```
/*Insertion Sort*/
T56K

/*M3*/
PFGKIFAFRDLFUFOFE@A6FG@E8FEZPF
/*Print program info*/
@&#!!!!!!*!!INSERTION!SORT
@&#!!!!!!*!!!!!!!!BY
@&#!!!!!!*CHRIS!TOWN#N!*C#M1999
@&@&*INITIALISING!DATA#MMM
..PK

/*set up parameters*/
T45K PF  /*set H to 0*/
T46K PF  /*set N to 0*/
T47K P16F  /*set M to 32*/
T48K P1F  /*set Delta(&) to 2*/



:start:
T start K
GK
/*Initialise Data*/
P65535F
A@
R1024F
R8F
/*
// Data in descending order
U0D
L1F U2D // note : L1F shifts left by TWO (use L0D to shift left by one)!
L1F U4D
L1F U6D
L1F U8D
L1F U10D
L1F U12D
L1F U14D
L1F U16D
L1F U18D
L1F U20D
L1F U22D
L1F U24D
L1F U26D
L1F U28D
L1F U30D
*/

// Data in near random order
U0D
L1024F U2D
R512F U4D
R1024F U6D
L128F U8D
L1024F U10D
R512F U12D
L16F U14D
L512F U16D
R1024F U18D
L512F U20D
L256F U22D
```

```
R1024F U24D
R512F U26D
L1024F U28D
L512F U30D


/* Print message */
O charCR F
O charLF F
O charLetterShift F
O charP F
O charR F
O charE F
O charS F
O charS F
O charSpace F
O charR F
O charE F
O charS F
O charE F
O charT F
O charSpace F
O charT F
O charO F
O charSpace F
O charS F
O charO F
O charR F
O charT F
ZF


/*Sort Data*/

:mainloop:
T vardump D /*clear acc*/

/*increment H by delta, i.e. 2*/
A45F A48F U45F
/*H marks boundary between sorted
 and unsorted part of data*/

/*if H >= 32 then we're finished*/
S47F
E finished F

/*copy H to N and decrement
by delta*/
T vardump D
A45F S48F T46F

:innerloop:

/*compare integers in N and N+2 by creating the instructions
 "A2#N" (addTwo) and "S#N" (SubtractN) for the current value of N */

/*first, clear the instruction, then write the correct value into the
accumulator and copy it*/
T vardump D //ensure that acc is clear
T varcode F // clear varcode
A varseven F //the order code
```

```
L1024F L1F //shift left to from order code
T varcode F // place it in varcode and clear acc
A46F //add N
A vartwo F  //add 2 to address
L0D //shift left 1 position
A varone F // add a one in the last position to indicate a long word
A varcode F // put order code back
T addTwo F //copy to "addTwo"

T varcode F // clear varcode
A varthree F //the order code
L1024F L1F //shift left to form order code
T varcode F // place it in varcode and clear acc
A46F //add N
L0D //shift left 1 position
A varone F // add a one in the last position to indicate a long word
A varcode F // put order code back
T SubtractN F //copy to "SubtractN"


/****self-modifying code****/
:addTwo:
A2#N
:SubtractN:
S#N
/****end of self-modifying code****/


/*swap if necessary, i.e. if acc
now <0,i.e. if val(N+2)-val(N)<0)*/
G swap D

/*else proceed with next main loop*/
T vardump D //ensure that acc is clear
E mainloop F


:swap:
/*perform the swap by creating and executing the following instructions
  for the current value of N :
      A2#N T vardump D
      A#N  T2#N
      A vardump D T#N
  Instructions "addTwo" and "SubtractN" are used where appropriate
*/

T vardump D /*clear acc*/
A addTwo F // copy addTwo to acc
U AA F // create AA from addTwo
S varfour F // modify it (subtract 2 from address field) ...
T BB F //...and copy to BB


// create instructions CC and DD
T varcode F // clear varcode
A varfive F //the order code
L1024F //shift left to from order code
T varcode F // place it in varcode and clear acc
A46F //add N
A vartwo F  //add 2 to address
L0D //shift left 1 position
```

```
A varone F // add a one in the last position to indicate a long word
A varcode F // put order code back
U CC F //copy to "CC"
S varfour F // modify it (subtract 2 from address field) ...
