

A Tool for Supporting Lectures on Programming Course

TAKUO OHTA

A LAN is being developed experimentally for supporting lectures in OGU. This system, as an ordinary CAI system, gives students problems appropriate for them, gets answers, and manages their performances. Students are able to work with the problems at any time and at any terminal after the lecture to review and supplement it. As the system does not aim to replace lecturers with machines but aims to support training required by the lectures, it was named CAT (Computer Aided Training) system. While students work at terminals, they can use several tools imbedded in the system to get data from some database, expand their computational ability, make a simulation, or refer some explanation.

In case of lectures on computer programming, it is important for students to understand algorithms. But for the beginners in programming, they often find difficulties in grasping the process that the algorithm means. Furthermore, they are requested to get ability to express an algorithm concisely with a kind of diagram or a programming language for a given problem. Tracing the process makes it easy for students to understand the algorithm. But it takes much time to trace it on a blackboard in the classroom. The author made a program which shows students the process that an algorithm means on a CRT display. The program, although it is a prototype at present, as a member of the tools imbedded in CAT system can be used while students work with problems in the course of computer programming.

1. The Function of the Program

This program visualizes the change of the contents of variables used in the algorithm. It behaves as if variables were on the display. Some

* Received Jan. 9, 1987.

considerations are taken to make it easy for the students to understand the meaning of the process.

A pointer type variable and the pointed variable are always connected by a dotted line. A variable on the display blinks when it has a new value. An arrow appears as a traject when a data is transferred from a variable to another one. Variables with meaningless value disappear on the display. Explanation of the present state can be displayed hierarchically. Also graphical illustrations can be set on the display.

Before the execution of the program, some procedures for setting up should be executed. For the first step, one should define the characters of variables used in it. The characters of a variable to be defined are listed as follows:

< identifier₁ > < location₂ > < type(1)₃ > < type(2)₄ > < color₅ >
< initial value₆ >

1. the identifier of the variable
2. the location of the variable in the display.
if it is array type, the shape:
(column or row)
3. the type of the variable:
(scalar or array),
(numeric, character string, or pointer)
4. if pointer type, the identifier of the variable to be pointed.
if numeric type, format for the field.
5. color code of the variable.
6. the initial value of the variable.

The procedure of the algorithm is expressed in the following statements.

1. assignment:
ASG < identifier > < expression >
2. transfer:
TRS < identifier of destination > < identifier of source >
3. movement:
MOV < identifier of destination > < identifier of source >

The source variable of TRS disappears after the operation while that of MOV preserves the content. If the variable is array type, The Index IA

(destination) and/or IB(source) should be set before operation.

2. Example

An example is shown using 'quick sort' by C.A.R.Hoar, as one of the most interesting algorithm. It seems to be effective for such an algorithm as one using stacks.

Figure 1 is one scene of the process.

Quick-Sort by C.A.R. Hoar

procedure of Quick-Sort

sort the interval on the stack while it is not empty

divide the working interval while it has length

push down the longer interval. (prolong the treatment)

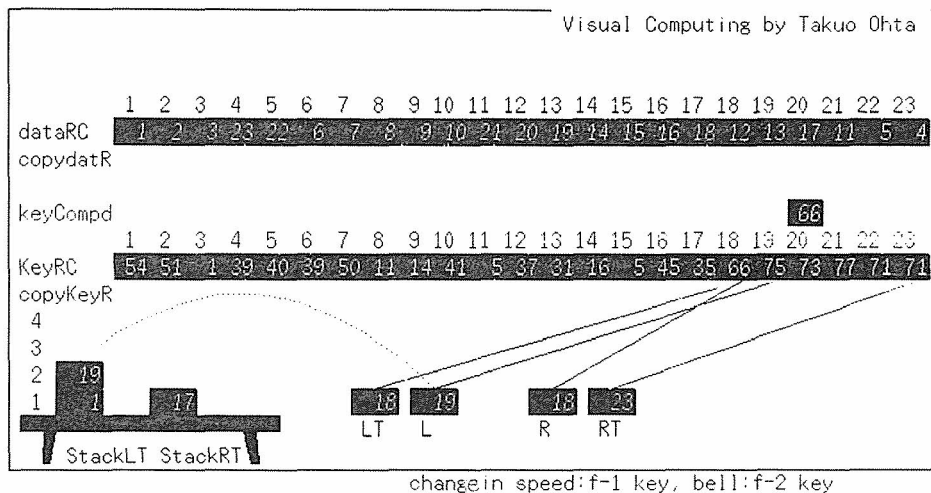


Figure 1

The source list of the program is in Appendix.

3. Comment

The program was written in Basic as a prototype, and the definition and statements described above were imbedded in Basic statements. These part of the statements should be separated from, and should be written in

order that anyone can construct them.

