Formula One program

AN IMPORTANT aspect of any computer graphics game on the UK101 is moving considerable amounts of data round the screen writes Toby Walsh of Leighton on Sea in Essex. Basic is too slow in most cases and one is left to either program totally in machine code or using machine-code subroutine, both of which are tedious and time-consuming.

If we look more closely at Basic, we see that there is already a built-in command for moving data around the screen, namely the print command.

When a line feed is executed by a print statement, the screen scrowls up which is, in fact, every character being moved back 64 positions. In the Formula One program, a car is Poked into a position on the top of the screen and cars and barriers marking the sides of the course are scrowled up at you, until either you crash or you cross the finishing line.

Line numbers 100-180 are the instructions

180 POKE 11,0 : POKE 12, 253: X = USR(X)

This jumps to a subroutine pointed to by the contents of locations 11 and 12 which, in this case, waits for a character to be input from the keyboard before returning. 190-220 initialises all the variables and draws the barriers for the start of the game. 230-280 is the game itself. 230-250 allows you to move left or right within the constraints of the barriers — the keys used for that are the two large shift keys. 160 randomly Pokes a car at random intervals on the bottom line of the screen. 290-350 prints the loose message. 360-370 Pokes the finishing line on to the bottom line of the screen. 380-400 prints the win message. 410-470 deals with another go, and the goodbye message.

Variables
Numerical

A	Loop variable
B	Position of left barrier on bottom line
C	Position of right barrier on bottom line
D	Counter for time of play
K	Location of keyboard
S	Position of player's car
SS	Position of top-left-hand corner of screen
X	Argument

Strings

A$ Answer string to an input.

```
10 REM    **************************************
20 REM    *****        FORMULA ONE        *****
30 REM    *****           14/5/80         *****
40 REM    *****       for the Compukit    *****
50 REM    *****        to run in 4K       *****
60 REM    **************************************
100 FORA=1TO16:PRINT:NEXT
110 PRINT"            FORMULA  ONE"
120 PRINT"           ============="
130 PRINT:PRINT:PRINT"You are driving";
140 PRINT" a Formula One car at Brands"
150 PRINT"Hatch.Left shift moves you left";
160 PRINT" and right "
165 PRINT"shift moves you right"
170 PRINT:PRINT"          Hit any key"
180 POKE11,0:POKE12,253:X=USR(X):K=57100:D=0
190 S=53281:B=54236:C=54247:SS=53261
200 FORA=1TO16:PRINT:NEXT:FORA=1TO15
210 POKESS+A*64-49,161:POKESS+A*64-38,161
220 NEXT
230 POKES,1:A=PEEK(K)
240 IFA=250ANDPEEK(S+63)=32THENS=S-1
250 IFA=252ANDPEEK(S+65)=32THENS=S+1
260 IFRND(1)>.4THENPOKEB+1+RND(1)*10,2
270 D=D+1:POKEB,161:POKEC,161:PRINT
280 IFPEEK(S)=32ANDD<200THEN230
290 IFPEEK(S)=61THEN380
300 IFD=200ANDPEEK(S)=32THEN360
310 FORA=1TO1000:NEXT:FORA=1TO16:PRINT:NEXT
320 PRINT "              CRASH !!!!"
330 PRINT:PRINT
340 PRINT"        Better luck next time"
350 GOTO420
360 FORA=1TO10:POKEB+A,61:NEXT:D=0
370 GOTO230
380 FORA=1TO1000:NEXT:FORA=1TO16:PRINT:NEXT
390 PRINT"            You win":PRINT
400 PRINT:PRINT
420 INPUT"          Another go";A$
430 IFLEFT$(A$,1)="Y"THENRUN
440 FORA=1TO16:PRINT:NEXT
450 PRINT "              BYE "
460 FORA=1TO16:FORB=1TO500:NEXT:PRINT:NEXT
470 END
```

Re-sequencing lines

THE PROGRAM will re-sequence the line numbers in a Basic program on the Superboard or UK101, writes Michael Whittle of Oxford. It is designed to be left in memory above the program being developed, and called into action by RUN63000.

The program re-sequences the line numbers in 10s and corrects all GOTO and GOSUB instructions, including ON . . . GOTO and ON . . . GOSUB. It may be adapted for use on any Microsoft Basic by changing the value of AD to the address of the fourth byte of the Basic storage area.

The program is restricted by re-numbering 100 lines, although for computers with sufficient memory, that could be changed by corrections of the values in lines 63010, 63050 and 63060. If the program needs to write a label into a space which is too small, for example, changing GOTO5 to GOTO10, the line number is given for subsequent re-typing.

```
63000 REM RENUMBERING PROGRAM
63010 DIMA(100):AD=771:FORI=1TO100
63020 GOSUB63230:IFLN>62999THEN63060
63030 N=I*10:NH=INT(N/256):NL=N-256*NH
63040 POKEAD,NL:POKEAD+1,NH:A(I)=LN
63050 AD=NA:NEXTI:PRINT"OVER 100 LINES"
63060 AD=771:FORB=1TO100:GOSUB63230
63070 IFLN>62999THENPRINT"COMPLETE":END
63080 FORJ=AD+2TONA-4:C=PEEK(J)
63090 IFC<>136ANDC<>140ANDC<>160THEN63220
63100 L=PEEK(J+1):IFL<48ORL>57THEN63220
63110 C$="":FORK=J+1TOJ+8:C=PEEK(K)
63120 IFC<48ORC>57THEN63140
63130 C$=C$+CHR$(C):NEXTK
63140 L=VAL(C$):FORH=1TOI:IFA(H)=LTHEN63160
63150 NEXTH:PRINT"LIN";B*10;"LAB";L:GOTO63210
63160 N$=STR$(H*10):R=LEN(N$):FORX=2TOR
63170 POKEK+X-R-1,ASC(MID$(N$,X,1)):NEXTX
63180 IFK-R<JTHENPRINT"OVERWRITTEN LINE";B*10
63190 IFK-J-R<=0THEN63210
63200 J=J+1:POKEJ,32:GOTO63190
63210 IFC=44THENJ=K:GOTO63110
63220 NEXTJ:AD=NA:NEXTB:PRINT"INCOMPLETE":END
63230 NA=PEEK(AD-1)*256+PEEK(AD-2)+2
63240 LN=PEEK(AD+1)*256+PEEK(AD):RETURN OK
```