Zinseszins	35
12	Data VArray	50-60
13		
14	AutoDoc	71
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		

2

87

```
0 \ d- split join clv06jul188
1
2 : d- dnegate d+ ;
3
4 Code split ( n-low high)
5     sp }y lda N sta
6     txa sp }y sta
7     N lda push0a jmp end-code
8
9 Code join   ( low high--n)
10    sp x} lda N sta sp zinc N lda
11    sp }y sta next jmp end-code
12
13
14
15
16
17
18
19
20
21
22
23
24
```

3

88

```

0 \ vArray - virtuelles Array clv06jul88 \ vArray - virtuelles Array clv03jul88
1
2 Create vRange 0 , 0 ,                                \ Beispiel:
3
4
5 : vfree ( --bytes)                                $1000 vArray memory
6 vRange 2+ @ vRange @ - ;                          $c00 Constant scrBeg
7
8 : vArray ( bytes--) \ name follows                $100 Constant scrLen
9 dup vfree u> abort" no vRoom"
10 Create vRange @ vRange +!                         : saveScreen ( no--)
11 does> ( bytes--addr)                           scrLen * memory
12 @ + $400 u/mod block + ;                      scrBeg swap scrLen cmove update ;
13
14 \| alle Adressen max 64kB !!!                  : getScreen ( no--)
15
16
17
18
19
20
21
22
23
24

```

4

89

5

90

0 \ col! col@ clv10jul88 VORNAME @
1 NAME @
2 ADR. ZUS NACH
3 \ Problem: bkgCol aendert ganzen Schirm ATZ Z.B. BEI MEIER, PF 0104 @ STRASSE &
4 \$53b Constant penCol HAUS-NR @ LAND @
5 \$ff15 Constant bkgCol
6
7 : col! { bkg pen--}
8 : penCol c! bkgCol' c! ; WOHNORT @
9 : col@ { --bkg pen} ZUSTELLBEZIRK @
10 : bkgCol c@ penCol c@ ; TELEFON-NR. @
11 : norm \$f6 \$80 col! ; ERSATZADR F. UMZUG (Z.B. ELTERN..) @ FIR
12 : BIS JAHR @
13 : .col col@ swap . ." col!" ;
14 : coltab BEMERKUNGEN @
15 : \$100 0 DO cr I penCol c! I &5 u.r LOOP STICHWORT 1 (FUER REGISTER)
16 :
17 :
18 :
19 : STICHWORT 2 (FUER REGISTER)
20 :
21 :
22 :
23 :
24 :
;

6

```

0 \ SU: scr@!
1
2 &40 Constant cols
3
4 : scr@ ( y x --n1 n2) swap cols * +
5   dup $800 + c@ swap $c00 + c@ ;
6
7 : scr! ( n1 n2 y x--) swap cols * +
8   under $c00 + c! $800 + c! ;
9
10 \\ fuer Vorsichtige:
11
12 : ?scr { adr--adr)
13   dup $800 $1000 uwithin ?exit
14   u. true abort" oh,n!!";
15 : scr! { n1 n2 y x--}
16   at? &15 0 at .s at
17   swap cols * +
18   under $c00 + ?scr c! $800 + ?scr c! ;
19
20
21
22
23
24

```

91

```

0 \ SU:
1
2 \\ will alles nicht
3 &40 Constant cols
4
5 forth forget cpush
6 : cpush 2dup 20 0 at
7   key 3 = abort" break"
8   2dup [ tools ] cpush 5 fill ;
9
10
11 : scrpush { y x h w--}
12   2swap swap cols * + $c00 + swap
13   ( h adr w )
14   BEGIN rot dup WHILE 1- -rot
15     2dup cpush under + swap
16     REPEAT 2drop drop ;
17
18
19
20 \\
21 debug scrpush
22
23 5 6 7 8 scrpush
24

```

92

```

0 \ MN: sample
1
2 &2 &06 &3 &10 Menue: doOne
3 ." 1 is One" ;
4
5
6 &12 &04 &6 &09 Menue: doTwo
7   0 4 at ." 2" 1 3 at ." is"
8   2 2 at ." two" ;
9
10 0 0 &25 &40 ww: wholeScreen
11 : edi wholeScreen wwSave r wwRestore ;
12
13 &12 &04 &10 &23 Menue: doFrosch
14   ." Froesche sind nett" ;
15
16
17 -->
18
19
20
21
22
23
24

```

93

9

0 \ clv10feb89
1
2
3 &8 &11 &7 &17 Menue: takeit
4 <s : " Choice One" s> doOne
5 cr <s : " Choice Two" s> doTwo
6 cr <s : " Information" s> info
7 cr <s : " Editor" s> edi
8 cr <s : " Froesche" s> dofrosch
9 3 3 2 2 ['] words sel! ;
10 &8 &11 &6 &17 Menue: t1
11 cr " .lahm, was?" takeit ;
12 &3 &5 &10 &25 Menue: big
13 cr cr " Dies ist ein fest" cr
14 cr cr " Na, wie ist das Fenster?"
15 t1 ;
16 : t : " Achtung! Screen &50-60 werden "
17 cr : " veraendert. RET to continue"
18 key fcr - ?exit big ;
19 t
20
21
22
23
24