In view of education, the effect of such a trial should be appreciated objectively. These subjects will be treated in the next report.

```

1 ' save "VCOM",a      ' Visual Computing (for sorting routines)
2                      ' prog. by Takuo Ohta, 86/06/28
3                      ' Rev.B.2,86/07/07(for N88 & N100)
1000      *VARLIST
1010 DATA "StackLT ",05,23,"V","N","####"      ,14, 4," "
1020 DATA "StackRT ",13,23,"V","N","####"      ,14, "", " "
1030 DATA "LT      ",30,22," ", "P","KeyRC " ,14, "", " "
1040 DATA "L       ",35,22," ", "P","KeyRC " , 6, "", " "
1050 DATA "R       ",45,22," ", "P","KeyRC " , 6, "", " "
1060 DATA "RT      ",50,22," ", "P","KeyRC " ,14, "", " "
1070 DATA "KeyRC   ",07,17,"H","N"," ##"      ,11, "", " "
1080 DATA "copyKeyR",07,18,"H","N"," ##"      ,11, "", "3"
1090 DATA "dataRC  ",07,12,"H","N"," ##"      ,13, "", " "
1100 DATA "copydatR",07,13,"H","N"," ##"      ,13, "", "3"
1110 DATA "keyCompd",07,15,"H","N"," ##"      ,15, "", "3"
1120 '
1130 DATA "maxp    ",20,22," ", "P","KeyRC " ,14, "", " "
1140 DATA "i       ",30,22," ", "P","KeyRC " , 6, "", " "
1150 DATA "right   ",40,22," ", "P","KeyRC " ,15, "", " "
1160 '
1170 DATA "EOF",60,02," ", "A","@ " ,0
1180 '
1190      *MSGLIST
2000 DATA "P0.0/4808*Visual Computing by Takuo Ohta"
2005 DATA "P1.0/0101*simple maximum selection/1,quick sort/2,exit/Q"
2010 DATA "P1.1/5102*speed(1:fast..9:slow)"
2020 DATA "P1.2/5103*      S:until keyin"
2030 DATA "P1.3/3525*change in speed:f-1 key, bell:f-2 key"
2040 DATA "P1.4/1501*number of sample data(1..23)="
2050 DATA "P1.5/5104*      Q:quit"
2060 DATA "P1.6/3525*                        any key"
2070 DATA "P1.7/5102*bell, silent/N      "
2080 '
2090 DATA "Q1.1/0101*Quick-Sort by C.A.R. Hoar"
2100 DATA "Q2.1/0302:generation of test data(KeyRC,dataRC)"
2110 DATA "Q2.2/0302:procedure of Quick-Sort"
2120 DATA "Q3.1/0503:push down the interval to be sorted"
2130 DATA "Q3.2/0503:sort the interval on the stack while it is not empty"
2140 DATA "Q4.1/0704:pop up the working interval(LT,RT)"
2150 DATA "Q4.2/0704:divide the working interval while it has length"
2160 DATA "Q5.1/0905:set temporary position of division(L,R)."
2170 DATA "Q5.2/0905:set value for comparison(keyCompd) as the data at center."
2180 DATA "Q5.3/0905:repeat division until the pointers L and R cross."
2190 DATA "Q6.1/1106:find the position of data to move to the left."
2200 DATA "Q6.2/1106:find the position of data to move to the right."

```

```

2210 DATA "Q6.3/1106:swap the keydata."
2220 DATA "Q6.4/1106:swap the data."
2230 DATA "Q5.4/0905:complete the division because pointers L and R cross."
2240 DATA "Q5.5/0905:push down the longer interval.(prolong the treatment.)"
2250 DATA "Q5.6/0905:set the interval for treatment(LT,RT) as the shorter."
2260 DATA "Q4.3/0704:this interval needs not to be divided because the length is
  less then 1 or equals 1."
2270 DATA "Q3.3/0503:completed the sorting because there is no interval to be so
  rted on the stack."
2280 '
2290 DATA "S1.1/0101:simple maximum selection"
2300 DATA "S2.1/0302:generation of sample data(KeyRC,dataRC)."
```

