Language Lab

To conclude our course, we take a critical look at the Basic language, and at some of the alternatives to it

As a postscript to our Basic Programming course, we would like to discuss briefly some of the strengths and weaknesses of BASIC compared with other programming languages.

BASIC is an off-shoot of FORTRAN, one of the earliest programming languages. Unlike most other languages, BASIC is interpreted. This means that when a BASIC program is executed, a special program elsewhere in the computer's memory interprets the code line by line and converts the BASIC statements into machine code. Here's what would happen in a short BASIC program like this:

```
10 CLS
20 PRINT "TYPE IN A NUMBER"
30 INPUT X
40 PRINT "TYPE IN A SECOND NUMBER"
50 INPUT Y
60 "PRINT "THE PRODUCT OF THE TWO NUMBERS
   IS: ";
70 PRINT X*Y
80 PRINT
90 PRINT "DO YOU WANT ANOTHER GO?"
100 PRINT "PRESS 'Y' TO TRY AGAIN"
110 PRINT "OR 'N' TO END"
120 FOR X = 1 TO 1
130 LET A$ = INKEY$
140 IF A$ < > "Y" AND A$ < > "N" THEN X = 0
150 NEXT X
160 IF A$ = "Y" THEN GOTO 10
170 END
```

When the BASIC interpreter encountered line 10 it would work out the machine code needed to clear the screen. For line 20 it would work out the machine code instructions necessary for sending the TYPE IN A NUMBER message to the screen. For line 30 it would set up the memory space needed to store a real number, wait for input from the keyboard and then convert the number typed in into binary and store it in the space allocated to variable X. All this would be repeated for lines 40 to 60. If the user wanted to repeat the program by typing Y, the interpreter would branch back to line 10 and repeat all the calculations and computations again.

Most languages other than BASIC are 'compiled'. This means that after the program has been written it is processed by a 'compiler' before it can be run. The compiler is a separate program that goes through the 'source code' (the original program) and produces a second version of it in machine code. When the compiled program is run, it is likely to work very much faster than an interpreted program because all the time-consuming translations into machine code have already been done.

If compiled programs work so much faster than interpreted programs, you might wonder why all programming languages don't use compilers. There are several advantages to using interpreted programs, such as BASIC. Most of these stem from the fact that it is an interactive language, which is one that can be tested and de-bugged 'at the keyboard' while the program is being developed. BASIC, for example, allows the STOP command to be inserted at any point in the program. When the interpreter encounters a STOP statement, it stops interpreting the program and allows 'commands' to be issued from the keyboard.

Commands are instructions that can be directly executed by the interpreter when the program is not running. BASIC is provided with a large number of these and they can be invaluable in de-bugging. After a BASIC program has been executed (i.e. the interpreter has encountered the END statement) or when the interpreter encounters a STOP statement, it is possible to PRINT the values of all the variables. Try running the address book program, for example. Run the program and type 9 to exit from the program. If it runs all the way through without any error messages appearing, it should end with the BASIC prompt (this is usually an OK, > or *). Then type PRINT RMOD<CR>. The interpreter should print a 0 on the screen (provided you have not added any records!). Then try PRINT SIZE<CR>. The interpreter will print a number on the screen one larger than the number of records you have in the data file.

Basic's Advantages

BASIC is often called the ideal language for the inexperienced programmer because it allows bugs to be removed at the keyboard. It has another great advantage: it is comparatively easy to learn. For example, in the Basic Programming course, we have covered all the fundamentals and many of the advanced aspects of BASIC in just 86 pages. Syntax errors such as 40 PRNT A(12) will usually result in easily understandable error messages when the program is executed, such as SYNTAX ERROR IN 40. A glance at the line number referred to usually makes it clear where the error lies, and rectifying the error is usually no more difficult than typing EDIT 40<CR> (followed by a