```
SCR # 21
  0 ( 16 bit Numerical Sort Demo                      20AUG82MIM)
  1
  2 FORTH DEFINITIONS  HEX
  3 7000 CONSTANT ARRAY1      ( address of data array)
  4 VARIABLE #ELEMENTS     ( number of 16 bit elements)
  5 VARIABLE DISTANCE      ( distance between elements)
  6 VARIABLE VI            ( temporary indexes for nested DO's)
  7 VARIABLE VJ
  8 VARIABLE SEED   HERE SEED !
  9
 10 : RND           ( random # generator)   ( n --- )
 11    SEED @ 103 * 3 + 7FFF AND
 12      DUP SEED !  7FFF */ ;
 13 : CLRS PAGE CR CR 17 SPACES ." FORTH SORTING DEMO" CR ;
 14 : KEYMSG  ." any key continues.." CR KEY DROP ;
 15
```

```
SCR # 22
  0 ( 16 bit Numerical Sort Demo                      20AUG82MIM)
  1
  2 : RANDOM                   ( create random pattern in ARRAY1)
  3    #ELEMENTS @ 2* 0      ( set loop limit and initial index)
  4       DO 3E8 RND         ( fetch random # between 0 and 999)
  5          I 3 MOD 0= IF NEGATE      ( negate 1 out of three)
  6       THEN I ARRAY1 + !               ( store in array)
  7          2 +LOOP ;                    ( increment loop)
  8
  9 : REVERSE               ( create reversed pattern in ARRAY1)
 10    #ELEMENTS @ 0         ( set loop limit and initial index)
 11       DO #ELEMENTS @ I -               ( compute value)
 12          I 2* ARRAY1 + !             ( store in array)
 13       LOOP ;                         ( decrement loop)
 14
 15
```

```
SCR # 23
  0 ( 16 bit Numerical Sort Demo                      20AUG82MIM)
  1
  2 : NUM(I) @ ARRAY1 + ;            ( array fetch)
  3 : NUMI@ NUM(I) @ ;               ( and store)
  4 : NUMI! NUM(I) ! ;               ( operators)
  5
  6 : COMPARE VI NUMI@ VJ NUMI@ > ;     ( true if #I > #J)
  7
  8 : NUMSWAP          ( swap elements of array)   ( --- )
  9    VI NUMI@ VJ NUMI@ VI NUMI! VJ NUMI! ;
 10
 11 : NUMLIST              ( output number array)   ( --- )
 12    #ELEMENTS @ 2* 0 DO I DUP
 13       1A MOD 0= IF CR THEN ARRAY1 + @
 14          6 .R 2 +LOOP CR CR ;
```

```
 15
```

```
SCR # 24
  0 ( 16 bit Numerical Sort Demo                      20AUG82MIM)
  1
  2 : BUBBLESORT              ( sort data array)   ( --- )
  3    #ELEMENTS @ 1- 2* 0 DO I VI !
  4       #ELEMENTS @ 2* I 2+ DO I VJ !
  5          COMPARE IF NUMSWAP
  6       THEN 2 +LOOP 2 +LOOP ;
  7
  8 : SHUTTLESORT             ( sort data array)   ( --- )
  9    #ELEMENTS @ 1- 2* 0 DO
 10       -2 I DO I DUP VI ! 2+ VJ !
 11          COMPARE IF NUMSWAP ELSE LEAVE
 12       THEN -2 +LOOP 2 +LOOP ;
 13
 14 ( For decending sorts change > in COMPARE to <)
 15
```

```
SCR #25
  0 ( 16 bit Numerical Sort demo                      20AUG82MIM)
  1
  2 : SETDIST              ( set initial distance)    ( --- )
  3    1 BEGIN 2* DUP #ELEMENTS @ >
  4        UNTIL 2- DISTANCE ! ;
  5
  6 : DECDIST             ( decrement distance)   ( --- flag)
  7     DISTANCE @ 2/ 2/ 2* DUP DISTANCE ! 2 < ;
  8
  9     ( Shell-Metzner sort)
 10 : SHELLSORT  SETDIST BEGIN    ( sort data array)  ( --- )
 11    #ELEMENTS @ 2* DISTANCE @ - 0 DO -2 I DO
 12       I DUP VI ! DISTANCE @ + VJ ! COMPARE IF
 13          NUMSWAP ELSE LEAVE THEN DISTANCE @ NEGATE
 14             +LOOP 2 +LOOP DECDIST UNTIL ;
 15
```

```
SCR # 26
  0 ( 16 bit Numerical Sort Demo                      20AUG82MIM)
  1    ( benchmark it)
  2 : #ELEMENTS?  CR ." How many elements?  " QUERY CR CR
  3    INTERPRET #ELEMENTS ! ." random array" RANDOM NUMLIST ;
  4 : REVIT  CR ." reversed array" REVERSE NUMLIST ;
  5
  6 : BUBBS  #ELEMENTS? ." random bubblesort.." CR BEEP
  7    BUBBLESORT BEEP NUMLIST KEYMSG ." sorting sorted array.."
  8    CR BEEP BUBBLESORT BEEP KEYMSG REVIT
  9    ." reverse bubblesort.." CR BEEP BUBBLESORT BEEP
```

```
 10    NUMLIST KEYMSG ;
 11 : SHUTS  #ELEMENTS? ." random shuttlesort.." CR BEEP
 12    SHUTTLESORT BEEP NUMLIST KEYMSG ." sorting sorted array.."
 13    CR BEEP SHUTTLESORT BEEP KEYMSG REVIT
 14    ." reverse shuttlesort.." CR BEEP SHUTTLESORT BEEP
 15    NUMLIST KEYMSG ;
```

```
SCR # 27
  0 ( 16 bit Numerical Sort Demo                      20AUG82MIM)
  1
  2 : SHELS  #ELEMENTS? ." random shellsort.." CR BEEP
  3    SHELLSORT BEEP NUMLIST KEYMSG ." sorting sorted array.."
  4    CR BEEP SHELLSORT BEEP KEYMSG REVIT
  5    ." reverse shellsort.." BEEP SHELLSORT BEEP
  6    NUMLIST KEYMSG ;
  7 : DECODE  DUP 31 = IF BUBBS ELSE DUP 32 = IF SHUTS ELSE
  8    DUP 33 = IF SHELS ELSE 34 = IF QUIT THEN THEN THEN THEN ;
  9 : MENU  CR CR ." Specify sort algorithm:" CR
 10    ." 1 - Bubblesort" CR ." 2 - Shuttlesort" CR
 11       ." 3 - Shellsort" CR ." 4 - Exit demo " BEGIN
 12    KEY DUP 30 > OVER 35 < AND NOT WHILE DROP REPEAT ;
 13
 14 : DEMO  BEGIN CLRS MENU DUP DECODE AGAIN ;
 15         DECIMAL
```