Original Author

It is possible to write computer programs that will themselves generate other programs, or correct errors in human coding

'If computers are so smart, how come they need humans to program them?' Experienced computer users will tend to shrug off a question like this from a sceptical newcomer, but it is not as silly as it seems. Much research is being undertaken in writing programs that can generate other programs, and operating systems that can correct bugs in code written by humans.

Consider the SYNTAX ERROR? message, which is frequently encountered by home computer users. This can be infuriating because the message gives you so little information. A compiler on a large mainframe computer will generally give far more information as to the nature of the error encountered. For example, the error message could read:

```
1090 LET A=(C*2+F$)*((FG—C)*TH+1))
                ↑                ↑
ERRORS: 1) MISMATCH — STRING VARIABLE F$
           NOT ALLOWED
        2) LAST CLOSE BRACKET NOT EXPECTED
```

There is no fundamental reason why such techniques cannot be used in an interpreter on a home computer — the cost of the extra ROM needed to store the routines would be minimal. But few home computers employ even cursory error monitoring procedures: most don't even check the syntax of the code as it is entered. However, it is often possible to buy additional ROM chips or plug-in software cartridges that will extend the range of BASIC commands available, particularly those related to the development and de-bugging of programs. These BASIC commands include:

HELP — prints out the program line and highlights the exact character position where program execution terminated. This will usually, but not always, indicate the source of the syntax error.

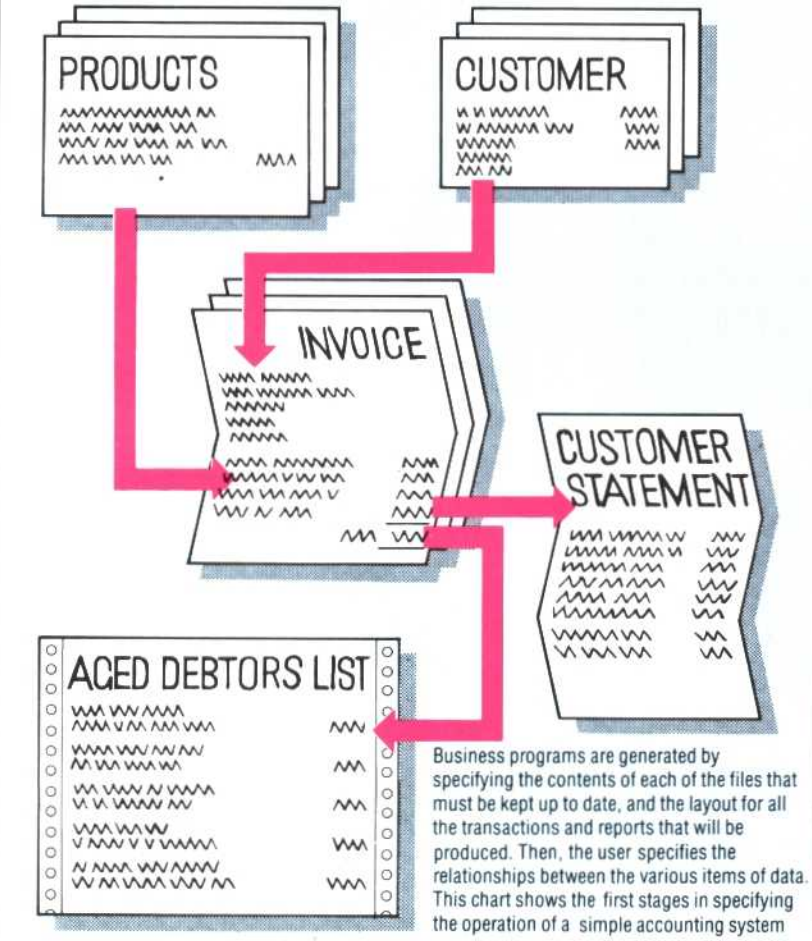
DUMP — prints a list of all variable names and their contents currently in use by the program. This is helpful in deducing how far the program had got in its task before the error occurred.

TRACE — displays (in a window in the corner of the screen) the line number (or numbers) currently being executed while the program is running. This helps the user to trace the flow of the program, and ensures, among other things that subroutines are being executed in the desired order.

Writing programs that allow a computer to correct human coding errors is, in general, not a simple task. But in the case of some errors it is fairly easy. For example, we know that all program lines have to start with a BASIC keyword (though some machines will allow you to drop the word LET). Therefore, if a line begins with PRUNT or PRONT, it would be easy to work out that it should say PRINT. In the Basic Programming course we have discussed the idea of fuzzy matching (algorithms for choosing the closest match to any phrase) and this could be applied to program keywords as well. Alternatively, the interpreter could simply include a list of common typing errors, and their correct equivalents. For safety, it would be desirable for the computer to check any alterations it makes with the operator.

But beyond these basic procedures, automatic correction becomes a great deal more difficult. In

Paper Money



Business programs are generated by specifying the contents of each of the files that must be kept up to date, and the layout for all the transactions and reports that will be produced. Then, the user specifies the relationships between the various items of data. This chart shows the first stages in specifying the operation of a simple accounting system

LIZ DIXON