10

```
0 | MN: Menue Loadscreen      cclv10jul88      clv06jul88
1
2
3
4      2 load \ d- split join
5      3 load \ vArray
6      4 load \ Listen / nil
7      6 load \ scre scr!
8      2 &14 +thru \ Windows
9      &20 &24 +thru \ Items
10
11 ' keySel Is select \ mouse not yet
12
13 \      1 +load \ sample
14
15
16 \\ Nachteil:
17 - bei Schachtelung des gleichen Menues
18 wird's immer kleiner
19 - bei falschen Cursorpositionen-> Crash
20 - Schreiben, Zeigen nur auf's aktuelle
21 Fenster moeglich.
22
23
24
```

11

95

```

0 \ MN: sample           clv10jul88 \ LL: sample           clv06jul188
1
2 &2 &06 &3 &10 Menue: doOne
3   "1 is One":
4 &12 &04 &6 &09 Menue: doTwo,
5   0 4 at ." 2" 1 3 at ." is"
6   2 2 at ." two".
7 0 0 &25 &40 ww: wholeScreen
8 : edi wholeScreen wwSave r wwRestore ;
9 &8 &11 &6 &17 Menue: takeit
10   <s ." Choice One" s> doOne
11   cr <s ." Choice Two" s> doTwo
12   cr <s ." Information" s> info
13   cr <s ." Editor" s> edi ;
14 &8 &11 &6 &17 Menue: t1
15   cr ." lahm, was?" takeit ;
16 &3 &5 &10 &25 Menue: big
17   cr cr ." Dies ist ein fest" cr
18   cr cr ." Ha, wie ist das Fenster?" cr
19   t1 ;
20 : t ." Achtung! Screen &50-60 werden"
21 cr ." veraendert. RET to continue"
22 key cr - ?exit big ;
23 t
24

```

5 Constant listlen
NewList myList listlen 2* allot
: nth (no--addr) 2* myList + ;
: linkall nil myList !
listlen i DO I nth myList link! LOOP ;
.l myList .link ;
\\
Format:
/-----| -----| -----|
x | ! | ? | - - - - |
adr nil

nil points to nil

12

```

0 \ MN: leap (>)select(ed)      clv10jul88
1 : leap rdrop rdrop rdrop rdrop ;
2 Defer select Create: >select select ;
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

97

13

```

0 \ MN: Variables            clv10jul88
1
2 Variable act      act off
3   \ zeigt auf mit Menue: deklarierte
4   \ Window-Parameter des aktuellen WW
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

98

14

```

0 \ MN: room               clv10jul88
1
2 &10 $400 * Constant wwLen
3 &50 $400 * vrangle !
4 &60 $400 * vrangle 2+
5   \ block 50 to 60 reserved for >room
6
7 wwLen VArray >room
8
9 \ pointer into the >room-space:
10
11 Variable wwp wwp off \ uni-pointer
12 Variable self self off
13   \ first selectable Item
14 Variable selected selected off \
15   \ now selected Item
16 ! Variable lastWW lastWW off
17   \ linked list of saved screenparts
18
19
20
21
22
23
24

```

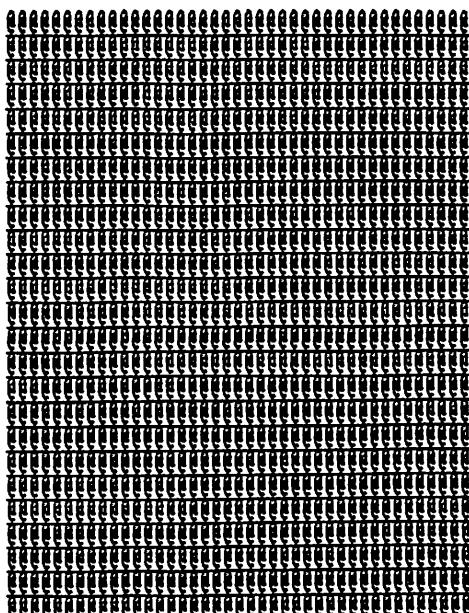
99

15

```

0 \ clv10jul88
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```


100

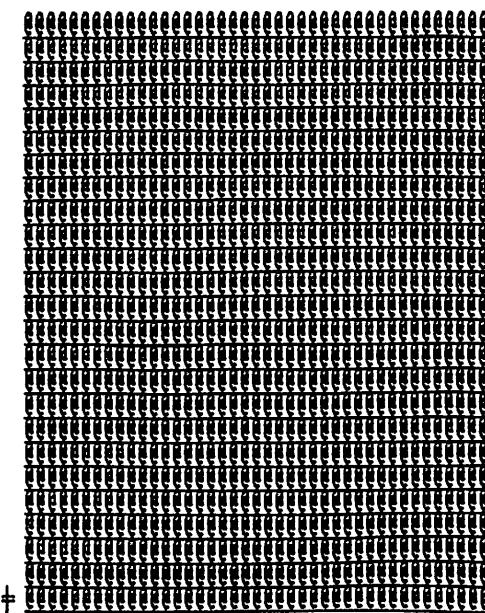
clv10jul88

101

```

0 \ MN: .wwBord           clv10jul88
1
2 : >act { n--adr) act @ + ;
3 : wwx 4 >act ; : wwy 6 >act ;
4 : www 0 >act ; : wwwH 2 >act ;
5
6 : Create wwBord
7   $b0 c, Ascii - c, $ae c, Ascii ! c,
8   $bd c, $ad c,
9
10 | : x+ 1 wwx +! ; | : y+ 1 wwy +! ;
11 | : x- -1 wwx +! ; | : y- -1 wwy +! ;
12
13 | : (.ww { bord#--)
14   wwy @ wwx @ at wwBord + c@ con! ;
15 : .wwBord
16   www @ 2 DO x+ 1 { .ww LOOP x+ 2 { .ww
17   www @ 2 DO y+ 3 { .ww LOOP y+ 4 { .ww
18   www @ 2 DO x- 1 { .ww LOOP x- 5 { .ww
19   www @ 2 DO y- 3 { .ww LOOP y- ;
20
21
22
23
24

