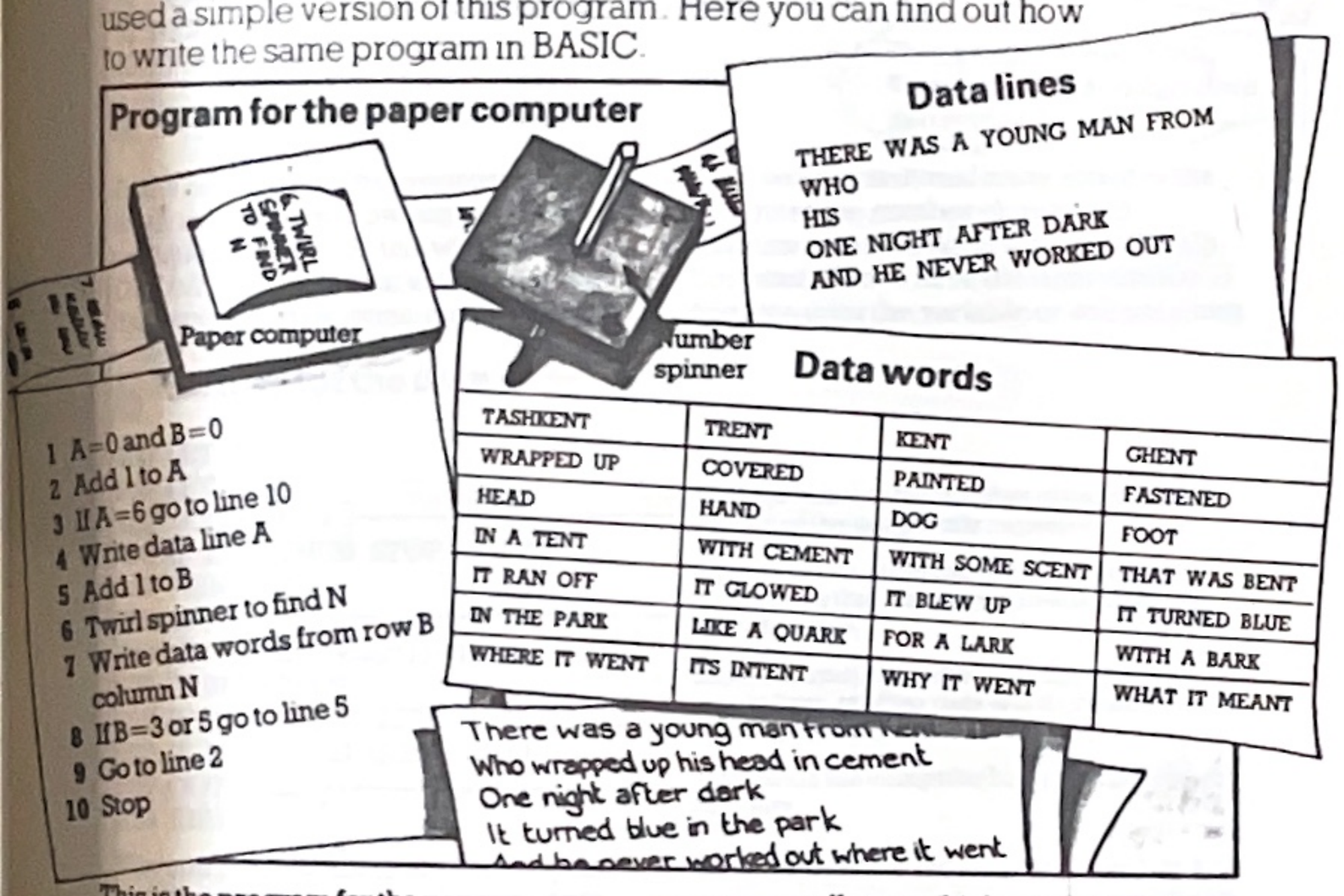
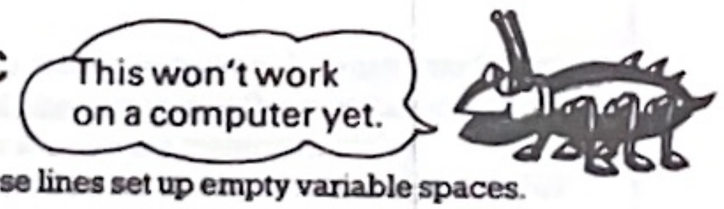
Funny poems program

The next few pages show you how to write a program which can compose lots of poems. A version of this program first appeared in the *Usborne Guide to Computers*. That book showed how to make a "paper computer" which used a simple version of this program. Here you can find out how to write the same program in BASIC.