T DD F //...and copy to BB

/****self-modifying code****/
:AA:
A2#N
  T vardump D
:BB:
A#N
:CC:
T2#N
  A vardump D
:DD:
T#N
/****end of self-modifying code****/


/*try decrementing N by delta*/
A46F S48F U46F
E innerloop F
/*else proceed with next main loop*/
T vardump D
E mainloop F


:finished:
ZF

// Constants used
:varone:CONST(1,F)
:vartwo:CONST(2,F)
:varthree:CONST(3,F)
:varfour:CONST(4,F)
:varfive:CONST(5,F)
:varseven:CONST(7,F)
:varcode:CONST(0,F) //variable used for self-modifying code
:vardump:CONST(0,D) //just to clear acc and for temp storage

// Character constants
:charA:CONST(114688,F)
:charB:CONST(118784,F)
:charC:CONST(122880,F)
:charD:CONST(77824,F)
:charE:CONST(12288,F)
:charF:CONST(69632,F)
:charG:CONST(110592,F)
:charH:CONST(86016,F)
:charI:CONST(32768,F)
:charJ:CONST(40960,F)
:charK:CONST(57344,F)
:charL:CONST(102400,F)
:charM:CONST(94208,F)
:charN:CONST(90112,F)
:charO:CONST(36864,F)
:charP:CONST(0,F)
:charQ:CONST(4096,F)
:charR:CONST(16384,F)
:charS:CONST(49152,F)
```

```
:charT:CONST(20480,F)
:charU:CONST(28672,F)
:charV:CONST(126976,F)
:charW:CONST(8192,F)
:charX:CONST(106496,F)
:charY:CONST(24576,F)
:charZ:CONST(53248,F)
:charCR:CONST(73728,F)
:charLF:CONST(98304,F)
:charSpace:CONST(81920,F)
:charLetterShift:CONST(61440,F)


E start KPF
```

```
/*Bubble Sort*/
T56K

/*M3*/
PFGKIFAFRDLFUFOFE@A6FG@E8FEZPF
/*Print program info*/
@&#!!!!!!*!!!BUBBLE!SORT
@&#!!!!!!*!!!!!!!!BY
@&#!!!!!!*CHRIS!TOWN#N!*C#M1999
@&@&*INITIALISING!DATA#MMM
..PK

/*set up parameters*/
T45K PD  /*set H to 1 initially*/
T46K PF  /*set N to 0*/
T47K P16F  /*set M to 32*/
T48K P1F  /*set Delta(&) to 2*/


:start:
T start K
GK
/*Initialise Data*/
P65535F
A@
R1024F
R8F
/*
// Data in descending order
U0D
L1F U2D // note : L1F shifts left by TWO (use L0D to shift left by one)!
L1F U4D
L1F U6D
L1F U8D
L1F U10D
L1F U12D
L1F U14D
L1F U16D
L1F U18D
L1F U20D
L1F U22D
L1F U24D
L1F U26D
L1F U28D
L1F U30D
*/
// Data in near random order
U0D
L1024F U2D
R512F U4D
R1024F U6D
L128F U8D
L1024F U10D
R512F U12D
L16F U14D
L512F U16D
R1024F U18D
L512F U20D
L256F U22D
R1024F U24D
R512F U26D
```

```
L1024F U28D
L512F U30D


/* Print message */
O charCR F
O charLF F
O charLetterShift F
O charP F
O charR F
O charE F
O charS F
O charS F
O charSpace F
O charR F
O charE F
O charS F
O charE F
O charT F
O charSpace F
O charT F
O charO F
O charSpace F
O charS F
O charO F
O charR F
O charT F
ZF


/*Sort Data*/

:mainloop:
T vardump D /*clear acc*/

/*If H=0 then we're finished, i.e. there were no swaps*/
A45F S varone F //check for zero by subtracting one
G finished F //<0?