Line	Text
2310	DATA "S2.2/0302:procedure of simple maximum selection."
2320	DATA "S3.1/0503:set the range for search of the maximum(1..right) from whol e interval to (1..2)."
2330	DATA "S4.1/0704:search the position of maximum(maxp) in the range."
2340	DATA "S5.1/0905:set the temporary position of the maximum(maxp)."
2350	DATA "S5.2/0905:modify the temporaty position of the maximum(maxp)."
2360	DATA "S4.2/0704:swap the data as the maximum locates at the right terminal of the range."
2370	DATA "S4.3/0704:swap the data."
2380	'
2390	DATA "EOF"
2400	'
3000	*GDATA
3010	DATA 02,23, 24,23, 24,23.6, 22,23.6, 22.2,24.8, 21.6,24.8, 21,23.6
3020	DATA 05,23.6, 4.5,24.8, 3.9,24.8, 04,23.6, 02,23.6, 02,23, 0,0
3030	'
5000	GOSUB *INITSCRN
5010	GOSUB *INITVAR
5020	GOSUB *PARAINPUT
5030	WHILE STYPE
5040	ON STYPE GOSUB *SIMPDEMO,*QSORTDEMO
5050	GOSUB *PARAINPUT
5060	WEND
5070	GOSUB *ENDPRC
5080	END
5090	'
6000	*PARAINPUT
6010	*U201
6015	M\$="P1.3": GOSUB *IREMV
6020	M\$="P1.0": GOSUB *MLIST
6030	Z\$=INPUT\$(1)
6040	STYPE=INSTR("12Qq",Z\$)
6050	IF STYPE=0 THEN U201

```
6060 GOSUB *IREMV
6070 IF STYPE>=3 THEN STYPE=0
6080 IF STYPE =0 THEN *ENDIF1107
6090   *U202
6100     MS="P1.4":GOSUB *ILIST
6110     INPUT N
6120     IF N>23 THEN *U202
6130     GOSUB *F1INT
6140 *ENDIF1107
6150 MS="P1.4":GOSUB *IREMV
6160 RETURN
6170 '
7000 *MLIST
7010 IF BEEPON THEN BEEP
7020 IF GWBASIC THEN KEY(1) STOP:KEY(2) STOP ELSE KEY STOP
7030 MS(MMAX+1)=MS:M=1
7040 WHILE MID$(MS(M),1,4)<>MS:M=M+1:WEND
7050 IF M=MMAX+1 THEN STOP
7060 H5=VAL(MID$(MS(M),6,2)):V5=VAL(MID$(MS(M),8,2))
7070 COLOR NRM
7080 H3=H5
7090 FOR V3=V5 TO 7:GOSUB *CURSOR:PRINT SPACE$(79-H3);:NEXT
7100 H3=H5:V3=V5 :GOSUB *CURSOR:PRINT MID$(MS(M),11);
7110 IF STEPBY THEN GOSUB *WAITUNTILKEY ELSE FOR I4=1 TO (Q-1)*500:NEXT
7120 IF GWBASIC THEN KEY(1) ON:KEY(2) STOP ELSE KEY ON
7130 RETURN
7140 '
7150 *IREMV
7160 MS(MMAX+1)=MS:M=1
7170 WHILE MID$(MS(M),1,4)<>MS:M=M+1:WEND
7180 IF M=MMAX+1 THEN STOP
7190 H3=VAL(MID$(MS(M),6,2)):V3=VAL(MID$(MS(M),8,2))
7200 GOSUB *CURSOR:PRINT SPACE$(79-H3);
7210 RETURN
7220 '
7230 *ILIST
7240 MS(MMAX+1)=MS:M=1
7250 WHILE MID$(MS(M),1,4)<>MS:M=M+1:WEND
7260 IF M=MMAX+1 THEN STOP
7270 IF GWBASIC THEN KEY(1) STOP :KEY(2) STOP ELSE KEY STOP
7280 H3=VAL(MID$(MS(M),6,2)):V3=VAL(MID$(MS(M),8,2))
7290 COLOR NRM
7300 GOSUB *CURSOR:PRINT MID$(MS(M),11);
7310 IF GWBASIC THEN KEY(1) ON :KEY(2) ON ELSE KEY ON
7320 RETURN
```

```

7330 '
8000 *GENERATEDATA '--generation of test data( keyboard) > n, key[],rc[]
8010 M$="Q2.1":GOSUB *MLIST
8020 RANDOMIZE VAL(MID$(TIMES$,7,2))
8030 FOR I=1 TO N
8040 KEYRC(I)=INT(100*RND)
8050 PS="ASG:KeyRC ":IB=I:D$=STR$(KEYRC(I)):GOSUB *OP
8060 DATARC(I)=I
8070 PS="ASG:dataRC ":IB=I:D$=STR$( I ) :GOSUB *OP
8080 NEXT
8090 RETURN
8100 '
9000 *DISP '-- set display(key[],rc[]) > display
9010 CLM=0
9020 PRINT
9030 FOR I=1 TO N
9040 PRINT USING "(####)###.## #### ";I,KEYRC(I),DATARC(I);
9050 CLM=CLM+1
9060 IF CLM=4 THEN PRINT:CLM=0
9070 NEXT
9080 PRINT
9090 PRINT STRING$(80,"-")
9100 RETURN
9110 '
10000 *QSORTDEMO
10010 GOSUB *QSETSCRN
10020 GOSUB *GENERATEDATA
10030 GOSUB *QSORT
10040 GOSUB *WAITUNTILKEY
10050 RETURN
10060 '
10070 *QSETSCRN
10080 IF GWBASIC THEN CLS ELSE CLS 3
10090 M$="Q1.1":GOSUB *MLIST
10100 GOSUB *MAKEUPSCRN
10140 *VARFORQLIST
10150 DATA "StackLT ","StackRT ","LT ","L ","R ","RT "
10160 DATA "KeyRC ","copyKeyR","dataRC ","copydatR","keyCompd","EOF"
10170 RESTORE *VARFORQLIST
10180 READ VARID1$
10190 WHILE VARID1$<>"EOF"
10200 IND1=0:GOSUB *ADRCAL
10210 J2=J1:GOSUB *DISPVARID
10220 READ VARID1$
10230 WEND

