That's life

More machine code madness as Auntie John Kennedy demonstrates the secrets of life itself. Next month it's back to the serious stuff.

Life must be one of the great computer programming games of all time. It was invented by the mathematician John Conway in the 1970s and is a program which, by means of a few simple rules, creates patterns of immense complexity, modelling life itself.

Life simulates a colony of cells, showing how they are born, grow and die in an idealized environment.

The rules which govern the game are as follows: (1) To survive to the next time cycle, each cell must have either two or three neighbours. Less than two and it dies from loneliness, greater than three and it starves. (2) If 3 cells get together, a new cell is born.

Although these rules may seem trivial, they were decided upon by Conway after studies of the many possibilities. They are not by any stretch of the imagination an accurate representation of a cell culture, only a means to explore such computer generated cellular automata. In other words, it's not realistic, but it looks good.

For your edification and delight, we present you with not one, but two listings for Life. The first is written in Locomotive Basic, mainly to allow you to get a general feel for the formulae involved. It creates a random colony and allows it to grow. Slowly.

The second listing is a specially written machine code masterpiece, operating many, many times faster. It is well worth the time and effort needed to type it in, but remember to save it before running, just in case you have made a mistake. Successive generations appear very quickly indeed, fast enough to watch the patterns grow and change without having to pop out for a cup of tea between frames. There is also an editor built in, so you can design your own cultures and watch them flourish or vanish.

Here are several common patterns that you can experiment with:

1. The Blob – boring, this one. Just sits and does nothing.

   oo
   oo

2. Spinner – it goes round, and round, and round . . .

   ooo

3. Glider – crawls its way off the screen.

   o
    o
   ooo

4. Honey Farm – this shape evolves into four beehives.

   ooooooo

Instructions for machine code Life: When first run, the program is in editing mode. From here you can:

– move the cursor around the screen with the arrow keys
– add or remove cells with the control and copy keys
– start the cells generating with the G key (stop with Space)
– clear away all the cells with the C key
– create a random pattern with the R key
– return to Basic by pressing the Escape key.