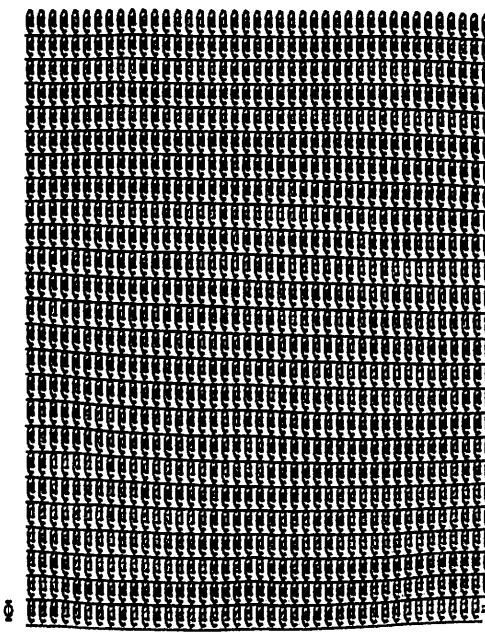
```


17

```

0 \ MN: wwP-Handling       clv10jul88
1
2
3 : room@ { --n) >room @ ;
4 : room! { n adr--) >room ! update ;
5
6 : wwHere { --adr) wwP @ ;
7 : wwAllot { n--) wwP +! ;
8 : ww? { n--) wwHere 2 wwAllot room! ;
9 : ww? { --n) wwHere >room @ 2 wwAllot ;
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```


102

18

```

0 \ clv10jul88
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

103

19

```

0 \ MN: wwOut           clv10jul88
1
2 : c64at? Alias absAt?
3 : c64at Alias absAt
4
5 : wwAt? absAt? wwy @ wwX @ d- ;
6 : wwAt  www @ 1- min swap
7   www @ 1- min swap
8   wwy @ wwX @ d+ absAt ;
9 : wwCr  row 1+ 0 at ;
10 : wwPage
11   www @ 0 DO I 0 at
12   www @ 0 DO bl emit LOOP LOOP 0 0 at ;
13 : wwDel col 0= ?exit
14   at? 1- at bl emit at? 1- at ;
15 : wwInit( char--)
16   col www @ u< IF c64emit exit THEN
17   drop ;
18 : wwType( adr count--)
19   bounds DO I c@ emit LOOP ;
20
21 Output: wwOut
22 wwInit wwCr wwType wwDel wwPage
23 wwAt wwAt? ;
24

```

104

20

```

0 \ MN: wwSave wwRestore    cclv10jul88 \ vArray - virtuelles Array clv03jul88
1
2 : wwSave
3   wwHere lastWW @ ww, lastWW ! \ link
4   output @ ww, \ ouput-vektor sichern
5   self @ ww, \ selecteable list
6   selected dup @ ww, off \ act. selected
7   absAt? ww, ww, \ cursor sichern
8   wwX @ ww, www @ ww,
9   wwy @ ww, www @ ww,
10  wwX @ www @ bounds DO
11  wwy @ www @ bounds DO
12  I J scr@ join ww,
13  LOOP LOOP wwHere self ! ;
14
15 : wwRestore
16   lastWW @ wwp ! wwp push ww? lastWW !
17   ww? output !
18   ww? self ! ww? selected !
19   ww? ww? swap absAt
20   ww? ww? bounds ww? ww? bounds 2swap
21   DO 2dup DO ww? split I J scr!
22   LOOP LOOP 2drop ;
23
24

```

\ Beispiel:

\$1000	vArray memory
\$c00	Constant scrBeg
\$100	Constant scrLen
:	
saveScreen(no--)	
scrLen	* memory
scrBeg	swap scrLen cmove update ;
:	
getScreen(no--)	
scrLen	* memory
scrBeg	scrLen cmove ;

105

21

```

0 \ MN: Format of savings      clv10jul88
1
2
3 \| The format which wwSave/Restore use
4
5 w:pointer to last saving OR 0
6 w:outputVektor
7 w:selfP - selectable-region-pointer
8 w:selected - last highlighted
9 2 w:CursorX CursorY
10 4 w:wwX www wwwH - Window-rectangle
11 H w:contents of window / H=www*www
12
13
14
15
16
17
18
19
20
21
22
23
24

```

106
22

```

0 \ MN: wwopen             clv10jul88
1
2 \| +! wegen .wwbord
3
4 : wwOpen
5   wwSave display .wwBord
6   1 wwX +! 1 wwY +! -2 www +! -2 wwwH +!
7   wwOut page ;
8
9 : wwClose
10  -1 wwX +! -1 wwY +! 2 www +! 2 wwwH +!
11  wwRestore ;
12
13
14
15
16
17
18
19
20
21
22
23
24

```

107
23

```

0 \
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

108

24

```

0 \ MN: (menue Menue:      clv10jul88
1
2 : (menue
3   r> dup act push act ! >select >r
4     8 + >r wopen ;
5
6
7 : Menue: { y x h w--}
8   : >r compile (menue , r) :
9   \r and r> saves security-flag of ':'
10
11 : ww: Create
12   does> r> act push'>r act ! ;
13
14
15
16
17
18
19
20
21
22
23
24