```

```

10232 RESTORE *GDATA
10234 GOSUB *MAKEG
10240 RETURN
10250 '
10260 *QSORT '(key[],rc[],n) > key[],rc[]
10270 'sort dataRC,array(size of n) according to KeyRC(ascending order)
10280 DIM STACKLT(16),STACKRT(16) 'stack of interval to be sorted
10290 M$="Q2.2":GOSUB *MLIST
10300 IKCO=(1+N)/2
10310 M$="Q3.1":GOSUB *MLIST
10320 SP=1 'initialize the stack
10330 STACKLT (SP)=1
10340 P$="ASG:StackLT ":D$=STR$(1):IB=SP:GOSUB *OP
10350 STACKRT (SP)=N
10360 P$="ASG:StackRT ":D$=STR$(N):IB=SP:GOSUB *OP
10370 M$="Q3.2":GOSUB *MLIST
10380 WHILE SP>0 'sort the interval on the top of the stack while not empty.
10390 M$="Q4.1":GOSUB *MLIST
10400 LT=STACKLT(SP) 'pop up the interval to be treated
10410 P$="TRS:LT :StackLT ":IA=SP:D$=STR$(LT):GOSUB *OP
10420 RT=STACKRT(SP)
10430 P$="TRS:RT :StackRT ":IA=SP:D$=STR$(RT):GOSUB *OP
10440 SP=SP-1
10450 M$="Q4.2":GOSUB *MLIST
10460 WHILE RT>LT 'sort while it has length 2
10470 M$="Q5.1":GOSUB *MLIST
10480 L=LT 'initialize the position to be divided
10490 P$="MOV:L :LT ":D$=STR$(L):GOSUB *OP
10500 R=RT
10510 P$="MOV:R :RT ":D$=STR$(R):GOSUB *OP
10520 KEYCOMP=KEYRC((LT+RT)/2) 'set value for comparison
10530 M$="Q5.2":GOSUB *MLIST
10540 IKC=(LT+RT)/2
10550 P$="ASG:keyCompd":IB=IKC:D$="":GOSUB *OP
10560 P$="MOV:keyCompd:KeyRC ":IA=IKC:IB=IKC:D$=STR$(KEYCOMP):GOSUB *OP
10570 IKCO=IKC
10580 M$="Q5.3":GOSUB *MLIST
10590 *U108 'swap until pointers cross
10600 M$="Q6.1":GOSUB *MLIST
10610 WHILE KEYCOMP<KEYRC(R) 'right position
10620 R=R-1
10630 P$="ASG:R ":D$=STR$(R):GOSUB *OP
10640 WEND
10650 M$="Q6.2":GOSUB *MLIST
10660 WHILE KEYCOMP>KEYRC(L) 'left position

