Message Understood

Press a key, and hidden layers of software spring into action – decoding your instructions, searching the memory and all the while scanning the keyboard for your next command

A computer is an assembly of metal, plastic and silicon which, without a program in its memory, is incapable of performing any sort of useful task — just like a record player with no record on the turntable. The process of getting the computer to perform the specific task you require is therefore known as 'programming'. Even the relative beginner to programming will be able to identify two distinct phases to solve a problem. First the problem must be translated and written down in a form which the computer can understand. Secondly, this program must be fed into the computer and 'run'. These two phases can be further subdivided into two stages — the programmer himself goes through the first stage, while in stage two the computer must take the actions (usually without the user's knowledge or intervention).

Suppose that you want to write a program to prepare a payroll. The first thing you need is a perfectly clear understanding of the problem. What do you require as your output from the computer? What information will the computer need to produce the weekly payslips? This may entail salaries, hours worked this week, etc. The next essential is to specify the process by which this output is going to be produced, for example: 'How are tax and pension contributions calculated?'

In a large business application, this may well be done by a trained 'Systems Analyst' — whose speciality is analysing the way a business operates and writing it down in a form which can easily be translated into a program. For home or educational programs, all this would usually be done by the programmer himself.

If computers could understand plain English then this 'program specification' could be run straight away; but unfortunately they cannot, yet. Many beginners experience difficulties because they try to write the program from start to finish — in the same way that you might translate an English essay into French. Experienced programmers, however, break this stage down even further. They might divide the payroll specification into four 'modules': for inputting the week's data, calculation, storing the cumulative results like 'tax paid this year', and printing out the payslips.

Each module can then be broken down into smaller structures. This is called 'structured programming'; each of these smaller sections is simple and can be expressed as one or two lines in a program. Finally, the whole collection of lines — the program listing — is typed into the computer.

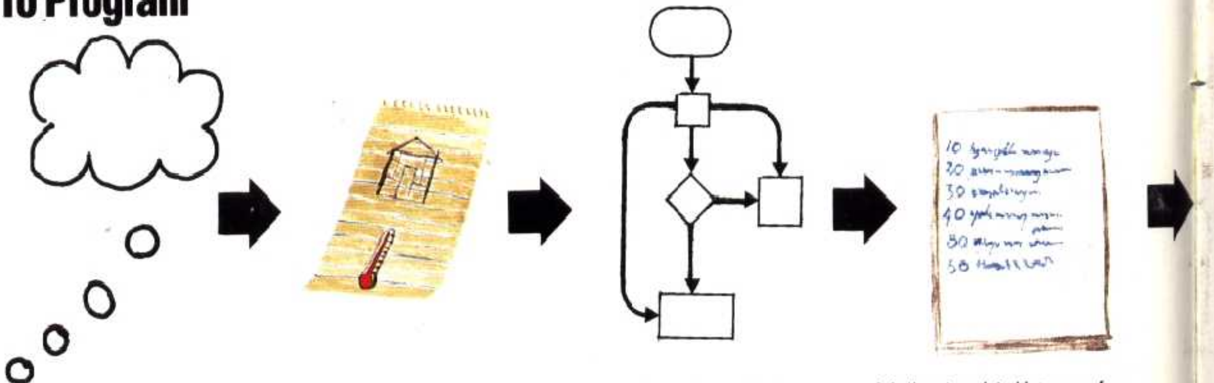
A good programmer always keeps notes from every stage and these reflect the many distinct levels from a problem written in English to a program written in a high-level language such as BASIC.

What happens from the moment you type RUN is entirely under the control of the computer and again involves many different layers or stages. However, the internal operations of the computer are 'hidden' — all the user is aware of is his program asking him for relevant information and producing the required output.

Because the microprocessor cannot understand a high-level language, the prime task ahead of the

From Problem To Program

The origin of a computer program begins with realising there is a problem that needs to be solved; in this case how to keep the temperature of the greenhouse at a constant level. In order to obtain the answer, this problem needs to progress through several layers of processing which results in the completed program



The problem arises . . .

The idea is scribbled down roughly on a piece of paper

A flowchart is formulated to analyse the problem and to develop the structure of the program . . .

It is then translated into one of the computer languages, for example, BASIC