```

109**25**

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

110**26**

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

111

27

112

0	00000000000000000000000000000000
1	00000000000000000000000000000000
2	00000000000000000000000000000000
3	00000000000000000000000000000000
4	00000000000000000000000000000000
5	00000000000000000000000000000000
6	00000000000000000000000000000000
7	00000000000000000000000000000000
8	00000000000000000000000000000000
9	00000000000000000000000000000000
10	00000000000000000000000000000000
11	00000000000000000000000000000000
12	00000000000000000000000000000000
13	00000000000000000000000000000000
14	00000000000000000000000000000000
15	00000000000000000000000000000000
16	00000000000000000000000000000000
17	00000000000000000000000000000000
18	00000000000000000000000000000000
19	00000000000000000000000000000000
20	00000000000000000000000000000000
21	00000000000000000000000000000000
22	00000000000000000000000000000000
23	00000000000000000000000000000000
24	00000000000000000000000000000000

28

113

0	00000000000000000000000000000000
1	00000000000000000000000000000000
2	00000000000000000000000000000000
3	00000000000000000000000000000000
4	00000000000000000000000000000000
5	00000000000000000000000000000000
6	00000000000000000000000000000000
7	00000000000000000000000000000000
8	00000000000000000000000000000000
9	00000000000000000000000000000000
10	00000000000000000000000000000000
11	00000000000000000000000000000000
12	00000000000000000000000000000000
13	00000000000000000000000000000000
14	00000000000000000000000000000000
15	00000000000000000000000000000000
16	00000000000000000000000000000000
17	00000000000000000000000000000000
18	00000000000000000000000000000000
19	00000000000000000000000000000000
20	00000000000000000000000000000000
21	00000000000000000000000000000000
22	00000000000000000000000000000000
23	00000000000000000000000000000000
24	00000000000000000000000000000000

29

114

30

```
0 \ IT:empty
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

clv10jul88

115

31

```
0 \ IT:sel! noSel sel@      clv10jul88
1
2 \\ used only for MOUSE (later):
3   not tested!!!
4
5 &8 &11 &6 &17 Menue: noSel
6 cr cr ." not possible"
7 cr ." press any key" key drop ;
8
9
10 : wrong? ( y x adr--flag)
11   \ adr points to x y w h cfa-field
12 dup 2 pick -
13 over 4 + 0< ?exit
14 dup 2+ 2 pick -
15 over 6 + 0< ?exit drop false ;
16 : sel@ ( y x--cfa)
17 wwp @ selP @
18 DO 2dup I wrong? not
19   IF I &10 + @ leap THEN
20     &10 +LOOP '['] noSel ;
21
22
23
24
```

116

32

```
0 \ IT:<s s>                  clv10jul88
1
2   \ 10 bytes: x y w h cfa
3
4 : <s absAt? ww, ww, ;           \ x y
5 : s> absAt? -4 wwp +! \ word follows
6   ww - 1+ swap ww? - 1+ swap
7   ww, ww, \ w h
8 r> dup 2+ >r @ ww, ; restrict \ cfa
9 : sel! [ h w y x cfa--]
10  >r ww, ww, ww, ww, r> ww, ;
11
12 \\ Used like:
13
14 5 5 6 6 Menue: name
15   ... <s ..outputWords... s> word
16   ...
17
18 declares the field of ..outWords..
19   to be selectable and to execute word
20   if selected. Word is only executed
21   when selected.
22
23 SEL! is another way to do it and not
24   tested yet.
```

117

33

```

0 \ IT:select-primitives      clv10jul88
1 | : revers ( char--char') $80 xor ;
2 | : highlight
3 | selected @ 0= ?exit
4 | wwp push selected @ wwp !
5 | ww? ww? ww? ( h w y x)
6 | rot swap bounds 2swap bounds
7 | ?DO 2dup ?DO I J scr@ revers I J scr!
8 |     LOOP LOOP 2drop ;
9 | : normlight highlight ;
10 | : firstSel ( --adr@0)
11 | selP @ dup wwhere u< ?exit drop 0 ;
12 | : nextSel ( --adr@0)
13 | selected @ dup 0= ?exit
14 | &10 + dup wwhere u< ?exit
15 | drop firstsel ;
16 | : action selected @ ?dup
17 | IF 8 + room@ execute THEN ;
18 |
19 |
20 |
21 |
22 |
23 |
24 |

```

118

34

```

0 \ IT:dokey keySel      clv10jul88
1 | $1b Constant #esc   $88 Constant #help
2 | &18 &04 &6 &17 Menue; info
3 | cr ." ESC = exit"
4 | cr ." CR = dothis"
5 | cr ." HELP = this"
6 | cr ." other = select"
7 |
8 | : doKey ( key-endflag) false swap
9 | #esc case? IF not exit THEN
10 | #cr case? IF action exit THEN
11 | #help case? IF info exit THEN
12 |     drop nextSel selected !
13 |
14 | : keySel firstSel selected !
15 | BEGIN highlight key normlight
16 |     dokey UNTIL wwclose ;
17 |
18 |
19 |
20 |
21 |
22 |
23 |
24 |

```

119

35

```

0 \ zinseszins      clv18aug88
1 | .( noch nix lauft ) quit
2 | \needs menue: 10 load
3 |
4 |
5 |
6 |
7 |
8 |
9 |
10 |
11 |
12 |
13 |
14 |
15 |
16 |
17 |
18 |
19 |
20 |
21 |
22 |
23 |
24 |

```

120

36