```

```

10670      L=L+1
10680      P$="ASG:L      ":D$=STR$(L):GOSUB *OP
10690  WEND
10700  IF L>R THEN *ENDIF90      'swap
10710      M$="Q6.3":GOSUB *MLIST
10720      SWAP KEYRC(L),KEYRC(R)
10730      P$="MOV:copyKeyR:KeyRC      ":IB=L:IA=L:D$=STR$(KEYRC(R)):GOSUB *OP
10740      P$="MOV:KeyRC      :KeyRC      ":IB=L:IA=R:D$=STR$(KEYRC(L)):GOSUB *OP
10750      P$="TRS:KeyRC      :copyKeyR":IB=R:IA=L:D$=STR$(KEYRC(R)):GOSUB *OP
10760      M$="Q6.4":GOSUB *MLIST
10770      SWAP DATARC(L),DATARC(R)
10780      P$="MOV:copydatR:dataRC      ":IB=L:IA=L:D$=STR$(DATARC(R)):GOSUB *OP
10790      P$="MOV:dataRC      :dataRC      ":IB=L:IA=R:D$=STR$(DATARC(L)):GOSUB *OP
10800      P$="TRS:dataRC      :copydatR":IB=R:IA=L:D$=STR$(DATARC(R)):GOSUB *OP
10810      R=R-1
10820      P$="ASG:R      ":D$=STR$(R):GOSUB *OP
10830      L=L+1
10840      P$="ASG:L      ":D$=STR$(L):GOSUB *OP
10850  *ENDIF90
10860  IF L<=R THEN *U108      'condition of termination of division
10870      'push down the longer, the shorter is on work
10880  M$="Q5.4":GOSUB *MLIST
10890  M$="Q5.5":GOSUB *MLIST
10900  IF (R-LT)<(RT-L) THEN *PUTOFFRIGHT
10910      SP=SP+1
10920      STACKLT(SP)=LT
10930      P$="TRS:StackLT :LT      ":IB=SP:D$=STR$(STACKLT(SP)):GOSUB *OP
10940      STACKRT (SP)=R
10950      P$="TRS:StackRT :R      ":IB=SP:D$=STR$(STACKRT(SP)):GOSUB *OP
10960      M$="Q5.6":GOSUB *MLIST
10970      LT=L
10980      P$="TRS:LT      :L      ":D$=STR$(LT):GOSUB *OP
10990      GOTO *ENDIF96
11000  *PUTOFFRIGHT
11010      SP=SP+1
11020      STACKLT(SP)=L
11030      P$="TRS:StackLT :L      ":IB=SP:D$=STR$(STACKLT(SP)):GOSUB *OP
11040      STACKRT (SP)=RT
11050      P$="TRS:StackRT :RT      ":IB=SP:D$=STR$(STACKRT(SP)):GOSUB *OP
11060      M$="Q5.6":GOSUB *MLIST
11070      RT=R
11080      P$="TRS:RT      :R      ":D$=STR$(RT):GOSUB *OP
11090  *ENDIF96
11100  WEND      'complete the sorting of the indicated interval
11110  M$="Q4.3":GOSUB *MLIST

```

```

11120 WEND                                'complete the initial interval
11130 M$="Q3.3":GOSUB *MLIST
11140 ERASE STACKLT ,STACKRT              'erase the stack
11150 RETURN
11160 '
20000 *SIMPDEMO
20010 GOSUB *SSETSCRN
20020 GOSUB *GENERATEDATA
20030 GOSUB *SSORT
20040 GOSUB *WAITUNTILKEY
20050 RETURN
20060 '
20070 *SSETSCRN
20080 IF GWBASIC THEN CLS ELSE CLS 3
20090 M$="S1.1":GOSUB *MLIST
20100 GOSUB *MAKEUPSCRN
20120 *VARFORSLIST
20130 DATA "maxp  ", "i      ", "right  "
20140 DATA "KeyRC  ", "copyKeyR", "dataRC  ", "copydatR"
20150 DATA "EOF"
20160 RESTORE *VARFORSLIST
20170 READ VARID1$
20180 WHILE VARID1$<>"EOF"
20190 IND1=0:GOSUB *ADRCAL
20200 J2=J1:GOSUB *DISPVARID
20210 READ VARID1$
20220 WEND
20230 RETURN
20240 '
20250 *SSORT
20260 M$="S2.2":GOSUB *MLIST
20265 M$="S3.1":GOSUB *MLIST
20270 FOR RIGHT=N TO 2 STEP -1
20280 M$="S4.1":GOSUB *MLIST
20290 P$="ASG:right  ":D$=STR$(RIGHT):GOSUB *OP
20300 M$="S5.1":GOSUB *MLIST
20310 MAXP=1
20320 P$="ASG:maxp   ":D$=STR$(MAXP):GOSUB *OP
20330 FOR I=2 TO RIGHT
20340 P$="ASG:i      ":D$=STR$(I):GOSUB *OP
20350 IF KEYRC(I)<=KEYRC(MAXP) THEN *ENDIF3025
20360 M$="S5.2":GOSUB *MLIST
20370 MAXP=I
20380 P$="MOV:maxp   :i      ":D$=STR$(MAXP):GOSUB *OP
20390 *ENDIF3025