This is the program for the paper computer. It looks a little like BASIC, but it would not work on a real computer. The words and phrases for the poem are "stored" on pieces of paper and the program tells you which to select. The number spinner is a random number generator to give random numbers between one and four.

1 Translating the program into BASIC




```
10 LET A=0
20 LET B=0
30 LET A=A+1
40 IF A=6 THEN STOP
50 Write data line A
60 LET B=B+1
70 LET N=INT(RND(1)*4+1)
80 Write data words from row B
   column N
90 IF B=3 THEN GOTO 60
100 IF B=5 THEN GOTO 60
110 GOTO 30
120 END
```

These lines set up empty variable spaces.

Lines 30 and 40 keep count of the number of data lines the computer has selected.

Lines 50 and 80 are not in BASIC yet.

Line 60 keeps count of the number of data words.

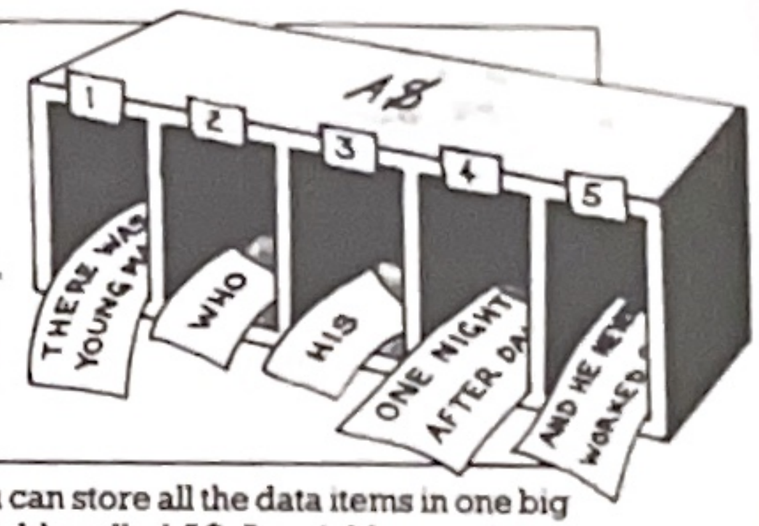
Gives a random number between 1 and 4

Lines 90 and 100 send it back to select another data line.

Most of the program is easy to translate into BASIC, but lines 50 and 80 are more difficult. The computer needs a way of storing and picking out the data lines and words which are needed for each line of the poem.

2 Giving the computer data

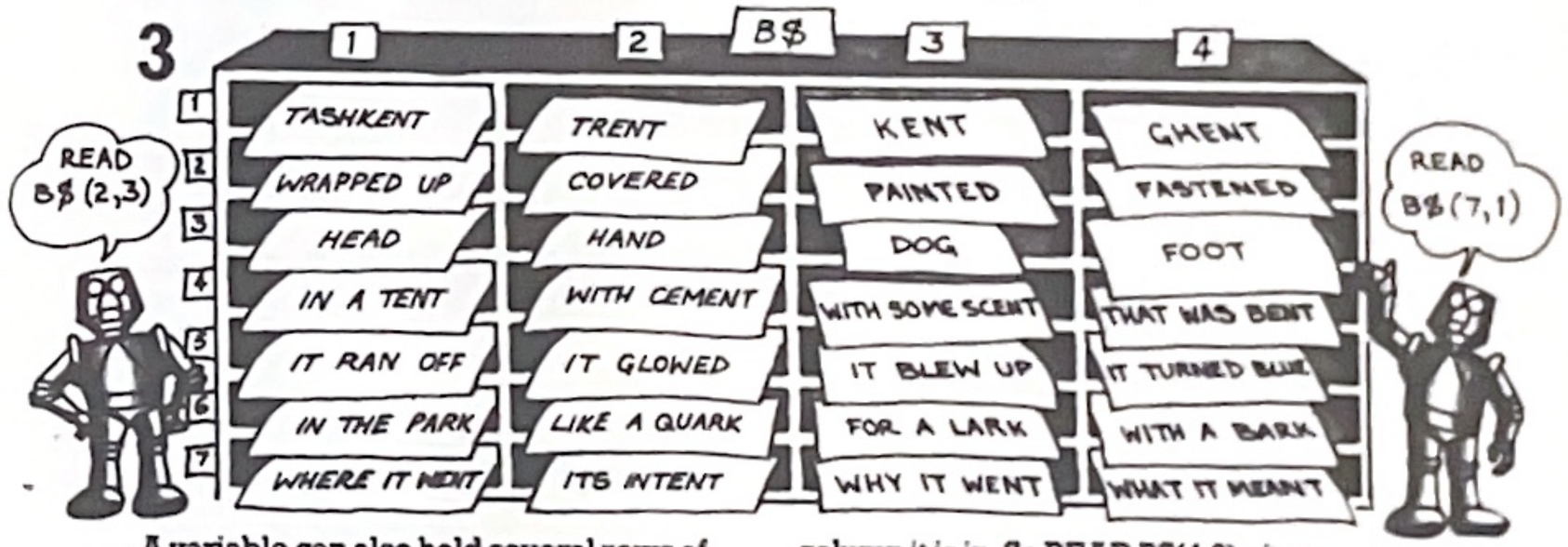
```
50 READ A$

180 DATA THERE WAS A YOUNG MAN FROM,
    WHO, HIS
190 DATA ONE NIGHT AFTER DARK, AND HE
    NEVER WORKED OUT
```