```

0 \
1
2 Create num 6 allot
3 &10 &12 &3 &30 Menue: getNumber ( --n )
4 " Enter a Number"
5 num 6 bl fill
6 num 6 expect
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

121

37

```

0 \
1
2 &8 &11 &12 &30 Menue: zinsen
3 <s ." Zinsberechnung " s> rechne
4 cr
5 cr d @ 6 u.r
6 <s ." DM Darlehen " s> chD
7 cr z @ 6 u.r
8 <s ." 0/00 Jahreszins" s> chZ
9 cr r @ 6 u.r
10 <s ." DM pro Rate " s> chR
11 cr r/z @ 6 u.r
12 <s ." Raten/a " s> chR/a
13 ;
14 ;
15
16
17
18
19
20
21
22
23
24

```

122

38

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

123

39

124

40

125

0 \ MC:Multiple CFA - Loadscr. clv07jul188
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

41

136

```
0 \ MC: clv07jul188
1
2 : Class: Create 0 , { super)
3           nil , { methods)
4 does> Create 0 , { class)
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

42

```

0 \ MC:           clv07jul88
1
2 Class: Item: Super: object
3   method: .It ?." has no name";
4   method: ?It ?." no help available";
5   method: xIt ?." no action";
6   method: plt highlight;
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

127

43

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

128

44

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

129

45

130

46

131

The image displays a large grid of binary digits (0s and 1s) arranged in 25 rows and 100 columns. The grid is composed of two distinct halves: the left half contains a repeating pattern of binary digits, while the right half is entirely filled with the digit 1. Each row is labeled with a number from 0 to 24 on its left side.

47

132

The image shows a large grid of handwritten digits, likely from a dataset like MNIST. The grid is organized into 5 columns and 5 rows. Each cell in the grid contains a single handwritten digit, ranging from '0' to '9'. The digits are written in a black, cursive-like font on a white background. The grid is centered and occupies most of the frame.

48

0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		

49

0	00000000000000000000000000000000
1	00000000000000000000000000000000
2	00000000000000000000000000000000
3	00000000000000000000000000000000
4	00000000000000000000000000000000
5	00000000000000000000000000000000
6	00000000000000000000000000000000
7	00000000000000000000000000000000
8	00000000000000000000000000000000
9	00000000000000000000000000000000
10	00000000000000000000000000000000
11	00000000000000000000000000000000
12	00000000000000000000000000000000
13	00000000000000000000000000000000
14	00000000000000000000000000000000
15	00000000000000000000000000000000
16	00000000000000000000000000000000
17	00000000000000000000000000000000
18	00000000000000000000000000000000
19	00000000000000000000000000000000
20	00000000000000000000000000000000
21	00000000000000000000000000000000
22	00000000000000000000000000000000
23	00000000000000000000000000000000
24	00000000000000000000000000000000

50

0 Eeexedea@) seeexedea@! .C1C C CRCRCRCRC C1C" C
 1 C C C C C C C C C &C CTC C C C C C C C &C C7
 2 C CSCSCSCSCSC C6C C C C C C C C C C C &C
 3 C.C.C.C.C.C C&CLCIC C" "C" "C" "C" C CICACTC
 4 C C C C C C C C7CTC C CCCCC1Cefc C CHMC C
 5 TCTCNCDRC CMC, CEC COCUFCFICUC CEC CNC CI
 6 CICUCHEC CMCWCU CCCCCCRCOOSC CUCAEC CEC
 7 ECMCRCCC CECSC .C C C CAC" CTC C :C? C C COCT
 8 CHC CEC C C "CHC CMCWCICSC" C CHC CAC CECOC
 9 UCYC C CICHCKC C" "CHC CSCAC CACEC C C C"
 10 CEC>CTC CKCIC CSCSC CDC C &C CECHC C>CSC
 11 ICDCAC CICAC C C C C CUCAT CHCTC CDCDC C
 12 CFCYC C CAC CUOCOCIC CRCEC C &E&PTEBEEJIEK
 13 HQTTFNQ CTC C C C C C C C C C C C C C C
 14 C C CIC C C C-CMC CSC C C -C-CEC CHC C C -CHC C
 15 C C C C-CUC CDC C C -CEC CAC C C -C-C C CSC C
 16 C-C C C C C C C-CBC CFC C C -CIC CEC C C CGC
 17 CHC C C-C C C CSC C C -C C C CHC C C -C C C EEC
 18 C C-C C TB& FBEQYJEKEHQFEGFC C! C! CIC! C= CIC
 19 -C C C C C -C-CBC C C C C C-CSC C C C C C C-C
 20 C-C C CLC C C C-CIC C CAC C C C-CSC C CTC C C-C
 21 HC-C CHC C C C C C C C C C C C C C C C C-C
 22 CYC-C CHC C C C-CNC C CAC C C C C C C CSC C C
 23 -CFC-C C7C C C C-CEC C C C C C C-CSC C C C C
 24 C -CHC+C CTC C! CIC! CTC LIEKEKAEE- EJKEKAEI