```

```

20400 NEXT
20410 M$="S4.2":GOSUB *MLIST
20420 SWAP KEYRC(MAXP),KEYRC(RIGHT)
20430 P$="MOV:copyKeyR:KeyRC ":IB=MAXP:IA=MAXP:D$=STR$(KEYRC(RIGHT)):GOSUB *
OP
20440 P$="MOV:KeyRC :KeyRC ":IB=MAXP:IA=RIGHT:D$=STR$(KEYRC(MAXP)):GOSUB *
OP
20450 P$="TRS:KeyRC :copyKeyR":IB=RIGHT:IA=MAXP:D$=STR$(KEYRC(RIGHT)):GOSUB
*OP
20460 M$="S4.3":GOSUB *MLIST
20470 SWAP DATARC(MAXP),DATARC(RIGHT)
20480 P$="MOV:copydatR:dataRC ":IB=MAXP:IA=MAXP:D$=STR$(DATARC(RIGHT)):GOSUB
*OP
20490 P$="MOV:dataRC :dataRC ":IB=MAXP:IA=RIGHT:D$=STR$(DATARC(MAXP)):GOSUB
*OP
20500 P$="TRS:dataRC :copydatR":IB=RIGHT:IA=MAXP:D$=STR$(DATARC(RIGHT)):GOSUB
*OP
20510 NEXT
20520 RETURN
20530 '
50000 *INITSCRN
50010 A=100000000000#:B=A+1:GWBASIC=(B>A) 'set working system
50030 ESC$=CHR$(27)+"["
50040 IF NOT GWBASIC THEN *N88BASIC1001
50050 ON KEY(1) GOSUB *F1INT: KEY(1) ON
50060 ON KEY(2) GOSUB *F2INT: KEY(2) ON
50070 PRINT ESC$+"=1h";
50080 SCREEN 1,0,0,0
50090 NRM=15:BLK=0
50100 VIEW (0,0)-(719,499)
50110 WINDOW SCREEN(1,1)-(80+8/9,25+19/20)
50140 LOCATE 1,1,0
50150 GOTO *ENDIF1001
50160 *N88BASIC1001
50170 ON KEY GOSUB *F1INT,*F2INT: KEY ON
50180 NRM= 7:BLK=0
50190 SCREEN 3,0:WIDTH 80,25:CONSOLE 0,25,0,1
50200 CLS 3
50210 VIEW (0,0)-(639,399)
50220 WINDOW (1,1)-(80+8/9,25+15/16)
50230 *ENDIF1001
50240 RETURN
50250 '
50260 *DRAWVAR '(x$(,h,v)) with color c, format f$
50264 COLOR C

```

```

50265 IF GWBASIC THEN PRINT ESC$+"7;3m";
50270 V3=V:H3=H:GOSUB *CURSOR
50280 FOR I11=1 TO Q
50290 LINE (H+1/8,V+1/16)-STEP(LEN(F$)-1/8,1-1/8),0,B:FOR I110=1 TO Q*10:NEXT
50300 LINE (H+1/8,V+1/16)-STEP(LEN(F$)-1/8,1-1/8),7,B:FOR I110=1 TO Q*10:NEXT
50310 NEXT
50340 IF INSTR(F$,"#")>0 THEN PRINT USING F$;VAL(X$);ELSE PRINT USING F$;X$;
50350 COLOR NRM
50360 IF GWBASIC THEN PRINT ESC$+"0m";
50370 LINE (H+1/8,V+1/16)-STEP(LEN(F$)-1/8,1-1/8),7*-(C<>0),B
50380 RETURN
50390 '
50400 *ARROW '(HA0,VA0)->(HB0,VB0)with color=c,curvature=ar0,STYLE=ST
50410 HW=HA0 :VW=VA0
50420 FOR T=1 TO NQ
50430 HWO=HW:VWO=VW
50440 HW=(HB0-HA0)/NQ*T+HA0
50450 VW=(VB0-VA0)/NQ*T+VA0 - T*(NQ-T)/(NQ/2)^2 * ABS(HB0-HA0)/10*AR0
50460 LINE(HWO,VWO)-(HW,VW),C,,ST
50470 NEXT
50480 RETURN
50490 '
50500 *ARROWSF '(HA,VA) -> (HB,VB) with color=c,ctrvature=ar
50510 AR0=AR:HA0=HA+2:HB0=HB+2:VA0=VA:VB0=VB
50520 IF ABS(HA-HB)<4 AND VA<VB THEN VA0=VA+1:VB0=VB:AR0=0
50530 IF ABS(HA-HB)<4 AND VA>VB THEN VA0=VA:VB0=VB+1:AR0=0
50540 NQ=Q*5:ST=&H2222:GOSUB *ARROW
50550 RETURN
50560 *ARROWLN '--(HA,VA) -> (HB,VB) with color=c
50570 AR0=0:HA0=HA+2:HB0=HB+2:VA0=VA:VB0=VB
50580 IF VA<VB THEN VA0=VA+1 ELSE VB0=VB+1
50590 NQ=1:ST=&HFFFF:GOSUB *ARROW
50600 RETURN
50610 '
60000 *INITVAR '--set characters of variables and screen
60010 DIM M$(51)
60020 MMAX=0
60030 RESTORE *MSGLIST
60040 READ M$
60050 WHILE M$<>"EOF"
60060 MMAX=MMAX+1
60070 M$(MMAX)=M$
60080 READ M$
60090 WEND
60100 NMAX=2000