Auntie John will be continuing his machine code tutorial in next month's issue. This month's effort is by way of a little light relief. Auntie sinned some what in the March tutorial in the listings. Line 20 in both 'Listing 1 – Excitement and really wild things' and 'Listing 2 – More excitement and really wild things' should read '20 READ X$', leaving off all the other gobbledegook.

```
1 REM ========================
2 REM =   Life by AJ  1990   =
3 REM ========================
4 REM
5 REM Remember to SAVE this
6 REM before running
7 REM
10 MEMORY &7FFF
20 add=&8000
30 READ a$
40 IF a$="stop" THEN CALL &8000
50 FOR b=1 TO LEN(a$) STEP 2
60 c=VAL("&"+MID$(a$,b,2))
70 POKE add,c
80 add=add+1
85 NEXT
90 GOTO 30
100 DATA 3E01CD0EBC210000
110 DATA 22AD8A21DD8222AF
120 DATA 8A011010CD38BCDD
130 DATA 21DD82FD21C5863E
140 DATA 32CD1EBBC482803E
150 DATA 34CD1EBBC489803E
160 DATA 3ECD1EBBC492803E
170 DATA 42CD1EBB205BCD04
180 DATA 81CD19BDCD19BDCD
190 DATA 0F81CD19BDCD19BD
200 DATA 3E00CD1EBBC49F80
210 DATA 3E02CD1EBBC4B880
220 DATA 3E01CD1EBBC4E280
230 DATA 3E08CD1EBBC4CE80
240 DATA 3E17CD1EBBC4F680
250 DATA 3E09CD1EBBC4FD80
260 DATA 188FCDF081CDBC81
270 DATA C9010101CD38BCC3
280 DATA 2681CD0F82CDBC81
290 DATA C93E01CD0EBCC92A
300 DATA AD8A7CFE00C82522
310 DATA AD8A2AAF8A012800
320 DATA 373FED4222AF8AC9
330 DATA 2AAD8A7CFE18C824
340 DATA 22AD8A2AAF8A0128
350 DATA 000922AF8AC92AAD
360 DATA 8A7DFE00C82D2D22
370 DATA AD8A2AAF8A2B22AF
380 DATA 8AC92AAD8A7DFE4E
390 DATA C82C2C22AD8A2AAF
400 DATA 8A2322AF8AC93E01
410 DATA 2AAF8A77C93E002A
420 DATA AF8A77C92AAD8ADD
430 DATA 21BD82CD2182C92A
440 DATA AD8AED5BAF8A1AFE
450 DATA 01CCE6811AFE00CC
460 DATA EB81CD2182C9DD21
470 DATA DD82FD21C586CDBC
480 DATA 81DD21DD82FD21C5
490 DATA 86CD60813E2FCD1E
500 DATA BBC0DD21C586FD21
510 DATA DD82CDBC81DD21C5
520 DATA 86FD21DD82CD6081
530 DATA 3E2FCD1EBBC018C6
540 DATA 01E803C5DDE5E101
550 DATA 2900373FED427E23
560 DATA 8623860126000986
570 DATA 2323860126000986
580 DATA 2386238647DD7E00
590 DATA FE01CCA281DD7E00
600 DATA FE00CCB181DD23FD
610 DATA 23C10B78B1FE0020
620 DATA C2C978FE02280FFE
630 DATA 03280B3E00FD7700
640 DATA C978FE0320F53E01
650 DATA FD7700C906000E00
660 DATA 6960DD7E00DDE5FE
670 DATA 00C4E681CCEB81C5
680 DATA CD2182C1DDE1DD23
690 DATA 0C0C79FE5020E104
700 DATA 78FE1920D9C9DD21
710 DATA AD82C9DD21CD82C9
720 DATA 11DD8201E803ED5F
730 DATA E610FE1028053E00
740 DATA 1218033E0112130B
750 DATA 78B1FE0020E8C911
760 DATA DD8201E8033E0012
770 DATA 130B78B1FE0020F5
780 DATA C9DDE55D7CCB2721
790 DATA 7B8206004F097E4F
800 DATA 237E476B260009D1
810 DATA 01FF071A7723131A
820 DATA 7713091A7723131A
830 DATA 7713091A7723131A
840 DATA 7713091A7723131A
850 DATA 7713091A7723131A
860 DATA 7713091A7723131A
870 DATA 7713091A7723131A
880 DATA 7713091A7723131A
890 DATA 7713C900C050C0A0
900 DATA C0F0C040C190C1E0
910 DATA C130C280C2D0C220
920 DATA C370C3C0C310C460
930 DATA C4B0C400C550C5A0
940 DATA C5F0C540C690C6E0
950 DATA C630C780C7010803
960 DATA 0C070E070E070E03
970 DATA 0C01080000F0F0F0
980 DATA F0F0F0F0F0F0F0F0
990 DATA F0F0F0F0F0000000
1000 DATA stop
```

```
10 REM The Game of Life
20 REM This version © 1990 John Kennedy
30 REM for Amstrad Computer User
40 REM
50 MODE 1
60 PRINT "Life"
70 WINDOW £0,10,30,3,22
80 DEFINT a-z
90 '
100 DIM c(21,21),d(21,21)
110 FOR a=1 TO 80:d(INT(RND*20)+1,INT(RND*20)+1)=1:NEXT a
120 '
130 GOSUB 170
140 GOSUB 250
150 GOTO 130
160 '
170 REM Draw Colony
180 LOCATE 1,1
190 FOR y=1 TO 20:FOR x=1 TO 20
200 IF d(x,y)=1 THEN PRINT "0"; ELSE PRINT " ";
210 c(x,y)=d(x,y)
220 NEXT:PRINT:NEXT
230 RETURN
240 '
250 REM Update Colony
260 FOR x=1 TO 20:FOR y=1 TO 20
270 n=0
280 FOR p=x-1 TO x+1:FOR q=y-1 TO y+1
290 n=n+c(p,q)
300 NEXT:NEXT
310 b=0
320 IF c(x,y)=0 AND n=3 THEN b=1
330 IF c(x,y)=1 AND (n=3 OR n=4) THEN b=1
340 d(x,y)=b
350 NEXT:BORDER 20-x:NEXT
360 RETURN
```