133

134

135

51

136

0 KELEAG~~T~~LELEGEAG~~T~~LEMELEAER~~T~~B&LICRDHAMED
 1 FQBRREFEC~~C~~COCEC C C C "CO CNC C C C CACDC C
 2 C CVCCCEC C C CECICRC C C CHCACHC C C CAC
 3 SC.C C C CECAC C C C C NCAC C C C C DCAC C
 4 C C CECACFC C C CRCACE~~C~~ C CHCACTC C C C
 5 .CAC C C C C C C CHC C C C C C C C C
 6 C C C C C CIC C CCCCCCCC C C C C C C C C C
 7 OCEC
 8 CRCAC~~C~~ COCNC C C C C C AC C C C C DC C C
 9 C C -C C C C CAC C CACDC C C C C C C C C C C
 10 CIC C CAC C C C C C C C C C C C C C C C C
 11 C C -C C CEC C C C C C C C C C C C C C C
 12 C
 13 TCRA~~CH~~ C C C C C C C C C C C C C C C C
 14 C&CdChC~~CH~~ CACSC C C C C C C C C C C C C C C
 15 CTC!C!C!C!C=CAC~~OC~~UCEC~~A~~C C C C C C C C C
 16 C C CSC C -CCCCC~~I~~CaFC C CAC~~CH~~ CAC C C C C
 17 C C C C C C CHC C CTCTC~~H~~CD~~CR~~ C C CGCD~~a~~
 18 C
 19 HC C !C~~E~~ CACFC C C C C C C C C C C C C C C
 20 CUCHCEC C C C C C C C C C C C C C C C C C C
 21 IC C -CCCCRC~~U~~CSC- CEC C S~~H~~ CACTC C C C C
 22 C C -C C NC C C C C C C C C C C C C C C C C
 23 C
 24 C C CHC C C C C C C C C C C C C C C C C C C

52

137

0 CEC-CSC CEC C COC C C C C C C C-C C CEC C
 1 -CNCWCIC C C-CHC CBC C(C C C C C C C C C C-
 2 C C CSC C-C ECO COC C C-C C C NCFC CCCC C C C
 3 C C C C-C C C CHC C-C C C CHC C C C CDC C CSC
 4 CUC
 5 AC CAC CFC CACNC C C C C C C C C C C C C C C
 6 C C C C-CSC C5C C C C C C C C C C C C C C C
 7 C C-C
 8 C C-C
 9 C
 10 C CACUC C C C C C C C C C C C CRC C C C C
 11 C C NC C CCCCCAC C C C C C C C C C C C C C
 12 CTC CTC CFC CCCSC C CUCAC C C C C C C C C C
 13 C
 14 CLC C CEC CCHCWCFC CSC C C C C C C C C C C
 15 C C C C C CVC C C C C C C C C C C C C C C C
 16 C C NC C C C C C C C C C C C C C C C C C C
 17 C
 18 C
 19 CAC
 20 C
 21 "C
 22 C
 23 C
 24 C

53

138

0 Sñyep' d"ÖICuWb[äCO]8vw R + ; + jTV; \$vNha 00000000000000000000000000000000
1 1D ja | K-He H1H+S; W|Gc-VsäJ SW?4E-W6U2 00000000000000000000000000000000
2 V19 >eIGHEN- R- D- + SæEH9oS+H+5\SW4+e 00000000000000000000000000000000
3 \+W LH- SyOe dc-K- CICRaddSg-1- B1S 2d 00000000000000000000000000000000
4 Gra- f- Hw1ZB- k- P- K- en-e SIK- 00000000000000000000000000000000
5 \+E Hdcs Wm 8 x!f+sh(S 1-1- 23! f-) 00000000000000000000000000000000
6 2 J \$ = \$ @strDi s *Pf "oo's ?h 00000000000000000000000000000000
7 mussbie :Bu d ite 00000000000000000000000000000000
8 c t a or te i a 00000000000000000000000000000000
9 e 00000 Hy Rl 20 e (I O es- 000 000000000000000000000000000000
10 e 00000 e 000 Rl 20 e (I O es- 000 000000000000000000000000000000
11 e- adæeet e- e- daçopl- o 4- ;I a ra2e 00000000000000000000000000000000
12 w - f- P J u ih n ! i tex 00000000000000000000000000000000
13 m dir g i C ud C 00000000000000000000000000000000
14 o oae 00000 i: n 00000000000000000000000000000000
15 r : g sruc d a e i: n 00000000000000000000000000000000
16 n nn a - p!id aw : nr 00000000000000000000000000000000
17 r 0 r 2 a 0 (p ac 8 e 00000000000000000000000000000000
18 r 0 r 2 a 0 L u dse 00000000000000000000000000000000
19 r r * a t c m a : u 00000000000000000000000000000000
20 r * o o a v 00000000000000000000000000000000
21 t d - a > t o b : 00000000000000000000000000000000
22 w . { ; o D ; ao> 00000000000000000000000000000000
23 d8-) : e ; 0 D 00000000000000000000000000000000
24

54

```

0 u e t t 0 a
1 4 ne w : ; lWA t e w w
2 w ? Ig r r
3 \n twe s :Ww w s P e s W w
4 e iy we;h H!ir la i c e i
5 v u w d
6 is n le x u c ,
7 -
8 -
9 u > >
10 S l s & r w
11 tw S "pe > w-w il0
12 6+ -P Ic u $ o w ai
13 c o, l a e - usw s oe o . e . : " u: h
14 c l p e x s e e n c
15 e: n & " d; e w n t
16 e a H e u w H e w 2 slc
17 r i t 3 W H S L 1 4
18 La :? r C C a + n [ 0?-cC3 N
19 G a G 4 " a z n- e e B O a [ C H - U Y O D I ^ 00 T &
20 E+Cv 5 $ GEG5+i^2, ET" (tr + T-SiH K [ bpi
21 T z [ p ] v H C H N y H - id
22 . N W C R V H e g e i , 6Pke- ge
23 > [ \ I 1 8 frá h q e J U E E D p e l 0^
24 +^2-@Be=^je'■^BIP<eamie-e:ti Blt J N-HHP[P'
```