To give the computer the data lines and words you can use READ . . . DATA. Each time the computer carries out the READ instruction it takes another item from the DATA line and stores it in the variable.

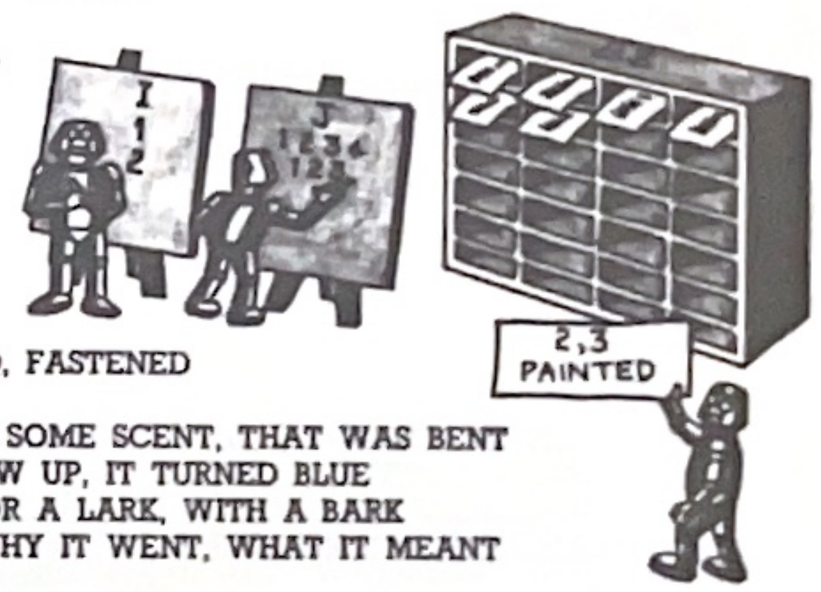
You can store all the data items in one big variable called A$. A variable containing more than one data item is called an array and each item is referred to by a number, e.g. READ A$(3) gives HIS.*

3



A variable can also hold several rows of data and you can store all the data words in a variable like this. It is called a two-dimensional array. Here, each data item is referred to by the number of the row and column it is in. So READ B$(4,2) gives WITH CEMENT and READ B$(6,3) gives FOR A LARK. You can store numbers in arrays, too, using a number variable, e.g. N(5,7).

4 Putting the data in the variables




```
10 FOR I=1 TO 7
20 FOR J=1 TO 4
30 READ B$(I, J)
40 NEXT J
50 NEXT I
60 DATA TASHKENT, TRENT, KENT, GHENT
70 DATA WRAPPED UP, COVERED, PAINTED, FASTENED
80 DATA HEAD, HAND, DOG, FOOT
90 DATA IN A TENT, WITH CEMENT, WITH SOME SCENT, THAT WAS BENT
100 DATA IT RAN OFF, IT GLOWED, IT BLEW UP, IT TURNED BLUE
110 DATA IN THE PARK, LIKE A QUARK, FOR A LARK, WITH A BARK
120 DATA WHERE IT WENT, ITS INTENT, WHY IT WENT, WHAT IT MEANT
```

I is the row number

J is the column number

To read each data item into the variable you need to be able to alter the numbers in brackets after READ. You can do this with loops. B$ needs nested loops as shown above with an I loop for the row number and a J loop for the column number. Each time the I loop is carried out the J loop is repeated four times – once for each of the columns in a row.

*Sinclair computers deal with variables in a different way and this program will not run on a Sinclair. You can find out more about this over the page.