```
10   REM ************************************************************
20   REM *                      ST Secrets series                     *
30   REM *                           by                               *
40   REM *                        COLM COX                            *
50   REM * ---------------------------------------------------------- *
60   REM *        GRAPHICS part one - ARTICLE TWO - LISTING TWO        *
70   REM *                        ST Basic                            *
80   REM *                 PAGE 6 MAGAZINE - ENGLAND                  *
90   REM ************************************************************
100  REM
110  REM This program uses the fact that numbers stored in an integer array
120  REM are stored sequentially.  The integer arrays are used to store the
130  REM Sprite Definition Block and used as the buffer for the background.
140  REM The address of the arrays are then passed to the Line A code.  You
150  REM even use this fact to store m/c programs, in an array!
160  REM
170  REM +-----------------------------+
180  REM !     INITALISE EVERYTHING      !
190  REM +-----------------------------+
200  MAX=2:DIM SPRITE%(37),BACKGROUND%(266*MAX),X(MAX),Y(MAX),XI(MAX),YI(MAX)
210  XI=3:YI=3:TOP=0:BOTTOM=383:LEFT=0:RIGHT=623:X=0:Y=0
220  LINEA=&H7FD00:INIT=0:DRAW=1:REMOVE=2
230  REM +-----------------------------+
240  REM !     READ IN SPRITE DEFINITION !
250  REM +-----------------------------+
260  RESTORE 610:FOR A=0 TO 36:READ A$:SPRITE%(A)=VAL("&H"+A$):NEXT A
270  FOR A=0 TO MAX-1:X(A)=INT(RND(1)*RIGHT):Y(A)=INT(RND(1)*BOTTOM)
280  XI(A)=INT(RND(1)*14)-7:YI(A)=INT(RND(1)*14)-7:NEXT A
290  GOSUB 31000:CALL LINEA(INIT):GOSUB 690
300  REM +-----------------------------+
310  REM !     MAIN LOOP               !
320  REM +-----------------------------+
330  GOSUB DRAW.SPRITES:REM                    DRAW  <-+
340  GOSUB MOVE.SPRITES:REM                    MOVE    !
350  GOSUB ERASE.SPRITES:GOTO 330:REM         ERASE ---+
360  REM +-----------------------------------------------+
370  REM !     DRAW SPRITES IN ORDER: LOW TO HIGH        !
380  REM +-----------------------------------------------+
390  DRAW.SPRITES:FOR A=0 TO MAX-1
400  SPADDR=VARPTR(SPRITE%(0)):BKADDR=VARPTR(BACKGROUND%(0))
410  CALL LINEA(DRAW,X(A),Y(A),SPADDR,BKADDR+A*266)
420  NEXT A:RETURN
430  REM +-----------------------------------------------+
440  REM !     ERASE SPRITES IN ORDER: HIGH TO LOW       !
450  REM +-----------------------------------------------+
460  ERASE.SPRITES:FOR A=MAX-1 TO 0 STEP -1
470  SPADDR=VARPTR(SPRITE%(0)):BKADDR=VARPTR(BACKGROUND%(0))
480  CALL LINEA(REMOVE,BKADDR+A*266)
490  NEXT A:RETURN
500  REM +-------------------------------------------------------------+
510  REM !     MOVE EACH SPRITE ACCORDING TO ITS XI AND YI INCREMENTS    !
520  REM +-------------------------------------------------------------+
530  MOVE.SPRITES:FOR A=0 TO MAX-1:X=X(A):Y=Y(A):XI=XI(A):YI=YI(A)
540  X=X+XI:IF X<LEFT OR X>RIGHT THEN XI=-XI:X=X+XI+XI
550  Y=Y+YI:IF Y<TOP OR Y>BOTTOM THEN YI=-YI:Y=Y+YI+YI
560  X(A)=X:Y(A)=Y:XI(A)=XI:YI(A)=YI:NEXT A
570  RETURN
580  REM +-----------------------------+
590  REM !     SPRITE DEFINITION BLOCK   !
600  REM +-----------------------------+
610  DATA 0,0,0,0,1
620  DATA FFFF,FFFF,FFFF,B001,C003,8001,C003,8001
630  DATA C003,8001,C003,B001,C003,8001,C003,8001
640  DATA C003,8001,C003,B001,C003,8001,C003,8001
650  DATA C003,8001,C003,B001,FFFF,8001,FFFF,FFFF
660  REM +-------------------------------------------------------+
670  REM !     TRASH ON THE SCREEN FOR THE SPRITES TO MOVE OVER    !
680  REM +-------------------------------------------------------+
690  FULLW 2:CLEARW 2:GOTOXY 0,0
700  ?"As you can see, the sprites move in a"
710  ?"non destructive manner. Their speed is"
720  ?"slow for the simple reason that Basic"
730  ?"is slow!"
740  FOR X=0 TO 300 STEP 3:LINEF X,100,150,180:NEXT X
750  RETURN
30990 REM
30991 REM ************************************************************
30992 REM * INSTALL CODE FOR LINE A SPRITES. CALLED BY: CALL (ARGS . .) *
30993 REM * ARG#1 - FUNCTION TO CARRY OUT:  0=INITALISE LINE A FOR USE. *
30994 REM * 1=DRAW SPRITE -> ARG#2=X POS, ARG#3=Y POS, ARG#4=ADDRESS OF *
30995 REM * S.D.B,    ARG#5=ADDRESS OF BUFFER FOR BACKGROUND.   3=ERASE  *
30996 REM * SPRITE - ARG#2=ADDRESS OF BUFFER FOR BACKGROUND. NO RANGE   *
30997 REM * CHECKING IS CARRIED OUT SO ANY OUT OF RANGE VALUES CAN AND  *
30998 REM * MOST LIKELY WILL CAUSE THE SYSTEM TO CRASH!                 *
30999 REM ************************************************************
31000 RESTORE 31003:DEF SEG=523520:ZZZZ=0
31001 READ D:IF D<>-1 THEN POKE ZZZZ,D:ZZZZ=ZZZZ+1:GOTO 31001
31002 RETURN
31003 DATA 78,86,0,0,72,231,255,254,34
31004 DATA 110,0,10,34,17,12,129,0,0
31005 DATA 0,0,103,0,0,30,12,129,0
31006 DATA 0,0,1,103,0,0,30,12,129
31007 DATA 0,0,0,2,103,0,0,46,76
31008 DATA 223,127,255,78,94,78,117,160,0
31009 DATA 76,223,127,255,78,94,78,117,32
31010 DATA 41,0,4,34,41,0,8,32,105
31011 DATA 0,12,36,105,0,16,160,13,76
31012 DATA 223,127,255,78,94,78,117,36,105
31013 DATA 0,4,160,12,76,223,127,255,78
31014 DATA 94,78,117,192,0,0,5,95,77
31015 DATA 65,73,78,0,128,0,5,133,128
31016 DATA 0,3,229,128,0,3,69,0,0
31017 DATA  -1
```