139

55

```

0 00000000000000000000000000000000
1 00000000000000000000000000000000
2 00000000000000000000000000000000
3 00000000000000000000000000000000
4 00000000000000000000000000000000
5 00000000000000000000000000000000
6 00000000000000000000000000000000
7 00000000000000000000000000000000
8 00000000000000000000000000000000
9 00000000000000000000000000000000
10 00000000000000000000000000000000
11 00000000000000000000000000000000
12 00000000000000000000000000000000
13 00000000000000000000000000000000
14 00000000000000000000000000000000
15 00000000000000000000000000000000
16 00000000000000000000000000000000
17 00000000000000000000000000000000
18 00000000000000000000000000000000
19 00000000000000000000000000000000
20 00000000000000000000000000000000
21 00000000000000000000000000000000
22 00000000000000000000000000000000
23 00000000000000000000000000000000
24 00000000000000000000000000000000
```

140

56

```

0 00000000000000000000000000000000
1 00000000000000000000000000000000
2 00000000000000000000000000000000
3 00000000000000000000000000000000
4 00000000000000000000000000000000
5 00000000000000000000000000000000
6 00000000000000000000000000000000
7 00000000000000000000000000000000
8 00000000000000000000000000000000
9 00000000000000000000000000000000
10 00000000000000000000000000000000
11 00000000000000000000000000000000
12 00000000000000000000000000000000
13 00000000000000000000000000000000
14 00000000000000000000000000000000
15 00000000000000000000000000000000
16 00000000000000000000000000000000
17 00000000000000000000000000000000
18 00000000000000000000000000000000
19 00000000000000000000000000000000
20 00000000000000000000000000000000
21 00000000000000000000000000000000
22 00000000000000000000000000000000
23 00000000000000000000000000000000
24 00000000000000000000000000000000
```

141

57

142

58

143

0	00000000000000000000000000000000
1	00000000000000000000000000000000
2	00000000000000000000000000000000
3	00000000000000000000000000000000
4	00000000000000000000000000000000
5	00000000000000000000000000000000
6	00000000000000000000000000000000
7	00000000000000000000000000000000
8	00000000000000000000000000000000
9	00000000000000000000000000000000
10	00000000000000000000000000000000
11	00000000000000000000000000000000
12	00000000000000000000000000000000
13	00000000000000000000000000000000
14	00000000000000000000000000000000
15	00000000000000000000000000000000
16	00000000000000000000000000000000
17	00000000000000000000000000000000
18	00000000000000000000000000000000
19	00000000000000000000000000000000
20	00000000000000000000000000000000
21	00000000000000000000000000000000
22	00000000000000000000000000000000
23	00000000000000000000000000000000
24	00000000000000000000000000000000

59

144

```
0 \\\ LL: sample           clv03jul188
1
2 5 Constant listlen
3 NewList myList      listlen 2* allot
4
5 : nth ( no--addr) 2* myList +
6
7 : linkall nil myList !
8 listlen 1 DO I nth myList link! LOOP ;
9
10 : .1 myList .link ;
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

60

145

61

146

0	00000000000000000000000000000000
1	00000000000000000000000000000000
2	00000000000000000000000000000000
3	00000000000000000000000000000000
4	00000000000000000000000000000000
5	00000000000000000000000000000000
6	00000000000000000000000000000000
7	00000000000000000000000000000000
8	00000000000000000000000000000000
9	00000000000000000000000000000000
10	00000000000000000000000000000000
11	00000000000000000000000000000000
12	00000000000000000000000000000000
13	00000000000000000000000000000000
14	00000000000000000000000000000000
15	00000000000000000000000000000000
16	00000000000000000000000000000000
17	00000000000000000000000000000000
18	00000000000000000000000000000000
19	00000000000000000000000000000000
20	00000000000000000000000000000000
21	00000000000000000000000000000000
22	00000000000000000000000000000000
23	00000000000000000000000000000000
24	00000000000000000000000000000000

62

147

63

148

64

149

0	00000000000000000000000000000000
1	00000000000000000000000000000000
2	00000000000000000000000000000000
3	00000000000000000000000000000000
4	00000000000000000000000000000000
5	00000000000000000000000000000000
6	00000000000000000000000000000000
7	00000000000000000000000000000000
8	00000000000000000000000000000000
9	00000000000000000000000000000000
10	00000000000000000000000000000000
11	00000000000000000000000000000000
12	00000000000000000000000000000000
13	00000000000000000000000000000000
14	00000000000000000000000000000000
15	00000000000000000000000000000000
16	00000000000000000000000000000000
17	00000000000000000000000000000000
18	00000000000000000000000000000000
19	00000000000000000000000000000000
20	00000000000000000000000000000000
21	00000000000000000000000000000000
22	00000000000000000000000000000000
23	00000000000000000000000000000000
24	00000000000000000000000000000000

65

150

The image displays a large grid of binary digits, consisting of 25 rows and 100 columns. The first column contains the row indices from 0 to 24. The subsequent 99 columns are filled with binary patterns. The patterns show a clear vertical gradient, transitioning from all zeros at the top to all ones at the bottom. Within each row, the binary digits appear to be randomly distributed, creating a noisy, high-contrast visual texture.

69

70

155

The image consists of a large grid of 24 rows by 24 columns. Each cell in the grid contains a small, identical symbol that looks like a stylized letter 'e' or a cross-like shape. The grid is set against a white background.

71

156