```

```

60110 DIM KEYRC(NMAX),DATARC(NMAX)
60120 MAXMAXVAR=100:MAXCONT=9
60130 DIM V$(MAXMAXVAR,MAXCONT)
60140 ' V$ (1) (2) (3) (4) (5) (6) (7) (8) (9)
60150 ' id column row shape type format color size initial value
60160 ' (shape=H|V) (type=P|N|A) (format=format|pointed var.)
60170 '
60180 MAXVAR=0
60190 RESTORE *VARLIST
60200 READ V$
60210 WHILE V$<>"EOF"
60220 MAXVAR=MAXVAR+1
60230 V$(MAXVAR,1)=V$
60240 FOR I=2 TO MAXCONT
60250 READ V$(MAXVAR,I)
60260 NEXT
60270 IF NOT GWBASIC THEN V$(MAXVAR,7)=STR$((VAL(V$(MAXVAR,7)) MOD 7)+1)
60280 READ V$
60290 WEND
60300 V$(MAXVAR+1,1)="EOF"
60310 RETURN
60320 '
60330 *DISPVARID
60340 IF GWBASIC THEN KEY(1) STOP:KEY(2) STOP ELSE KEY STOP
60350 H=VAL(V$(J2,2)):V=VAL(V$(J2,3))
60360 IF V$(J2,4)="H" THEN V0=V:H0=H-5 ELSE V0=V+1:H0=H+1
60370 V3=V0:H3=H0:GOSUB *CURSOR
60380 COLOR VAL(V$(J2,7)):PRINT V$(J2,1);
60390 IF V$(J2,8)=" " THEN IMAX6=N ELSE IMAX6=VAL(V$(J2,8))
60400 FOR I=1 TO IMAX6
60410 IF V$(J2,4)="H" THEN H0=H+LEN(V$(J2,6))*I:V0=V-1 ELSE H0=H-3:V0=V-I
60420 V3=V0:H3=H0:GOSUB *CURSOR
60430 PRINT USING "##";I;
60440 NEXT
60450 IF GWBASIC THEN KEY(1) ON :KEY(2) STOP ELSE KEY ON
60460 RETURN
60470 '
60480 *OP '(P$,D$,IA,IB)
60490 ' P$: 1..3 5..12 14..21
60500 ' op.code to from
60510 ' (op.code=ASG|MOV|TRS|SWP)
60520 ' D$: data to be transforred
60530 ' IA: index of 'from'
60540 ' IB: index of 'to'
60550 '

```

```

60560 IF GWBASIC THEN KEY(1) STOP:KEY(2) STOP ELSE KEY STOP
60570 ON (3+INSTR("MOV ASG TRS SWP ",MID$(P$,1,3)))¥4 GOSUB *MOV,*ASG,*TRS,*SWP
60580 IF GWBASIC THEN KEY(1) ON:KEY(2) ON ELSE KEY ON
60590 RETURN
60600 '
60610 *ASG
60620 VARID$=MID$(P$,5,8):IND=IB:D1$=D$:GOSUB *ASIGN
60630 RETURN
60640 *MOV
60650 VARID1$=MID$(P$,14, 8):IND1=IA:GOSUB *ADRCAL:JA=J1:HA=H1:VA=V1
60660 VARID1$=MID$(P$, 5, 8):IND1=IB:GOSUB *ADRCAL:JB=J1:HB=H1:VB=V1
60670 C=VAL(V$(JA,7)):AR=1:GOSUB *ARROWSF
60680 VARID$ =MID$(P$, 5, 8):IND=IB:D1$=D$:GOSUB *ASIGN
60690 VARID1$=MID$(P$,14, 8):IND1=IA:GOSUB *ADRCAL:JA=J1:HA=H1:VA=V1
60700 VARID1$=MID$(P$, 5, 8):IND1=IB:GOSUB *ADRCAL:JB=J1:HB=H1:VB=V1
60710 C=0:AR=1:GOSUB *ARROWSF
60720 RETURN
60730 *TRS
60740 VARID1$=MID$(P$,14, 8):IND1=IA:GOSUB *ADRCAL:JA=J1:HA=H1:VA=V1
60750 VARID1$=MID$(P$, 5, 8):IND1=IB:GOSUB *ADRCAL:JB=J1:HB=H1:VB=V1
60760 C=VAL(V$(JA,7)):AR=1:GOSUB *ARROWSF
60770 VARID$ =MID$(P$, 5, 8):IND=IB:D1$=D$:GOSUB *ASIGN
60780 VARID$ =MID$(P$,14, 8):IND=IA:D1$="":GOSUB *ASIGN
60790 VARID1$=MID$(P$,14, 8):IND1=IA:GOSUB *ADRCAL:JA=J1:HA=H1:VA=V1
60800 VARID1$=MID$(P$, 5, 8):IND1=IB:GOSUB *ADRCAL:JB=J1:HB=H1:VB=V1
60810 C=0:AR=1:GOSUB *ARROWSF
60820 RETURN
60830 '
60840 *ADRCAL ' (VARID1$,IND1) => J1,(H1,V1)
60850 V$(MAXVAR+1,1)=VARID1$:J1=1
60860 WHILE V$(J1,1)<>VARID1$:J1=J1+1:WEND
60870 H1=VAL(V$(J1,2))
60880 V1=VAL(V$(J1,3))
60890 IF V$(J1,4)="H" THEN H1=H1+IND1*LEN(V$(J1,6))
60900 IF V$(J1,4)="V" THEN V1=V1-IND1
60910 RETURN
60920 '
60930 *ASIGN ' (VARID$,IND,D1$)
60940 VARID1$=VARID$:IND1=IND:GOSUB *ADRCAL:J=J1:H=H1:V=V1
60950 F$=V$(J,6):C=VAL(V$(J,7))
60960 IF D1$="" THEN C0=0 ELSE C0=VAL(V$(J,7))
60970 IF V$(J,5)="P" THEN F$="####"
60980 X$=D1$:C=C0
60990 GOSUB *DRAWVAR
61000 IF V$(J,5)<>"P" THEN *ENDIF310