/*else reset H to 0*/
T vardump D /*clear acc*/
T45F /*H keeps a count of the swaps made during each pass*/

/*set N to -delta*/
T46F S48F T46F
// N points to the pair of values currently being considered


:innerloop:

//add value of delta to N
A46F A48F U46F

/*if N >= 32 then go to next outer loop*/
S47F
E mainloop F

/*compare integers in N and N+2 by creating the instructions
 "A2#N" (addTwo) and "S#N" (SubtractN) for the current value of N */

/*first, clear the instruction, then write the correct value into the
```

```
accumulator and copy it*/
T vardump D //ensure that acc is clear
T varcode F // clear varcode
A varseven F //the order code
L1024F L1F //shift left to from order code
T varcode F // place it in varcode and clear acc
A46F //add N
A vartwo F  //add 2 to address
L0D //shift left 1 position
A varone F // add a one in the last position to indicate a long word
A varcode F // put order code back
T addTwo F //copy to "addTwo"

T varcode F // clear varcode
A varthree F //the order code
L1024F L1F //shift left to form order code
T varcode F // place it in varcode and clear acc
A46F //add N

L0D //shift left 1 position
A varone F // add a one in the last position to indicate a long word
A varcode F // put order code back
T SubtractN F //copy to "SubtractN"


/****self-modifying code****/
:addTwo:
A2#N
:SubtractN:

S#N
/****end of self-modifying code****/


/*swap if necessary, i.e. if acc
now <0,i.e. if val(N+2)-val(N)<0)*/
G swap D

/*else proceed with next inner loop*/
T vardump D //ensure that acc is clear
E innerloop F


:swap:
T vardump D /*clear acc*/

// increase number of swaps
A45F A varone F T45F //increase H by 1

/*perform the swap by creating and executing the following instructions
  for the current value of N :
      A2#N T vardump D
      A#N  T2#N
      A vardump D T#N
  Instructions "addTwo" and "SubtractN" are used where appropriate
*/

T vardump D /*clear acc*/
A addTwo F // copy addTwo to acc
U AA F // create AA from addTwo
S varfour F // modify it (subtract 2 from address field) ...
```

```
T BB F //...and copy to BB


// create instructions CC and DD
T varcode F // clear varcode
A varfive F //the order code
L1024F //shift left to from order code
T varcode F // place it in varcode and clear acc
A46F //add N
A vartwo F  //add 2 to address
L0D //shift left 1 position
A varone F // add a one in the last position to indicate a long word
A varcode F // put order code back
U CC F //copy to "CC"
S varfour F // modify it (subtract 2 from address field) ...
T DD F //...and copy to BB

/****self-modifying code****/
:AA:
A2#N
  T vardump D
:BB:
A#N
:CC:
T2#N
  A vardump D
:DD:
T#N
/****end of self-modifying code****/

//loop
E innerloop F



:finished:
ZF

// Constants used
:varone:CONST(1,F)
:vartwo:CONST(2,F)
:varthree:CONST(3,F)
:varfour:CONST(4,F)
:varfive:CONST(5,F)
:varseven:CONST(7,F)
:varcode:CONST(0,F) //variable used for self-modifying code
:vardump:CONST(0,D) //just to clear acc and for temp storage

// Character constants
:charA:CONST(114688,F)
:charB:CONST(118784,F)
:charC:CONST(122880,F)
:charD:CONST(77824,F)
:charE:CONST(12288,F)
:charF:CONST(69632,F)
:charG:CONST(110592,F)
:charH:CONST(86016,F)
:charI:CONST(32768,F)
:charJ:CONST(40960,F)
:charK:CONST(57344,F)
:charL:CONST(102400,F)
```

```
:charM:CONST(94208,F)
:charN:CONST(90112,F)
:charO:CONST(36864,F)
:charP:CONST(0,F)
:charQ:CONST(4096,F)
:charR:CONST(16384,F)
:charS:CONST(49152,F)
:charT:CONST(20480,F)
:charU:CONST(28672,F)
:charV:CONST(126976,F)
:charW:CONST(8192,F)
:charX:CONST(106496,F)
:charY:CONST(24576,F)
:charZ:CONST(53248,F)
:charCR:CONST(73728,F)
:charLF:CONST(98304,F)
:charSpace:CONST(81920,F)
:charLetterShift:CONST(61440,F)


E start KPF
```