```
0 \ AD: Autdoc Lodscreen      clv10aug88
1
2 empty \ laesst sich grad laden
3 \ nix output
4 Onlyforth    Vocabulary Doc
5 Doc also definitions
6
7 | : F Forth ; immediate
8 | : D Doc ; immediate
9 | : docquit F end-trace
10 | '[' no.extensions Is notfound
11 Onlyforth ;
12
13
14 1 7 +thru
15
16
17 Onlyforth
18
19 | : Onlydoc
20   Only Doc also definitions seal
21   '[' drop Is notfound ;
22   : docload
23     onlydoc [ Doc ] load docquit ;
24 doc debug :: clear
```

72

```

0 \
1           clv10aug88 00000000000000000000000000000000
2 | : .info ." blk" blk @ dup u. block
3 |   " line" >in @ c/l / dup
4 |   { blkadr lin } c/l * + c/l cr type ;
5 |
6 | : name bl parse cr type ;
7
8
9
10 \ : pushinput r> blk push >in push >r ;
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

157

73

```

0 \
1           clv10aug88 00000000000000000000000000000000
2 | Variable state state off
3 | 2Variable lastDef
4 |
5 | : lastDef! >in @ here count - blk @
6 |   lastDef 2@ block >in !
7 | Defer ':: \ forward
8
9
10 | : defPass \ ungets last name and def
11 |   lastDef 2@ block >in !
12 |   ':: D state on ] ;
13
14
15
16
17
18
19
20
21
22
23
24

```

158

74

```

0 \
1           clv10aug88 00000000000000000000000000000000
2 | : 2^ ( n -- 2^n )
3 |   1 swap 0 ?DO 2^ LOOP ;
4
5 | : Ask: F Create D ( n--n+1 ) dup , 1+
6 |   does > ( adr -- adr flag ) @ 2
7 |   ( adr mask ) over @ and ;
8
9
10
11
12 0 | Ask: creates?    | Ask: setsState?
13 | Ask: parses(?)   | Ask: parses"?
14 | Ask: eatsWord?   | Ask: resetsState?
15 | Ask: eatsChar?   |
16 | Ask: compiles?
17 drop
18
19
20
21
22
23
24

```

159

75

clv10aug8

```

1 \ 
2 | :: F Create D 0 ,
3 | does> >r clearstack r>
4 | \ Zahlen werden noch verarbeitet !
5 | F state @ D state @ 0= and
6 | \ IF defPass exit THEN
7 | eatsWord? IF bl parse THEN
8 | eatsChar? IF 1 >in +! THEN
9 | parses( ? IF Ascii ) parse THEN
10 | parses( ? IF Ascii " parse THEN
11 | creates? IF name .info THEN
12 | setsState? IF ] THEN
13 | resetsState? IF [compile] [ THEN
14 | drop ;
15 
16 ' :: Is '::
17 
18 
19 
20 
21 
22 
23 
24

```

A dense grid of small, identical symbols arranged in a 20x20 pattern. The symbols are black and have a complex, symmetrical, and slightly organic shape, resembling stylized leaves or floral motifs. They are packed closely together, creating a textured and intricate visual effect.

76

```

0 \
1 |
2 | :: x
3 |
4 | : ?:?: { cfa -- pfa }
5 | dup @ ['] x @- abort" not ::"
6 | >body ;
7 |
8 |
9 | : Set: F Create ( n -- n+1 ) dup , 1+
10 | does> @ 2` ( bitmask )
11 | last @ name> ?:??
12 | ( mask cfa )
13 | dup @ under or swap ! ;
14 |
15 0 | Set: creates      | Set: setsState
16 | Set: parses(       | Set: parses"
17 | Set: eatsWord      | Set: resetsState
18 | Set: eatsChar
19 | Set: compiles
20 drop
21 |
22 | : vererbt ; \ nur zur Information
23 |
24

```

A dense grid of small, identical symbols arranged in a 10x10 pattern. The symbols are black and appear to be a stylized letter 'E' or a similar character, repeated across the entire area.

77

```
0 \
1
2
3 :: : creates setsState vererbt compiles
4 :: Variable creates vererbt
5 :: Constant creates vererbt
6 :: Create creates vererbt
7 :: " parses" immediate
8 :: " parses" immediate
9 :: " parses" vererbt
10 :: abort" parses" immediate
11 :: ( parses( immediate
12 :: ( parses( immediate
13 :: Ascii eatsChar immediate
14
15 \\ [compile] '[' [ ]
16
17
18
19
20
21
22
23
24
```

The image consists of a 10x10 grid of small, identical symbols. Each symbol is a black, stylized character or mark that looks like a combination of a 'C' and a 'G'. The symbols are arranged in a regular, repeating pattern across the entire grid.

78

```

0 \
1
2
3
4 F' load Alias load
5 F' +load Alias +load
6 F' thru Alias thru
7 F' +thru Alias +thru
8
9 \\
10 | : [ D state off F [compile] [ ;
11 | immediate
12 | ; [compile] [ ; immediate
13
14
15
16
17
18
19
20
21
22
23
24

```

163

79

```

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

164

80

```

0 \ test
1
2 Variable xxx
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

```

165

84

169

8

0

1

8