```

```

61010 IND1=VAL(V$(J,MAXCONT)):VARID1$=V$(J,6):GOSUB *ADRCAL
61020 HA=H:VA=V:HB=H1:VB=V1:C=0:GOSUB *ARROWLN
61030 IND1=VAL(D1$):VARID1$=V$(J,6):GOSUB *ADRCAL
61040 HA=H:VA=V:HB=H1:VB=V1:C=C0:GOSUB *ARROWLN
61050 V$(J,MAXCONT)=D$
61060 *ENDIF310
61070 RETURN
61080 '
63000 *CURSOR
63010 IF NOT GWBASIC THEN *N88BASIC3031
63020 LOCATE V3,H3
63030 GOTO *ENDIF3031
63040 *N88BASIC3031
63050 LOCATE H3-1,V3-1
63060 *ENDIF3031
63070 RETURN
63080 '
63090 *F1INT
63100 H9=POS(0):V9=CSRLIN
63110 LINE(50,1)-(80,5.5),7,B
63120 M$="P1.6":GOSUB *IREMV
63130 M$="P1.1":GOSUB *ILIST
63140 M$="P1.2":GOSUB *ILIST
63150 M$="P1.5":GOSUB *ILIST
63160 *U631
63170 Q=INSTR("123456789Qs",INPUT$(1))
63180 IF Q=0 THEN *U631
63190 IF Q>11 THEN Q=3:STEPBY=-1 ELSE STEPBY=0
63200 IF Q=10 OR Q=11 THEN *QUIT
63210 LINE(50,1)-(80,5),0,B
63220 M$="P1.1":GOSUB *IREMV
63230 M$="P1.2":GOSUB *IREMV
63240 M$="P1.5":GOSUB *IREMV
63250 H3=H9:V3=V9:GOSUB *CURSOR
63260 LINE(50,1)-(80,5.5),0,B
63270 M$="P1.3":GOSUB *ILIST
63280 GOTO *ENDIF401
63290 *QUIT
63300 GOSUB *ENDPRC
63310 *ENDIF401
63320 RETURN
63330 '
63340 *F2INT
63350 H9=POS(0):V9=CSRLIN
63360 LINE(50,1)-(80,5.5),7,B

```

```
63370 M$="P1.6":GOSUB *IREMV
63380 M$="P1.7":GOSUB *ILIST
63390 *U635
63400 B6=INSTR("NnBb",INPUT$(1))
63410 IF B6=0 THEN *U635
63420 IF B6>2 THEN BEEPON=-1 ELSE BEEPON=0
63430 LINE(50,1)-(80,5),0,B
63440 M$="P1.7":GOSUB *IREMV
63450 H3=H9:V3=V9:GOSUB *CURSOR
63460 LINE(50,1)-(80,5.5),0,B
63470 M$="P1.3":GOSUB *ILIST
63480 RETURN
63490 '
63500 '
63510 *MAKEG
63520 READ X50,Y50
63530 PSET (X50,Y50)
63540 READ X5,Y5
63550 WHILE X5<>0
63560 LINE -(X5,Y5),5
63570 READ X5,Y5
63580 WEND
63590 PAINT (X50+.2,Y50+.2),5,5
63600 RETURN
63610 '
63700 *MAKEUPSCRN
63710 M$="P1.3":GOSUB *MLIST
63720 LINE (1,8)-(80,25),7,B
63730 LINE (47,8)-(79,8),0
63740 M$="P0.0":GOSUB *ILIST
63750 RETURN
63755 '
63760 *ENDPRC
63770 CLS 3
63775 IF GWBASIC THEN KEY(1) ON:KEY(2) ON ELSE KEY ON
63780 END
63790 '
63800 *WAITUNTILKEY
63810 M$="P1.6":GOSUB *ILIST
63820 Z$=INPUT$(1)
63830 M$="P1.3":GOSUB *ILIST
63840 RETURN
```